Tema principal:
Inteligencia Artificial aplicada al análisis de datos educativos.

Monografía principal.

Alumno:
Lic. A.S. Alfredo Poggio Manzor
Tutor:
Prof. Ricardo Nagel Rodrigues.

UTEC
Universidad Tecnológica del Uruguay - ITR Norte - Rivera, Uruguay.

FURG
Universidade Federal do Rio Grande do Sul - RS - Brasil.

Septiembre, 2022

Indices general:

Indices general: 2
Capitulo 1 - Introducción....4
Objetivos generales:....4
Objetivos específicos:....4
Justificación:....5
Caracterización del problema:....5
Contextualización: 6
Estructura de Trabajo: 7
Capitulo 2 - Marco Teórico 8
Trabajos relacionados:....8
Antecedentes académicos y de la investigación. 8
Fundamentación teórica: 8
Bases teóricas....8
Revisión bibliográfica, Referencias bibliográficas:....8
Metodologías, técnicas y herramientas a utilizar:....9
¿Cómo debo leer la predicción realizada con inteligencia artificial de cada dimensión de la educación ?: 9
Proceso y Tratamiento de los datos:.... 10
Entrenamiento de una Red Neuronal Utilizando Perceptron Multi Layer con Keras TensorFlow: 13
Comparación entre las tecnologías utilizadas: SKLearn y Tensor Flow Keras: 16
Resultados esperados 16
Público implicado 16
Capitulo 3 - Desarrollo del Proyecto. Predicción para el 2022 y el 2023 para la educación, en forma cuantitativa. 17
Primera categoría de conjuntos de datos a analizar por IA de la realidad:.... 17
Por formas de Administración: 17
Cantidad de ingresos y egresos en carreras universitarias de grado según forma de administración. Total país 17
Cantidad de alumnos matriculados en universidades según forma de administración. Total país 22
Prioridad macroeconómica del Gasto Público en Educación 24
Tasa de desempleo según nivel educativo. Total país.... 25
Segunda categoría de conjuntos de datos a analizar por IA de la realidad: 28
Por Sexo: 28
Porcentaje de jóvenes que finalizaron primaria según sexo. Total país 28
Porcentaje de jóvenes de 18 y más años que finalizaron secundaria (6° de liceo/UTU) según sexo. Total país31
Tasa de analfabetismo de las persona de 15 y más años según sexo. Total país 33
Tasa de actividad según sexo por nivel educativo. Total país 35
Tercera categoría de conjuntos de datos a analizar por IA de la realidad:.... 41
Por ascendencia racial: 41
Tasa de desempleo según sexo por ascendencia afro. Total país.... 41
Porcentaje de personas en situación de pobreza según sexo por ascendencia afro. Total país 43
Tasa de empleo según ascendencia afro. Total país 46
Cuarta categoría de conjuntos de datos a analizar por IA de la realidad: 47
Situación de pobreza: 47
Tasa de empleo por sexo y situación de pobreza 47
Tasa de desempleo según sexo por situación de pobreza. Total país 50
Capitulo 4 - Conclusiones - Predicción para el 2022 y el 2023 para la educación, en forma cualitativa.... 53
Cantidad de ingresos y egresos en carreras universitarias de grado según forma de administración. Total país 53
Cantidad de alumnos matriculados en universidades según forma de administración. Total país 53
Prioridad macroeconómica del Gasto Público en Educación 54
Tasa de desempleo según nivel educativo. Total país.... 54
Porcentaje de jóvenes que finalizaron primaria según sexo. Total país 54
Porcentaje de jóvenes de 18 y más años que finalizaron secundaria (6° de liceo/UTU) según sexo. Total país54
Tasa de analfabetismo de las persona de 15 y más años según sexo. Total país 54

Tasa de actividad según sexo por nivel educativo. Total país 55
Tasa de desempleo según sexo por ascendencia afro. Total país 55
Porcentaje de personas en situación de pobreza según sexo por ascendencia afro. Total país 56
Tasa de empleo según ascendencia afro. Total país 56
Tasa de empleo por sexo y situación de pobreza 56
Tasa de desempleo según sexo por situación de pobreza. Total país 56
Conclusiones finales obtenidas con el entrenamiento de modelos con inteligencia artificial. 57
Anexo, Trabajo futuro 58
Anexo, Códigos Fuentes. 58
Cantidad de ingresos y egresos en carreras universitarias de grado según forma de administración. Total país 58
Cantidad de alumnos matriculados en universidades según forma de administración. Total país 66
Prioridad macroeconómica del Gasto Público en Educación 71
Tasa de desempleo según nivel educativo. Total país 73
Porcentaje de jóvenes que finalizaron primaria según sexo. Total país 85
Porcentaje de jóvenes de 18 y más años que finalizaron secundaria (6° de liceo/UTU) según sexo. Total país 90
Tasa de analfabetismo de las persona de 15 y más años según sexo. Total país 95
Tasa de actividad según sexo por nivel educativo. Total país 100
Tasa de desempleo según sexo por ascendencia afro. Total país 126
Porcentaje de personas en situación de pobreza según sexo por ascendencia afro. Total país 136
Tasa de empleo por sexo y situación de pobreza 143
Tasa de desempleo según sexo por situación de pobreza. Total país 148

Capitulo 1 - Introducción

Objetivos generales:

Predecir estados futuros del sistema educativo público y privado, mediante técnicas de inteligencia artificial. Recopilar todos los datos del pasado y luego buscar y predecir o estimar variables para el futuro, que pertenecen a la dimensión social de la Educación.
Estimar el futuro de diversas dimensiones de la educación y su relación sobre la situación social y económica, en base a los datos e información del gobierno uruguayo.
Anticipar variables que pertenecen al domino de los estudiantes de la educación uruguaya, en la medida que se puedan descubrir predecir el futuro, mediante técnicas de inteligencia artificial, del sistema educativo.
Una vez que se encuentren conocimientos de predicción del futuro de la Educación en Uruguay, mediante entrenamiento con Inteligencia Artificial, se pueda proyectar y afirmar tendencias sobre la educación en los años venideros.

Objetivos específicos:

Recopilar todos los datos del pasado para luego buscar tendencias futuras, entre distintas variables y dimensiones, para realizar una predicción lo más certera posible de la situación futura desde la situación social y económica, sobre la educación en los próximos años, puntualmente para 2022 y 2023.[6][7]
La educación en su dimensión social y económica, en todas sus ramas y niveles académicos, será analizada con Inteligencia Artificial en las siguientes métricas:

- Por Formas de Administración
- Por sexo
- Por afrodescendencia
- Por Situación de pobreza

Las siguientes dimensiones serán analizadas con inteligencia artificial:

- Cantidad de alumnos matriculados en universidades según forma de administración.
- Cantidad de ingresos y egresos en carreras universitarias de grado según forma de administración.
- Distribución porcentual de las personas según máximo nivel educativo por sexo y ascendencia racial.
- Porcentaje de jóvenes de 18 y más años que finalizaron secundaria (6° de liceoUTU) según sexo.
- Porcentaje de jóvenes que finalizaron primaria según sexo.
- Porcentaje de personas en situación de pobreza según sexo por ascendencia afro.
- Prioridad MacroEconómica en Educación.
- Tasa de actividad según sexo por nivel educativo.
- Tasa de analfabetismo de las persona de 15 y más años según sexo.
- Tasa de desempleo según nivel educativo.
- Tasa de desempleo según sexo por ascendencia afro.
- Tasa de desempleo según sexo por situación de pobreza.
- Tasa de empleo por ascendencia afro.
- Tasa de empleo por sexo y situación de pobreza.

Justificación:

Existe una necesidad imperiosa de determinar y predecir el futuro del rumbo de la educación en Uruguay, inmersa dentro de un contexto social, cultural y sobretodo económico-político, que atañen a las decisiones políticas, y que impactan de manera muy significativa en el camino a recorrer por la educación primaria, secundaria y terciaria, así como también la universitaria de nuestro país.
Así, de esta manera, una visión global es necesario para saber el futuro de la educación y para tal efecto, se puede utilizar algoritmos de inteligencia artificial [1], aplicado a las bases de datos estadísticas de datos abiertos, que dispone el gobierno del Poder Ejecutivo, para proyectar el futuro, en diversas dimensiones que lo contiene.[4]

Caracterización del problema:

El problema se abordará aplicando algoritmos de Machine Learning a diversas dimensiones de la educación en el Uruguay. Para ello, es necesario contar con una base de datos estadística cuya fuente sea confiable, en donde han participado actores con un trabajo serio y responsable para su elaboración [1]. Esta base de datos es la que se aplicarán la tecnología de inteligencia artificial. Se trata de predecir tendencias en la educación uruguaya para el futuro, a través de Machine Learning.
Como tal, se conseguirá predecir una imagen general de la educación en Uruguay en el futuro 2022 y 2023, en varias dimensiones que la componen.

Todas las instituciones educativas, (Universidades, colegios, liceos y escuelas primarias) brindan datos en formatos de archivos .csv, de ingresos y egresos de alumnos, tasas de alfabetización, entre varias dimensiones de la educación. La idea es predecir para el futuro estas dimensiones.
El gobierno uruguayo dispone de datos abiertos para ser analizados por investigadores, profesionales, empresas y estudiantes. Estas son las fuentes de datos:

Fuente de datos:

Las fuentes de datos que se usarán serán del Catálogo Abierto de Datos, https://catalogodatos.gub.uy/, [4], que es un programa que lleva adelante el Poder Ejecutivo. Dicho catálogo es gestionado por el equipo de gobierno AGESIC, (Agencia de Gobierno Electrónico y Sociedad de la Información y del Conocimiento):

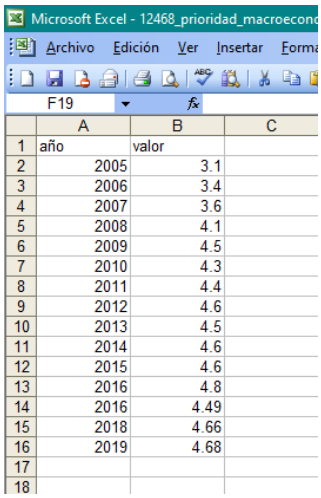
"El Catálogo Nacional de Datos Abiertos es una herramienta que permite acceder a datos abiertos de organismos públicos, academia, organizaciones de sociedad civil y empresas privadas. Cualquier persona puede utilizar los datos publicados libremente para contar historias, desarrollar investigaciones, visualizaciones, aplicaciones cívicas y emprendimientos."[4]

Actualmente contiene más de 2.000 conjuntos de datos en formatos de archivos .csv, .json, .xml y .xls y aplicaciones, mediante CKAN, https://ckan.org/, (CKAN es una plataforma de datos, de código abierto, líder a nivel mundial).

Los archivos .csv que AGESIC recolecta y lleva a cabo, se realizan mediante los siguientes órganos del Estado uruguayo y sus actores intervinientes:

- **Ministerio de Desarrollo Social (MIDES)**: Germán Barcelona, Manuel Piriz, Luis Lagarxio.
- **Gestión y Evaluación- OPP (AGEV)**: Gabriela Delfino, Diego Gonnet.
- **Facultad de Ingeniería (FING)**: Fernando Carpani, Lorena Echeverry
- **Instituto Nacional de Estadística (INE)**: Federico Segui, Lucía Perez
- **Red de Gobierno Abierto (REDGA-Sociedad Civil)**: Fernando Uval, Román Sugo
- **Intendencia de Montevideo (IM)**: Juan Prada
- **Agencia para el Desarrollo de Gobierno Abierto y Sociedad de la Información (AGESIC)**: Victoria Köster, Laura Nahabetián
- (https://www.gub.uy/agencia-gobierno-electronico-sociedad-informacion-conocimiento/comunicacion/publicaciones/grupo-trabajo-datos-abiertos)

Los datasets .csv, con sus respectivas dimensiones, se encuentran en: https://catalogodatos.gub.uy/dataset
Cada dataset, contiene filas y columnas, y, por ejemplo, para el Gasto Público en Educación (el porcentaje del PBI utilizado para la educación) es:



	A	B	C
1	año	valor	
2	2005	3.1	
3	2006	3.4	
4	2007	3.6	
5	2008	4.1	
6	2009	4.5	
7	2010	4.3	
8	2011	4.4	
9	2012	4.6	
10	2013	4.5	
11	2014	4.6	
12	2015	4.6	
13	2016	4.8	
14	2016	4.49	
15	2018	4.66	
16	2019	4.68	
17			
18			

Como se puede apreciar, la features o variables de entrada son los años, y las variables de salida, son los datos a predecir.

Contextualización:

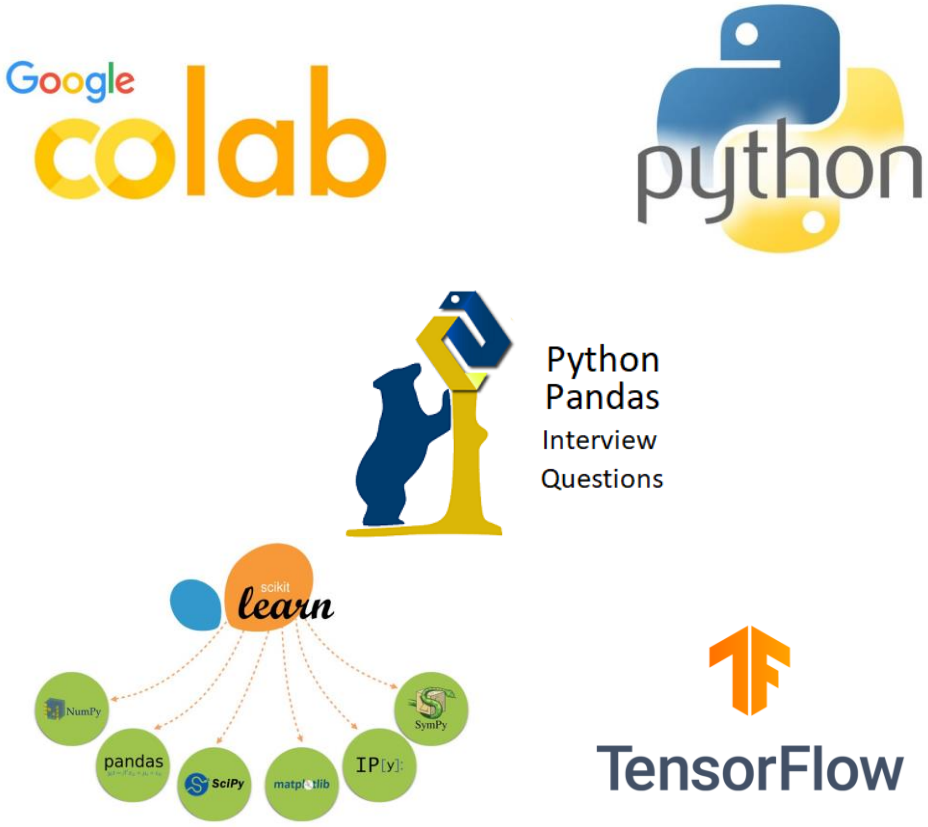
El contexto en la cual se aplicarán las tecnologías y algoritmos de inteligencia artificial serán y recaerán sobre los estudiantes y sus carreras y cursos, desde varias dimensiones del tejido social, cultural y económico, en lo que concierne a la educación pública y privada en la República Oriental del Uruguay.
Se analizarán además de las dimensiones, los actores involucrados, períodos en el tiempo, expresado en años, en las instituciones educativas públicas:

- de primaria que pertenecen al Consejo de Educación y Primaria (DGEIP),
- a los liceos y colegios del Consejo de Educación Secundaria (DGES),
- a las Escuelas Técnicas del Consejo de Educación Técnico Profesional de la Universidad del Trabajo del Uruguay (DGETP),
- a los institutos de formación docente de maestros y centros regionales de profesores del Consejo de Formación en Educación (DFE),
- todas dentro del CODICEN (Consejo Directivo Central) que está subordinado a la ANEP (Administración Nacional de Educación Pública)
- las facultades que pertenecen a la UDELAR (Universidad de la República), todas estas bajo el MEC (Ministerio de Educación y Cultura).

Así como también las universidades Privadas, Universidad de la Empresa UDE, Universidad ORT, Universidad Católica, Universidad de Montevideo y colegios privados de todo el país.

Estructura de Trabajo:

Se utilizará el lenguaje Python, conjuntamente con bibliotecas de SKLearn, entre otras, y los ambientes de desarrollo siguientes:



El Python Pandas servirá para la manipulación entre distintos conjuntos de datos .csv, en caso que sea necesario y la librería SKLearn es la base, ya que implementa los algoritmos de Inteligencia Artificial de Regression Linear, y TensorFlow para las redes neuronales multicapa perceptron, para la predicción de distintos tipos de variables sobre la educación uruguaya. [2][5]

Capitulo 2 - Marco Teórico

Trabajos relacionados:

El trabajo relacionado que se conecta con este proyecto es la monografía realizada por el estudiante Alfredo Poggio Manzor en el Instituto Universitario Autónomo del Sur IUAS, actualmente Universidad de la Empresa UDE (www.ude.edu.uy), que es un trabajo de conclusión de la carrera de Licenciatura en Informática denominado proyecto "DeCobSol", en que consistía de descubrir patrones de descubrimiento en los datos en bases de datos open source, aplicando Data Mining, en el año 2008. Este proyecto DeCobSol, no tenía una base sólida teórica sobre redes neuronales. El "trabajo futuro" del proyecto DeCobSol era profundizar académicamente en redes neuronales, Perceptron, supervisados y no supervisados, en su estructura algorítmica y lógica. Por lo tanto, este proyecto de conclusión del PRIA es el "trabajo futuro" del proyecto DeCobSol.

Antecedentes académicos y de la investigación.

Este trabajo se basa como punta pie inicial, el proyecto DeCobSol, proyecto eéste que realicé (el que escribe, Alfredo Poggio Manzor) sobre DataMining, que consiste en el descubrimiento de patrones de información sobre bases de datos. Pero en su trabajo futuro, se debería hacer incapié en la redes neuronales. Se utilizaron varios algoritmos de creación de modelos exploratorios sobre los datos de estas bases de datos. Algoritmos como Redes Bayesianas y árboles de clasificación y árboles de decisión.

Fundamentación teórica:

Es de vital importancia el uso de los algoritmos de inteligencia artificial, supervisados, ya que se compone de las últimas tecnologías para predecir datos y hechos futuros. Se elijarán los algoritmos de redes Perceptron para clasificar y predecir, en sus variantes como Regression Linear, para crear y entrenar modelos predictivos para encontrar y predecir incluso datos escondidos en la información [2].

Bases teóricas.

Como base teórica para la creación de modelos predictivos utilizando Linear Regression, se tomaron puntos críticos y fundamentos de la ciencia de la predicción: los modelos lineales pueden predecir, sin mucha exactitud, pero con precisión aceptada por la mayoría de los usuarios y cientistas de datos [1].

Como se trata de predecir variables que dependen de otras variables de entrada, se optó por manejar redes neuronales multicapa, con capas de entrada, capas ocultas y capas de salida. En general, la capa de salida predice el comportamiento de una dimensión de la realidad, basada en la variable de entrada año, que acompaña los otros features de entrada. El Perceptron puede generar una buena predicción de esta variable de salida [2].

Revisión bibliográfica, Referencias bibliográficas:

1. Dhar, V. Prediction in financial markets: The case for small disjuncts. ACM Transactions on Intelligent Systems and Technologies 2, 3 (Apr. 2011).
2. Dhar, V. and Stein, R. Seven Methods for Transforming Corporate Data Into Business Intelligence. Prentice-Hall, Englewood Cliffs, NJ, 1997
3. Frawley, W. and Piatetsky-Shapiro, G., Eds. Knowledge Discovery in Databases. AAAI/MIT Press, Cambridge, MA, 1991.
4. Fuente de datos para el proyecto: https://catalogodatos.gub.uy/about
5. Gartner (2012). Big Data. IT Glossary.
6. González Díaz, I. (2017). Big data para CEOs y directores de marketing
7. Greenberg, P. (2008). CRM at the Speed of Light (4.ª ed.). Nueva York: McGraw-Hill.
8. Data Science For Business (2019): What You Need to Know About Data Mining & Data-Analytic Thinking
9. https://www.datascience-pm.com/crisp-dm-2/ - Metodología CRISP – DM

Metodologías, técnicas y herramientas a utilizar:

Se pretende realizar predicciones futuras, descubrimiento de nuevos patrones (Minería de datos y DataScience), aplicando a los datasets del Catálogo Abierto, mediante el entrenamiento con machine learning:

- Redes neuronales estilo Perceptron, utilizando Python Google Colab y librerías de Keras TensorFlow.
- Visualización de las relaciones entre las dimensiones con Matplotlib de Python, y biblitecas de numpy.
- Utilización de algoritmos de clasificación de la librería SKLearn.
- Biblioteca PANDAS del Python, para la realización de la operaciónes sobre los datasets,
- Software a utilizar: Google Colaboratory Colab, Python, ambiente y biblioteca SciKitLearn

Al aplicar los algoritmos sobre los conjuntos de datos abiertos y al utilizar las mencionadas herramientas, se compararán los resultados, unos con otros para una mayor precición y acuracia (accuracy) [3].

La metodología, líneas generales, en resumen será:

1. Elección del dataset .csv adecuado para analizar según la dimensión a estudiar.
2. Se aplica las bibliotecas de numpy y pandas para la extracción y lectura de los datos.
3. Se realiza una depuración (preprocesamiento de los datos a analizar), como ser, eliminación de campos null, fechas incorrectas, valores inadecuados "garbage" en columnas
4. Selección de las features adecuadas para el análisis
5. Creación de conjuntos de entradas para entrenar una red neuronal (conjunto de datos de entrenamiento)
6. Creación de un conjunto de test de la red
7. Entrenamiento de la red neuronal (puede ser Perceptron o Linear Regresión)
8. Elección de la cantidad de capas de entrada, capas ocultas y capa de salida, y el paso de aprendizaje adecuado.
9. Análisis de la curva de aprendizaje, para verificar si la Función de Error tiende a 0 a través de las épocas, utilizando el set de test.
10. Obtención de una red entrenada y la predicción futura de cuantitativa
11. Obtención de una conclusión cualitativa

¿Cómo debo leer la predicción realizada con inteligencia artificial de cada dimensión de la educación ?:

En este presente documento de monografía de investigación del PRIA UTEC FURG, cada dimensión de la educación tiene un título, por ejemplo, "Tasa de analfabetismo de las personas de 15 y más años".
En cada dimensión, se presenta una breve descripción de qué se trata esta dimensión, el conjunto de datos a analizar, (por ejemplo: 10394_tasa_de_analfabetismo_de_las_persona_de_15_y_mas_aos_segun_sexo-_total_pais.csv) y luego se presenta en forma de gráfico estos datos.
Luego se muestra la predicción futura para el 2022 y el 2023 que se realiza con a inteligencia artificial.

Luego de cada dimensión predicha, se pasa a las conclusiones cualitativas de cada dimensión. Al final se hace una conclusión final.

Por último, hay un anexo con todos los códigos fuentes Python que fueron necesarios utilizar.

Proceso y Tratamiento de los datos:

¿Cómo se logra la predicción de una dimension de la educación?

A continuación, se muestra un ejemplo de cómo se hizo para entrenar con SKLearn y Keras TensorFlow, para predecir la tasa de analfabetismo para el 2022 y el 2023:

Importamos las bibliotecas:

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import datetime
```

Cargamos desde el drive, el dataset en cuestión:

```
dataset = pd.read_csv('/content/drive/MyDrive/PRIA Alfredo Poggio Manzor 2
```

```
'7000_tasa_de_analfabetismo_de_las_personas_de_15_y_mas_aos-_total_pais.csv',  encoding='latin-1')
```

Como se puede apreciar, para el tratamiento de los datos, se trabaja con una codificación latin-1.

- Mostramos el dataset:

```
dataset
```

	año	valor
0	2006	2.2
1	2007	2.1
2	2008	1.9
3	2009	1.8
4	2010	2.0
5	2011	1.7
6	2012	1.6
7	2013	1.6
8	2014	1.5
9	2015	1.5
10	2016	1.4
11	2017	1.4

Graficamos el dataset:

```
filtered_df2 = dataset
filtered_df2
#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Tasa de analfabetismo de las personas de 15 y más años")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Tasa de analfabetismo de las personas de 15 y más años'])
```

Entrenamos la red Multi Linear Regression, con las bibliotecas SKLearn:

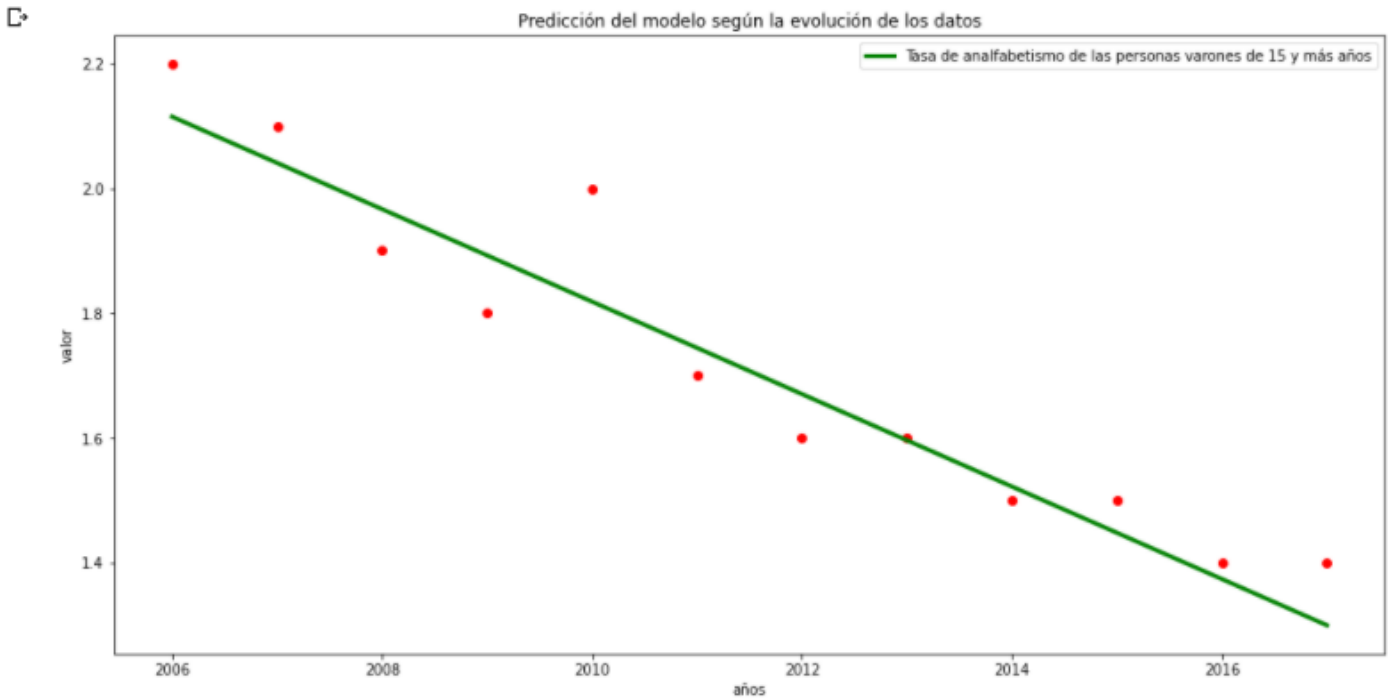
```
from sklearn import linear_model, metrics

rl = linear_model.LinearRegression()

año =  filtered_df2["año"]

#como el linear model del sklearn espera que se le pase una matriz , hay que hacer un reshape:
X = año[:, np.newaxis]
y = filtered_df2["valor"]

from sklearn.model_selection import train_test_split

while True:
  X_train, X_test, y_train, y_test = train_test_split(X,y)
  rl.fit(X_train, y_train)
  if rl.score(X_train, y_train) > 0.95:
    break

print("Score of Test set", rl.score(X_test, y_test))

print(rl.predict([[2022]]))
print(rl.predict([[2023]]))
```

Vemos la accuracy (índice de aciertos sobre el conjunto de test):

```
0.8982952271362883
0.8795922914113908
0.8992681524397308
0.9179793590439963
0.8441860465116133
0.9077649077649146
0.9413298565840885
0.8740946261682403
0.9374999999999954
0.8834549071617946
0.8991890639480976
0.9359153951710388
0.9043635981608691
0.9523809523809574
Score of Test set 0.7434597295708535
[0.92962963]
[0.85555556]
```

Graficamos la predicción:

```
plt.figure(figsize=(16,8))
plt.scatter(X, y, color = "red")
plt.plot(X, rl.predict(X), color = "green", linewidth =3)
plt.title("Predicción del modelo según la evolución de los datos")
plt.xlabel('años')
plt.ylabel('valor')
plt.legend(["Tasa de analfabetismo de las personas varones de 15 y más años"])
plt.show()
```



Entrenamiento de una Red Neuronal Utilizando Perceptron Multi Layer con Keras TensorFlow:

```
dataframeIII = filtered_df2["año"]

X = np.array(dataframeIII, dtype=float)

dataframeIV = filtered_df2["valor"]
Y = np.array(dataframeIV, dtype=float)

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,Y)
```

Utilizando Perceptron Multi Layer:

```
capa1 = tf.keras.layers.Dense(units = 1, input_shape = [1])
capa2 = tf.keras.layers.Dense(units=8)
capa3 = tf.keras.layers.Dense(units=4)
salida = tf.keras.layers.Dense(units=1)
modelo = tf.keras.Sequential([capa1, capa2, salida])

print(modelo.summary())

modelo.compile(
    optimizer = tf.keras.optimizers.Adam(0.01),
    loss = 'mean_squared_error'
)
```

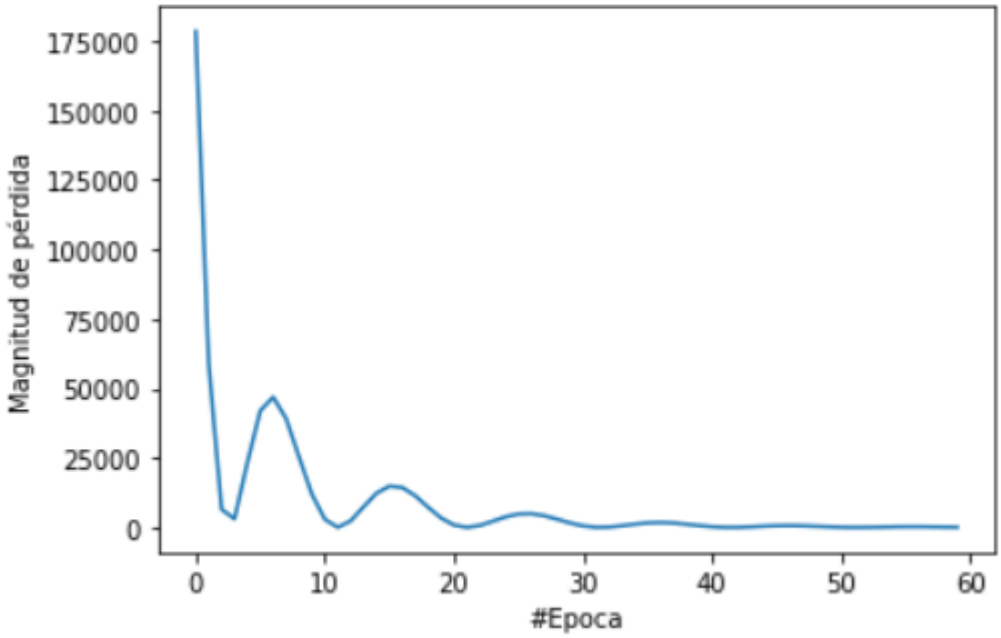
- Utilizando Perceptron Multi Layer:

Con un accuracy mayor a 94%

```
historial = modelo.fit(X_train, y_train, epochs=600, verbose=False)

acc1 = modelo.evaluate(X_test, y_test)
print(acc1)
print(f"\nAccuracy is {acc1*100}")

print("Modelo Entrenado")
```

```
import matplotlib.pyplot as plt
plt.xlabel('#Epoca')
plt.ylabel('Magnitud de pérdida')
plt.plot(historial.history['loss'])
```

```
[<matplotlib.lines.Line2D at 0x7ff134d3d510>]
```



Utilizando Perceptron Multi Layer:

```
print("Hagamos una Testeo")
resultado = modelo.predict(X_test)
print("Los valores calculados por la red neuronal son de: ")
print(str(resultado))
print("El valor esperado es de: ")
print(y_test)
print("Hagamos una predicción para el 2022: ")
resultado = modelo.predict([[2022]])
print(resultado)
print("Hagamos una predicción para el 2023: ")
resultado = modelo.predict([[2023]])
print(resultado)
```

```
Hagamos una Testeo
Los valores calculados por la red neuronal son de:
[[1.7356741]
 [1.7298785]
 [1.735075 ]]
El valor esperado es de:
[1.4 2.2 1.5]
Hagamos una predicción para el 2022:
[[1.7392153]]
Hagamos una predicción para el 2023:
[[1.7396863]]
```

Comparación entre las tecnologías utilizadas: SKLearn y Tensor Flow Keras:

Según los resultados obtenidos con redes neuronales artificiales:
La utilización de la librería SKLearn, para redes MLRegression dio resultados de una predicción de una regresión lineal bastante aceptable, la mayoría de las veces por encima del 85%, y hasta algunos casos por encima del 95% como se puede ver en los códigos fuentes .ipynb.
La utilización del TensorFlow permitió redes neuronales com muy bajo error, en la mayoría de los casos-:

Resultados esperados

Se espera que a partir de los datos estadísticos históricos, se pueda predecir con buena precisión, en el futuro, para los años 2022 y 2023 las facetas y características de la educación en Uruguay.

Público implicado

El público implicado es la gran masa de alumnos de primaria, secundaria y universidad de la educación pública y privada del Uruguay.

Como se puede apreciar, al comienzo en enero del 2021, se comienza con la identificación de los principales requisitos (qué es lo que se quiere predecir). En febrero se hace la recolecta, la depuración y la transformación de los datasets necesarios, como un anteproyecto.
En marzo se preparan los datasets para que puedan ser procesados por Python y sus bibliotecas, como el SKLearn y el SciKitLearn, sobre los datasets.
En abril y Mayo se aplican los algoritmos de inteligencia artificial, como ser: redes neuronales perceptron y regresion lineal. A fines de Agosto se arman las predicciones que se han obtenido.

Capitulo 3 - Desarrollo del Proyecto. Predicción para el 2022 y el 2023 para la educación, en forma cuantitativa.

A continuación, se detallarán los datasets a ser analizados por las técnicas de inteligencia artificial. Se aplicarán algoritmos de IA, para predecir y pronosticar, en el futuro, las distintas dimensiones de la educación en Uruguay y la situación social del empleo y del desempleo, de muchos uruguayos. Al aplicar algoritmos de IA, se realiza un proceso de deep learning a través de redes neuronales multicapa, para que haya o se logre un correcto machine learning, es decir, de una red entrenada con los datos suministrados por el gobierno uruguayo. Luego de entrenada una red neuronal multicapa, se ingresa el año de un futuro cercano, como por ejemplo, 2022, 2023, y la red estimará cómo serán los valores en estos futuros años.

Primera categoría de conjuntos de datos a analizar por IA de la realidad:

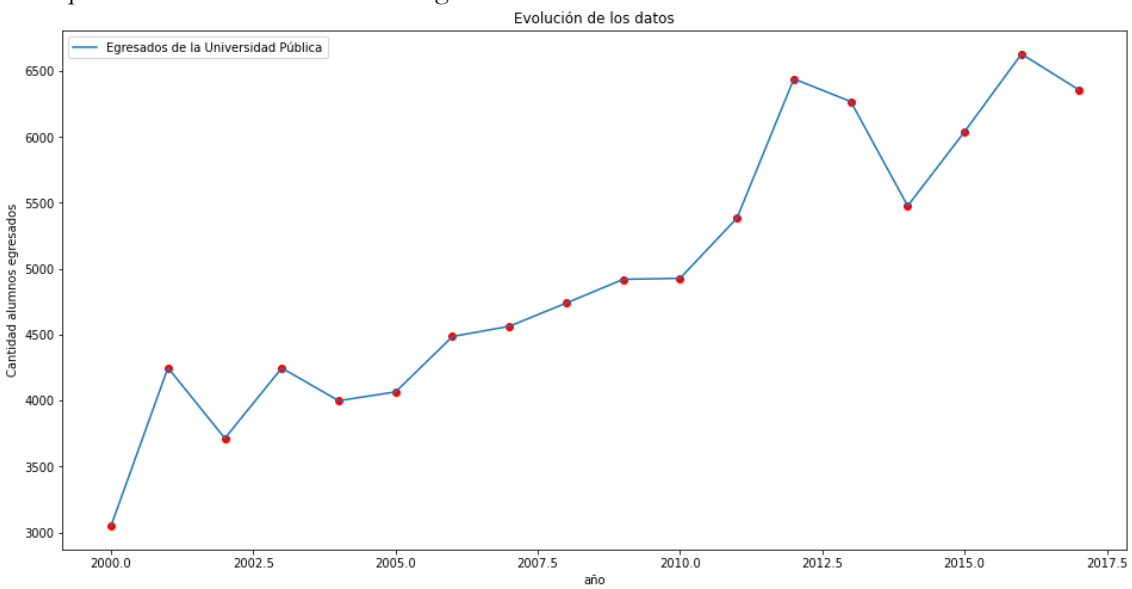
Por formas de Administración:

Cantidad de ingresos y egresos en carreras universitarias de grado según forma de administración. Total país

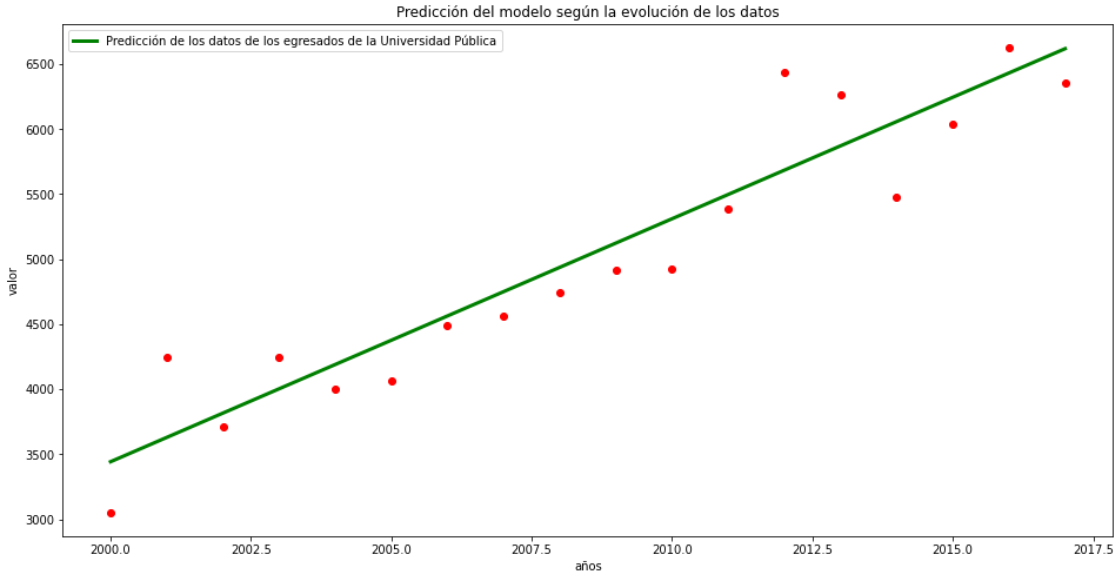
Ingresos: Todos los individuos que se inscribieron para comenzar a cursar una determinada oferta de curso universitario por primera vez. Dicha inscripción debe haber tenido lugar en algún momento del año de referencia, y el comienzo de los cursos para los que se inscribió también debe estar previsto para el mismo año. Egresados: Alumnos que han cumplido con la totalidad de los requisitos del currículum para la obtención del título o certificación terminal (Área de Investigación y Estadística-MEC).

Nombre del conjunto de datos a analizar por Inteligencia Artificial:
10447_cantidad_de_ingresos_y_egresos_en_carreras_universitarias_de_grado_segun_forma_de_administ.csv

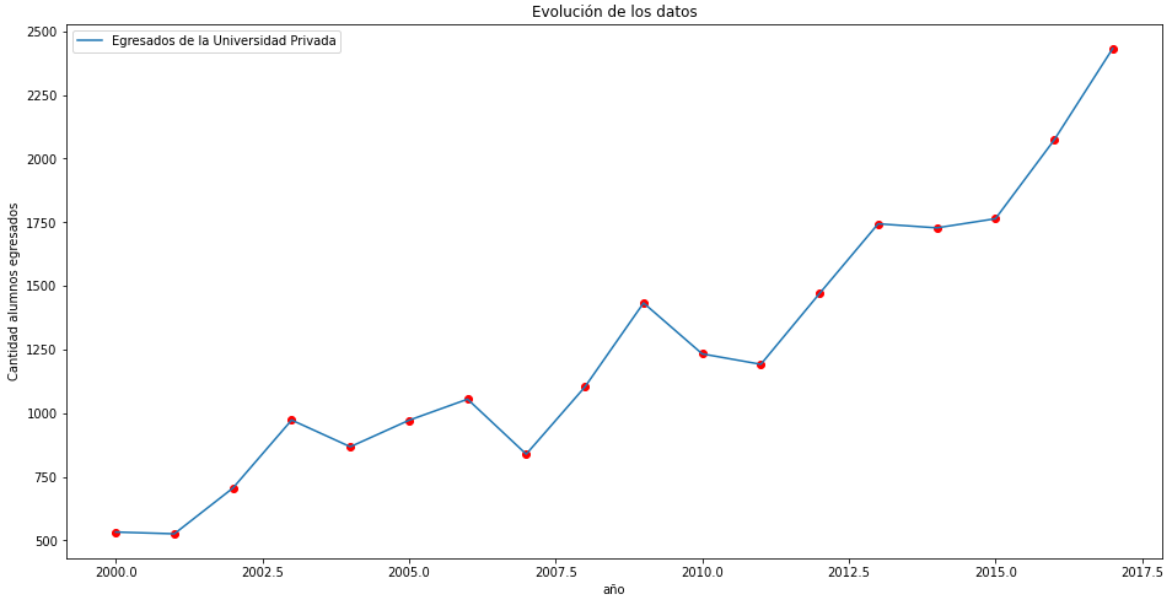
Grafiquemos la cantidad de alumnos egresados en la Universidad Pública:



Predicciones:
Se predice según Inteligencia Artificial utilizando una red neuronal Linear Regression:

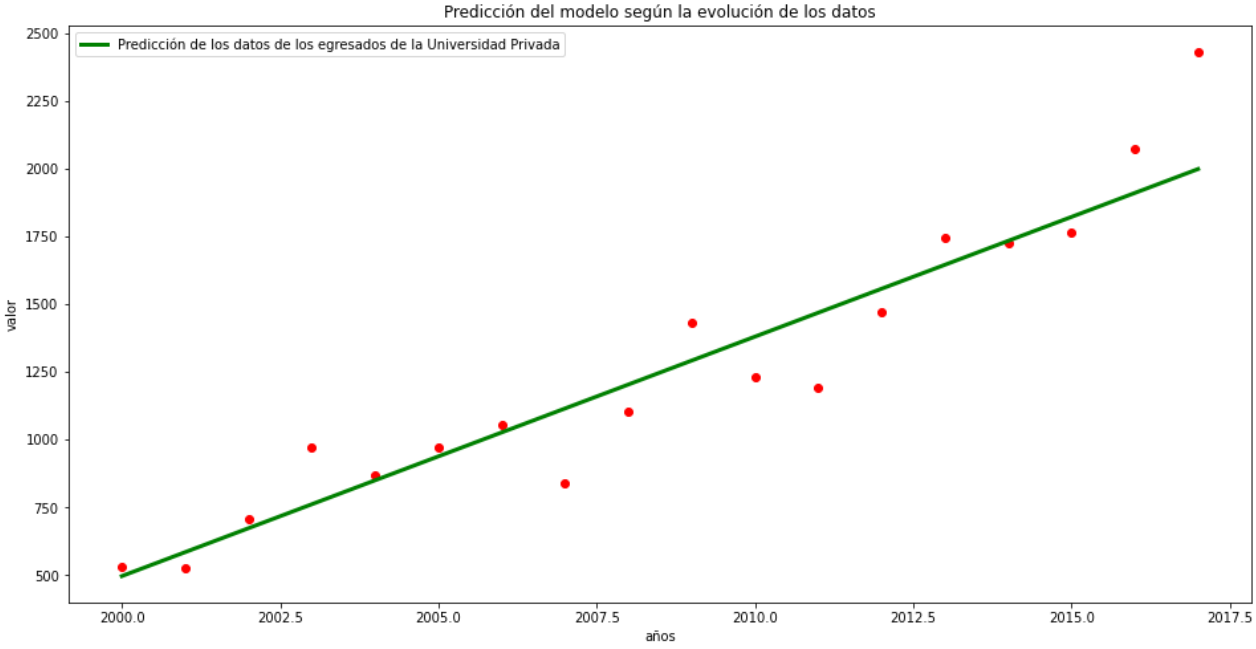


Ahora vamos a graficar la cantidad de egresados en Universidad Privada:

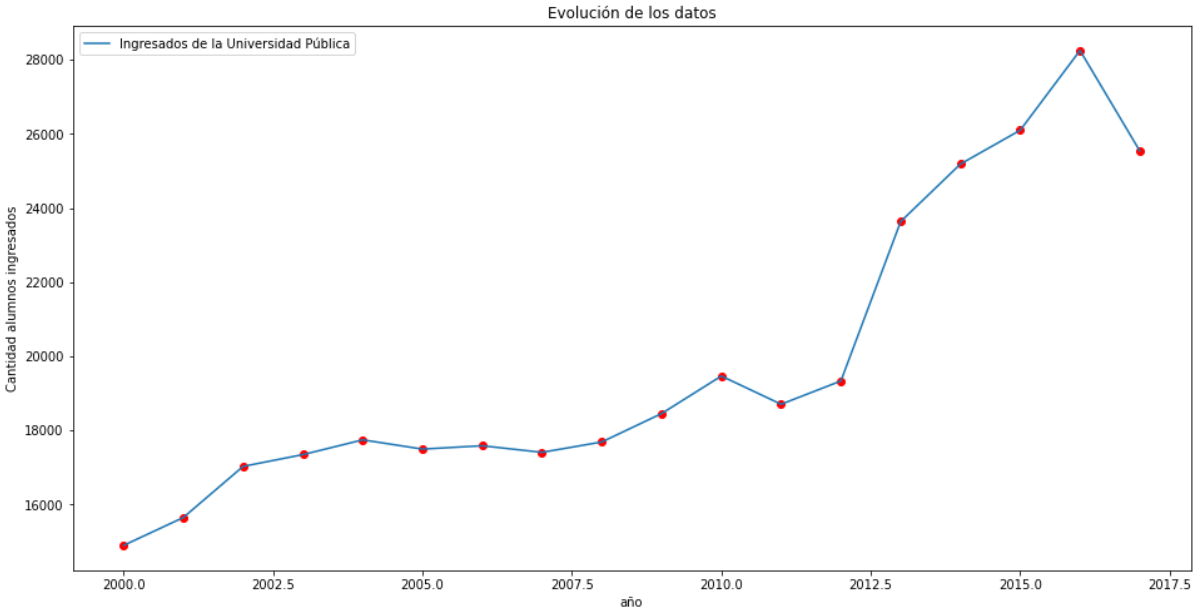


Se predice o se pronostica según Inteligencia Artificial utilizando una red neuronal Linear Regression SKLearn, que para 2022 y 2023:
[2616.6071569]
[2715.87927645]
de alumnos respectivamente, con una efectividad por encima del 88%.

Modelo utilizando LinearRegression, entrenado para los egresados de la Universidad Privada:



Ahora vamos a trabajar con los datos de los alumnos que ingresaron a las Universidades Públicas en Uruguay:

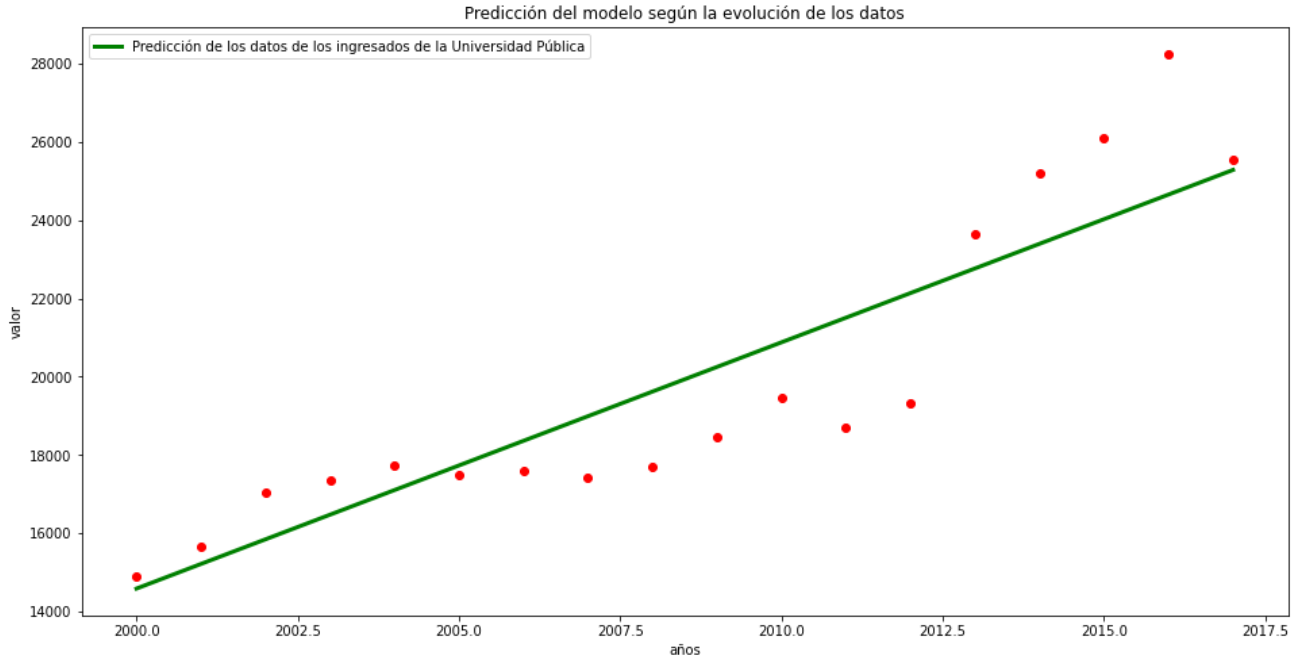


Predicciones:
Se predice o se pronostica según Inteligencia Artificial utilizando una red neuronal Linear Regression SKLearn que va haber de ingresados en la Universidad Pública para el 2022 y el 2023:

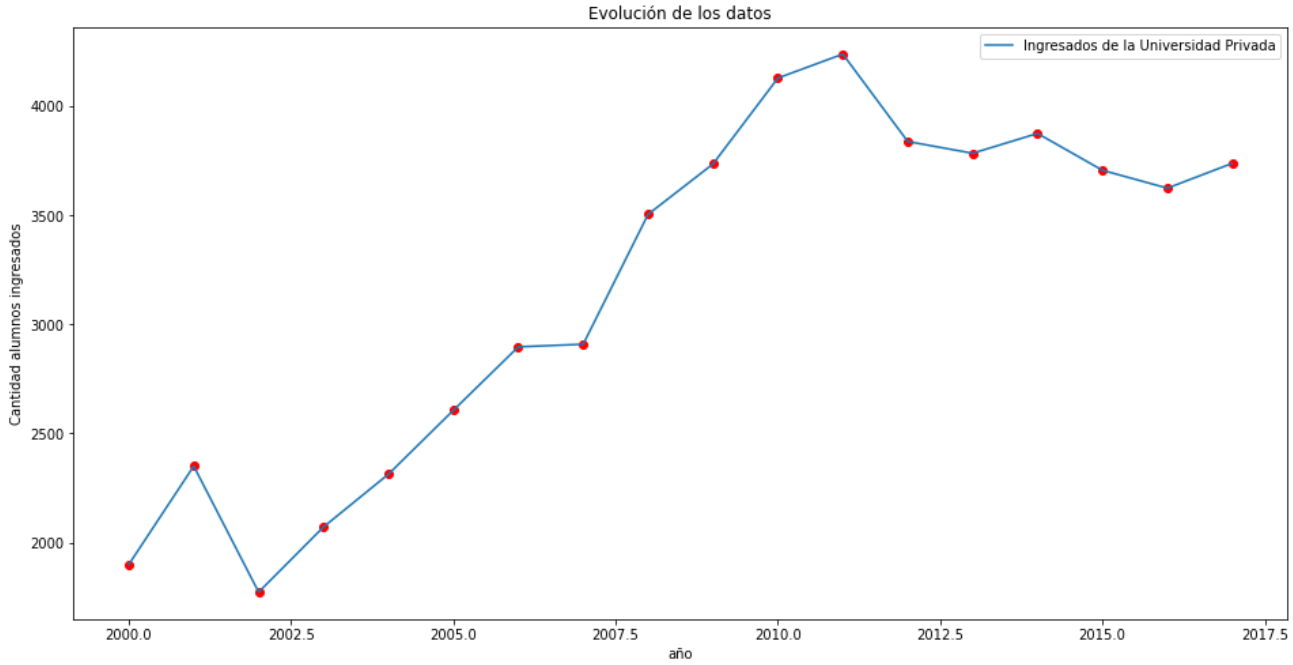
[28431.79118059]
[29061.02613159]

de alumnos

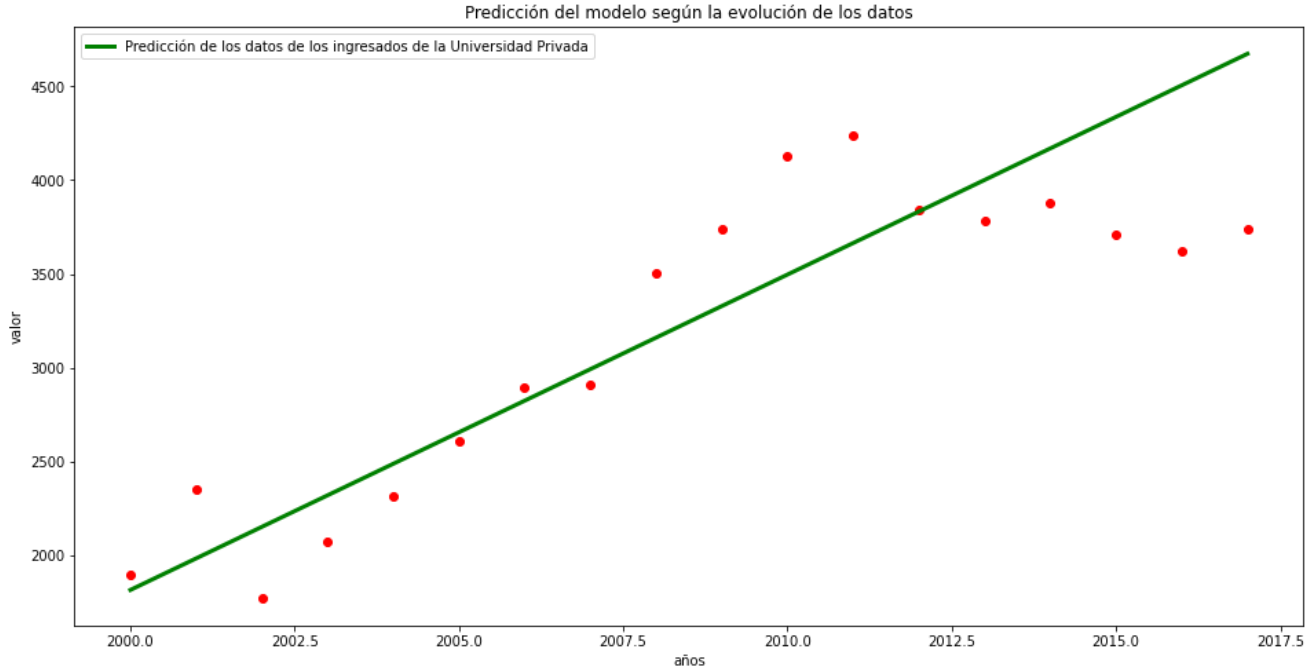
Modelo entrenado usando LinearRegression para ingresados para la Universidad Pública:



Ahora vamos a trabajar para ver la cantidad de Ingresos para las Universidades Privadas:



Modelo entrenado usando LinearRegression para ingresados para la Universidad Privada:



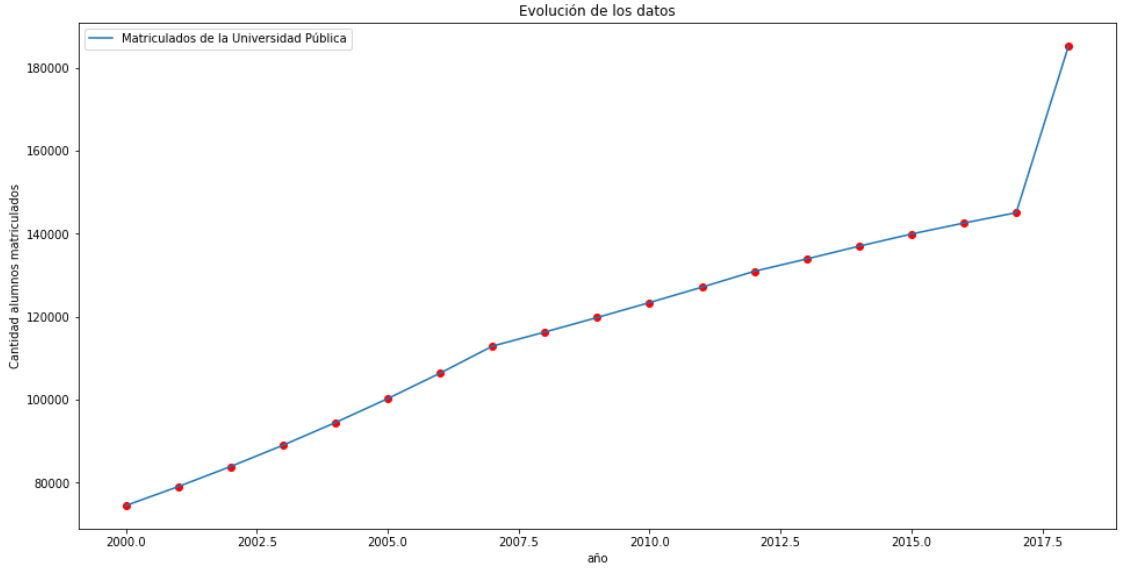
La predicción para el año 2022 es de: [5513.81427742]
La predicción para el año 2023 es de: [5681.90220546]
Desde el 2012 se nota un notorio descenso en los ingresos en las Universidades Privadas, hasta el 2017 en que hubo un leve aumento. Sin embargo desde el 2012, hubo un notorio aumento de ingresos en las Universidades públicas, hasta el 2017 en que hubo un leve descenso.

Cantidad de alumnos matriculados en universidades según forma de administración. Total país

La matricula corresponde al total de estudiantes inscriptos en carreras universitarias. La expresión Estudiantes, hace referencia a relaciones entre personas e instituciones y no al conteo de personas físicas, de manera que una persona puede ser estudiante de más de una institución ostentando inserciones múltiples y por tanto siendo computada más de una vez.

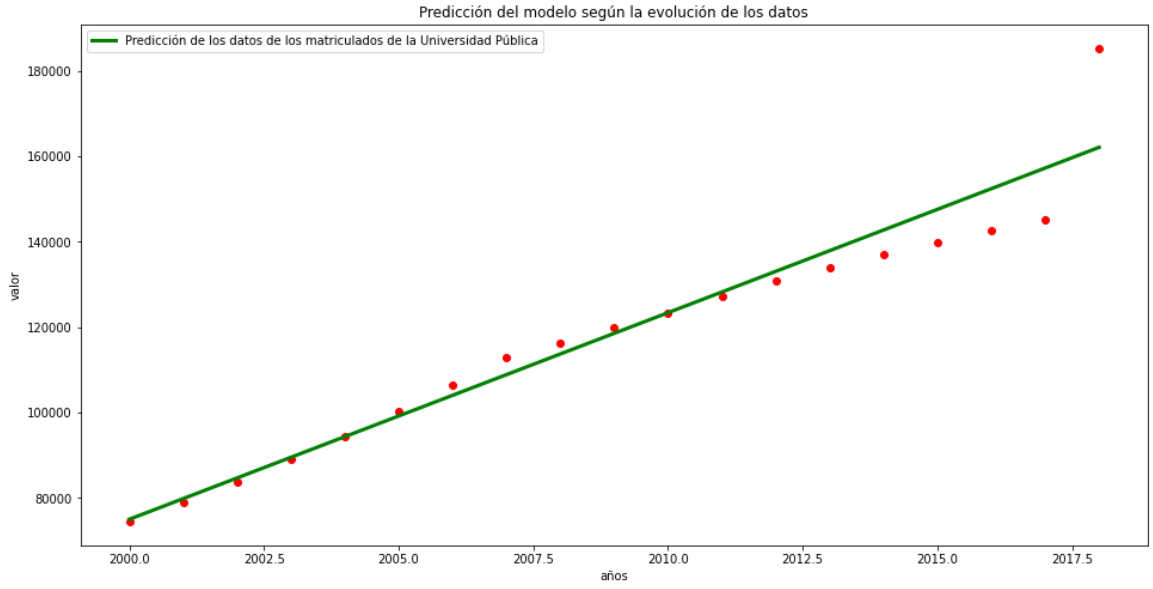
Nombre del conjunto de datos a analizar por Inteligencia Artificial:
11633_cantidad_de_alumnos_matriculados_en_universidades_segun_forma_de_administracion-_total_pai.csv

Cantidad de alumnos matriculados en universidades Públicas según forma de administración :



Predicciones:

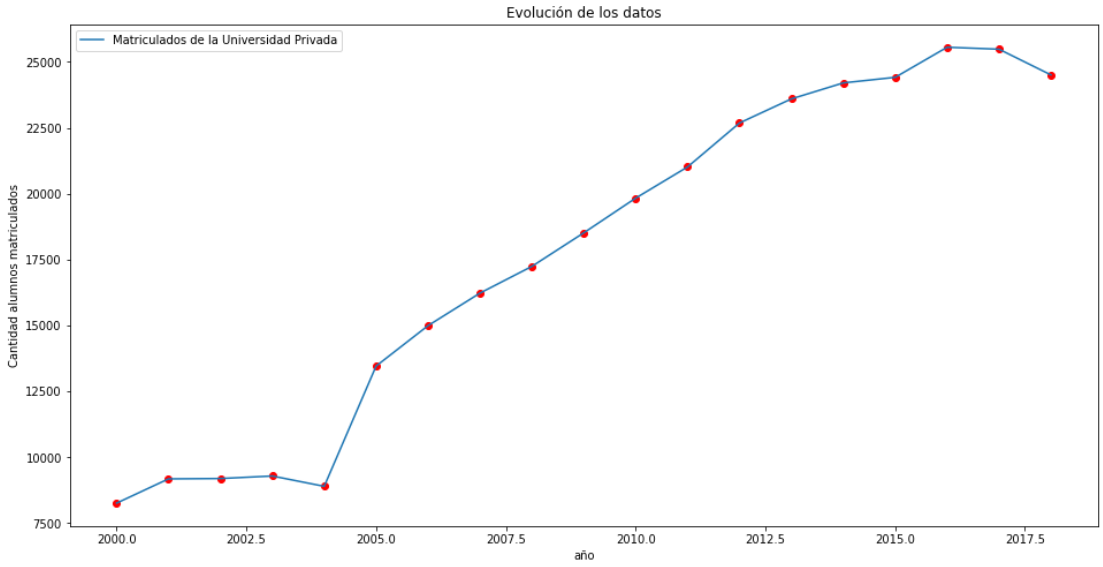
Predicciones de los datos de los alumnos matriculados en Universidades Públicas:



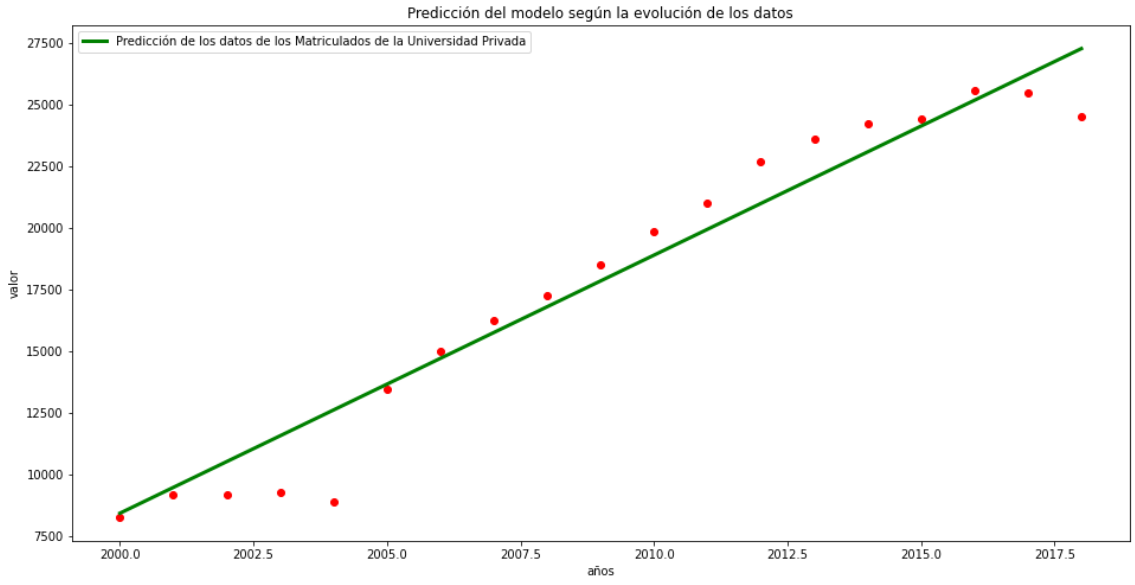
Hagamos una predicción: La predicción para el año 2022 es de: [172496.30898876] y para el 2023 es de: [176825.77422753] alumnos respectivamente
(usando un Multi layer Regression Linear).

De alumnos para universidades públicas.

Cantidad de alumnos matriculados en universidades Privadas según forma de administración :



Hagamos una predicción usando un Linear Multi layer Regression :

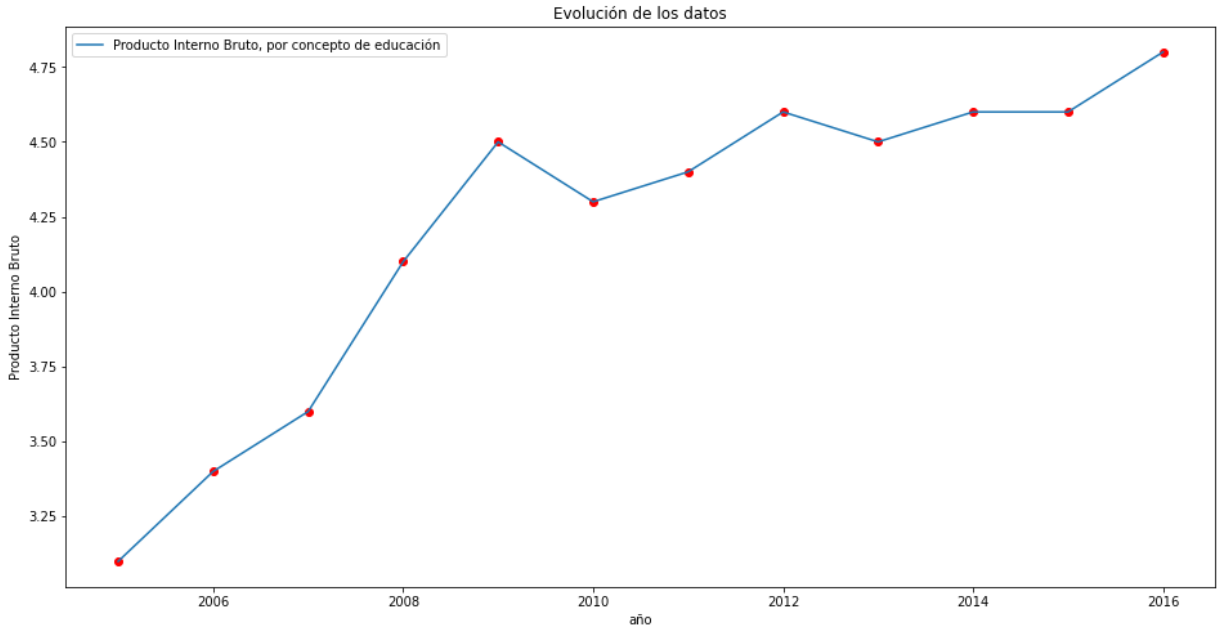


Hagamos una predicción usando un Linear Multi layer Regression :
La predicción para el año 2022 es de: [31782.94204852]
Y para el 2023 es de:
[32868.69912399] (usando MLR SkLearn)
Alumnos para las universidades privadas.

Prioridad macroeconómica del Gasto Público en Educación

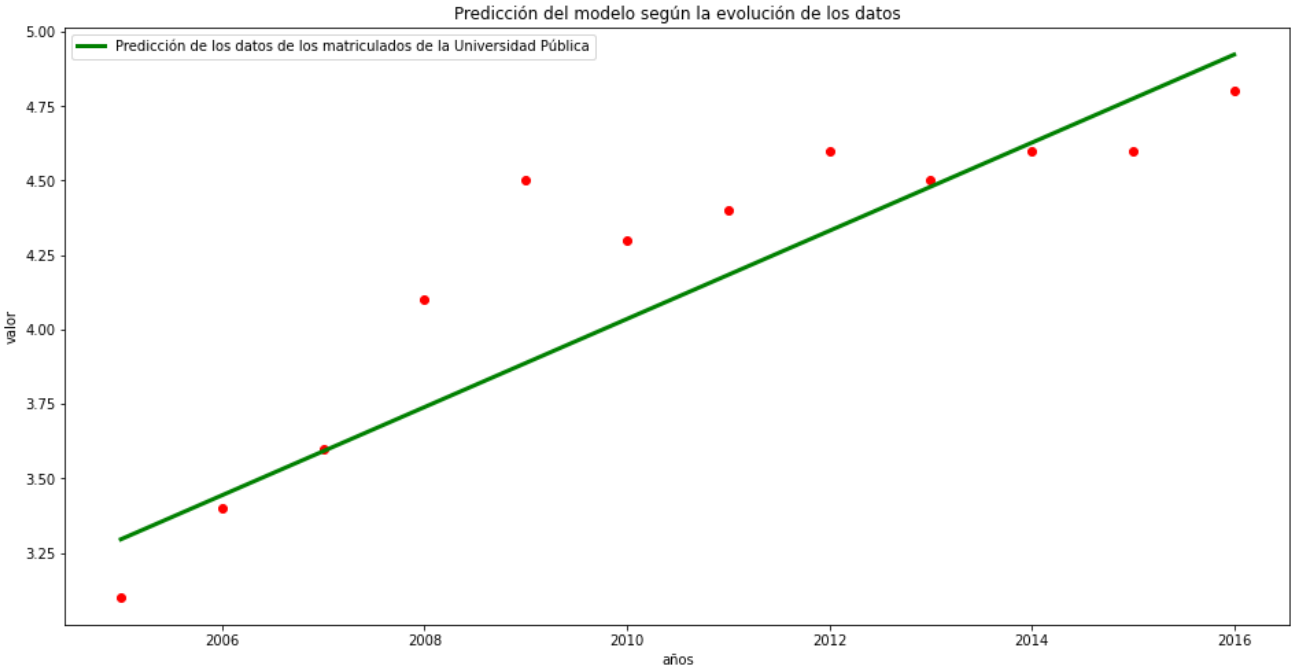
La prioridad macroeconómica del Gasto Público en Educación refiere a la proporción del Producto Bruto Interno que se invierte en Educación a través del gasto público.

Nombre del conjunto de datos a analizar por Inteligencia Artificial:
12468_prioridad_macroeconomica_del_gasto_publico_en_educacion.csv



Predicciones:

Hagamos una predicción:
La predicción para el año 2022 es de:
[5.81103679]
y para el 2023 es de:
[5.9590301]
Porcentaje sobre el PBI, respectivamente



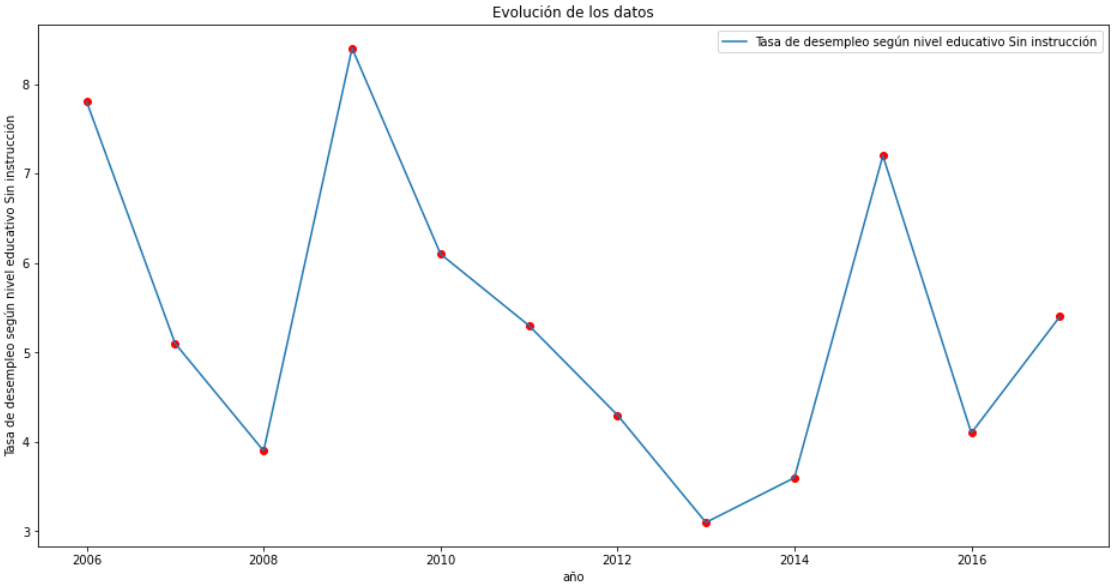
Conclusiones obtenidas al predecir para el futuro con Inteligencia Artificial:
Se pronostica un aumento del gasto público en educación, con respecto al PBI en los próximos años.

Tasa de desempleo según nivel educativo. Total país

Proporción de personas que buscan empleo y no lo tienen, en relación a la población económicamente activa, según nivel educativo.

Nombre del conjunto de datos a analizar por Inteligencia Artificial:
10805_tasa_de_desempleo_segun_nivel_educativo-_total_pais.csv

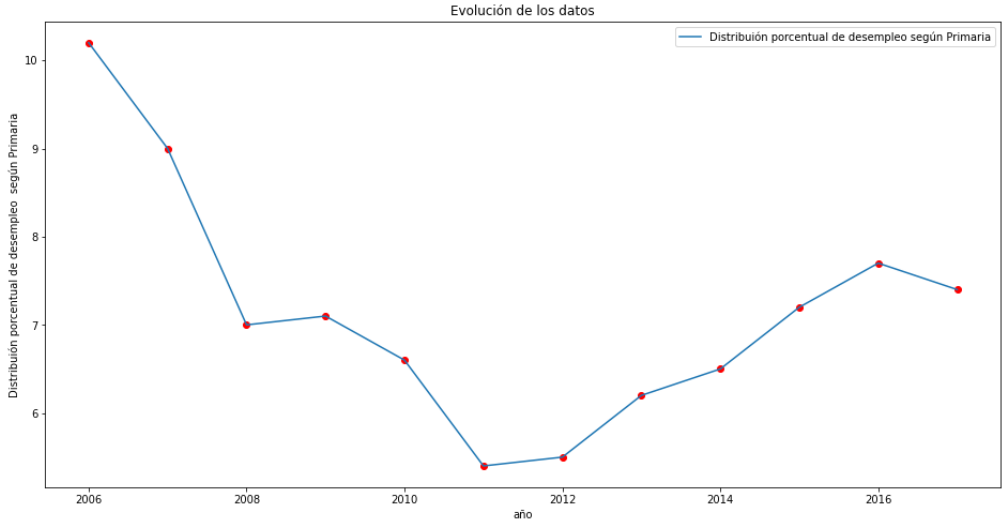
Tasa de desempleo según nivel educativo Sin instrucción:



Predicciones:

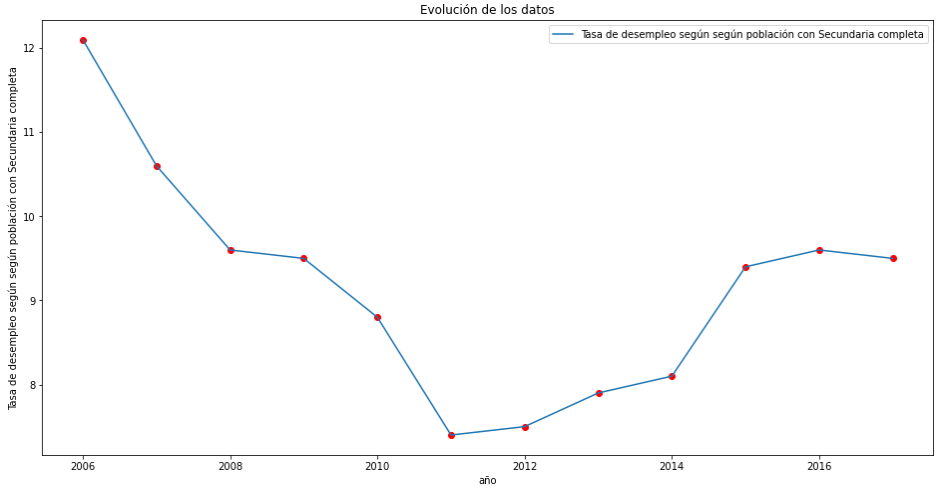
Hagamos una predicción:
La predicción para el año 2022 es de:
[[5.021358]]
y para el 2023 es de:
[[5.0238075]]

Tasa de desempleo según nivel educativo nivel educativo Primaria completa:



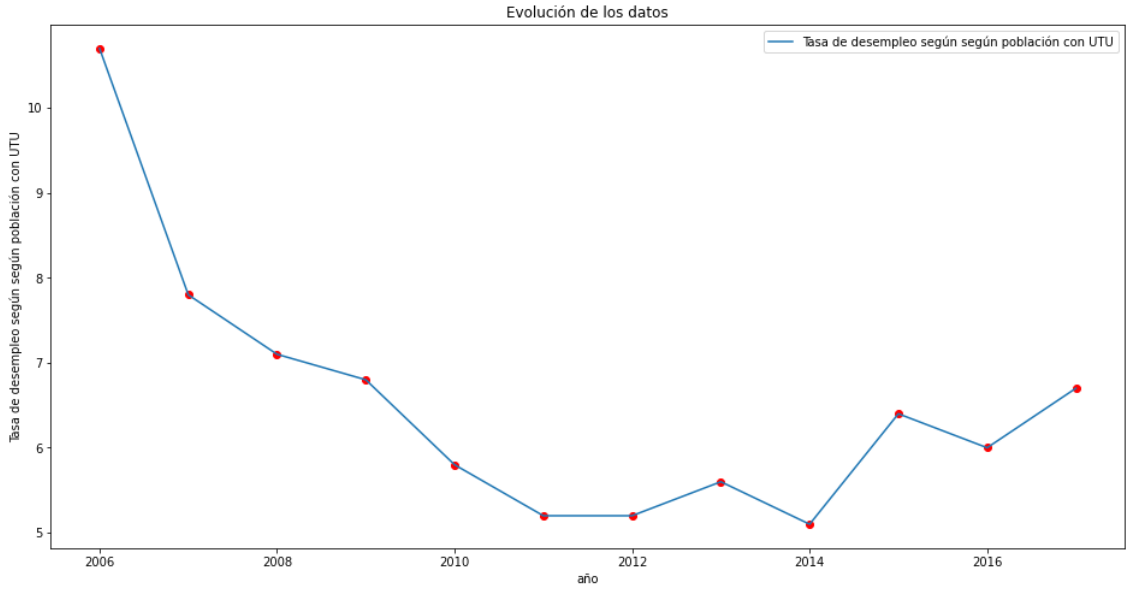
Hagamos una predicción:
La predicción para el año 2022 es de:
[[7.0943546]]
y para el 2023 es de:
[[7.097851]]

Tasa de desempleo según nivel educativo Secundaria completa:



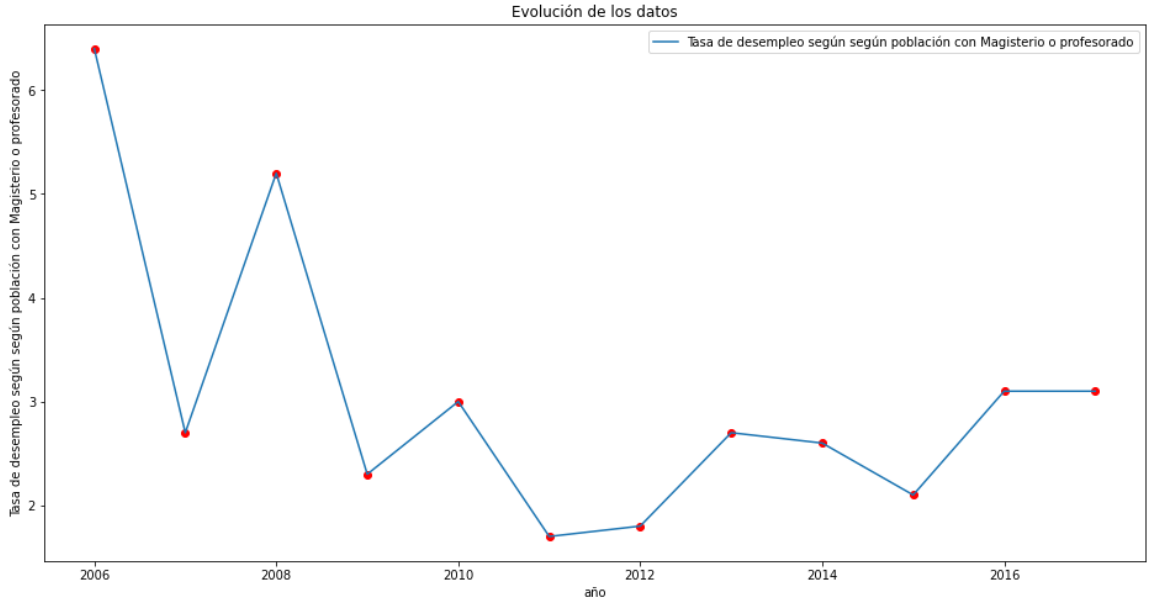
Hagamos una predicción:
La predicción para el año 2022 es de:
[[16.659025]]
y para el 2023 es de:
[[16.667227]]

Tasa de desempleo según nivel educativo UTU completa:



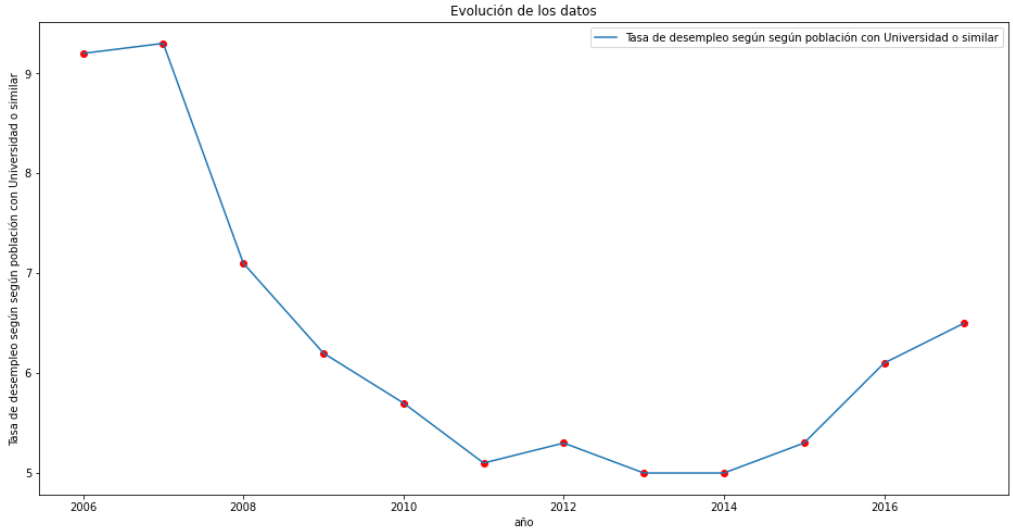
Hagamos una predicción:
La predicción para el año 2022 es de:
[[9.987397]]
y para el 2023 es de:
[[9.992335]]

Tasa de desempleo según nivel educativo Magisterio o Profesorado completa:



Hagamos una predicción:
La predicción para el año 2022 es de:
[[3.0111113]]
y para el 2023 es de:
[[3.0111113]]

Tasa de desempleo según nivel educativo Universidad o similar completa:



Hagamos una predicción:
La predicción para el año 2022 es de:
[[5.688259]]
y para el 2023 es de:
[[5.688259]]

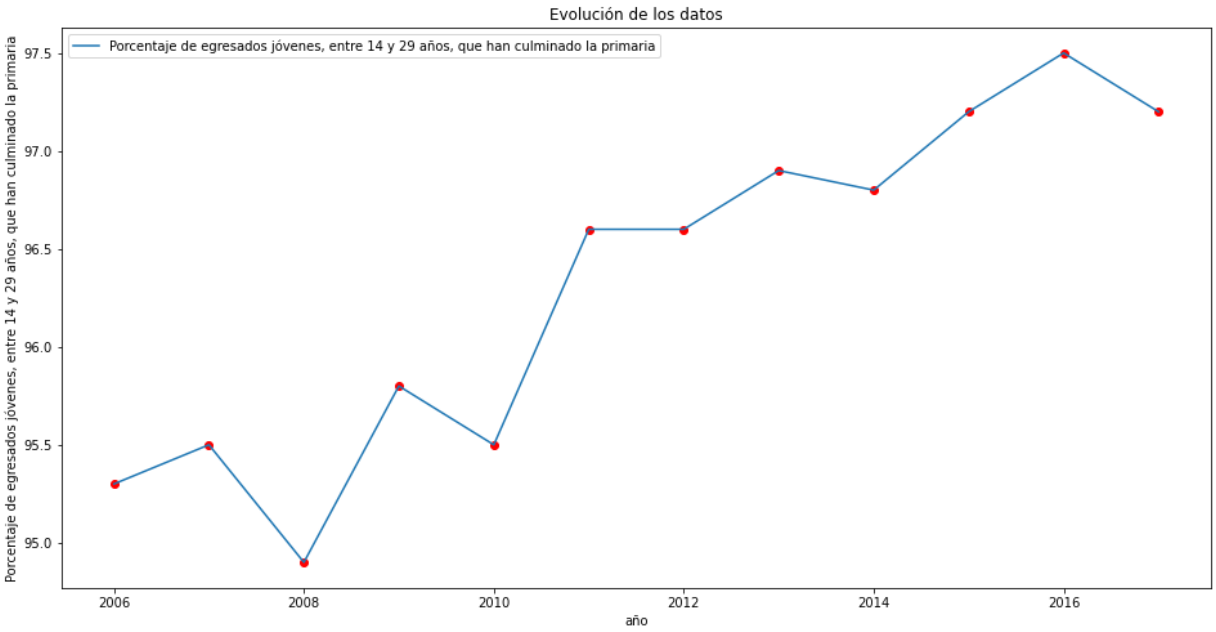
Segunda categoría de conjuntos de datos a analizar por IA de la realidad:

Por Sexo:

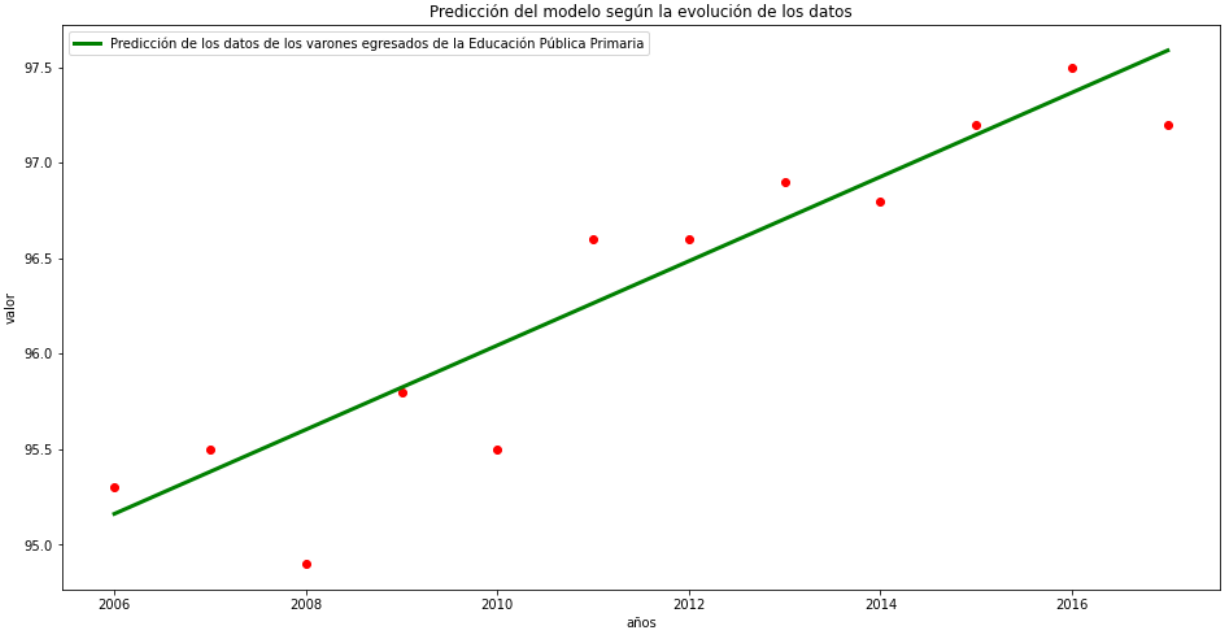
Porcentaje de jóvenes que finalizaron primaria según sexo. Total país

Para cada sexo se define como la cantidad de jóvenes que finalizaron primaria expresado en porcentaje de la cantidad de jóvenes. Se considera a los jóvenes de 14 a 29 años.

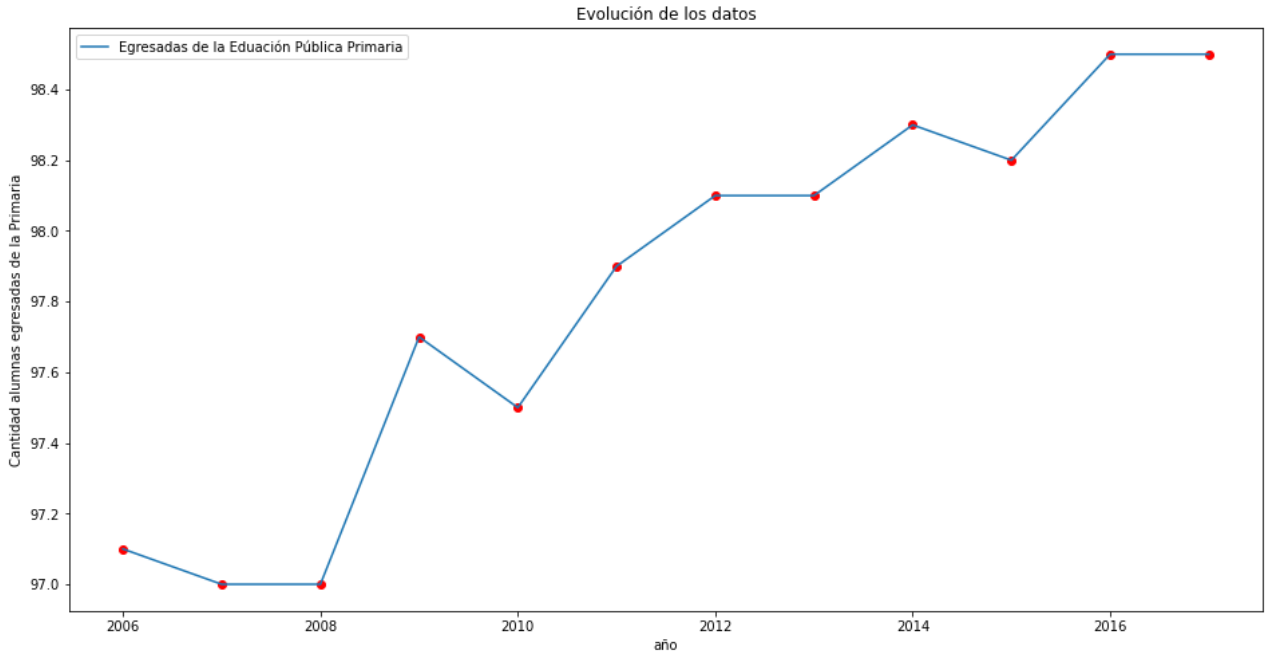
Nombre del conjunto de datos a analizar por Inteligencia Artificial:
7829_porcentaje_de_jovenes_que_finalizaron_primaria_segun_sexo-_total_pais.csv



Hagamos una predicción:
La predicción para el año 2022 es de:
[98.69036458]
y para el 2023 es de:
[98.9109375]
de porcentaje de alumnos varones respectivamente
(usando un Multi layer Linear Regression):

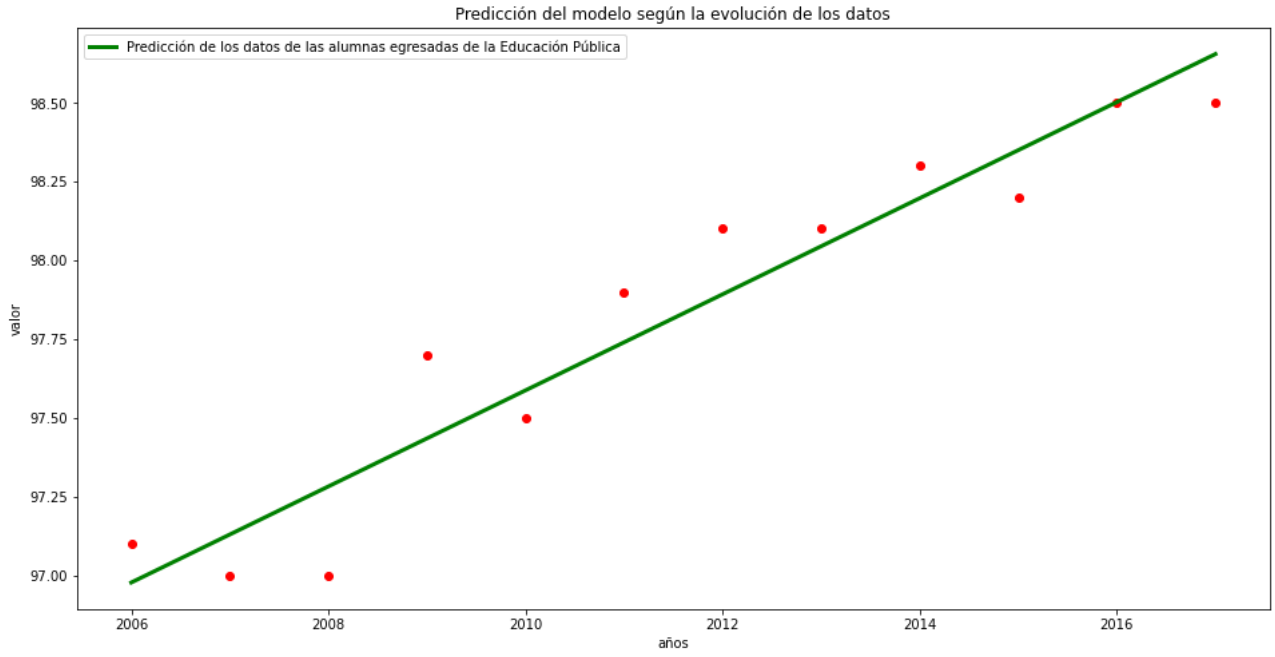


Porcentaje de jóvenes que finalizaron primaria según sexo Mujeres :



Predicciones:

Hagamos una predicción:
La predicción para el año 2022 es de:
[99.31015326]
y para el 2023 es de:
[99.44894636]
de alumnos mujeres respectivamente
(usando un Multi layer Linear Regression):



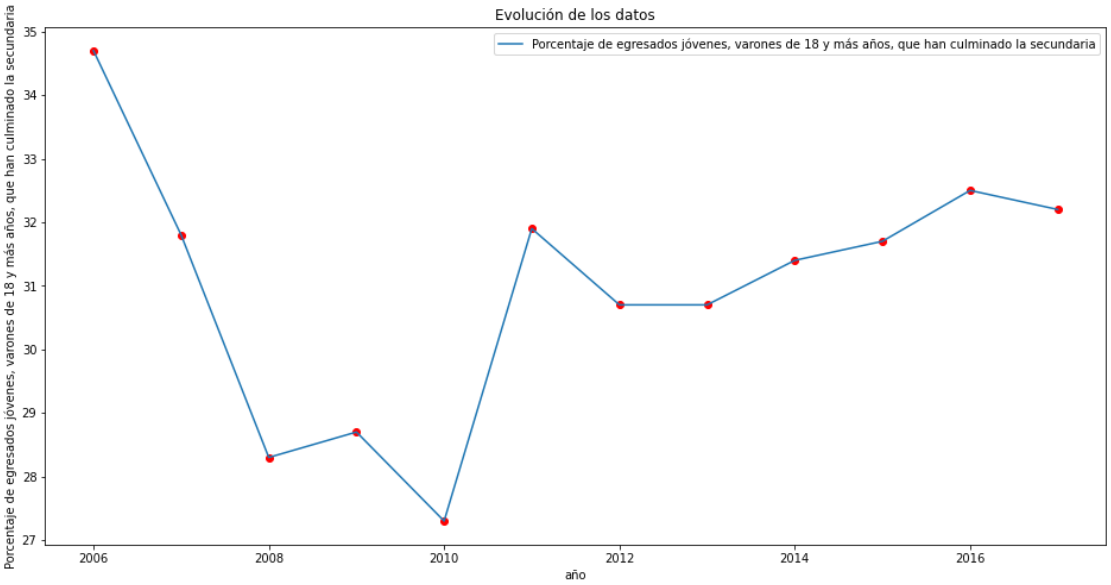
Conclusiones obtenidas al predecir para el futuro con Inteligencia Artificial:
A partir del 2010 en adelante y para el futuro 2022 y 2023, hay un crecimiento importante de la población de personas del sexo varón, que culminaron primaria.
De una forma similar, para el sexo mujer, a partir del 2010, hay un crecimiento importante de ciudadanas que culminan Primaria y para el futuro 2022 y 2023 esta tendencia tiende a crecer.

Porcentaje de jóvenes de 18 y más años que finalizaron secundaria (6° de liceo/UTU) según sexo. Total país

Para cada sexo se define como la cantidad de personas jóvenes de 18 y más años que finalizaron secundaria (6° de liceo/UTU), expresado en porcentaje del número de jóvenes de 18 y más años.

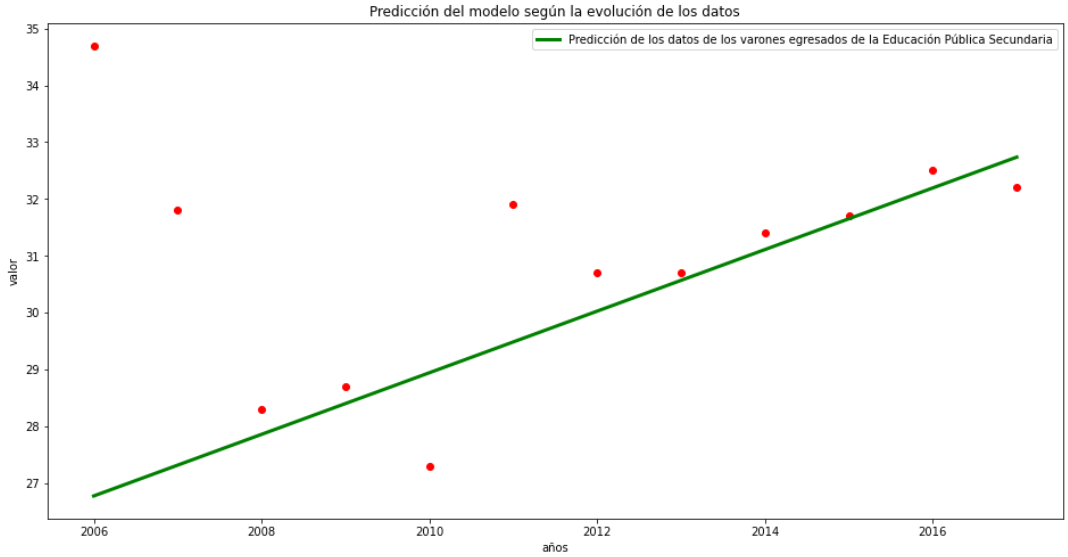
Nombre del conjunto de datos a analizar por Inteligencia Artificial:
7866_porcentaje_de_jovenes_de_18_y_mas_aos_que_finalizaron_secundaria_-6_de_liceo-utu-_segun_sex.csv

Predicciones:
Predicciones - Porcentaje de jóvenes que finalizaron secundaria según sexo Varones:



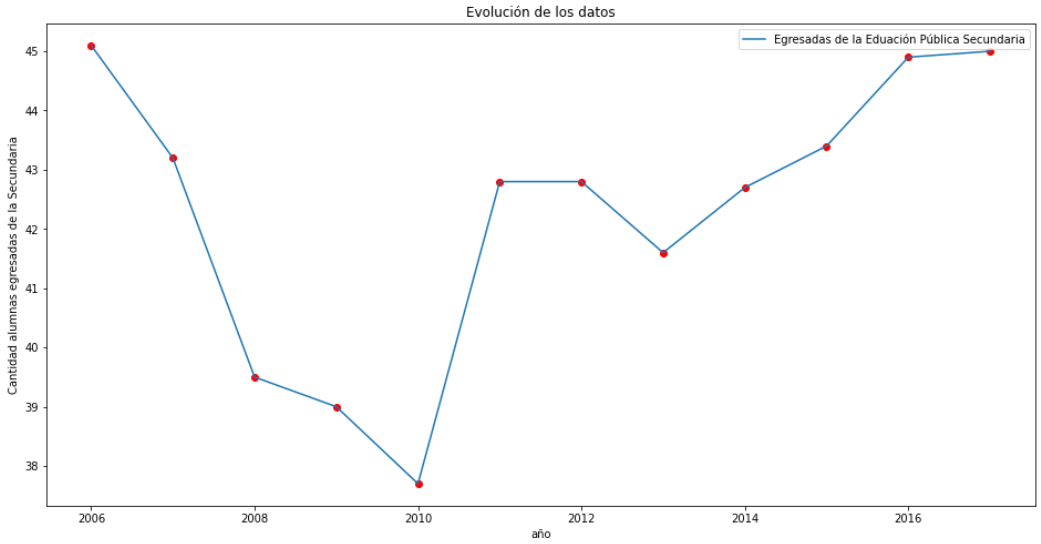
Hagamos una predicción:
La predicción para el año 2022 es de:
[35.44833333]
y para el 2023 es de:
[35.99041667]
de alumnos varones respectivamente
(usando un Multi layer Linear Regression):

Graficamente, el modelo para los Varones según Multi Linear Regression:



Hagamos una predicción para el sexo Varón, usando un perceptron multicapa:
La predicción para el año 2022 es de:
[[46.01648]]
La predicción para el año 2023 es de: [[46.038986]]

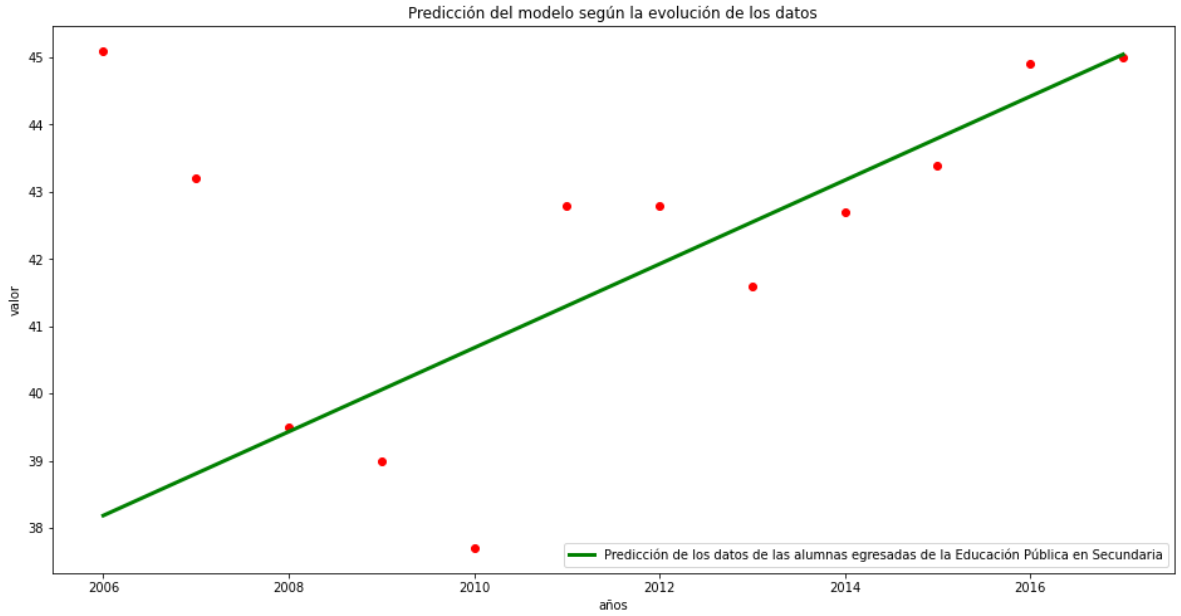
Porcentaje de jóvenes que finalizaron secundaria según sexo Mujeres :



Predicciones:

Hagamos una predicción:
La predicción para el año 2022 es de:
[48.16279412]
y para el 2023 es de:
[48.78647059]
de alumnos mujeres respectivamente
(usando un Multi layer Linear Regression)

Graficamente, el modelo para los mujeres según Multi Linear Regression:



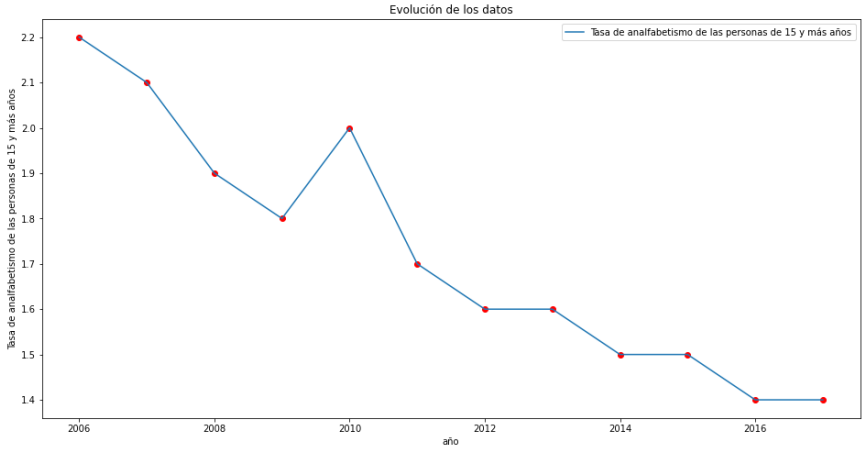
Hagamos una predicción para el sexo Mujer, usando un perceptron multicapa:
La predicción para el año 2022 es de:
[[45.62272]]
La predicción para el año 2023 es de: [[45.645206]]

Tasa de analfabetismo de las persona de 15 y más años según sexo. Total país

Porcentaje de personas de 15 y más años que no saben leer ni escribir, según sexo

Nombre del conjunto de datos a analizar por Inteligencia Artificial:
10394_tasa_de_analfabetismo_de_las_persona_de_15_y_mas_aos_segun_sexo-_total_pais.csv

Tasa de analfabetismo de las personas Varones de 15 y más años:



Predicciones:

Predicciones - Tasa de analfabetismo de las persona de 15 y más años según sexo:

Las predicciones para el 2022:
0.92 % de la población analfabeta
Para el 2023:
0.85 % de la población será analfabeta.

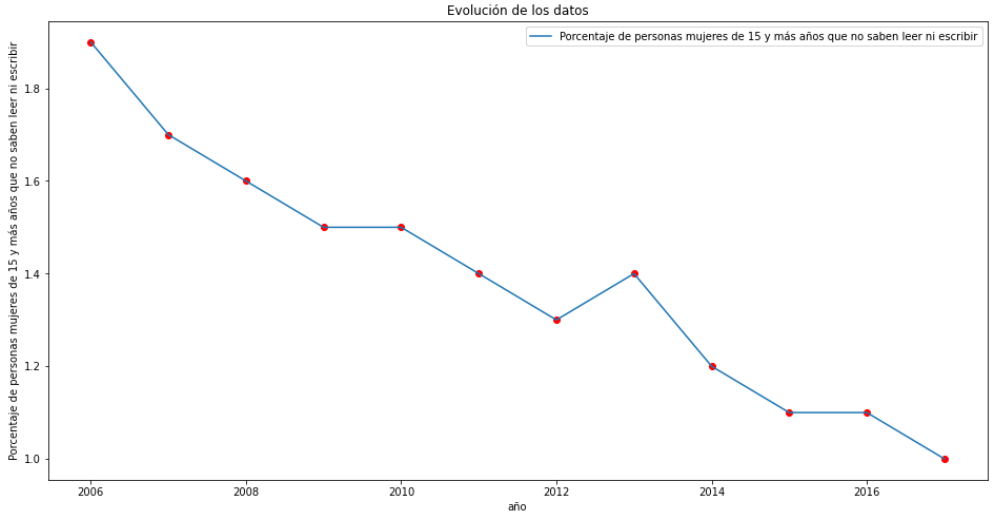
(efectividad por encima del 95% sobre el conjunto de entrenamiento)
(efectividad por encima del 74% sobre el conjunto de test)

En forma cualitativa, se puede apreciar que la tasa de analfabetismo se espera que decaiga en el presente año 2022 y más aún en el 2023

- A partir del año 2010 y en adelante para el futuro 2022 y 2023, esta tasa de analfabetismo comienza a descender de una forma muy importante, para el sexo varón.
- Para el sexo mujer, esta caída es pronunciada y continua y para el futuro 2022 y 2023 tiende a continuar decayendo.

Para los varones:
Hagamos una predicción para los Varones:
La predicción para el año 2022 es de:
[1.33367556] (MLR) y [[2.2109563]]
y para el 2023 es de:
[1.25924025] (MLR) y [[2.2119613]]
de alumnos varones respectivamente

Tasa de analfabetismo de las personas Mujeres de 15 y más años:



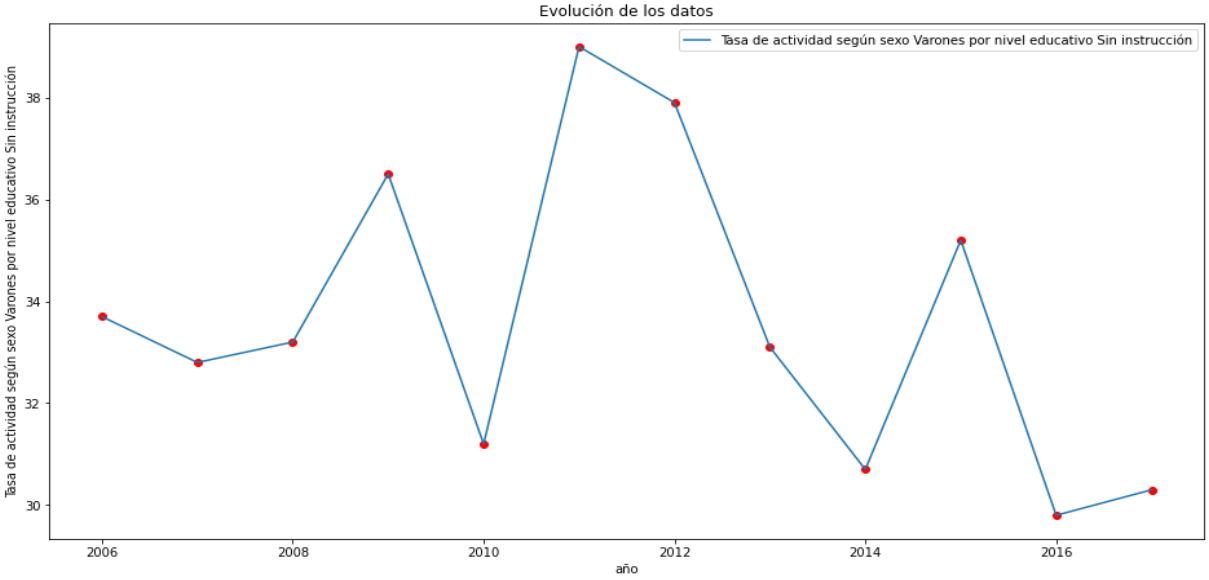
Predicciones:
Hagamos una predicción para los Mujeres:
La predicción para el año 2022 es de:
[0.61681985]
y para el 2023 es de:
[0.54099265]
de alumnos mujeres respectivamente

Tasa de actividad según sexo por nivel educativo. Total país

Proporción de varones y mujeres que se encuentran activos (trabajan o buscan trabajo) entre aquellos de 14 y más años, según nivel educativo. Nos permite medir el grado de participación de las mujeres y varones en el mercado de trabajo con dichas características.

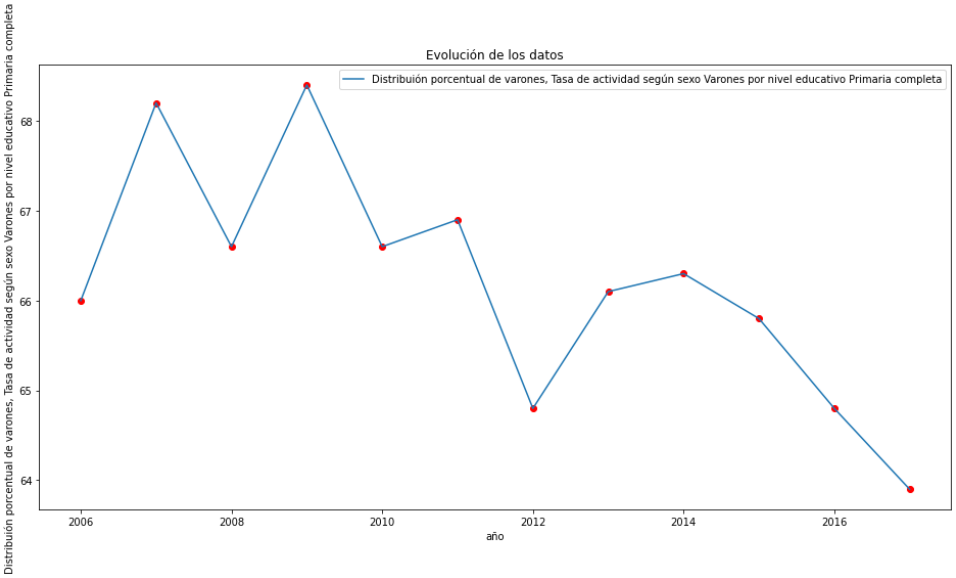
Nombre del conjunto de datos a analizar por Inteligencia Artificial:
12559_tasa_de_actividad_segun_sexo_por_nivel_educativo-_total_pais.csv

Tasa de actividad según sexo Varón por nivel educativo Sin instrucción:



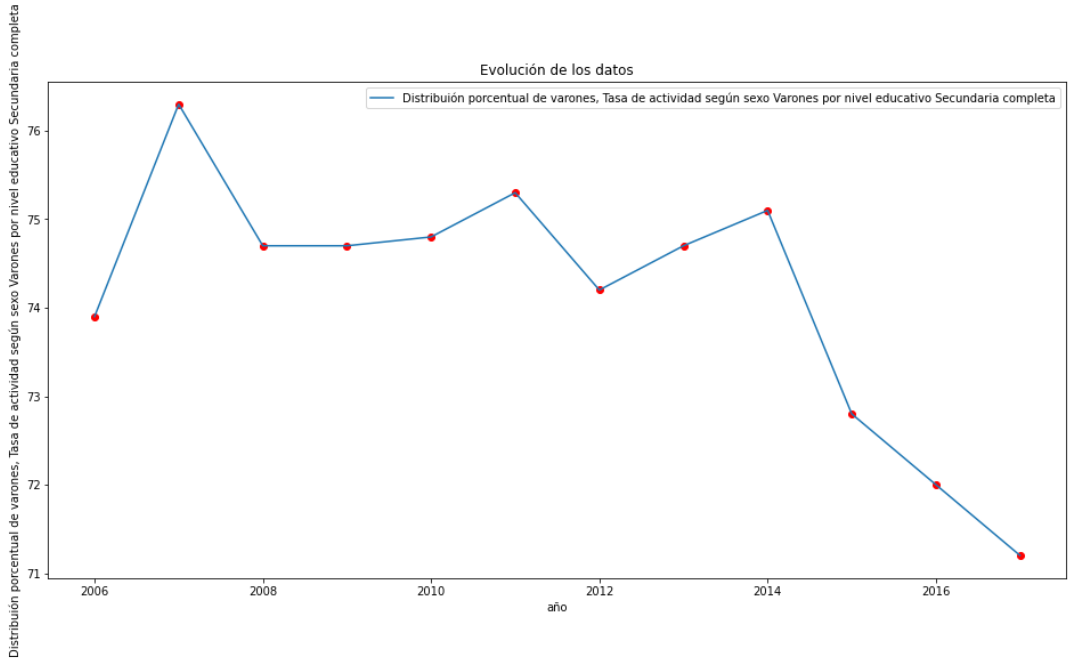
Predicciones:
Hagamos una predicción:
La predicción para el año 2022 es de:
[28.94866071] (MLR) y [[31.174189]]
y para el 2023 es de:
[28.56863839] (MLR) y [[31.18962]]
de alumnos varones respectivamente

Tasa de actividad según sexo Varón por nivel educativo Primaria completa:



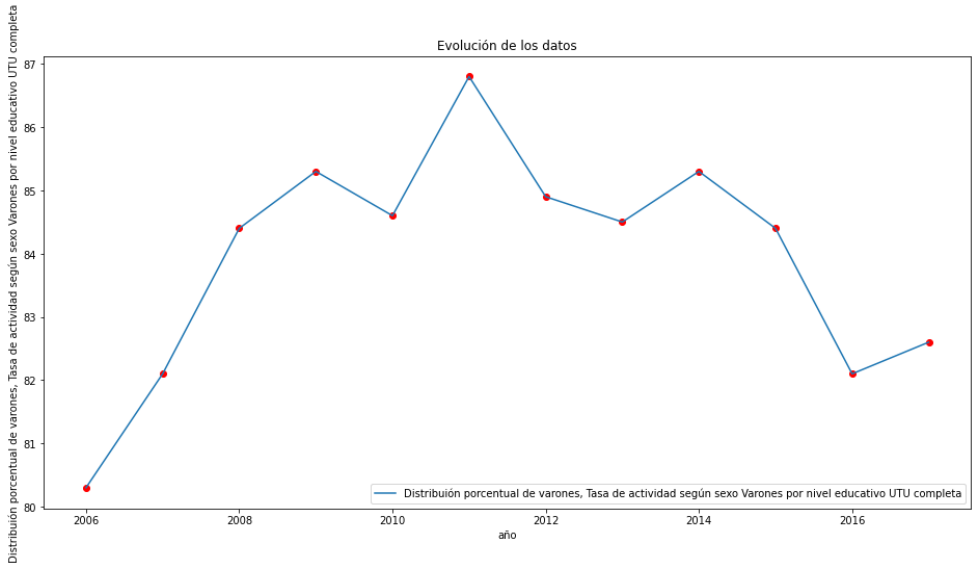
Hagamos una predicción:
La predicción para el año 2022 es de:
[62.51686747] (MLR) y [[67.49758]]
y para el 2023 es de:
[62.11746988] (MLR) y [[67.53087]]
de alumnos varones respectivamente

Tasa de actividad según sexo Varón por nivel educativo Secundaria completa:



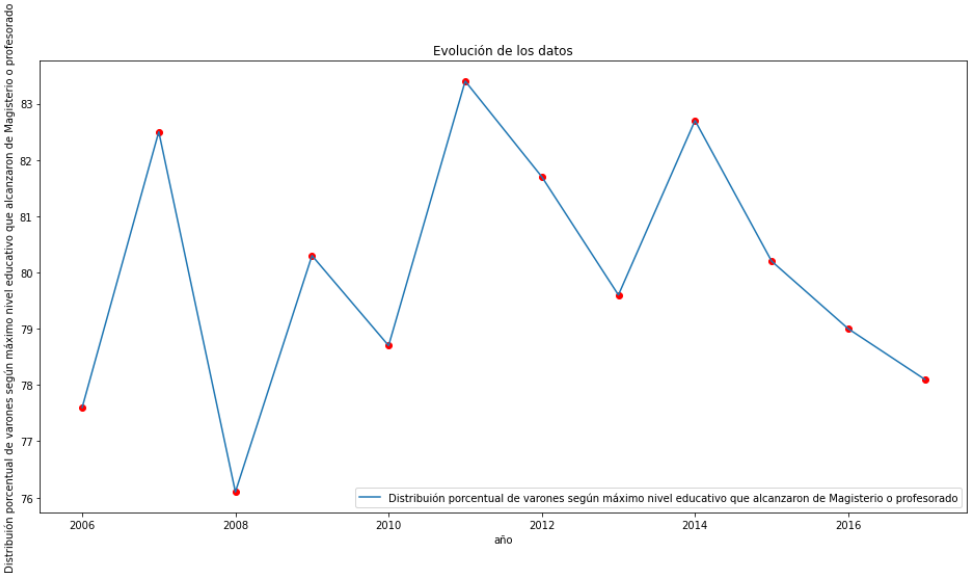
Hagamos una predicción:
La predicción para el año 2022 es de:
[69.70768156] (MLR) y [[70.24415]]
y para el 2023 es de:
[69.23743017] (MLR) y [[70.27893]]
de alumnos varones respectivamente

Tasa de actividad según sexo Varón por nivel educativo UTU completa:



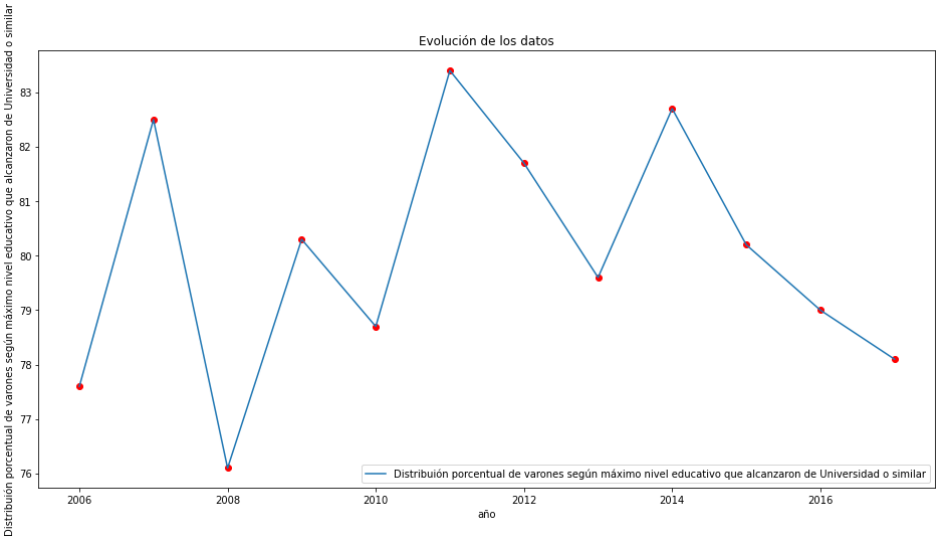
Hagamos una predicción:
La predicción para el año 2022 es de:
[81.91486486] (MLR)
y para el 2023 es de:
[81.64702703] (MLR)
de alumnos varones respectivamente

Tasa de actividad según sexo Varón por nivel educativo Magisterio o Profesorado completa:



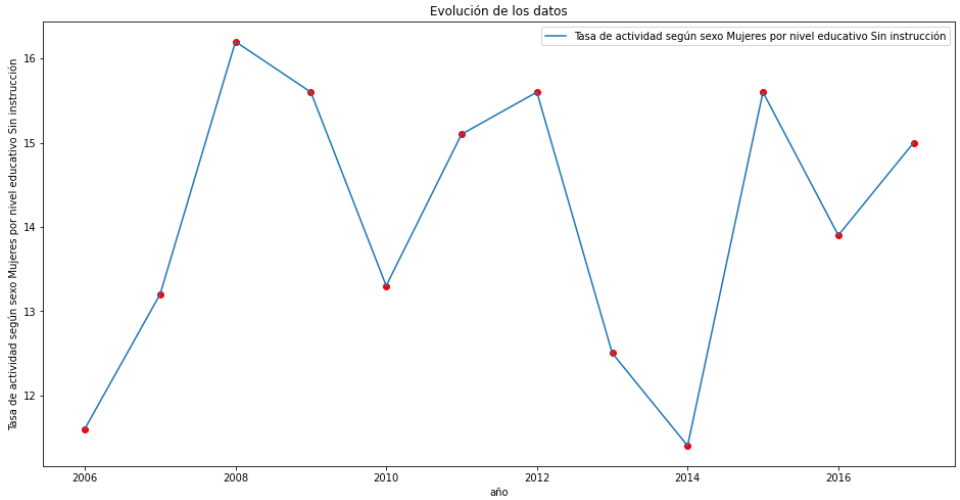
Hagamos una predicción:
La predicción para el año 2022 es de:
[85.55483871] (MLR)
y para el 2023 es de:
[86.05177419] (MLR)
de alumnos varones respectivamente

Tasa de actividad según sexo Varón por nivel educativo Universidad o similar completa:



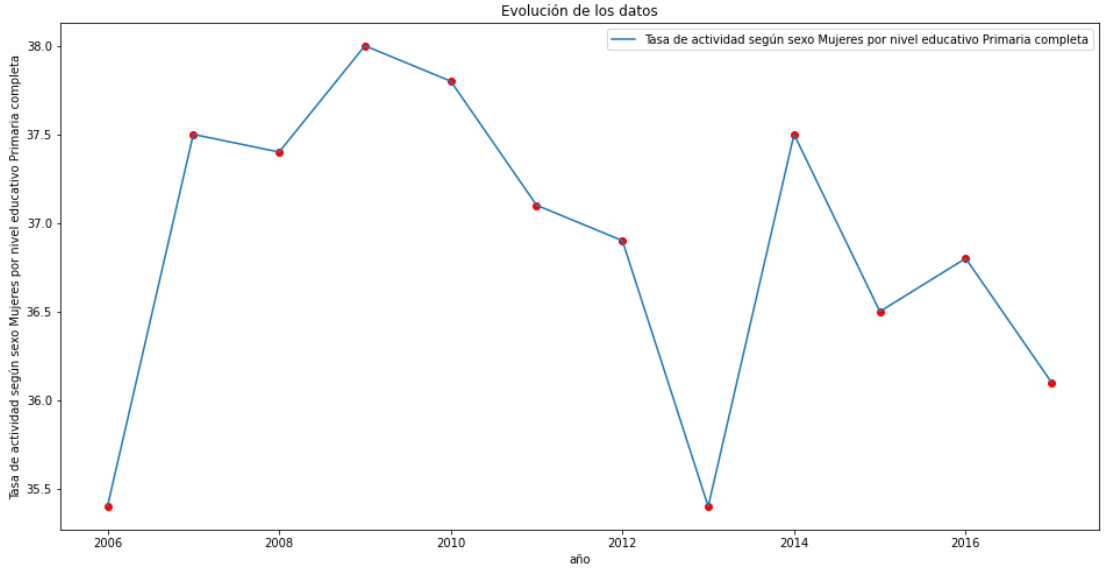
Hagamos una predicción:
La predicción para el año 2022 es de:
[77.26472868] (MLR) y [[70.636055]]
y para el 2023 es de:
[77.26472868] (MLR) y [[70.67114]]
de alumnos varones respectivamente

Tasa de actividad según sexo Mujer por nivel educativo Sin instrucción:



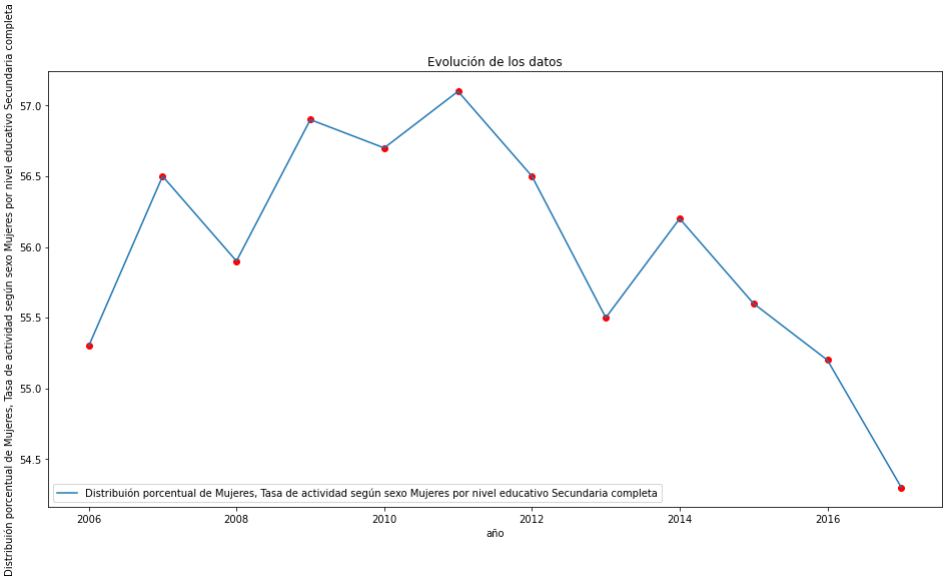
Hagamos una predicción:
La predicción para el año 2022 es de:
[16.27680608] (MLR) y [[18.262402]]
y para el 2023 es de:
[16.50418251] (MLR) y [[18.27168]]
de alumnos mujeres respectivamente

Tasa de actividad según sexo Mujer por nivel educativo Primaria completa:



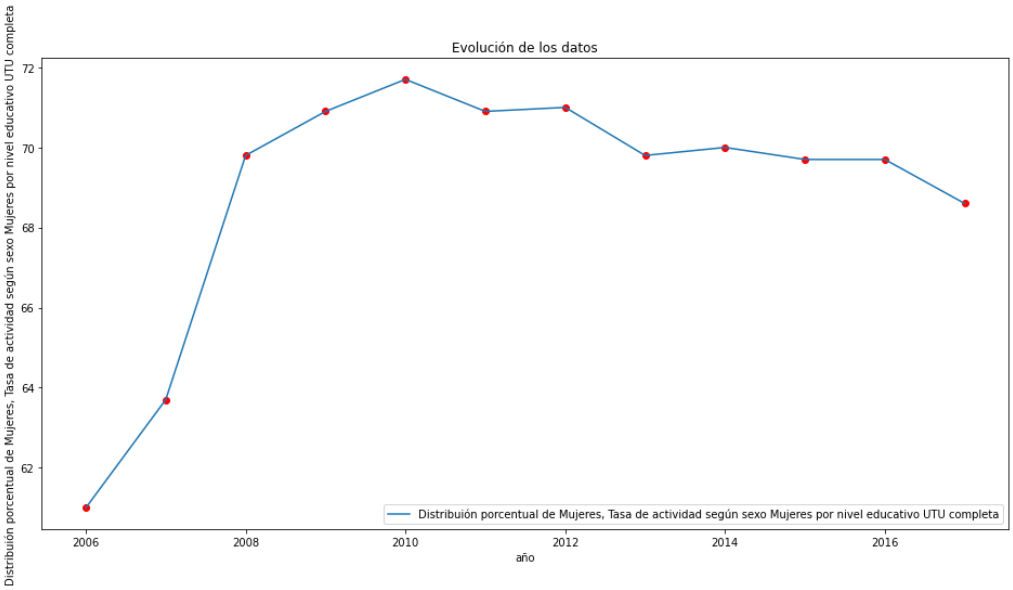
Hagamos una predicción:
La predicción para el año 2022 es de:
[35.32614379] (MLR)
y para el 2023 es de:
[35.17320261] (MLR)
de alumnos varones respectivamente

Tasa de actividad según sexo Mujer por nivel educativo Secundaria completa:



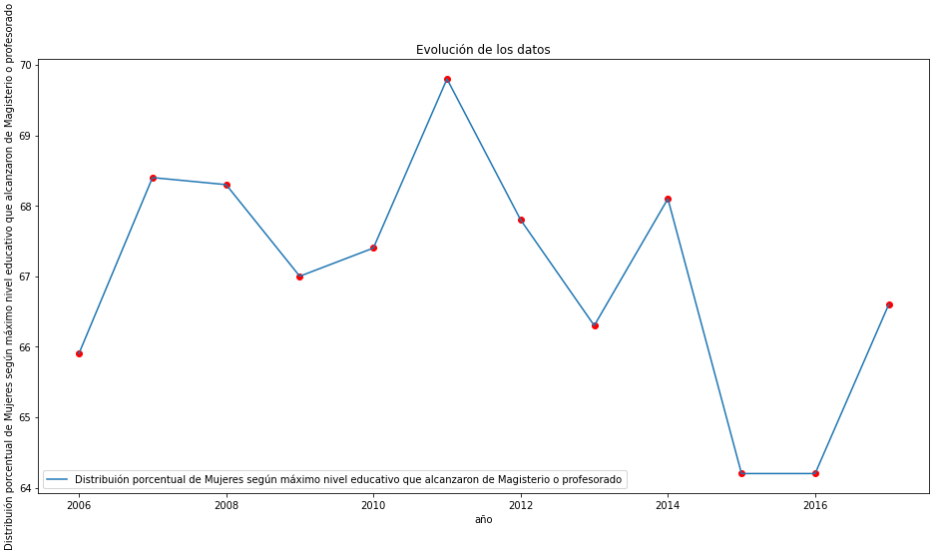
Hagamos una predicción:
La predicción para el año 2022 es de:
[53.27] (MLR)
para el 2023 es de:
[52.96666667] (MLR)
de alumnos varones respectivamente

Tasa de actividad según sexo Mujer por nivel educativo UTU completa:



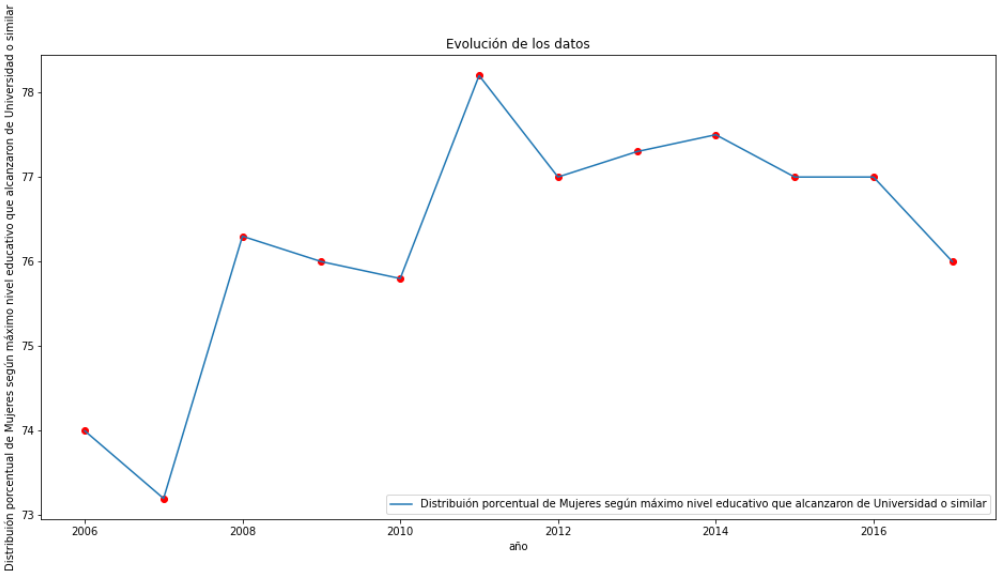
Hagamos una predicción:
La predicción para el año 2022 es de:
[[65.586525]]
para el 2023 es de:
[[65.61888]]
de alumnos varones respectivamente

Tasa de actividad según sexo Mujer por nivel educativo Magisterio o Profesorado completa:



Hagamos una predicción:
La predicción para el año 2022 es de:
[63.25603814] (MLR) y [[58.779606]]
para el 2023 es de:
[62.91652542] (MLR) y [[58.80884]]
de alumnos varones respectivamente

Tasa de actividad según sexo Mujer por nivel educativo Universidad o similar completa:



Hagamos una predicción:
La predicción para el año 2022 es de:
[79.25662651] (MLR)
para el 2023 es de:
[79.52409639] (MLR)
de alumnos varones respectivamente

Tercera categoría de conjuntos de datos a analizar por IA de la realidad:

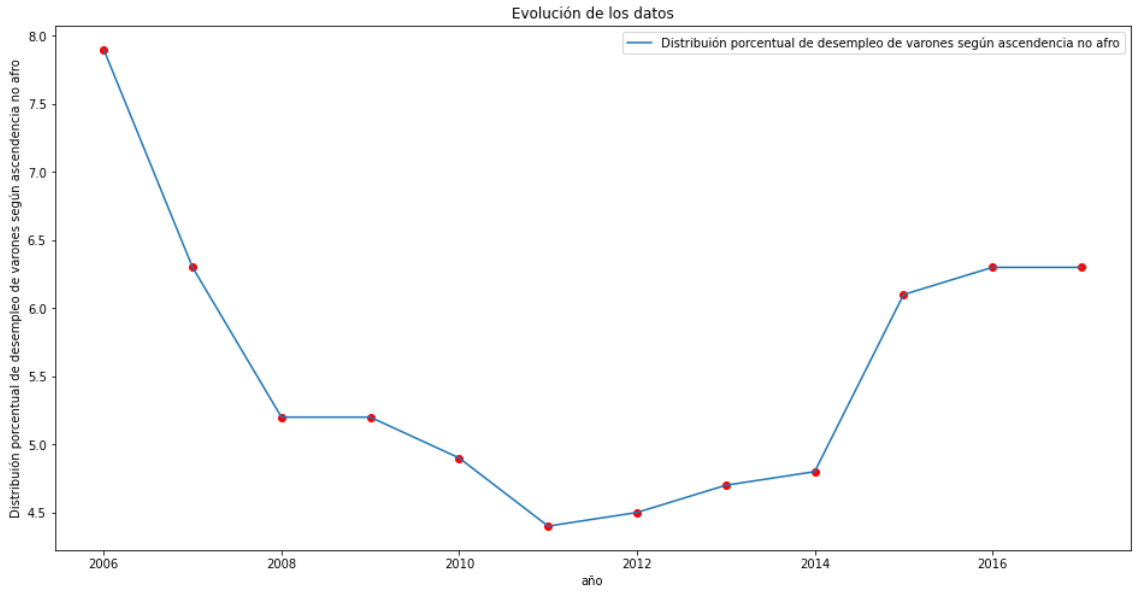
Por ascendencia racial:

Tasa de desempleo según sexo por ascendencia afro. Total país

Proporción de varones y mujeres que buscan empleo y no tienen, en relación a la población económicamente activa, según ascendencia afro.

Nombre del conjunto de datos a analizar por Inteligencia Artificial:
10213_tasa_de_desempleo_segun_sexo_por_ascendencia_afro-_total_pais.csv

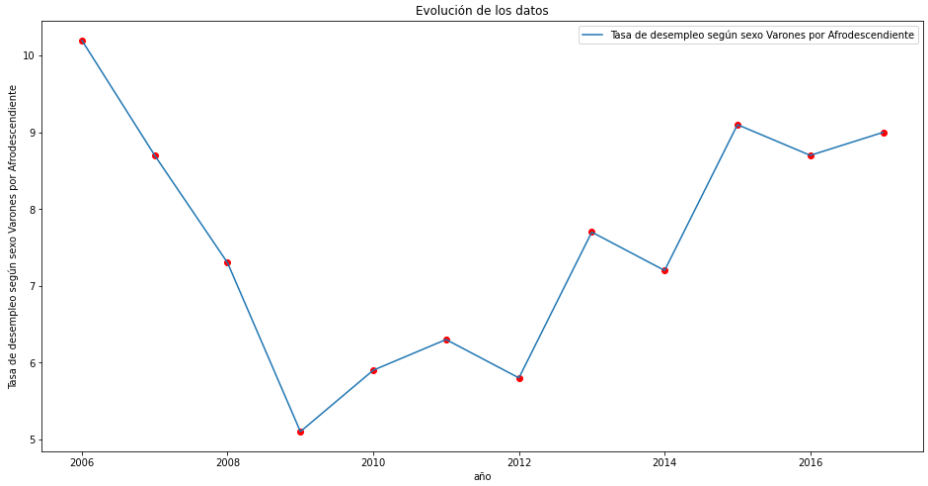
Tasa de desempleo según sexo Varón por ascendencia no afro :



Predicciones:

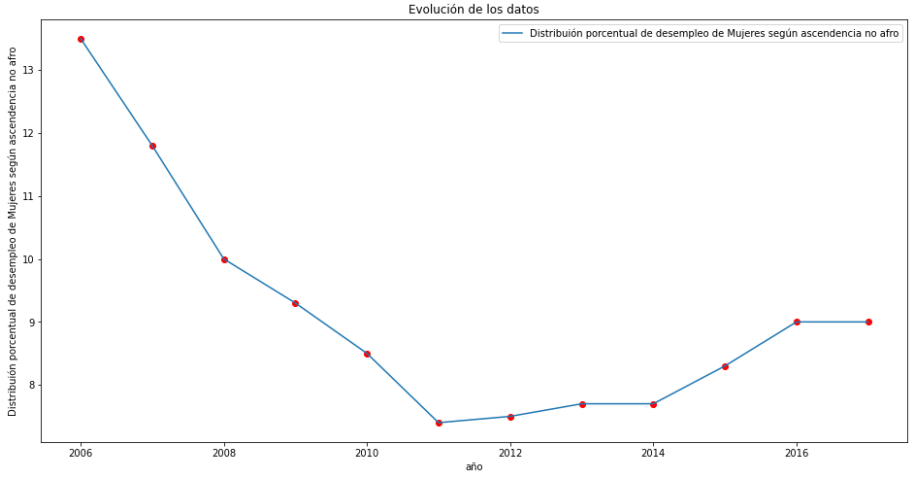
Hagamos una predicción:
La predicción para el año 2022 es de:
[[5.059894]]
y para el 2023 es de:
[[5.0626025]]
de alumnos varones respectivamente

Tasa de desempleo según sexo Varón por ascendencia afro:



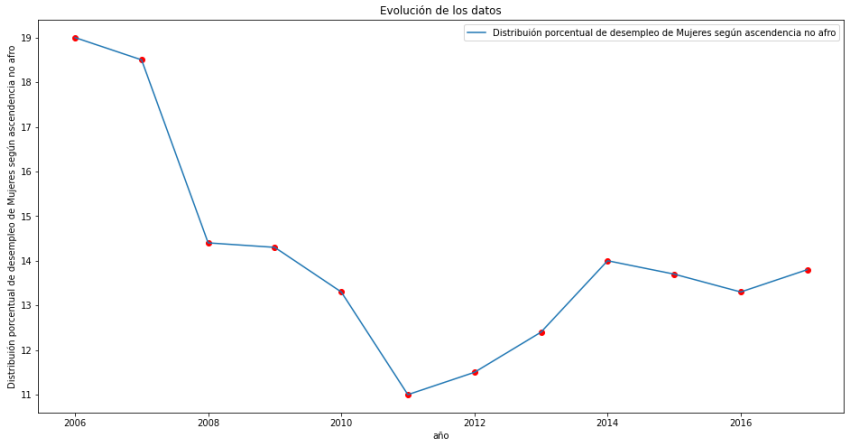
Hagamos una predicción:
La predicción para el año 2022 es de:
[11.85] (MLR)
y para el 2023 es de:
[12.36666667] (MLR)
de alumnos varones respectivamente

Tasa de desempleo según sexo Mujeres por ascendencia no afro:



Hagamos una predicción:
La predicción para el año 2022 es de:
[4.88367347] (MLR)
y para el 2023 es de:
[4.46520408] (MLR)
de mujeres respectivamente

Tasa de desempleo según sexo Mujeres por ascendencia afro:



Hagamos una predicción:
La predicción para el año 2022 es de:
[10.39622951] (MLR)
y para el 2023 es de:
[9.99278689] (MLR)
de mujeres respectivamente

Porcentaje de personas en situación de pobreza según sexo por ascendencia afro. Total país

Porcentaje de varones y mujeres que habitan en hogares cuyo ingreso per cápita es inferior a la línea de pobreza (metodología INE 2006), según ascendencia afro.

Nombre del conjunto de datos a analizar por Inteligencia Artificial:
10885_porcentaje_de_personas_en_situacion_de_pobreza_segun_sexo_por_ascendencia_afro-_total_pais.csv

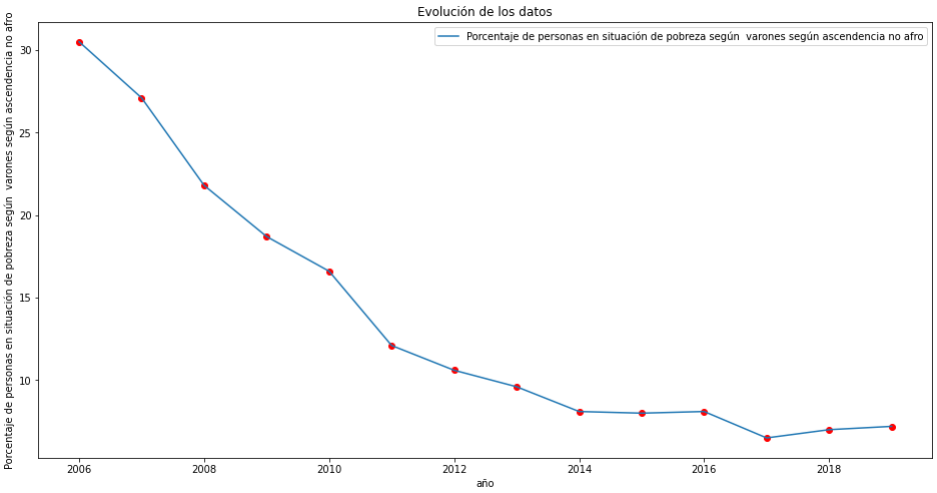
Conclusiones obtenidas al predecir para el futuro con Inteligencia Artificial:
Para los varones masculinos no afrodescendientes, la situación de pobreza venía decayendo, pero para el futuro, se pronostica un aumento.
De forma similar, para los varones masculinos afrodescendientes, la situación de pobreza venía decayendo, pero para el futuro, se pronostica un aumento.

Para las mujeres no afrodescendientes, la situación de pobreza venía decayendo, y para el futuro, se pronostica que continúe esta caída.

Sin embargo, para las mujeres afrodescendientes, la situación de pobreza venía decayendo, pero para el futuro, se pronostica un aumento.

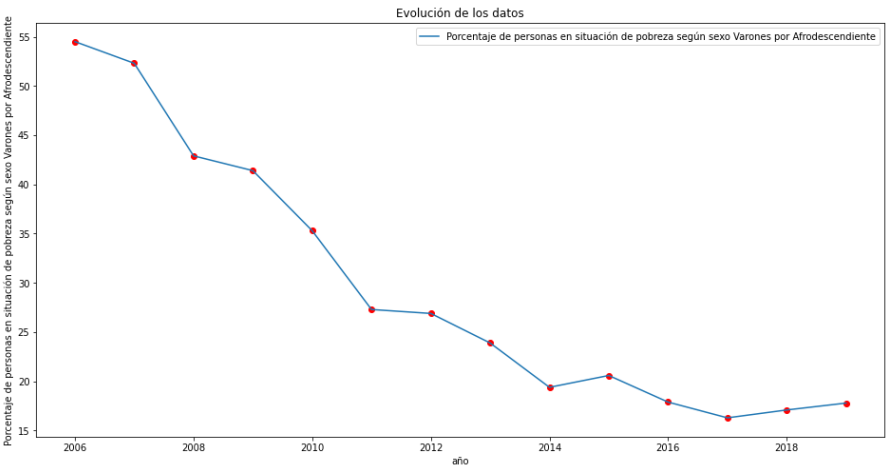
Predicciones:

Porcentaje de personas en situación de pobreza según sexo Varón por ascendencia no afro :



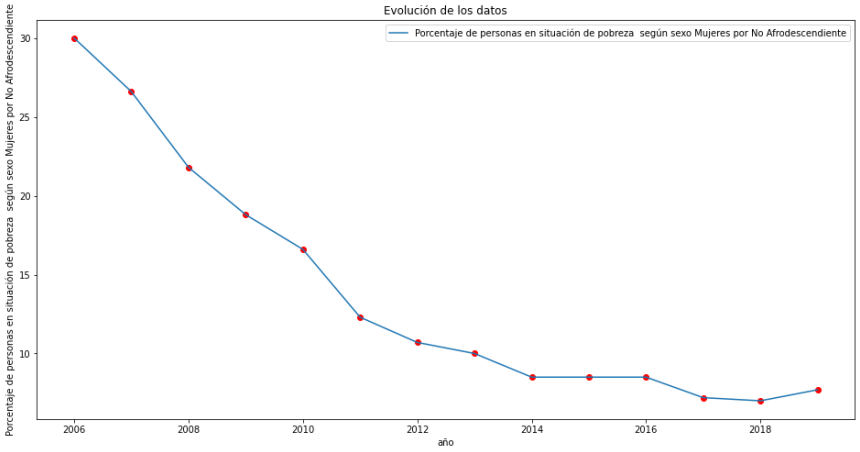
Hagamos una predicción:
La predicción para el año 2022 es de:
[[17.991688]]
y para el 2023 es de:
[[18.000467]]

Porcentaje de personas en situación de pobreza según sexo Varón por ascendencia afro:



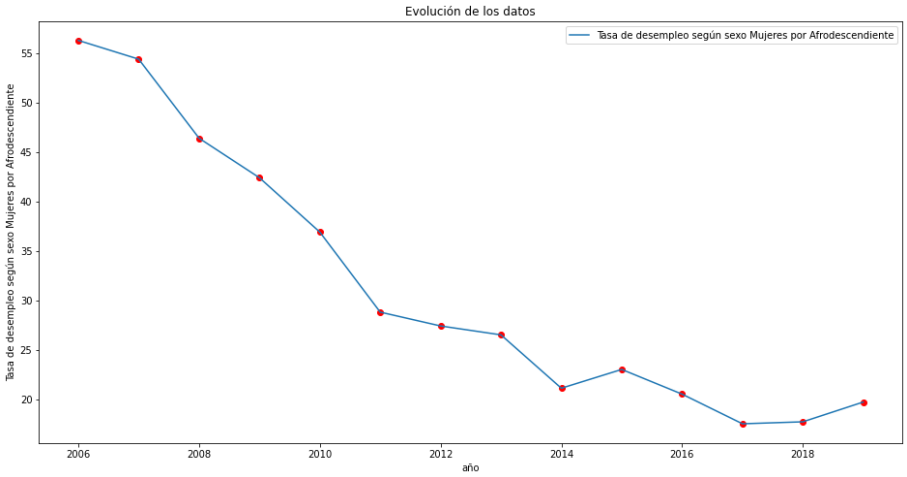
Hagamos una predicción:
La predicción para el año 2022 es de:
[[38.976444]]
y para el 2023 es de:
[[38.99552]]

Porcentaje de personas en situación de pobreza según sexo Mujeres por ascendencia no afro:



Hagamos una predicción:
La predicción para el año 2022 es de:
[[8.006077]]
y para el 2023 es de:
[[8.009811]]
de mujeres respectivamente

Porcentaje de personas en situación de pobreza según sexo Mujeres por ascendencia afro:



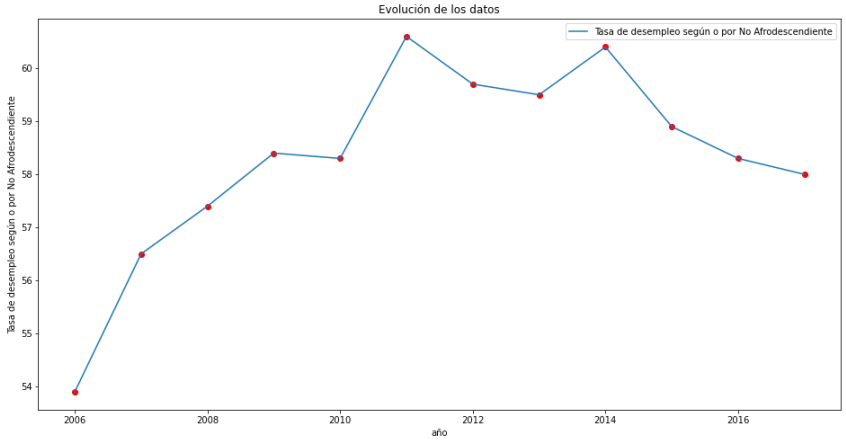
Hagamos una predicción:
La predicción para el año 2022 es de:
[[46.512768]]
y para el 2023 es de:
[[46.535812]]
de mujeres respectivamente

Tasa de empleo según ascendencia afro. Total país

Proporción de personas que declaran trabajar en relación a aquellos de 14 y más años, según ascendencia afro.

Nombre del conjunto de datos a analizar por Inteligencia Artificial:
10227_tasa_de_empleo_segun_ascendencia_afro-_total_pais.csv

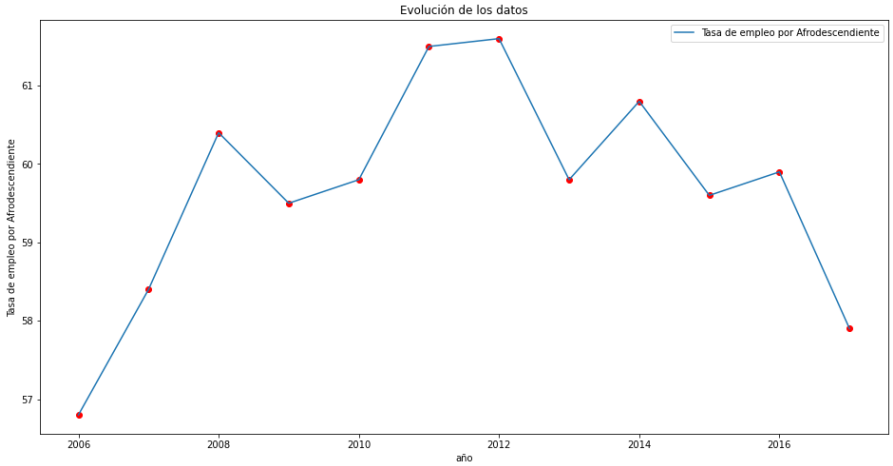
Tasa de empleo según ascendencia no afro :



Predicciones:

Hagamos una predicción:
La predicción para el año 2022 es de:
[[60.87546]]
y para el 2023 es de:
[[60.905518]]

Tasa de empleo según ascendencia afro:



Hagamos una predicción:
La predicción para el año 2022 es de:
[[57.572514]]
y para el 2023 es de:
[[57.600872]]

Conclusiones obtenidas al predecir para el futuro con Inteligencia Artificial:
Para las personas no afrodescendientes, la tendencia en el futuro es que el empleo crezca, sin embargo para los afrodescendientes, la tendencia es de crecimiento.

Cuarta categoría de conjuntos de datos a analizar por IA de la realidad:

Situación de pobreza:

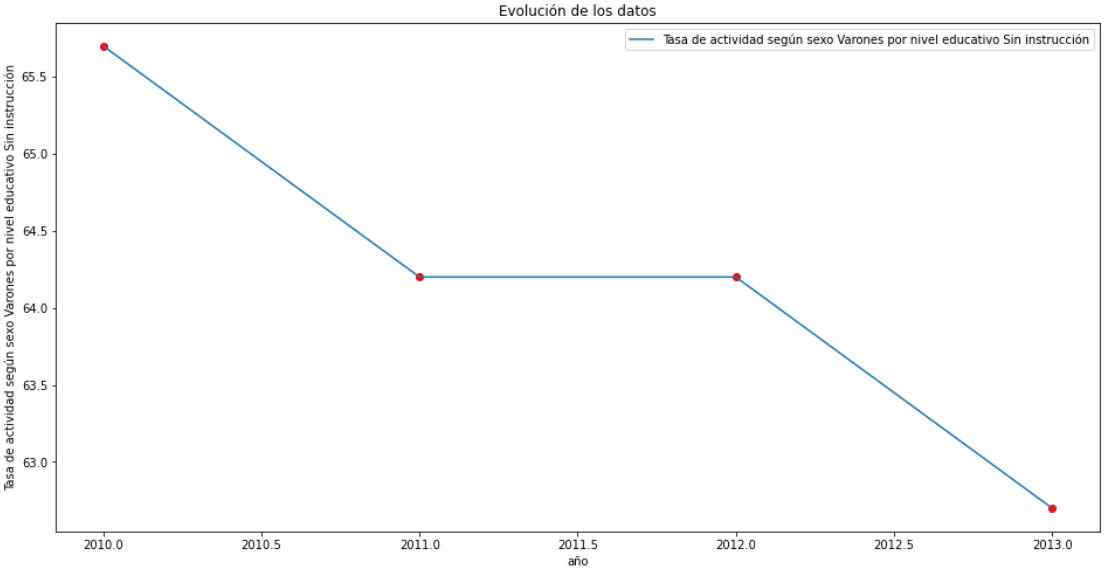
Tasa de empleo por sexo y situación de pobreza

Proporción de personas trabajando en relación a todas las personas de 14 y más años, sexo y situación de pobreza.

Conclusiones obtenidas al predecir para el futuro con Inteligencia Artificial:
Para las mujeres pobres tiende a decaer la tasa, y para las mujeres no pobres, también tienden a decaer.
Para los varones pobres, tiende a decaer la tasa, y para los varones no pobres, tiende a crecer.

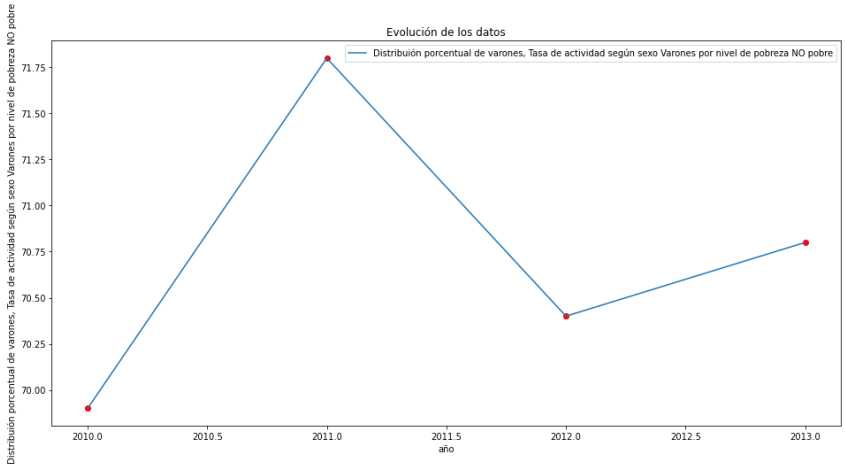
Nombre del conjunto de datos a analizar por Inteligencia Artificial:
3688_tasa_de_empleo_por_sexo_y_situacion_de_pobreza.csv

Tasa de empleo según sexo Masculino y situación de pobreza Pobre:



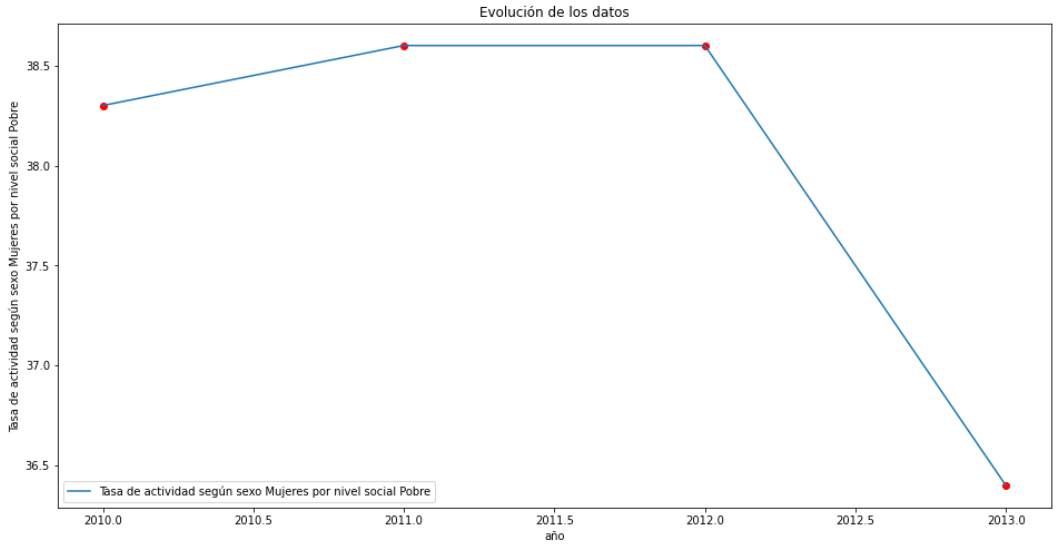
Hagamos una predicción:
La predicción para el año 2022 es de:
[53.91428571]
y para el 2023 es de:
[52.95] de varones pobres respectivamente
(usando un Multi layer Linear Regression):

Tasa de empleo según sexo Masculino y situación de pobreza No Pobre:



Hagamos una predicción:
La predicción para el año 2022 es de:
[73.39285714]
y para el 2023 es de:
[73.68571429] de varones NO pobres respectivamente
(usando un Multi layer Linear Regression):

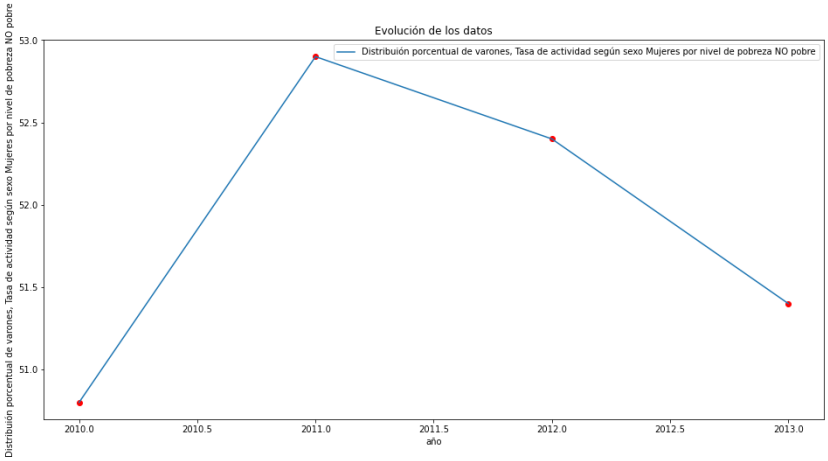
Tasa de empleo según sexo Femenino y situación de pobreza Pobre:



Hagamos una predicción:
La predicción para el año 2022 es de:
[40.15]
y para el 2023 es de:

[40.3] de Mujeres pobres respectivamente
(usando un Multi layer Linear Regression)

Tasa de empleo según sexo Femenino y situación de pobreza NO Pobre:



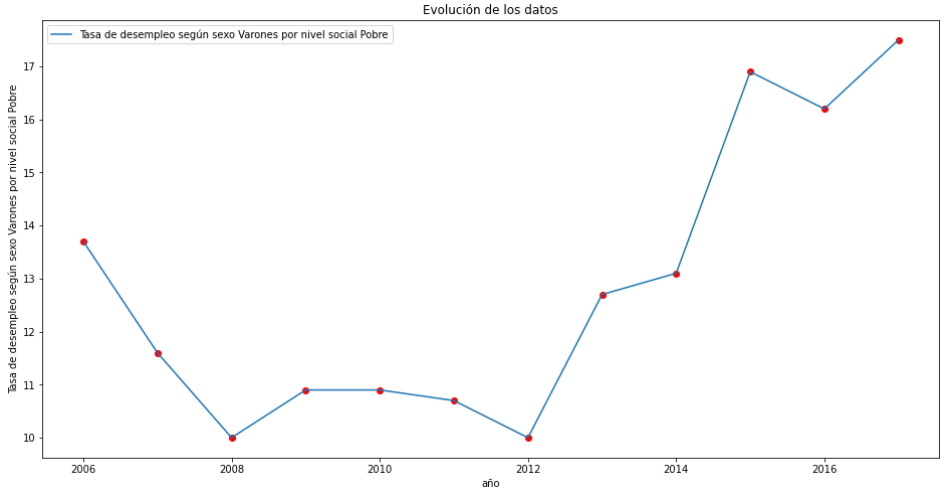
Hagamos una predicción:
La predicción para el año 2022 es de:
[44.73333333]
y para el 2023 es de:
[43.98333333] de Mujeres NO pobres respectivamente
(usando un Multi layer Linear Regression)

Tasa de desempleo según sexo por situación de pobreza. Total país

Proporción de varones y mujeres que buscan empleo y no tienen, en relación a la población económicamente activa, según situación de pobreza.

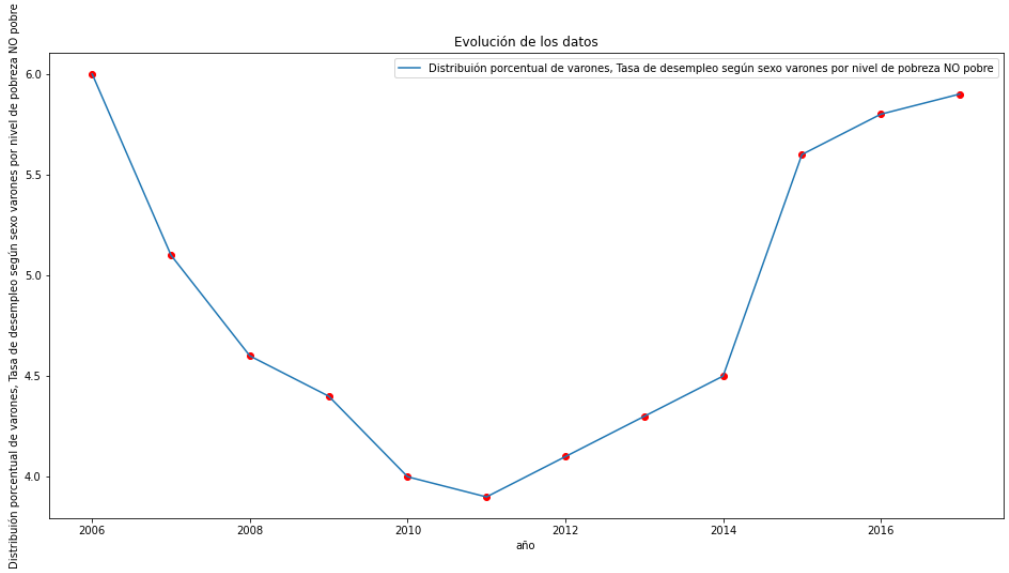
Conclusiones obtenidas al predecir para el futuro con Inteligencia Artificial:
Para las mujeres pobres, tienden a crecer su tasa de desempleo, y para las mujeres no pobres tienden también a mantenerse a una caída.
Para los hombres pobres, tienden a crecer su tasa de desempleo, y para los hombres no pobres, tienden a crecer su tasa de desempleo.
Nombre del conjunto de datos a analizar por Inteligencia Artificial:
9404_tasa_de_desempleo_segun_sexo_por_situacion_de_pobreza-_total_pais.csv

Tasa de desempleo según sexo Varón por situación de pobreza Pobre:



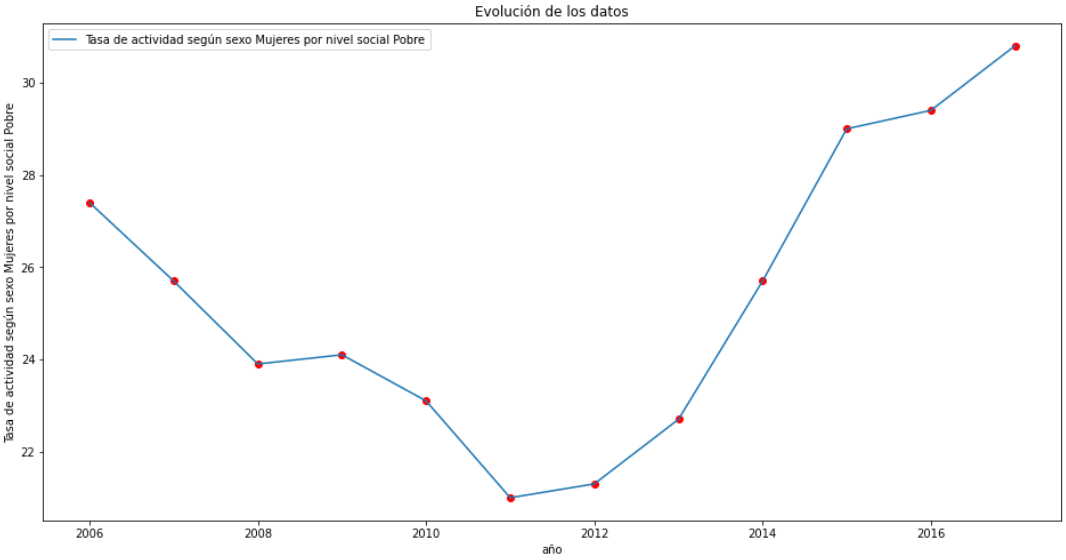
Hagamos una predicción:
La predicción para el año 2022 es de:
[21.32567568] (MLR) y [[21.486763]]
y para el 2023 es de:
[22.18486486] (MLR) y [[21.497564]]
de alumnos varones respectivamente

Tasa de desempleo según sexo Varón por situación de pobreza NO Pobre:



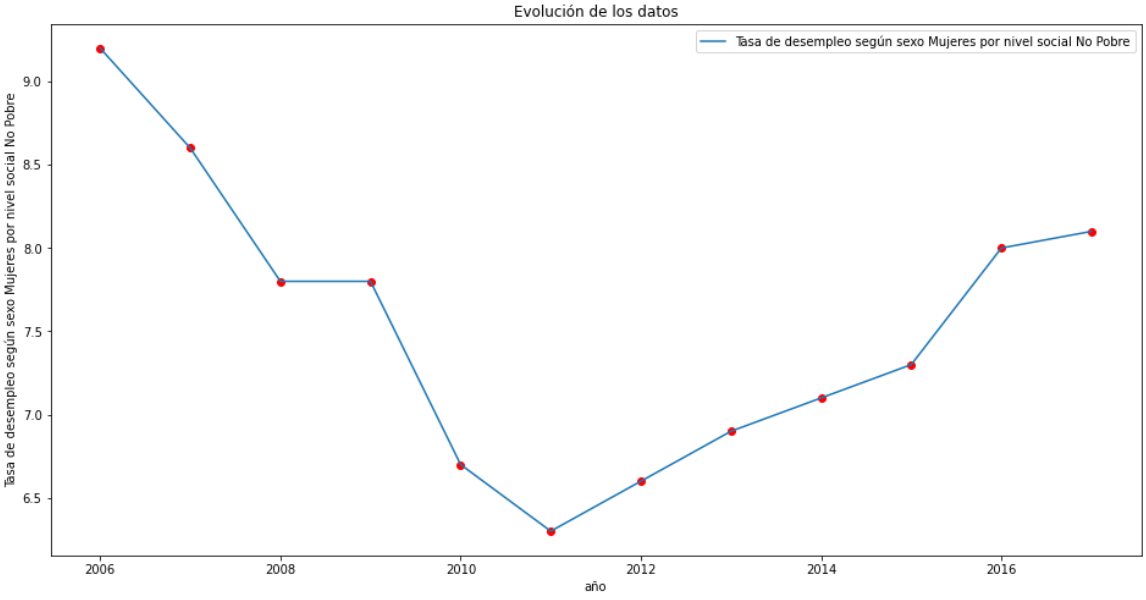
Hagamos una predicción:
La predicción para el año 2022 es de:
[7.00222222] (MLR)
y para el 2023 es de:
[7.25555556] (MLR)
de alumnos varones respectivamente

Tasa de desempleo según sexo Mujer por situación de pobreza Pobre:



Hagamos una predicción:
La predicción para el año 2022 es de:
[30.13625] (MLR) y [[37.447395]]
y para el 2023 es de:
[30.5375] (MLR) y [[37.46581]]
de mujeres respectivamente

Tasa de desempleo según sexo Mujer por situación de pobreza NO Pobre:



Hagamos una predicción:
La predicción para el año 2022 es de:
[3.96444444] (MLR) y [[8.773492]]
y para el 2023 es de:
[3.67444444] (MLR) y [[8.777791]]
de mujeres respectivamente

Capitulo 4 - Conclusiones - Predicción para el 2022 y el 2023 para la educación, en forma cualitativa.

Conclusiones cualitativas obtenidas al predecir para el futuro con Inteligencia Artificial:

En forma cualitativa, se puede obtener una red neuronal "interesante", que predice en forma adecuada para el 2022 y el 2023, en forma coherente los datos futuros en base a los datos históricos, adquiriendo un patrón de comportamiento en los datos.

Cantidad de ingresos y egresos en carreras universitarias de grado según forma de administración. Total país

Comparando las Universidades Públicas con las Universidades Privadas:

Para los ingresos:
Va a ver un crecimiento en los ingresos de alumnos en las Universidades públicas así como también en las privadas.
Para los ingresos en la Universidad Pública:
Se pronostica ingresados en la Universidad Pública para el 2022 y el 2023: 28431 y 29061 de alumnos.
Para los ingresos en la Universidades privadas:
Se pronostica ingresados en la Universidades privadas para el 2022 y el 2023: 5513 y 5681 de alumnos.

Para los egresos:
En resumen y en síntesis: para los años 2022 y 2023, crecerá el número de estudiantes egresados en las Universidades públicas así como también en las Universidades privadas.
Para los egresos en la Universidad Pública:
7678 alumnos egresados para el 2022 y 7879 alumnos egresados para el 2023 para la Universidad pública.
Para los egresos en las Universidades privadas:
2616 alumnos egresados para el 2022 y 2715 alumnos egresados para el 2023 para las Universidades privadas.

Sin embargo, aumenta la cantidad de alumnos inscriptos, pero la gran mayoría de ellos no culminan y no egresan.

Cantidad de alumnos matriculados en universidades según forma de administración. Total país

En las Universidades públicas:
En lo que respecta a los alumnos matriculados, para el año 2022 es de: 172496 y para el 2023 es de: 176825 de alumnos respectivamente. Hay una tendencia a la caída. Menos estudiantes culminando la Universidad.

En lo que respecta a las Universidades Privadas:

La predicción para el año 2022 es de: 31782 y para el 2023 es de: 32868 alumnos para las universidades privadas. Hay un crecimiento de alumnos matriculados (cursando) universidades privadas.

Prioridad macroeconómica del Gasto Público en Educación

Se pronostica un aumento en el gasto público con la educación pública, con respecto al PBI en los próximos años: la predicción para el año 2022 es de: 5.81% y para el 2023 es de:
5.95% Porcentaje sobre el PBI, respectivamente, (usando un Multi layer Linear Regression).

Tasa de desempleo según nivel educativo. Total país

En general, tiende a decaer, a pesar de sus variaciones, pero muy lentamente, para las personas que no tienen instrucción alguna o formación alguna en educación.
Tiende haber un crecimiento muy leve, cercano a una estabilidad, en el desempleo para aquellos que tienen solamente Primaria completa,.
En el futuro, tiende haber más desempleo para aquellos ciudadanos que tienen Secundaria completa, cercano al 16% de la población desempleada, para el 2022 y el 2023.
De forma similar, aquellos que tienen UTU completada tienden haber un crecimiento en el desempleo pero de una forma más moderada, quizás por tener un oficio específico, cercano al 10% de la población sin empleo.

Para los que tienen Magisterio o Profesorado, hubo un pequeño descenso en el desempleo (una mejora) y tiende a mantenerse.
Para los que tienen Universidad completa, afortunadamente hubo una fuerte caída en el desempleo. En un futuro, se mantiene a disminuir el desempleo. Se pronostica el 5,6% de la población universitario no tendrá trabajo.

Porcentaje de jóvenes que finalizaron primaria según sexo. Total país

Se nota que habrá un crecimiento en la cantidad de personas varones que culminen primaria. Se pronostica cerca del 98% para el 2022 y el 2023.
Para el sexo Mujer, habrá un crecimiento en la cantidad de personas que terminen primaria, cerca del 99% para el 2022 y el 2023. Las mujeres terminan más primaria que los hombres.

Porcentaje de jóvenes de 18 y más años que finalizaron secundaria (6º de liceo/UTU) según sexo. Total país

A partir del 2012 y para el futuro 2022 y 2023 tiende a mantenerse estable el porcentaje de egresados de educación media para el sexo varón, se pronostica el 35% de la población (46% con Peceptron).
Para el sexo Mujer, tiende a crecer lentamente las egresadas desde el 2012 y para el futuro 2022 y 2023, este crecimiento se acentúa, se pronostica el 48% de la población femenina (46% con Pereceptron)

Tasa de analfabetismo de las persona de 15 y más años según sexo. Total país

A partir del año 2010 y en adelante para el futuro 2022 y 2023, esta tasa de analfabetismo comienza a descender de una forma muy importante, para el sexo varón. Para el sexo mujer, esta caída es pronunciada y continua y para el futuro 2022 y 2023 tiende a continuar decayendo.
Las predicciones para el 2022:
0.92 % de la población analfabeta

Para el 2023:
0.85 % de la población será analfabeta.
(efectividad por encima del 95% sobre el conjunto de entrenamiento)
(efectividad por encima del 74% sobre el conjunto de test)
Para los varones:
Hagamos una predicción para los Varones:
La predicción para el año 2022 es de:
[1.33367556] (MLR) y [[2.2109563]]
y para el 2023 es de:
[1.25924025] (MLR) y [[2.2119613]]
de alumnos varones respectivamente

Hagamos una predicción para los Mujeres:
La predicción para el año 2022 es de:
[0.61681985]
y para el 2023 es de:
[0.54099265]
de alumnos mujeres respectivamente

Tasa de actividad según sexo por nivel educativo. Total país

Para las mujeres:
Para las mujeres sin instrucción, la tendencia es mantenerse al alza, con una taza del 16% para el 2022 y el 2023.
Para las que tienen Primaria completa, la tendencia es que decaigan un poco la tasa de actividad, se pronostica el 35%.
Para aquellas que tienen Secundaria completa, lamentablemente la caída se va a ver más pronunciada, se pronostica 53% para el 2022 y el 52% para el 2023.
Para las que tienen la UTU completa, la tendencia es caer, con una taza del 65%.
Para las que se formaron como maestras (Magisterio), lamentablemente hay una caída importante en los próximos años, 63% para el 2022 y el 58% para el 2023.
Para las que tienen formación universitaria, la tendencia es de crecimiento en los próximos años, con una taza del 79%.
Para los hombres:
Para los hombres sin instrucción, la tendencia es a la caída, se pronostica para el 2022 y el 2023, una taza del 28%.
Para los que tienen Primaria completa, la tendencia es que decaigan un poco la tasa de actividad, con un pronóstico del 62%.
Para aquellos que tienen Secundaria completa, lamentablemente la caída se va a ver más pronunciada, se pronostica entre el 69 y 70%.
Para los que tienen la UTU completa, la tendencia es caer lentamente, se pronostica el 81%.
Para los que se formaron como maestros (Magisterio), abrá un aumento en los próximos años, para el 2022 y el 2023, la taza será del 85% y el 86% respectivamente.
Para los que tienen formación universitaria, lamentablemente la tendencia es de decaída en los próximos años, del 77%

Tasa de desempleo según sexo por ascendencia afro. Total país

Para las mujeres no afrodescendientes, la tendencia es de mantenerse a caída, para un 4.88% para el 2022 y un 4.46% para el 2023 de pronóstico. Para las mujeres afrodescendientes, la tendencia es de caída en el desempleo para un 10.39% para el 2022 y un 9.99% para el 2023 de pronóstico.

Para los varones, no afrodescendientes, la tendencia es de caer un poco el desempleo, pronosticando un 5.05% y 5.06% para 2022 y 2023 respectivamente. Para aquellos afrodescendientes, la tendencia es de una crecida en el desempleo, pronosticando el 11.85% y el 12.36% para el 2022 y el 2023 respectivamente.

Porcentaje de personas en situación de pobreza según sexo por ascendencia afro. Total país

Para la situación de pobreza, para los varones no afrodescendientes, la situación de pobreza venía decayendo, pero para el futuro, se pronostica un aumento para el 17% y del 18%, 2022 y 2023, respectivamente. Para los varones afrodescendientes, la situación de pobreza venía decayendo, pero para el futuro, se pronostica un aumento para el 38.97% y el 38.99%, para el 2022 y el 2023 respectivamente.
Para las mujeres no afrodescendientes, la situación de pobreza venía decayendo, y para el futuro, se pronostica que continúe esta caída: el 8% de mujeres no afrodescendientes estarían en la situación de pobreza para el 2022 y el 2023, pero para las mujeres afrodescendientes, la situación de pobreza venía decayendo, pero para el futuro, se pronostica un aumento bastante significativo: el 46% de las mujeres afrodescendientes para el 2022 y el 2023 estarían bajo la situación de pobreza.

Tasa de empleo según ascendencia afro. Total país

Se pronostica para el 2022 y el 2023, un aumento en la tasa de empleo, para las personas no afrodescendientes: el 60% estarían con trabajo activo.
Lamentablemente, para el 2022 y el 2023, se pronostica una caída en la tasa de empleo para las personas afrodescendientes: el 57% estarían con trabajo activo.

Tasa de empleo por sexo y situación de pobreza

Para las mujeres pobres tiende a crecer la tasa, el 40% para el 2022 y el 2023
Para las mujeres no pobres, también tienden a decaer: el 44% para el 2022 y el 43% para el 2023.
Para los varones pobres, tiende a decaer la tasa bruscamente: el 53% para el 2022 y el 52.9% para el 2023
Para los varones no pobres, tiende a crecer: el 73% para el 2022 y el 2023

Tasa de desempleo según sexo por situación de pobreza. Total país

Para las mujeres pobres, tienden a crecer su tasa de desempleo.
Para las mujeres no pobres tienden también a mantenerse a una caída, para el 3.96% el 2022 y el 3.67% el 2023.
Para los hombres pobres, tienden a crecer su tasa de desempleo para el 21% y el 22%, 2022 y el 2023 respectivamente.
Para los hombres no pobres, tienden a crecer su tasa de desempleo, para el 7.0% y el 7.2%, para el 2022 y el 2023, respectivamente.

Conclusiones finales obtenidas con el entrenamiento de modelos con inteligencia artificial.

Como se puede apreciar, gran cantidad de alumnos se inscriben en la universidades públicas en Uruguay, pero pocos son los que egresan, una razón de 3,7 alumnos. Eso quiere decir, que de 4 alumnos que ingresan, solamente 1 egresa.
En el caso de las universidades privadas, la razón es de 2,10. Eso quiere decir, que de cada 2 alumnos que ingresan, 1 egresa.
También se puede apreciar, que decae a través del tiempo, la cantidad de matriculados, es decir, crece la deserción universitaria, en las universidades públicas. En las universidades privadas, crece el número de inscriptos, por lo que indica, según la relación entre ingresados y egresados, que estos alumnos van cursando materias por año.
A su vez, por lo que se puede ver, que el número de desertados crece, al mismo tiempo, el gobierno aumenta el gasto público en la Educación, aumentando a cada año.
Por otro lado, para las personas que tienen primaria y secundaria completa, crece el desempleo, sin embargo, parecería que aquellos que completaron UTU, por tener un oficio, tienen mayores posibilidades de conseguir trabajo, ya que éstos presentan una tasa de desempleo menor, en relación con los que culminaron Secundaria.
El panorama es mejor para los que se forman en la rama de la educación: aquellos que hicieron la carrera para profesor y maestro, tienen mejores posibilidades de trabajo, quizás debe ser por el motivo que una vez que culminan la carrera, ya entran en el sistema educativo en forma directa, sin esperar por llamados a concursos.
Para los ciudadanos que tienen formación universitaria, son los que tienen más posibilidades de trabajo, y es el sector de la sociedad que menos desempleo tienen. Esto se puede ver al analizar los conjuntos de datos de la taza de actividad, cuanto mayor instrucción, mayor serán los ocupados. Un dato curioso: entre los hombres con formación UTU y los universitarios, aquellos que aprendieron un oficio en UTU tendrán más empleo que los universitarios.
Las mujeres tienden a terminar más la escuela Primaria que los hombres. Ya el panorama para la educación Secundaria, es preocupante: se pronostica que solamente entre un 35% a 46% de la población joven del sexo masculino, termine el Liceo o colegios, habiendo una gran deserción y tiene a crecer al misma. Para el caso del sexo femenino, es igual a un poco más preocupante, el 48% de las mujeres jóvenes terminarían el liceo.
Afortunadamente, la tasa de analfabetismo en Uruguay se pronostica que caiga. Mujeres y hombres sabrán leer y escribir, en casi la totalidad de la población uruguaya, llegando a tan sólo 0.5% de la población analfabeta.

Desafortunadamente, la tasa de desempleo y la descendencia afro tienen una relación estrecha: las mujeres afrodescendientes son más desempleadas que las no afrodescendientes (10.4% contra el 4.46%). Para el caso de los varones, la tendencia es similar: la población afrodescendiente es mayor desempleada que la no afrodescendiente (12.36% contra el 5.05% respectivamente).
Esto se refleja también en la situación social de pobreza: la población varón no afrodescendiente en menos pobre que la afrodescendiente, entre el 17% y el 18% para los primeros contra el 38% de los segundos. Entre las mujeres esta diferencia es aún mayor y más preocupante: el 8% de las mujeres no afrodescendientes son pobres contra el 46% de las mujeres afrodescendientes serán pobres.
De forma similar, la población no afrodescendiente, se pronostica que su tasa de empleo crecerá y para la población afrodescendiente su tasa de empleo decrecerá.

Vemos también que las mujeres que no son pobres tienen una tasa de empleo mayor de aquellas que sí lo son. Es de apreciar que los varones que no son pobres tienen una tasa de empleo casi el doble de las mujeres no pobres. Los hombres tienen una cota mucho mayor dentro del mercado laboral. Hay una tendencia a decaer la tasa de desempleo para las mujeres y para los hombre tiende a crecer la tasa de desempleo, todo esto para el 2022 y 2023.

Anexo, Trabajo futuro

Como trabajo futuro, se espera que el Catálogo de Datos Abiertos disponga de datos correspondientes desde febrero y marzo del 2020 en delante. Así, en base a los datos del período que corresponde a la pandemia del COVID-19, se pueda analizarlos con precisión y se pueda predecir el futuro de la educación con su impacto debido a la pandemia. El trabajo de este proyecto se basa a los datos que se recabaron hasta enero del 2020.

Anexo, Códigos Fuentes.

Cantidad de ingresos y egresos en carreras universitarias de grado según forma de administración. Total país

Código fuente:

```
# -*- coding: utf-8 -*-
"""CantIngrEgrsUniver con Perceptron Keras.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/1AWKnmwtpNkNGo6UJAnHfgijgj0KYmNlk
"""

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

from keras.models import Sequential
from keras.layers.core import Dense

import tensorflow as tf

dataframe = pd.read_csv('/content/drive/MyDrive/PRIA Alfredo Poggio Manzor 24-08-2020 17:00
hrs/ProyectoFinal/DataSets/10447_cantidad_de_ingresos_y_egresos_en_carreras_universitarias_d
e_grado_segun_forma_de_administ.csv',  encoding='latin-1')

dataframe

selecciona1 = ['Egresos']
filtered_df = dataframe[dataframe['Promoción o Desvinculación'].isin(selecciona1)]
filtered_df

selecciona2 = ['Pública']
filtered_df2 = filtered_df[filtered_df['Tipo de administración'].isin(selecciona2)]

dataframeIII = filtered_df2["año"]

X = np.array(dataframeIII, dtype=float)

dataframeII = filtered_df2["valor"]
Y = np.array(dataframeII, dtype=float)

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,Y)

print(X_test)
print(y_test)
```

```
filtered_df2

capa1 = tf.keras.layers.Dense(units = 3, input_shape = [1])
capa2 = tf.keras.layers.Dense(units=6)
capa3 = tf.keras.layers.Dense(units=3)
salida = tf.keras.layers.Dense(units=1)
modelo = tf.keras.Sequential([capa1, capa2, capa3,salida])

print(modelo.summary())

modelo.compile(
    optimizer = tf.keras.optimizers.Adam(0.01),
    loss = 'mean_squared_error'
)

historial = modelo.fit(X, Y, epochs=60, verbose=False)
print("Modelo Entrenado")

import matplotlib.pyplot as plt
plt.xlabel('#Epoca')
plt.ylabel('Magnitud de pérdida')
plt.plot(historial.history['loss'])

print("Hagamos una Testeo")
resultado = modelo.predict(X_test)
print("Los valores calculados por la red neuronal son de: ")
print(str(resultado))
print("El valor esperado es de: ")
print(y_test)
print("Hagamos una predicción para el 2022: ")
resultado = modelo.predict([[2022]])
print(resultado)
print("Hagamos una predicción para el 2023: ")
resultado = modelo.predict([[2023]])
print(resultado)

selecciona1 = ['Ingresos']
filtered_df = dataframe[dataframe['Promoción o Desvinculación'].isin(selecciona1)]

selecciona2 = ['Pública']
filtered_df2 = filtered_df[filtered_df['Tipo de administración'].isin(selecciona2)]

dataframeIII = filtered_df2["año"]

X = np.array(dataframeIII, dtype=float)

dataframeII = filtered_df2["valor"]
Y = np.array(dataframeII, dtype=float)

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,Y)

print(X_test)
print(y_test)

filtered_df2

capa1 = tf.keras.layers.Dense(units = 3, input_shape = [1])
capa2 = tf.keras.layers.Dense(units=6)
```

```
capa3 = tf.keras.layers.Dense(units=3)
salida = tf.keras.layers.Dense(units=1)
modelo = tf.keras.Sequential([capa1, capa2, capa3,salida])

print(modelo.summary())

modelo.compile(
    optimizer = tf.keras.optimizers.Adam(0.01),
    loss = 'mean_squared_error'
)

historial = modelo.fit(X_train, y_train, epochs=60, verbose=False)
print("Modelo Entrenado")

import matplotlib.pyplot as plt
plt.xlabel('#Epoca')
plt.ylabel('Magnitud de pérdida')
plt.plot(historial.history['loss'])

print("Hagamos una Testeo")
resultado = modelo.predict(X_test)
print("Los valores calculados por la red neuronal son de: ")
print(str(resultado))
print("El valor esperado es de: ")
print(y_test)
print("Hagamos una predicción para el 2022: ")
resultado = modelo.predict([[2022]])
print(resultado)
print("Hagamos una predicción para el 2023: ")
resultado = modelo.predict([[2023]])
print(resultado)

"""# Ahora vamos analizar para Egresos de Universidades Privadas"""

selecciona1 = ['Egresos']
filtered_df = dataframe[dataframe['Promoción o Desvinculación'].isin(selecciona1)]

selecciona2 = ['Privada']
filtered_df2 = filtered_df[filtered_df['Tipo de administración'].isin(selecciona2)]

dataframeIII = filtered_df2["año"]

X = np.array(dataframeIII, dtype=float)

dataframeII = filtered_df2["valor"]
Y = np.array(dataframeII, dtype=float)

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,Y)

print(X_test)
print(y_test)

capa1 = tf.keras.layers.Dense(units = 3, input_shape = [1])
capa2 = tf.keras.layers.Dense(units=6)
capa3 = tf.keras.layers.Dense(units=3)
salida = tf.keras.layers.Dense(units=1)
modelo = tf.keras.Sequential([capa1, capa2, capa3,salida])

print(modelo.summary())
```

```
modelo.compile(
    optimizer = tf.keras.optimizers.Adam(0.01),
    loss = 'mean_squared_error'
)

historial = modelo.fit(X_train, y_train, epochs=60, verbose=False)
print("Modelo Entrenado")

import matplotlib.pyplot as plt
plt.xlabel('#Epoca')
plt.ylabel('Magnitud de pérdida')
plt.plot(historial.history['loss'])

print("Hagamos una Testeo")
resultado = modelo.predict(X_test)
print("Los valores calculados por la red neuronal son de: ")
print(str(resultado))
print("El valor esperado es de: ")
print(y_test)
print("Hagamos una predicción para el 2022: ")
resultado = modelo.predict([[2022]])
print(resultado)
print("Hagamos una predicción para el 2023: ")
resultado = modelo.predict([[2023]])
print(resultado)

selecciona1 = ['Ingresos']
filtered_df = dataframe[dataframe['Promoción o Desvinculación'].isin(selecciona1)]

selecciona2 = ['Privada']
filtered_df2 = filtered_df[filtered_df['Tipo de administración'].isin(selecciona2)]

dataframeIII = filtered_df2["año"]

X = np.array(dataframeIII, dtype=float)

dataframeII = filtered_df2["valor"]
Y = np.array(dataframeII, dtype=float)

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,Y)

print(X_test)
print(y_test)

capa1 = tf.keras.layers.Dense(units = 3, input_shape = [1])
capa2 = tf.keras.layers.Dense(units=6)
capa3 = tf.keras.layers.Dense(units=3)
salida = tf.keras.layers.Dense(units=1)
modelo = tf.keras.Sequential([capa1, capa2, capa3,salida])

print(modelo.summary())

modelo.compile(
    optimizer = tf.keras.optimizers.Adam(0.01),
    loss = 'mean_squared_error'
)

historial = modelo.fit(X_train, y_train, epochs=60, verbose=False)
```

```
print("Modelo Entrenado")

import matplotlib.pyplot as plt
plt.xlabel('#Epoca')
plt.ylabel('Magnitud de pérdida')
plt.plot(historial.history['loss'])

print("Hagamos una Testeo")
resultado = modelo.predict(X_test)
print("Los valores calculados por la red neuronal son de: ")
print(str(resultado))
print("El valor esperado es de: ")
print(y_test)
print("Hagamos una predicción para el 2022: ")
resultado = modelo.predict([[2022]])
print(resultado)
print("Hagamos una predicción para el 2023: ")
resultado = modelo.predict([[2023]])
print(resultado)

# -*- coding: utf-8 -*-
"""CantIngrEgrsUniver con Perceptron SKLearn.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/1zSzXMABnKgktQvA7-JmgQebzygattD_w
"""

import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import datetime

dataset = pd.read_csv('/content/drive/MyDrive/PRIA Alfredo Poggio Manzor 24-08-2020 17:00
hrs/ProyectoFinal/DataSets/10447_cantidad_de_ingresos_y_egresos_en_carreras_universitarias_d
e_grado_segun_forma_de_administ.csv',  encoding='latin-1')

dataset.head()

"""# Egresados de las Universidades Públicas."""

selecciona1 = ['Egresos']

selecciona2 = ['Pública']
filtered_df2 = dataset[dataset['Promoción o Desvinculación'].isin(selecciona1) &
dataset['Tipo de administración'].isin(selecciona2)]
filtered_df2

#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Cantidad alumnos egresados")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Egresados de la Universidad Pública'])

from sklearn import linear_model
```

```
rl = linear_model.LinearRegression()

año =  filtered_df2["año"]
print(filtered_df2)
#como el linear model del sklearn espera que se le pase una matriz , hay que hacer un
reshape:
X = año[:, np.newaxis]

y = filtered_df2["valor"]
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_text = train_test_split(X,y)

while True:
  X_train, X_test, y_train, y_text = train_test_split(X,y)

  rl.fit(X_train, y_train)
  print(rl.score(X_train, y_train))
  if rl.score(X_train, y_train) > 0.85:
    break
print(rl.score(X_test, y_text))

print(rl.predict([[2022]]))
print(rl.predict([[2023]]))

plt.figure(figsize=(16,8))
plt.scatter(X, y, color = "red")
plt.plot(X, rl.predict(X), color = "green", linewidth =3)
plt.title("Predicción del modelo según la evolución de los datos")
plt.xlabel('años')
plt.ylabel('valor')
plt.legend(["Predicción de los datos de los egresados de la Universidad Pública"])
plt.show()

"""# Ahora con Egresos con las Universidades Privadas."""

selecciona1 = ['Egresos']
selecciona2 = ['Privada']
filtered_df2 = dataset[dataset['Promoción o Desvinculación'].isin(selecciona1) &
dataset['Tipo de administración'].isin(selecciona2)]
filtered_df2

#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Cantidad alumnos egresados")
plt.legend(["Egresados de la Universidad Privada"])
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")

from sklearn import linear_model

rl = linear_model.LinearRegression()

año =  filtered_df2["año"]
print(filtered_df2)
#como el linear model del sklearn espera que se le pase una matriz 2x2, hay que hacer un
reshape:
X = año[:, np.newaxis]
```

```
y = filtered_df2["valor"]
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_text = train_test_split(X,y)

while True:
  X_train, X_test, y_train, y_text = train_test_split(X,y)

  rl.fit(X_train, y_train)
  print(rl.score(X_train, y_train))
  if rl.score(X_train, y_train) > 0.88:
    break


print(rl.predict([[2022]]))
print(rl.predict([[2023]]))

plt.figure(figsize=(16,8))
plt.scatter(X, y, color = "red")
plt.plot(X, rl.predict(X), color = "green", linewidth =3)
plt.title("Predicción del modelo según la evolución de los datos")
plt.xlabel('años')
plt.ylabel('valor')
plt.legend(["Predicción de los datos de los egresados de la Universidad Privada"])
plt.show()

"""# Ahora con ingresos con las Universidades Públicas."""

selecciona1 = ['Ingresos']
selecciona2 = ['Pública']
filtered_df2 = dataset[dataset['Promoción o Desvinculación'].isin(selecciona1) &
dataset['Tipo de administración'].isin(selecciona2)]
filtered_df2

#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Cantidad alumnos ingresados")
plt.legend(["Ingresados de la Universidad Pública"])
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")

from sklearn import linear_model

rl = linear_model.LinearRegression()

año =  filtered_df2["año"]
print(filtered_df2)
#como el linear model del sklearn espera que se le pase una matriz 2x2, hay que hacer un
reshape:
X = año[:, np.newaxis]

y = filtered_df2["valor"]
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_text = train_test_split(X,y)

while True:
```

```
  X_train, X_test, y_train, y_text = train_test_split(X,y)

  rl.fit(X_train, y_train)
  print(rl.score(X_train, y_train))
  if rl.score(X_train, y_train) > 0.88:
    break


print(rl.predict([[2022]]))
print(rl.predict([[2023]]))

plt.figure(figsize=(16,8))
plt.scatter(X, y, color = "red")
plt.plot(X, rl.predict(X), color = "green", linewidth =3)
plt.title("Predicción del modelo según la evolución de los datos")
plt.xlabel('años')
plt.ylabel('valor')
plt.legend(["Predicción de los datos de los ingresados de la Universidad Pública"])
plt.show()

"""# Ahora con ingresos con las Universidades Privadas."""

selecciona1 = ['Ingresos']
selecciona2 = ['Privada']
filtered_df2 = dataset[dataset['Promoción o Desvinculación'].isin(selecciona1) &
dataset['Tipo de administración'].isin(selecciona2)]
filtered_df2

#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Cantidad alumnos ingresados")
plt.legend(["Ingresados de la Universidad Privada"])
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")

from sklearn import linear_model

rl = linear_model.LinearRegression()

año =  filtered_df2["año"]
print(filtered_df2)
#como el linear model del sklearn espera que se le pase una matriz 2x2, hay que hacer un
reshape:
X = año[:, np.newaxis]

y = filtered_df2["valor"]
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_text = train_test_split(X,y)

while True:
  X_train, X_test, y_train, y_text = train_test_split(X,y)

  rl.fit(X_train, y_train)
  print(rl.score(X_train, y_train))
  if rl.score(X_train, y_train) > 0.88:
    break
```

```
plt.figure(figsize=(16,8))
plt.scatter(X, y, color = "red")
plt.plot(X, rl.predict(X), color = "green", linewidth =3)
plt.title("Predicción del modelo según la evolución de los datos")
plt.xlabel('años')
plt.ylabel('valor')
plt.legend(["Predicción de los datos de los ingresados de la Universidad Privada"])
plt.show()



print("La predicción para el año 2022 es de: ", rl.predict([[2022]]))
print("La predicción para el año 2023 es de: ", rl.predict([[2023]]))
```

Cantidad de alumnos matriculados en universidades según forma de administración. Total país

Código fuente:

```
# -*- coding: utf-8 -*-
"""CantMatricUniver con Perceptron SKLearn.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/1EQdlg9geHd3c2u0PnnfRqbHtBMmoMnOe
"""

import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import datetime

dataset = pd.read_csv('/content/drive/MyDrive/PRIA Alfredo Poggio Manzor 24-08-2020 17:00
hrs/ProyectoFinal/DataSets/11633_cantidad_de_alumnos_matriculados_en_universidades_segun_for
ma_de_administracion-_total_pai.csv',  encoding='latin-1')

dataset.head()

"""# Alumnos matriculados de las Universidades Públicas. Cabe destacar que un alumno
matriculado no necesariamente se lo considera como ingresado, ya que éste puede inscribirse
en la Universidad pero nunca llegar a cursarla."""

selecciona1 = ['Público']
filtered_df2 = dataset[dataset['Nombre'].isin(selecciona1)]
filtered_df2

#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Cantidad alumnos matriculados")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Matriculados de la Universidad Pública'])

from sklearn import linear_model
```

```
rl = linear_model.LinearRegression()

año =  filtered_df2["año"]

#como el linear model del sklearn espera que se le pase una matriz , hay que hacer un
reshape:
X = año[:, np.newaxis]

y = filtered_df2["valor"]
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_text = train_test_split(X,y)

while True:
  X_train, X_test, y_train, y_text = train_test_split(X,y)

  rl.fit(X_train, y_train)
  print(rl.score(X_train, y_train))
  if rl.score(X_train, y_train) > 0.85:
    break

print(rl.score(X_test, y_text))

print(rl.predict([[2022]]))
print(rl.predict([[2023]]))

plt.figure(figsize=(16,8))
plt.scatter(X, y, color = "red")
plt.plot(X, rl.predict(X), color = "green", linewidth =3)
plt.title("Predicción del modelo según la evolución de los datos")
plt.xlabel('años')
plt.ylabel('valor')
plt.legend(["Predicción de los datos de los matriculados de la Universidad Pública"])
plt.show()

"""# Alumnos matriculados de las Universidades Privadas. Cabe destacar que un alumno
matriculado no necesariamente se lo considera como ingresado, ya que éste puede inscribirse
en la Universidad pero nunca llegar a cursarla"""

selecciona2 = ['Privado']
filtered_df2 = dataset[dataset['Nombre'].isin(selecciona2)]
filtered_df2

#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Cantidad alumnos matriculados")
plt.legend(["Matriculados de la Universidad Privada"])
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")

from sklearn import linear_model

rl = linear_model.LinearRegression()

año =  filtered_df2["año"]
```

```
#como el linear model del sklearn espera que se le pase una matriz 2x2, hay que hacer un
reshape:
X = año[:, np.newaxis]

y = filtered_df2["valor"]
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_text = train_test_split(X,y)

while True:
  X_train, X_test, y_train, y_text = train_test_split(X,y)

  rl.fit(X_train, y_train)
  print(rl.score(X_train, y_train))
  if rl.score(X_train, y_train) > 0.88:
    break


print(rl.predict([[2022]]))
print(rl.predict([[2023]]))

plt.figure(figsize=(16,8))
plt.scatter(X, y, color = "red")
plt.plot(X, rl.predict(X), color = "green", linewidth =3)
plt.title("Predicción del modelo según la evolución de los datos")
plt.xlabel('años')
plt.ylabel('valor')
plt.legend(["Predicción de los datos de los Matriculados de la Universidad Privada"])
plt.show()

# -*- coding: utf-8 -*-
"""CantMatricUniver con Perceptron Keras.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/1A3tfNbgCLdBuN-YhU_2Ang-ZXKSWvRoo
"""

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

from keras.models import Sequential
from keras.layers.core import Dense

import tensorflow as tf

dataset = pd.read_csv('/content/drive/MyDrive/PRIA Alfredo Poggio Manzor 24-08-2020 17:00
hrs/ProyectoFinal/DataSets/11633_cantidad_de_alumnos_matriculados_en_universidades_segun_for
ma_de_administracion-_total_pai.csv',  encoding='latin-1')

dataset

"""# Alumnos matriculados de las Universidades Públicas. Cabe destacar que un alumno
matriculado no necesariamente se lo considera como ingresado, ya que éste puede inscribirse
en la Universidad pero nunca llegar a cursarla."""

#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
```

```
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Cantidad alumnos matriculados")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Matriculados de la Universidad Pública'])

selecciona1 = ['Público']
filtered_df2 = dataset[dataset['Nombre'].isin(selecciona1)]

dataframeIII = filtered_df2["año"]

X = np.array(dataframeIII, dtype=float)

dataframeII = filtered_df2["valor"]
Y = np.array(dataframeII, dtype=float)

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,Y)

print(X_test)
print(y_test)

capa1 = tf.keras.layers.Dense(units = 3, input_shape = [1])
capa2 = tf.keras.layers.Dense(units=6)
capa3 = tf.keras.layers.Dense(units=3)
salida = tf.keras.layers.Dense(units=1)
modelo = tf.keras.Sequential([capa1, capa2, capa3,salida])

print(modelo.summary())

modelo.compile(
    optimizer = tf.keras.optimizers.Adam(0.05),
    loss = 'mean_squared_error'
)

historial = modelo.fit(X_train, y_train, epochs=60, verbose=False)
print("Modelo Entrenado")

import matplotlib.pyplot as plt
plt.xlabel('#Epoca')
plt.ylabel('Magnitud de pérdida')
plt.plot(historial.history['loss'])

print("Hagamos una Testeo")
resultado = modelo.predict(X_test)
print("Los valores calculados por la red neuronal son de: ")
print(str(resultado))
print("El valor esperado es de: ")
print(y_test)
print("Hagamos una predicción para el 2022: ")
resultado = modelo.predict([[2022]])
print(resultado)
print("Hagamos una predicción para el 2023: ")
resultado = modelo.predict([[2023]])
print(resultado)

"""# Ahora con los matriculados en la Universidad Privada"""

selecciona1 = ['Privado']
```

```
filtered_df2 = dataset[dataset['Nombre'].isin(selecciona1)]

dataframeIII = filtered_df2["año"]

X = np.array(dataframeIII, dtype=float)

dataframeII = filtered_df2["valor"]
Y = np.array(dataframeII, dtype=float)

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,Y)

print(X_test)
print(y_test)

#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Cantidad alumnos matriculados")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Matriculados de la Universidad Privada'])

capa1 = tf.keras.layers.Dense(units = 3, input_shape = [1])
capa2 = tf.keras.layers.Dense(units=6)
capa3 = tf.keras.layers.Dense(units=3)
salida = tf.keras.layers.Dense(units=1)
modelo = tf.keras.Sequential([capa1, capa2, capa3,salida])

print(modelo.summary())

modelo.compile(
    optimizer = tf.keras.optimizers.Adam(0.05),
    loss = 'mean_squared_error'
)

historial = modelo.fit(X_train, y_train, epochs=60, verbose=False)
print("Modelo Entrenado")

import matplotlib.pyplot as plt
plt.xlabel('#Epoca')
plt.ylabel('Magnitud de pérdida')
plt.plot(historial.history['loss'])

print("Hagamos una Testeo")
resultado = modelo.predict(X_test)
print("Los valores calculados por la red neuronal son de: ")
print(str(resultado))
print("El valor esperado es de: ")
print(y_test)
print("Hagamos una predicción para el 2022: ")
resultado = modelo.predict([[2022]])
print(resultado)
print("Hagamos una predicción para el 2023: ")
resultado = modelo.predict([[2023]])
print(resultado)
```

Prioridad macroeconómica del Gasto Público en Educación

Código fuente:

```
# -*- coding: utf-8 -*-
"""Prioridad MacroEconómica en Educación con Perceptron SKLearn.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/1PYvL6x1eNBw7PPtk6bZhavUj8UaZSUvc
"""

import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import datetime

dataset = pd.read_csv('/content/drive/MyDrive/PRIA Alfredo Poggio Manzor 24-08-2020 17:00
hrs/ProyectoFinal/DataSets/12468_prioridad_macroeconomica_del_gasto_publico_en_educacion.csv
',  encoding='latin-1')

dataset

"""# Prioridad Macroeconómica del gasto público en educación. Esto representa el gasto del
PBI (Producto Bruto Interno, que es la suma de todas las riquezas producidas por el país)
por concepto de Educación Pública."""

#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(dataset["año"], dataset['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Producto Interno Bruto")
plt.scatter(dataset["año"], dataset['valor'], color = "red")
plt.legend(['Producto Interno Bruto, por concepto de educación'])

from sklearn import linear_model

rl = linear_model.LinearRegression()

año =  dataset["año"]

#como el linear model del sklearn espera que se le pase una matriz , hay que hacer un
reshape:
X = año[:, np.newaxis]

y = dataset["valor"]
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_text = train_test_split(X,y)

while True:
  X_train, X_test, y_train, y_text = train_test_split(X,y)

  rl.fit(X_train, y_train)
  print(rl.score(X_train, y_train))
  if rl.score(X_train, y_train) > 0.85:
    break

print(rl.score(X_test, y_text))
```

```
print(rl.predict([[2022]]))
print(rl.predict([[2023]]))

plt.figure(figsize=(16,8))
plt.scatter(X, y, color = "red")
plt.plot(X, rl.predict(X), color = "green", linewidth =3)
plt.title("Predicción del modelo según la evolución de los datos")
plt.xlabel('años')
plt.ylabel('valor')
plt.legend(["Predicción de los datos de los matriculados de la Universidad Pública"])
plt.show()


# -*- coding: utf-8 -*-
"""Prioridad MacroEconómica en Educación con Perceptron Keras.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/18dsUSLD5bp5GxvOmUHFW7QtVoPO0-7GH
"""

import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import datetime

from keras.models import Sequential
from keras.layers.core import Dense

import tensorflow as tf

dataset = pd.read_csv('/content/drive/MyDrive/PRIA Alfredo Poggio Manzor 24-08-2020 17:00
hrs/ProyectoFinal/DataSets/12468_prioridad_macroeconomica_del_gasto_publico_en_educacion.csv
',  encoding='latin-1')

dataset

"""# Prioridad Macroeconómica del gasto público en educación. Esto representa el gasto del
PBI (Producto Bruto Interno, que es la suma de todas las riquezas producidas por el país)
por concepto de Educación Pública."""

#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(dataset["año"], dataset['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Producto Interno Bruto")
plt.scatter(dataset["año"], dataset['valor'], color = "red")
plt.legend(['Producto Interno Bruto, por concepto de educación'])

dataframeIII = dataset["año"]

X = np.array(dataframeIII, dtype=float)

dataframeII = dataset["valor"]
Y = np.array(dataframeII, dtype=float)

from sklearn.model_selection import train_test_split
```

```
X_train, X_test, y_train, y_test = train_test_split(X,Y)

print(X_test)
print(y_test)

capa1 = tf.keras.layers.Dense(units = 2, input_shape = [1])
capa2 = tf.keras.layers.Dense(units=4)
capa3 = tf.keras.layers.Dense(units=2)
salida = tf.keras.layers.Dense(units=1)
modelo = tf.keras.Sequential([capa1, capa2, capa3,salida])

print(modelo.summary())

modelo.compile(
    optimizer = tf.keras.optimizers.Adam(0.01),
    loss = 'mean_squared_error'
)

historial = modelo.fit(X_train, y_train, epochs=60, verbose=False)
print("Modelo Entrenado")

import matplotlib.pyplot as plt
plt.xlabel('#Epoca')
plt.ylabel('Magnitud de pérdida')
plt.plot(historial.history['loss'])

print("Hagamos una Testeo")
resultado = modelo.predict(X_test)
print("Los valores calculados por la red neuronal son de: ")
print(str(resultado))
print("El valor esperado es de: ")
print(y_test)
print("Hagamos una predicción para el 2022: ")
resultado = modelo.predict([[2022]])
print(resultado)
print("Hagamos una predicción para el 2023: ")
resultado = modelo.predict([[2023]])
print(resultado)
```

Tasa de desempleo según nivel educativo. Total país

Código fuente:

```
# -*- coding: utf-8 -*-
"""Tasa de desempleo según nivel educativo, Perceptron Keras.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/1vTNbHnPhfiX_RkoyUhpFZAeLq9ShzdB0

# Proporción de personas que buscan empleo y no tienen, en relación a la población
económicamente activa, según nivel educativo.
"""

import numpy as np
```

```
import matplotlib.pyplot as plt
import pandas as pd

from keras.models import Sequential
from keras.layers.core import Dense

import tensorflow as tf

dataset = pd.read_csv('/content/drive/MyDrive/PRIA Alfredo Poggio Manzor 24-08-2020 17:00
hrs/ProyectoFinal/DataSets/10805_tasa_de_desempleo_segun_nivel_educativo-_total_pais.csv',
encoding='latin-1')

dataset

"""# Tasa de desempleo según según población sin instrucción"""

selecciona1 = ['Sin instrucción']

filtered_df2 = dataset[dataset['nivel_educa'].isin(selecciona1)]
filtered_df2
#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Tasa de desempleo según según población sin instrucción")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Tasa de desempleo según según población sin instrucción'])

"""# Preparo los conjuntos de entradas y salidas deseadas:"""

dataframeIII = filtered_df2["año"]

X = np.array(dataframeIII, dtype=float)

dataframeIV = filtered_df2["valor"]
Y = np.array(dataframeIV, dtype=float)

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,Y)

print(X_test)
print(y_test)

capa1 = tf.keras.layers.Dense(units = 2, input_shape = [1])
capa2 = tf.keras.layers.Dense(units=4)
capa3 = tf.keras.layers.Dense(units=2)
salida = tf.keras.layers.Dense(units=1)
modelo = tf.keras.Sequential([capa1, capa2, capa3,salida])

print(modelo.summary())

modelo.compile(
    optimizer = tf.keras.optimizers.Adam(0.01),
    loss = 'mean_squared_error'
)

historial = modelo.fit(X_train, y_train, epochs=60, verbose=False)
```

```
acc1 = modelo.evaluate(X_test, y_test)
print(acc1)
print(f"\nAccuracy is {acc1*100}")

print("Modelo Entrenado")

import matplotlib.pyplot as plt
plt.xlabel('#Epoca')
plt.ylabel('Magnitud de pérdida')
plt.plot(historial.history['loss'])

print("Hagamos una Testeo")
resultado = modelo.predict(X_test)
print("Los valores calculados por la red neuronal son de: ")
print(str(resultado))
print("El valor esperado es de: ")
print(y_test)
print("Hagamos una predicción para el 2022: ")
resultado = modelo.predict([[2022]])
print(resultado)
print("Hagamos una predicción para el 2023: ")
resultado = modelo.predict([[2023]])
print(resultado)

"""#  Tasa de desempleo según según población con Primaria completa"""

selecciona1 = ['Primaria']

filtered_df2 = dataset[dataset['nivel_educa'].isin(selecciona1)]
filtered_df2
#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Tasa de desempleo según según población con Primaria completa")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Tasa de desempleo según según población con Primaria completa'])

dataframeIII = filtered_df2["año"]

X = np.array(dataframeIII, dtype=float)

dataframeIV = filtered_df2["valor"]
Y = np.array(dataframeIV, dtype=float)

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,Y)

print(X_test)
print(y_test)

capa1 = tf.keras.layers.Dense(units = 2, input_shape = [1])
capa2 = tf.keras.layers.Dense(units=4)
capa3 = tf.keras.layers.Dense(units=2)
salida = tf.keras.layers.Dense(units=1)
modelo = tf.keras.Sequential([capa1, capa2, capa3,salida])

print(modelo.summary())
```

```
modelo.compile(
    optimizer = tf.keras.optimizers.Adam(0.01),
    loss = 'mean_squared_error'
)

historial = modelo.fit(X_train, y_train, epochs=60, verbose=False)
print("Modelo Entrenado")

import matplotlib.pyplot as plt
plt.xlabel('#Epoca')
plt.ylabel('Magnitud de pérdida')
plt.plot(historial.history['loss'])

print("Hagamos una Testeo")
resultado = modelo.predict(X_test)
print("Los valores calculados por la red neuronal son de: ")
print(str(resultado))
print("El valor esperado es de: ")
print(y_test)
print("Hagamos una predicción para el 2022: ")
resultado = modelo.predict([[2022]])
print(resultado)
print("Hagamos una predicción para el 2023: ")
resultado = modelo.predict([[2023]])
print(resultado)

"""#  Tasa de desempleo según según población con Secundaria completa"""

selecciona1 = ['Secundaria']

filtered_df2 = dataset[dataset['nivel_educa'].isin(selecciona1)]
filtered_df2
#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Tasa de desempleo según según población con Secundaria completa")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Tasa de desempleo según según población con Secundaria completa'])

dataframeIII = filtered_df2["año"]

X = np.array(dataframeIII, dtype=float)

dataframeIV = filtered_df2["valor"]
Y = np.array(dataframeIV, dtype=float)

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,Y)

print(X_test)
print(y_test)

capa1 = tf.keras.layers.Dense(units = 4, input_shape = [1])
capa2 = tf.keras.layers.Dense(units=8)
capa3 = tf.keras.layers.Dense(units=4)
salida = tf.keras.layers.Dense(units=1)
modelo = tf.keras.Sequential([capa1, capa2, capa3,salida])
```

```
print(modelo.summary())

modelo.compile(
    optimizer = tf.keras.optimizers.Adam(0.01),
    loss = 'mean_squared_error'
)

historial = modelo.fit(X_train, y_train, epochs=60, verbose=False)
print("Modelo Entrenado")

import matplotlib.pyplot as plt
plt.xlabel('#Epoca')
plt.ylabel('Magnitud de pérdida')
plt.plot(historial.history['loss'])

print("Hagamos una Testeo")
resultado = modelo.predict(X_test)
print("Los valores calculados por la red neuronal son de: ")
print(str(resultado))
print("El valor esperado es de: ")
print(y_test)
print("Hagamos una predicción para el 2022: ")
resultado = modelo.predict([[2022]])
print(resultado)
print("Hagamos una predicción para el 2023: ")
resultado = modelo.predict([[2023]])
print(resultado)

"""#  Tasa de desempleo según según población con UTU"""

selecciona1 = ['UTU']

filtered_df2 = dataset[dataset['nivel_educa'].isin(selecciona1)]
filtered_df2
#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Tasa de desempleo según según población con UTU")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Tasa de desempleo según según población con UTU'])

dataframeIII = filtered_df2["año"]

X = np.array(dataframeIII, dtype=float)

dataframeIV = filtered_df2["valor"]
Y = np.array(dataframeIV, dtype=float)

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,Y)

print(X_test)
print(y_test)

capa1 = tf.keras.layers.Dense(units = 4, input_shape = [1])
capa2 = tf.keras.layers.Dense(units=8)
capa3 = tf.keras.layers.Dense(units=4)
```

```
salida = tf.keras.layers.Dense(units=1)
modelo = tf.keras.Sequential([capa1, capa2, capa3,salida])

print(modelo.summary())

modelo.compile(
    optimizer = tf.keras.optimizers.Adam(0.01),
    loss = 'mean_squared_error'
)

historial = modelo.fit(X_train, y_train, epochs=60, verbose=False)
print("Modelo Entrenado")

import matplotlib.pyplot as plt
plt.xlabel('#Epoca')
plt.ylabel('Magnitud de pérdida')
plt.plot(historial.history['loss'])

print("Hagamos una Testeo")
resultado = modelo.predict(X_test)
print("Los valores calculados por la red neuronal son de: ")
print(str(resultado))
print("El valor esperado es de: ")
print(y_test)
print("Hagamos una predicción para el 2022: ")
resultado = modelo.predict([[2022]])
print(resultado)
print("Hagamos una predicción para el 2023: ")
resultado = modelo.predict([[2023]])
print(resultado)

"""#  Tasa de desempleo según según población con Magisterio o profesorado"""

selecciona1 = ['Magisterio o profesorado']

filtered_df2 = dataset[dataset['nivel_educa'].isin(selecciona1)]
filtered_df2
#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Tasa de desempleo según según población con Magisterio o profesorado")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Tasa de desempleo según según población con Magisterio o profesorado'])

dataframeIII = filtered_df2["año"]

X = np.array(dataframeIII, dtype=float)

dataframeIV = filtered_df2["valor"]
Y = np.array(dataframeIV, dtype=float)

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,Y)

print(X_test)
print(y_test)

capa1 = tf.keras.layers.Dense(units = 1, input_shape = [1], activation = 'sigmoid')
```

```
capa2 = tf.keras.layers.Dense(units=2, activation = 'sigmoid')
capa3 = tf.keras.layers.Dense(units=1, activation = 'sigmoid')
salida = tf.keras.layers.Dense(units=1)
modelo = tf.keras.Sequential([capa1, capa2, capa3,salida])

print(modelo.summary())

modelo.compile(
    optimizer = tf.keras.optimizers.Adam(0.01),
    loss = 'mean_squared_error'
)

historial = modelo.fit(X_train, y_train, epochs=500, verbose=False)
print("Modelo Entrenado")

import matplotlib.pyplot as plt
plt.xlabel('#Epoca')
plt.ylabel('Magnitud de pérdida')
plt.plot(historial.history['loss'])

print("Hagamos una Testeo")
resultado = modelo.predict(X_test)
print("Los valores calculados por la red neuronal son de: ")
print(str(resultado))
print("El valor esperado es de: ")
print(y_test)
print("Hagamos una predicción para el 2022: ")
resultado = modelo.predict([[2022]])
print(resultado)
print("Hagamos una predicción para el 2023: ")
resultado = modelo.predict([[2023]])
print(resultado)

"""#  Tasa de desempleo según según población con Universidad o similar"""

selecciona1 = ['Universidad o similar']

filtered_df2 = dataset[dataset['nivel_educa'].isin(selecciona1)]
filtered_df2
#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Tasa de desempleo según según población con Universidad o similar")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Tasa de desempleo según según población con Universidad o similar'])

dataframeIII = filtered_df2["año"]

X = np.array(dataframeIII, dtype=float)

dataframeIV = filtered_df2["valor"]
Y = np.array(dataframeIV, dtype=float)

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,Y)

print(X_test)
print(y_test)
```

```
capa1 = tf.keras.layers.Dense(units = 1, input_shape = [1], activation = 'sigmoid')
capa2 = tf.keras.layers.Dense(units=2, activation = 'sigmoid')
capa3 = tf.keras.layers.Dense(units=1, activation = 'sigmoid')
salida = tf.keras.layers.Dense(units=1)
modelo = tf.keras.Sequential([capa1, capa2, capa3,salida])

print(modelo.summary())

modelo.compile(
    optimizer = tf.keras.optimizers.Adam(0.01),
    loss = 'mean_squared_error'
)

historial = modelo.fit(X_train, y_train, epochs=500, verbose=False)
print("Modelo Entrenado")

import matplotlib.pyplot as plt
plt.xlabel('#Epoca')
plt.ylabel('Magnitud de pérdida')
plt.plot(historial.history['loss'])

print("Hagamos una Testeo")
resultado = modelo.predict(X_test)
print("Los valores calculados por la red neuronal son de: ")
print(str(resultado))
print("El valor esperado es de: ")
print(y_test)
print("Hagamos una predicción para el 2022: ")
resultado = modelo.predict([[2022]])
print(resultado)
print("Hagamos una predicción para el 2023: ")
resultado = modelo.predict([[2023]])
print(resultado)

# -*- coding: utf-8 -*-
"""Tasa de desempleo según nivel educativo, Perceptron SKLearn.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/1OzWTdtj4Qjq2-qPRoVTR_gHLguTtU4-r

# Proporción de personas que buscan empleo y no tienen, en relación a la población
económicamente activa, según nivel educativo..
"""

import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import datetime

dataset = pd.read_csv('/content/drive/MyDrive/PRIA Alfredo Poggio Manzor 24-08-2020 17:00
hrs/ProyectoFinal/DataSets/10805_tasa_de_desempleo_segun_nivel_educativo-_total_pais.csv',
encoding='latin-1')

from google.colab import drive
drive.mount('/content/drive')

dataset
```

```
"""# Tasa de desempleo según Sin instrucción

"""

selecciona1 = ['Sin instrucción']


filtered_df2 = dataset[dataset['nivel_educa'].isin(selecciona1)]
filtered_df2
#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Tasa de desempleo según nivel educativo Sin instrucción")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Tasa de desempleo según nivel educativo Sin instrucción'])

from sklearn import linear_model

rl = linear_model.LinearRegression()

año =  filtered_df2["año"]
print(filtered_df2)
#como el linear model del sklearn espera que se le pase una matriz , hay que hacer un
reshape:
X = año[:, np.newaxis]

y = filtered_df2["valor"]
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_text = train_test_split(X,y)

while True:
  X_train, X_test, y_train, y_text = train_test_split(X,y)

  rl.fit(X_train, y_train)
  print(rl.score(X_train, y_train))
  if rl.score(X_train, y_train) > 0.50:
    break


print(rl.predict([[2022]]))
print(rl.predict([[2023]]))

plt.figure(figsize=(16,8))
plt.scatter(X, y, color = "red")
plt.plot(X, rl.predict(X), color = "green", linewidth =3)
plt.title("Predicción del modelo según la evolución de los datos")
plt.xlabel('años')
plt.ylabel('valor')
plt.legend(["Distribuión porcentual de desempleo de Mujeres por ascedencia no afro"])
plt.show()

"""# Tasa de desempleo según sexo Primaria     """

selecciona1 = ['Primaria']
```

```
filtered_df2 = dataset[dataset['nivel_educa'].isin(selecciona1)]
filtered_df2
#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Distribuión porcentual de desempleo  según Primaria")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Distribuión porcentual de desempleo según Primaria'])

from sklearn import linear_model

rl = linear_model.LinearRegression()

año =  filtered_df2["año"]
print(filtered_df2)
#como el linear model del sklearn espera que se le pase una matriz , hay que hacer un
reshape:
X = año[:, np.newaxis]

y = filtered_df2["valor"]
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_text = train_test_split(X,y)

while True:
  X_train, X_test, y_train, y_text = train_test_split(X,y)

  rl.fit(X_train, y_train)
  print(rl.score(X_train, y_train))
  if rl.score(X_train, y_train) > 0.55:
    break


print(rl.predict([[2022]]))
print(rl.predict([[2023]]))

plt.figure(figsize=(16,8))
plt.scatter(X, y, color = "red")
plt.plot(X, rl.predict(X), color = "green", linewidth =3)
plt.title("Predicción del modelo según la evolución de los datos")
plt.xlabel('años')
plt.ylabel('valor')
plt.legend(["Distribuión porcentual de Mujeres, Tasa de desempleo según sexo Mujeres por
descendencia afro"])
plt.show()

"""#Tasa de desempleo según según población con Secundaria completa

"""

selecciona1 = ['Secundaria']


filtered_df2 = dataset[dataset['nivel_educa'].isin(selecciona1)]
filtered_df2
#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
```

```
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Tasa de desempleo según nivel educativo Secundaria")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Tasa de desempleo según nivel educativo Secundaria'])

from sklearn import linear_model

rl = linear_model.LinearRegression()

año =  filtered_df2["año"]
print(filtered_df2)
#como el linear model del sklearn espera que se le pase una matriz , hay que hacer un
reshape:
X = año[:, np.newaxis]

y = filtered_df2["valor"]
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_text = train_test_split(X,y)

while True:
  X_train, X_test, y_train, y_text = train_test_split(X,y)

  rl.fit(X_train, y_train)
  print(rl.score(X_train, y_train))
  if rl.score(X_train, y_train) > 0.65:
    break


print(rl.predict([[2022]]))
print(rl.predict([[2023]]))

"""#  Tasa de desempleo según  población con UTU completa"""

selecciona1 = ['UTU']

filtered_df2 = dataset[dataset['nivel_educa'].isin(selecciona1)]
filtered_df2
#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Tasa de desempleo según según población con UTU")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Tasa de desempleo según según población con UTU'])

from sklearn import linear_model

rl = linear_model.LinearRegression()

año =  filtered_df2["año"]
print(filtered_df2)
#como el linear model del sklearn espera que se le pase una matriz , hay que hacer un
reshape:
X = año[:, np.newaxis]

y = filtered_df2["valor"]
from sklearn.model_selection import train_test_split
```

```
X_train, X_test, y_train, y_text = train_test_split(X,y)

while True:
  X_train, X_test, y_train, y_text = train_test_split(X,y)

  rl.fit(X_train, y_train)
  print(rl.score(X_train, y_train))
  if rl.score(X_train, y_train) > 0.70:
    break


print(rl.predict([[2022]]))
print(rl.predict([[2023]]))

"""# Tasa de desempleo según según población con Magisterio o profesorado"""

selecciona1 = ['Magisterio o profesorado']

filtered_df2 = dataset[dataset['nivel_educa'].isin(selecciona1)]
filtered_df2
#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Tasa de desempleo según según población con Magisterio o profesorado")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Tasa de desempleo según según población con Magisterio o profesorado'])

from sklearn import linear_model

rl = linear_model.LinearRegression()

año =  filtered_df2["año"]
print(filtered_df2)
#como el linear model del sklearn espera que se le pase una matriz , hay que hacer un
reshape:
X = año[:, np.newaxis]

y = filtered_df2["valor"]
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_text = train_test_split(X,y)

while True:
  X_train, X_test, y_train, y_text = train_test_split(X,y)

  rl.fit(X_train, y_train)
  print(rl.score(X_train, y_train))
  if rl.score(X_train, y_train) > 0.50:
    break


print(rl.predict([[2022]]))
print(rl.predict([[2023]]))

"""# Tasa de desempleo según según población con Universidad o similar"""

selecciona1 = ['Universidad o similar']
```

```
filtered_df2 = dataset[dataset['nivel_educa'].isin(selecciona1)]
filtered_df2
#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Tasa de desempleo según según población con Universidad o similar")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Tasa de desempleo según según población con Universidad o similar'])

from sklearn import linear_model

rl = linear_model.LinearRegression()

año =  filtered_df2["año"]
print(filtered_df2)
#como el linear model del sklearn espera que se le pase una matriz , hay que hacer un
reshape:
X = año[:, np.newaxis]

y = filtered_df2["valor"]
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_text = train_test_split(X,y)

while True:
  X_train, X_test, y_train, y_text = train_test_split(X,y)

  rl.fit(X_train, y_train)
  print(rl.score(X_train, y_train))
  if rl.score(X_train, y_train) > 0.75:
    break


print(rl.predict([[2022]]))
print(rl.predict([[2023]]))
```

Porcentaje de jóvenes que finalizaron primaria según sexo. Total país

Código fuente:

```
# -*- coding: utf-8 -*-
"""PorcentEgrsPrimaria con Perceptron SKLearn.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/1WUMnXDulHc7QgLY7Z6-0vw66gNXdaKjf
"""

import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import datetime
```

```
dataset = pd.read_csv('/content/drive/MyDrive/PRIA Alfredo Poggio Manzor 24-08-2020 17:00
hrs/ProyectoFinal/DataSets/7829_porcentaje_de_jovenes_que_finalizaron_primaria_segun_sexo-
_total_pais.csv',  encoding='latin-1')

dataset.head()

"""# Para cada sexo se define como la cantidad de jóvenes que finalizaron primaria expresado
en porcentaje de la cantidad de jóvenes. Se considera a los jóvenes de 14 a 29 años."""

selecciona1 = ['Varones']

filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1)]
filtered_df2

#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Porcentaje de egresados jóvenes, entre 14 y 29 años, que han culminado la
primaria")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Porcentaje de egresados jóvenes, entre 14 y 29 años, que han culminado la
primaria'])

from sklearn import linear_model

rl = linear_model.LinearRegression()

año =  filtered_df2["año"]
print(filtered_df2)
#como el linear model del sklearn espera que se le pase una matriz , hay que hacer un
reshape:
X = año[:, np.newaxis]

y = filtered_df2["valor"]
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_text = train_test_split(X,y)

while True:
  X_train, X_test, y_train, y_text = train_test_split(X,y)

  rl.fit(X_train, y_train)
  print(rl.score(X_train, y_train))
  if rl.score(X_train, y_train) > 0.85:
    break

print(rl.score(X_test, y_text))

print(rl.predict([[2022]]))
print(rl.predict([[2023]]))

plt.figure(figsize=(16,8))
plt.scatter(X, y, color = "red")
plt.plot(X, rl.predict(X), color = "green", linewidth =3)
plt.title("Predicción del modelo según la evolución de los datos")
plt.xlabel('años')
plt.ylabel('valor')
```

```
plt.legend(["Predicción de los datos de los varones egresados de la Educación Pública
Primaria"])
plt.show()

"""# Ahora con las mujeres Egresadas de la Educación Pública Primaria."""

selecciona2 = ['Mujeres']
filtered_df2 = dataset[dataset['Sexo'].isin(selecciona2)]
filtered_df2

#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Cantidad alumnas egresadas de la Primaria")
plt.legend(["Egresadas de la Eduación Pública Primaria"])
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")

from sklearn import linear_model

rl = linear_model.LinearRegression()

año =  filtered_df2["año"]
print(filtered_df2)
#como el linear model del sklearn espera que se le pase una matriz 2x2, hay que hacer un
reshape:
X = año[:, np.newaxis]

y = filtered_df2["valor"]
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_text = train_test_split(X,y)

while True:
  X_train, X_test, y_train, y_text = train_test_split(X,y)

  rl.fit(X_train, y_train)
  print(rl.score(X_train, y_train))
  if rl.score(X_train, y_train) > 0.88:
    break


print(rl.predict([[2022]]))
print(rl.predict([[2023]]))

plt.figure(figsize=(16,8))
plt.scatter(X, y, color = "red")
plt.plot(X, rl.predict(X), color = "green", linewidth =3)
plt.title("Predicción del modelo según la evolución de los datos")
plt.xlabel('años')
plt.ylabel('valor')
plt.legend(["Predicción de los datos de las alumnas egresadas de la Educación Pública"])
plt.show()

# -*- coding: utf-8 -*-
"""PorcentEgrsPrimaria con Perceptron Keras.ipynb

Automatically generated by Colaboratory.
```

```
Original file is located at
    https://colab.research.google.com/drive/1HbEnUaBzehrP_D6806dcTq3g9l1vfp9g
"""

import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

from keras.models import Sequential
from keras.layers.core import Dense

import tensorflow as tf

dataset = pd.read_csv('/content/drive/MyDrive/PRIA Alfredo Poggio Manzor 24-08-2020 17:00
hrs/ProyectoFinal/DataSets/7829_porcentaje_de_jovenes_que_finalizaron_primaria_segun_sexo-
_total_pais.csv',  encoding='latin-1')

dataset.head()

"""# Para cada sexo se define como la cantidad de jóvenes que finalizaron primaria expresado
en porcentaje de la cantidad de jóvenes. Se considera a los jóvenes de 14 a 29 años."""

selecciona1 = ['Varones']

filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1)]
filtered_df2

#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Porcentaje de egresados jóvenes, entre 14 y 29 años, que han culminado la
primaria")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Porcentaje de egresados jóvenes, entre 14 y 29 años, que han culminado la
primaria'])

"""# Preparo los conjuntos de entradas y salidas deseadas:"""

dataframeIII = filtered_df2["año"]

X = np.array(dataframeIII, dtype=float)

dataframeIV = filtered_df2["valor"]
Y = np.array(dataframeIV, dtype=float)

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,Y)

print(X_test)
print(y_test)

capa1 = tf.keras.layers.Dense(units = 2, input_shape = [1])
capa2 = tf.keras.layers.Dense(units=4)
capa3 = tf.keras.layers.Dense(units=2)
salida = tf.keras.layers.Dense(units=1)
modelo = tf.keras.Sequential([capa1, capa2, capa3,salida])

print(modelo.summary())
```

```
modelo.compile(
    optimizer = tf.keras.optimizers.Adam(0.01),
    loss = 'mean_squared_error'
)

historial = modelo.fit(X_train, y_train, epochs=60, verbose=False)
print("Modelo Entrenado")

import matplotlib.pyplot as plt
plt.xlabel('#Epoca')
plt.ylabel('Magnitud de pérdida')
plt.plot(historial.history['loss'])

print("Hagamos una Testeo")
resultado = modelo.predict(X_test)
print("Los valores calculados por la red neuronal son de: ")
print(str(resultado))
print("El valor esperado es de: ")
print(y_test)
print("Hagamos una predicción para el 2022: ")
resultado = modelo.predict([[2022]])
print(resultado)
print("Hagamos una predicción para el 2023: ")
resultado = modelo.predict([[2023]])
print(resultado)

"""# Ahora con las mujeres Egresadas de la Educación Pública Primaria."""

selecciona2 = ['Mujeres']
filtered_df2 = dataset[dataset['Sexo'].isin(selecciona2)]
filtered_df2

#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Cantidad alumnas egresadas de la Primaria")
plt.legend(["Egresadas de la Eduación Pública Primaria"])
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")

dataframeIII = filtered_df2["año"]

X = np.array(dataframeIII, dtype=float)

dataframeIV = filtered_df2["valor"]
Y = np.array(dataframeIV, dtype=float)

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,Y)

print(X_test)
print(y_test)

capa1 = tf.keras.layers.Dense(units = 2, input_shape = [1])
capa2 = tf.keras.layers.Dense(units=4)
capa3 = tf.keras.layers.Dense(units=2)
salida = tf.keras.layers.Dense(units=1)
modelo = tf.keras.Sequential([capa1, capa2, capa3,salida])
```

```
print(modelo.summary())

modelo.compile(
    optimizer = tf.keras.optimizers.Adam(0.01),
    loss = 'mean_squared_error'
)

historial = modelo.fit(X_train, y_train, epochs=60, verbose=False)
print("Modelo Entrenado")

import matplotlib.pyplot as plt
plt.xlabel('#Epoca')
plt.ylabel('Magnitud de pérdida')
plt.plot(historial.history['loss'])

print("Hagamos una Testeo")
resultado = modelo.predict(X_test)
print("Los valores calculados por la red neuronal son de: ")
print(str(resultado))
print("El valor esperado es de: ")
print(y_test)
print("Hagamos una predicción para el 2022: ")
resultado = modelo.predict([[2022]])
print(resultado)
print("Hagamos una predicción para el 2023: ")
resultado = modelo.predict([[2023]])
print(resultado)
```

Porcentaje de jóvenes de 18 y más años que finalizaron secundaria (6º de liceo/UTU) según sexo. Total país

Código fuente:

```
# -*- coding: utf-8 -*-
"""CantEgrsEdMedia según sexo, con Perceptron SKLearn.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/1rBopAmJViz2wnegHA-o3Nmx92CibdkVE
"""

import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import datetime

from google.colab import drive
drive.mount('/content/drive')

dataset = pd.read_csv('/content/drive/MyDrive/PRIA Alfredo Poggio Manzor 24-08-2020 17:00
hrs/ProyectoFinal/DataSets/7866_porcentaje_de_jovenes_de_18_y_mas_aos_que_finalizaron_secund
aria_-6_de_liceo-utu-_segun_sex.csv',  encoding='latin-1')

dataset.head()
```

```
"""# Porcentaje de jóvenes de 18 y más años que finalizaron secundaria (6° de liceo/UTU)
según sexo. Total país

#Para cada sexo se define como la cantidad de personas jóvenes de 18 y más años que
finalizaron secundaria (6º de liceo/UTU), expresado en porcentaje del número de jóvenes de
18 y más años.
"""

selecciona1 = ['Varones']

filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1)]
filtered_df2

#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Porcentaje de egresados jóvenes, varones de 18 y más años, que han culminado la
secundaria")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Porcentaje de egresados jóvenes, varones de 18 y más años, que han culminado la
secundaria'])

from sklearn import linear_model


rl = linear_model.LinearRegression()

año =  filtered_df2["año"]
print(filtered_df2)
#como el linear model del sklearn espera que se le pase una matriz , hay que hacer un
reshape:
X = año[:, np.newaxis]

y = filtered_df2["valor"]
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_text = train_test_split(X,y)

while True:
  X_train, X_test, y_train, y_text = train_test_split(X,y)
  rl.fit(X_train, y_train)
  print(rl.score(X_train, y_train))
  print("La accuracy del modelo es de:", rl.score(X_test, y_text))
  if rl.score(X_train, y_train) > 0.85:
    break

print("La accuracy del modelo es de:", rl.score(X_test, y_text))

print(rl.predict([[2022]]))
print(rl.predict([[2023]]))

plt.figure(figsize=(16,8))
plt.scatter(X, y, color = "red")
plt.plot(X, rl.predict(X), color = "green", linewidth =3)
plt.title("Predicción del modelo según la evolución de los datos")
plt.xlabel('años')
plt.ylabel('valor')
```

```
plt.legend(["Predicción de los datos de los varones egresados de la Educación Pública
Secundaria"])
plt.show()

"""# Ahora con Egresos con las Mujeres en la educación secundaria."""

selecciona2 = ['Mujeres']
filtered_df2 = dataset[dataset['Sexo'].isin(selecciona2)]
filtered_df2

#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Cantidad alumnas egresadas de la Secundaria")
plt.legend(["Egresadas de la Eduación Pública Secundaria"])
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")

from sklearn import linear_model

rl = linear_model.LinearRegression()

año =  filtered_df2["año"]
print(filtered_df2)
#como el linear model del sklearn espera que se le pase una matriz 2x2, hay que hacer un
reshape:
X = año[:, np.newaxis]

y = filtered_df2["valor"]
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_text = train_test_split(X,y)

while True:
  X_train, X_test, y_train, y_text = train_test_split(X,y)

  rl.fit(X_train, y_train)
  print(rl.score(X_train, y_train))
  if rl.score(X_train, y_train) > 0.78:
    break


print(rl.predict([[2022]]))
print(rl.predict([[2023]]))

plt.figure(figsize=(16,8))
plt.scatter(X, y, color = "red")
plt.plot(X, rl.predict(X), color = "green", linewidth =3)
plt.title("Predicción del modelo según la evolución de los datos")
plt.xlabel('años')
plt.ylabel('valor')
plt.legend(["Predicción de los datos de las alumnas egresadas de la Educación Pública en
Secundaria"])
plt.show()

# -*- coding: utf-8 -*-
"""CantEgrsEdMedia según sexo, con Perceptron Keras.ipynb

Automatically generated by Colaboratory.
```

```
Original file is located at
    https://colab.research.google.com/drive/1i0Y7m-rkX-OVkaZgLERB2UucvShsvKjB
"""

import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

from keras.models import Sequential
from keras.layers.core import Dense

import tensorflow as tf

dataset = pd.read_csv('/content/drive/MyDrive/PRIA Alfredo Poggio Manzor 24-08-2020 17:00
hrs/ProyectoFinal/DataSets/7866_porcentaje_de_jovenes_de_18_y_mas_aos_que_finalizaron_secund
aria_-6_de_liceo-utu-_segun_sex.csv',  encoding='latin-1')

dataset.head()

"""# Para cada sexo se define como la cantidad de jóvenes que finalizaron primaria expresado
en porcentaje de la cantidad de jóvenes. Se considera a los jóvenes de 14 a 29 años."""

selecciona1 = ['Varones']

filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1)]
filtered_df2

#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Porcentaje de egresados jóvenes, entre 14 y 29 años, que han culminado la
primaria")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Porcentaje de egresados jóvenes, entre 14 y 29 años, que han culminado la
primaria'])

"""# Preparo los conjuntos de entradas y salidas deseadas:"""

dataframeIII = filtered_df2["año"]

X = np.array(dataframeIII, dtype=float)

dataframeIV = filtered_df2["valor"]
Y = np.array(dataframeIV, dtype=float)

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,Y)

print(X_test)
print(y_test)

capa1 = tf.keras.layers.Dense(units = 2, input_shape = [1])
capa2 = tf.keras.layers.Dense(units=4)
capa3 = tf.keras.layers.Dense(units=2)
salida = tf.keras.layers.Dense(units=1)
modelo = tf.keras.Sequential([capa1, capa2, capa3,salida])
```

```
print(modelo.summary())

modelo.compile(
    optimizer = tf.keras.optimizers.Adam(0.01),
    loss = 'mean_squared_error'
)

historial = modelo.fit(X_train, y_train, epochs=60, verbose=False)
print("Modelo Entrenado")

import matplotlib.pyplot as plt
plt.xlabel('#Epoca')
plt.ylabel('Magnitud de pérdida')
plt.plot(historial.history['loss'])

print("Hagamos una Testeo")
resultado = modelo.predict(X_test)
print("Los valores calculados por la red neuronal son de: ")
print(str(resultado))
print("El valor esperado es de: ")
print(y_test)
print("Hagamos una predicción para el 2022: ")
resultado = modelo.predict([[2022]])
print(resultado)
print("Hagamos una predicción para el 2023: ")
resultado = modelo.predict([[2023]])
print(resultado)

"""# Ahora con las mujeres Egresadas de la Educación Pública Primaria."""

selecciona2 = ['Mujeres']
filtered_df2 = dataset[dataset['Sexo'].isin(selecciona2)]
filtered_df2

#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Cantidad alumnas egresadas de la Primaria")
plt.legend(["Egresadas de la Eduación Pública Primaria"])
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")

dataframeIII = filtered_df2["año"]

X = np.array(dataframeIII, dtype=float)

dataframeIV = filtered_df2["valor"]
Y = np.array(dataframeIV, dtype=float)

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,Y)

print(X_test)
print(y_test)

capa1 = tf.keras.layers.Dense(units = 2, input_shape = [1])
capa2 = tf.keras.layers.Dense(units=4)
capa3 = tf.keras.layers.Dense(units=2)
salida = tf.keras.layers.Dense(units=1)
```

```
modelo = tf.keras.Sequential([capa1, capa2, capa3,salida])

print(modelo.summary())

modelo.compile(
    optimizer = tf.keras.optimizers.Adam(0.01),
    loss = 'mean_squared_error'
)

historial = modelo.fit(X_train, y_train, epochs=60, verbose=False)
print("Modelo Entrenado")

import matplotlib.pyplot as plt
plt.xlabel('#Epoca')
plt.ylabel('Magnitud de pérdida')
plt.plot(historial.history['loss'])

print("Hagamos una Testeo")
resultado = modelo.predict(X_test)
print("Los valores calculados por la red neuronal son de: ")
print(str(resultado))
print("El valor esperado es de: ")
print(y_test)
print("Hagamos una predicción para el 2022: ")
resultado = modelo.predict([[2022]])
print(resultado)
print("Hagamos una predicción para el 2023: ")
resultado = modelo.predict([[2023]])
print(resultado)
```

Tasa de analfabetismo de las persona de 15 y más años según sexo. Total país

Código fuente:

```
# -*- coding: utf-8 -*-
"""Tasa de analfabetismo de las persona de 15 y más años según sexo, Perceptron
SKLearn.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/1qsG-FnoETWrhFwcHbxabDCC0WdYrqUvm

# Porcentaje de personas de 15 y más años que no saben leer ni escribir, según sexo.
"""

import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import datetime

dataset = pd.read_csv('/content/drive/MyDrive/PRIA Alfredo Poggio Manzor 24-08-2020 17:00
hrs/ProyectoFinal/DataSets/10394_tasa_de_analfabetismo_de_las_persona_de_15_y_mas_aos_segun_
sexo-_total_pais.csv',  encoding='latin-1')

dataset
```

```
"""# Porcentaje de personas varones de 15 y más años que no saben leer ni escribir"""

selecciona1 = ['Varones']
filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1)]
filtered_df2
#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Tasa de analfabetismo de las personas varones de 15 y más años según sexo,")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Tasa de analfabetismo de las personas varones de 15 y más años según sexo,'])

from sklearn import linear_model, metrics

rl = linear_model.LinearRegression()

año =  filtered_df2["año"]
print(filtered_df2)
#como el linear model del sklearn espera que se le pase una matriz , hay que hacer un
reshape:
X = año[:, np.newaxis]

y = filtered_df2["valor"]
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_text = train_test_split(X,y)

while True:
  X_train, X_test, y_train, y_text = train_test_split(X,y)

  rl.fit(X_train, y_train)
  print(rl.score(X_train, y_train))
  if rl.score(X_train, y_train) > 0.90:
    break

print("Score of Test set", rl.score(X_test, y_text))

print(rl.predict([[2022]]))
print(rl.predict([[2023]]))

plt.figure(figsize=(16,8))
plt.scatter(X, y, color = "red")
plt.plot(X, rl.predict(X), color = "green", linewidth =3)
plt.title("Predicción del modelo según la evolución de los datos")
plt.xlabel('años')
plt.ylabel('valor')
plt.legend(["Tasa de analfabetismo de las personas varones de 15 y más años según sexo,"])
plt.show()

"""# Porcentaje de personas mujeres de 15 y más años que no saben leer ni escribir"""

selecciona1 = ['Mujeres']
filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1)]
filtered_df2

#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
```

```
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Porcentaje de personas mujeres de 15 y más años que no saben leer ni escribir")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Porcentaje de personas mujeres de 15 y más años que no saben leer ni
escribir'])

from sklearn import linear_model

rl = linear_model.LinearRegression()

año =  filtered_df2["año"]
print(filtered_df2)
#como el linear model del sklearn espera que se le pase una matriz , hay que hacer un
reshape:
X = año[:, np.newaxis]

y = filtered_df2["valor"]
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_text = train_test_split(X,y)

while True:
  X_train, X_test, y_train, y_text = train_test_split(X,y)

  rl.fit(X_train, y_train)
  print(rl.score(X_train, y_train))
  if rl.score(X_train, y_train) > 0.95:
    break

print("Score of Test set", rl.score(X_test, y_text))

print(rl.predict([[2022]]))
print(rl.predict([[2023]]))

plt.figure(figsize=(16,8))
plt.scatter(X, y, color = "red")
plt.plot(X, rl.predict(X), color = "green", linewidth =3)
plt.title("Predicción del modelo según la evolución de los datos")
plt.xlabel('años')
plt.ylabel('valor')
plt.legend(["Porcentaje de personas mujeres de 15 y más años que no saben leer ni
escribir"])
plt.show()

# -*- coding: utf-8 -*-
"""Tasa de analfabetismo de las persona de 15 y más años según sexo, Perceptron  Keras.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/1R5GCzbeajg35rJRhyy5Ccg2FzBTbzhUP

# Porcentaje de personas de 15 y más años que no saben leer ni escribir, según sexo.
"""

import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

```
from keras.models import Sequential
from keras.layers.core import Dense

import tensorflow as tf

dataset = pd.read_csv('/content/drive/MyDrive/PRIA Alfredo Poggio Manzor 24-08-2020 17:00
hrs/ProyectoFinal/DataSets/10394_tasa_de_analfabetismo_de_las_persona_de_15_y_mas_aos_segun_
sexo-_total_pais.csv',  encoding='latin-1')

dataset

"""# Porcentaje de personas varones de 15 y más años que no saben leer ni escribir"""

selecciona1 = ['Varones']
filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1)]
filtered_df2
#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Porcentaje de personas Varones de 15 y más años que no saben leer ni escribir")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Porcentaje de personas varones de 15 y más años que no saben leer ni
escribir'])

"""# Preparo los conjuntos de entradas y salidas deseadas:"""

dataframeIII = filtered_df2["año"]

X = np.array(dataframeIII, dtype=float)

dataframeIV = filtered_df2["valor"]
Y = np.array(dataframeIV, dtype=float)

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,Y)

print(X_test)
print(y_test)

capa1 = tf.keras.layers.Dense(units = 1, input_shape = [1])
capa2 = tf.keras.layers.Dense(units=2)
capa3 = tf.keras.layers.Dense(units=1)
salida = tf.keras.layers.Dense(units=1)
modelo = tf.keras.Sequential([capa1, capa2, capa3,salida])

print(modelo.summary())

modelo.compile(
    optimizer = tf.keras.optimizers.Adam(0.001),
    loss = 'mean_squared_error'
)

historial = modelo.fit(X_train, y_train, epochs=600, verbose=False)

acc1 = modelo.evaluate(X_test, y_test)
print(acc1)
print(f"\nAccuracy is {acc1*100}")
```

```
print("Modelo Entrenado")

import matplotlib.pyplot as plt
plt.xlabel('#Epoca')
plt.ylabel('Magnitud de pérdida')
plt.plot(historial.history['loss'])

print("Hagamos una Testeo")
resultado = modelo.predict(X_test)
print("Los valores calculados por la red neuronal son de: ")
print(str(resultado))
print("El valor esperado es de: ")
print(y_test)
print("Hagamos una predicción para el 2022: ")
resultado = modelo.predict([[2022]])
print(resultado)
print("Hagamos una predicción para el 2023: ")
resultado = modelo.predict([[2023]])
print(resultado)

"""# Porcentaje de personas mujeres de 15 y más años que no saben leer ni escribir"""

selecciona1 = ['Mujeres']
filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1)]
filtered_df2
#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Porcentaje de personas mujeres de 15 y más años que no saben leer ni escribir")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Porcentaje de personas mujeres de 15 y más años que no saben leer ni
escribir'])

dataframeIII = filtered_df2["año"]

X = np.array(dataframeIII, dtype=float)

dataframeIV = filtered_df2["valor"]
Y = np.array(dataframeIV, dtype=float)

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,Y)

print(X_test)
print(y_test)

capa1 = tf.keras.layers.Dense(units = 1, input_shape = [1])
capa2 = tf.keras.layers.Dense(units=2)
capa3 = tf.keras.layers.Dense(units=1)
salida = tf.keras.layers.Dense(units=1)
modelo = tf.keras.Sequential([capa1, capa2, capa3,salida])

print(modelo.summary())

modelo.compile(
    optimizer = tf.keras.optimizers.Adam(0.05),
    loss = 'mean_squared_error'
)
```

```
historial = modelo.fit(X_train, y_train, epochs=60, verbose=False)
print("Modelo Entrenado")

import matplotlib.pyplot as plt
plt.xlabel('#Epoca')
plt.ylabel('Magnitud de pérdida')
plt.plot(historial.history['loss'])

print("Hagamos una Testeo")
resultado = modelo.predict(X_test)
print("Los valores calculados por la red neuronal son de: ")
print(str(resultado))
print("El valor esperado es de: ")
print(y_test)
print("Hagamos una predicción para el 2022: ")
resultado = modelo.predict([[2022]])
print(resultado)
```

Tasa de actividad según sexo por nivel educativo. Total país

Código fuente:

```
# -*- coding: utf-8 -*-
"""Tasa de actividad según sexo Masculino por nivel educativo, Perceptron SKLearn.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/14bMxi8jmXIvFHH687BAD609TWphKRyLB

# Proporción de varones que se encuentran activos (trabajan o buscan trabajo) entre aquellos
de 14 y más años, según nivel educativo. Nos permite medir el grado de participación de los
varones en el mercado de trabajo con dichas características.
"""

import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import datetime

dataset = pd.read_csv('/content/drive/MyDrive/PRIA Alfredo Poggio Manzor 24-08-2020 17:00
hrs/ProyectoFinal/DataSets/12559_tasa_de_actividad_segun_sexo_por_nivel_educativo-
_total_pais.csv',  encoding='latin-1')

dataset

"""# Tasa de actividad según sexo Varones por nivel educativo Sin instrucción"""

selecciona1 = ['Varones']
selecciona2 = ['Sin instrucción']
filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1) &
dataset['nivel_educa'].isin(selecciona2)]
filtered_df2
#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
```

```
plt.xlabel("año")
plt.ylabel("Tasa de actividad según sexo Varones por nivel educativo Sin instrucción")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Tasa de actividad según sexo Varones por nivel educativo Sin instrucción'])

from sklearn import linear_model

rl = linear_model.LinearRegression()

año =  filtered_df2["año"]
print(filtered_df2)
#como el linear model del sklearn espera que se le pase una matriz , hay que hacer un
reshape:
X = año[:, np.newaxis]

y = filtered_df2["valor"]
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_text = train_test_split(X,y)

while True:
  X_train, X_test, y_train, y_text = train_test_split(X,y)

  rl.fit(X_train, y_train)
  print(rl.score(X_train, y_train))
  if rl.score(X_train, y_train) > 0.20:
    break

print(rl.score(X_test, y_text))

print(rl.predict([[2022]]))
print(rl.predict([[2023]]))

plt.figure(figsize=(16,8))
plt.scatter(X, y, color = "red")
plt.plot(X, rl.predict(X), color = "green", linewidth =3)
plt.title("Predicción del modelo según la evolución de los datos")
plt.xlabel('años')
plt.ylabel('valor')
plt.legend(["Distribuión porcentual de varones por nivel educativo Sin instrucción"])
plt.show()

"""# Tasa de actividad según sexo Varones por nivel educativo Primaria completa"""

selecciona1 = ['Varones']
selecciona2 = ['Primaria']
filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1) &
dataset['nivel_educa'].isin(selecciona2)]
filtered_df2

#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Distribuión porcentual de varones, Tasa de actividad según sexo Varones por
nivel educativo Primaria completa")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Distribuión porcentual de varones, Tasa de actividad según sexo Varones por
nivel educativo Primaria completa'])
```

```
from sklearn import linear_model

rl = linear_model.LinearRegression()

año =  filtered_df2["año"]
print(filtered_df2)
#como el linear model del sklearn espera que se le pase una matriz , hay que hacer un
reshape:
X = año[:, np.newaxis]

y = filtered_df2["valor"]
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_text = train_test_split(X,y)

while True:
  X_train, X_test, y_train, y_text = train_test_split(X,y)

  rl.fit(X_train, y_train)
  print(rl.score(X_train, y_train))
  if rl.score(X_train, y_train) > 0.85:
    break

print(rl.score(X_test, y_text))

print(rl.predict([[2022]]))
print(rl.predict([[2023]]))

plt.figure(figsize=(16,8))
plt.scatter(X, y, color = "red")
plt.plot(X, rl.predict(X), color = "green", linewidth =3)
plt.title("Predicción del modelo según la evolución de los datos")
plt.xlabel('años')
plt.ylabel('valor')
plt.legend(["Distribuión porcentual de varones, Tasa de actividad según sexo Varones por
nivel educativo Primaria completa"])
plt.show()

"""
# Tasa de actividad según sexo Varones por nivel educativo Secundaria completa"""

selecciona1 = ['Varones']
selecciona2 = ['Secundaria']
filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1) &
dataset['nivel_educa'].isin(selecciona2)]
filtered_df2

#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Distribuión porcentual de varones, Tasa de actividad según sexo Varones por
nivel educativo Secundaria completa")
plt.legend(["Distribuión porcentual de varones, Tasa de actividad según sexo Varones por
nivel educativo Secundaria completa"])
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")

from sklearn import linear_model
```

```
rl = linear_model.LinearRegression()

año =  filtered_df2["año"]
print(filtered_df2)
#como el linear model del sklearn espera que se le pase una matriz 2x2, hay que hacer un
reshape:
X = año[:, np.newaxis]

y = filtered_df2["valor"]
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_text = train_test_split(X,y)

while True:
  X_train, X_test, y_train, y_text = train_test_split(X,y)

  rl.fit(X_train, y_train)
  print(rl.score(X_train, y_train))
  if rl.score(X_train, y_train) > 0.75:
    break


print(rl.predict([[2022]]))
print(rl.predict([[2023]]))

plt.figure(figsize=(16,8))
plt.scatter(X, y, color = "red")
plt.plot(X, rl.predict(X), color = "green", linewidth =3)
plt.title("Distribuión porcentual de varones, Tasa de actividad según sexo Varones por nivel
educativo Secundaria completa")
plt.xlabel('años')
plt.ylabel('valor')
plt.legend(["Predicción del modelo según la evolución de los datos"])
plt.show()

"""# Tasa de actividad según sexo Varones por nivel educativo UTU completa"""

selecciona1 = ['Varones']
selecciona2 = ['UTU']
filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1) &
dataset['nivel_educa'].isin(selecciona2)]
filtered_df2

#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Distribuión porcentual de varones, Tasa de actividad según sexo Varones por
nivel educativo UTU completa")
plt.legend(["Distribuión porcentual de varones, Tasa de actividad según sexo Varones por
nivel educativo UTU completa"])
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")

from sklearn import linear_model

rl = linear_model.LinearRegression()

año =  filtered_df2["año"]
```

```
print(filtered_df2)

#como el linear model del sklearn espera que se le pase una matriz 2x2, hay que hacer un
reshape:
X = año[:, np.newaxis]

y = filtered_df2["valor"]
from sklearn import linear_model

rl = linear_model.LinearRegression()

from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_text = train_test_split(X,y)

while True:
  X_train, X_test, y_train, y_text = train_test_split(X,y)

  rl.fit(X_train, y_train)
  print(rl.score(X_train, y_train))
  if rl.score(X_train, y_train) > 0.20:
    break

print("La predicción para el año 2022 es de: ", rl.predict([[2022]]))
print("La predicción para el año 2023 es de: ", rl.predict([[2023]]))

plt.figure(figsize=(16,8))
plt.scatter(X, y, color = "red")
plt.plot(X, rl.predict(X), color = "green", linewidth =3)
plt.title("Distribuión porcentual de varones, Tasa de actividad según sexo Varones por nivel
educativo UTU completa")
plt.xlabel('años')
plt.ylabel('valor')
plt.legend(["Predicción del modelo según la evolución de los datos"])
plt.show()

"""# Tasa de actividad según sexo Varones por nivel educativo Magisterio o profesorado"""

selecciona1 = ['Varones']
selecciona2 = ['Magisterio o profesorado']
filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1) &
dataset['nivel_educa'].isin(selecciona2)]
filtered_df2

#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Distribuión porcentual de varones según máximo nivel educativo que alcanzaron de
Magisterio o profesorado")
plt.legend(["Distribuión porcentual de varones según máximo nivel educativo que alcanzaron
de Magisterio o profesorado"])
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")

from sklearn import linear_model

rl = linear_model.LinearRegression()

año =  filtered_df2["año"]
```

```
print(filtered_df2)

#como el linear model del sklearn espera que se le pase una matriz 2x2, hay que hacer un
reshape:
X = año[:, np.newaxis]

y = filtered_df2["valor"]
from sklearn import linear_model

rl = linear_model.LinearRegression()

from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_text = train_test_split(X,y)

while True:
  X_train, X_test, y_train, y_text = train_test_split(X,y)

  rl.fit(X_train, y_train)
  print(rl.score(X_train, y_train))
  if rl.score(X_train, y_train) > 0.35:
    break

print("La predicción para el año 2022 es de: ", rl.predict([[2022]]))
print("La predicción para el año 2023 es de: ", rl.predict([[2023]]))

plt.figure(figsize=(16,8))
plt.scatter(X, y, color = "red")
plt.plot(X, rl.predict(X), color = "green", linewidth =3)
plt.title("Distribuión porcentual de varones, Tasa de actividad según Magisterio Completo")
plt.ylabel('valor')
plt.legend(["Predicción del modelo según la evolución de los datos"])
plt.show()

"""# Tasa de actividad según sexo Varones por nivel educativo Universidad o similar"""

selecciona1 = ['Varones']
selecciona2 = ['Magisterio o profesorado']
filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1) &
dataset['nivel_educa'].isin(selecciona2)]
filtered_df2

#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Distribuión porcentual de varones según máximo nivel educativo que alcanzaron de
Universidad o similar")
plt.legend(["Distribuión porcentual de varones según máximo nivel educativo que alcanzaron
de Universidad o similar"])
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")

from sklearn import linear_model

rl = linear_model.LinearRegression()

año =  filtered_df2["año"]
print(filtered_df2)
```

```
#como el linear model del sklearn espera que se le pase una matriz 2x2, hay que hacer un
reshape:
X = año[:, np.newaxis]

y = filtered_df2["valor"]
from sklearn import linear_model

rl = linear_model.LinearRegression()

from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_text = train_test_split(X,y)

while True:
  X_train, X_test, y_train, y_text = train_test_split(X,y)

  rl.fit(X_train, y_train)
  print(rl.score(X_train, y_train))
  if rl.score(X_train, y_train) > 0.35:
    break

print("La predicción para el año 2022 es de: ", rl.predict([[2022]]))
print("La predicción para el año 2023 es de: ", rl.predict([[2023]]))

plt.figure(figsize=(16,8))
plt.scatter(X, y, color = "red")
plt.plot(X, rl.predict(X), color = "green", linewidth =3)
plt.title("Distribuión porcentual de varones, Tasa de actividad según sexo Varones por nivel
educativo Universidad completa")
plt.xlabel('años')
plt.ylabel('valor')
plt.legend(["Predicción del modelo según la evolución de los datos"])
plt.show()

# -*- coding: utf-8 -*-
"""Tasa de actividad según sexo Femenino por nivel educativo, Perceptron SKLearn.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/1tIfYh0i2I2XzFpC4jB0te4eBx6nQCdNX

# Proporción de mujeres que se encuentran activos (trabajan o buscan trabajo) entre aquellos
de 14 y más años, según nivel educativo. Nos permite medir el grado de participación de las
mujeres en el mercado de trabajo con dichas características.
"""

import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import datetime

dataset = pd.read_csv('/content/drive/MyDrive/PRIA Alfredo Poggio Manzor 24-08-2020 17:00
hrs/ProyectoFinal/DataSets/12559_tasa_de_actividad_segun_sexo_por_nivel_educativo-
_total_pais.csv',  encoding='latin-1')

dataset

"""# Tasa de actividad según sexo Mujeres por nivel educativo Sin instrucción"""
```

```
selecciona1 = ['Mujeres']
selecciona2 = ['Sin instrucción']
filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1) &
dataset['nivel_educa'].isin(selecciona2)]
filtered_df2
#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Tasa de actividad según sexo Mujeres por nivel educativo Sin instrucción")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Tasa de actividad según sexo Mujeres por nivel educativo Sin instrucción'])

from sklearn import linear_model

rl = linear_model.LinearRegression()

año =  filtered_df2["año"]
print(filtered_df2)
#como el linear model del sklearn espera que se le pase una matriz , hay que hacer un
reshape:
X = año[:, np.newaxis]

y = filtered_df2["valor"]
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_text = train_test_split(X,y)

while True:
  X_train, X_test, y_train, y_text = train_test_split(X,y)

  rl.fit(X_train, y_train)
  print(rl.score(X_train, y_train))
  if rl.score(X_train, y_train) > 0.35:
    break


print(rl.predict([[2022]]))
print(rl.predict([[2023]]))

plt.figure(figsize=(16,8))
plt.scatter(X, y, color = "red")
plt.plot(X, rl.predict(X), color = "green", linewidth =3)
plt.title("Predicción del modelo según la evolución de los datos")
plt.xlabel('años')
plt.ylabel('valor')
plt.legend(["Distribuión porcentual de Mujeres por nivel educativo Sin instrucción"])
plt.show()

"""# Tasa de actividad según sexo Mujeres por nivel educativo Primaria completa"""

selecciona1 = ['Mujeres']
selecciona2 = ['Primaria']
filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1) &
dataset['nivel_educa'].isin(selecciona2)]
filtered_df2

#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
```

```
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Distribuión porcentual de Mujeres, Tasa de actividad según sexo Mujeres por
nivel educativo Primaria completa")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Distribuión porcentual de Mujeres, Tasa de actividad según sexo Mujeres por
nivel educativo Primaria completa'])

from sklearn import linear_model

rl = linear_model.LinearRegression()

año =  filtered_df2["año"]
print(filtered_df2)
#como el linear model del sklearn espera que se le pase una matriz , hay que hacer un
reshape:
X = año[:, np.newaxis]

y = filtered_df2["valor"]
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_text = train_test_split(X,y)

while True:
  X_train, X_test, y_train, y_text = train_test_split(X,y)

  rl.fit(X_train, y_train)
  print(rl.score(X_train, y_train))
  if rl.score(X_train, y_train) > 0.45:
    break


print(rl.predict([[2022]]))
print(rl.predict([[2023]]))

plt.figure(figsize=(16,8))
plt.scatter(X, y, color = "red")
plt.plot(X, rl.predict(X), color = "green", linewidth =3)
plt.title("Predicción del modelo según la evolución de los datos")
plt.xlabel('años')
plt.ylabel('valor')
plt.legend(["Distribuión porcentual de Mujeres, Tasa de actividad según sexo Mujeres por
nivel educativo Primaria completa"])
plt.show()

"""
# Tasa de actividad según sexo Mujeres por nivel educativo Secundaria completa"""

selecciona1 = ['Mujeres']
selecciona2 = ['Secundaria']
filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1) &
dataset['nivel_educa'].isin(selecciona2)]
filtered_df2

#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
```

```
plt.ylabel("Distribuión porcentual de Mujeres, Tasa de actividad según sexo Mujeres por
nivel educativo Secundaria completa")
plt.legend(["Distribuión porcentual de Mujeres, Tasa de actividad según sexo Mujeres por
nivel educativo Secundaria completa"])
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")

from sklearn import linear_model

rl = linear_model.LinearRegression()

año =  filtered_df2["año"]
print(filtered_df2)
#como el linear model del sklearn espera que se le pase una matriz 2x2, hay que hacer un
reshape:
X = año[:, np.newaxis]

y = filtered_df2["valor"]
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_text = train_test_split(X,y)

while True:
  X_train, X_test, y_train, y_text = train_test_split(X,y)

  rl.fit(X_train, y_train)
  print(rl.score(X_train, y_train))
  if rl.score(X_train, y_train) > 0.75:
    break


print(rl.predict([[2022]]))
print(rl.predict([[2023]]))

plt.figure(figsize=(16,8))
plt.scatter(X, y, color = "red")
plt.plot(X, rl.predict(X), color = "green", linewidth =3)
plt.title("Distribuión porcentual de Mujeres, Tasa de actividad según sexo Mujeres por nivel
educativo Secundaria completa")
plt.xlabel('años')
plt.ylabel('valor')
plt.legend(["Predicción del modelo según la evolución de los datos"])
plt.show()

"""# Tasa de actividad según sexo Mujeres por nivel educativo UTU completa"""

selecciona1 = ['Mujeres']
selecciona2 = ['UTU']
filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1) &
dataset['nivel_educa'].isin(selecciona2)]
filtered_df2

#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Distribuión porcentual de Mujeres, Tasa de actividad según sexo Mujeres por
nivel educativo UTU completa")
plt.legend(["Distribuión porcentual de Mujeres, Tasa de actividad según sexo Mujeres por
nivel educativo UTU completa"])
```

```
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")

from sklearn import linear_model

rl = linear_model.LinearRegression()

año =  filtered_df2["año"]
print(filtered_df2)

#como el linear model del sklearn espera que se le pase una matriz 2x2, hay que hacer un
reshape:
X = año[:, np.newaxis]

y = filtered_df2["valor"]
from sklearn import linear_model

rl = linear_model.LinearRegression()

from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_text = train_test_split(X,y)

while True:
  X_train, X_test, y_train, y_text = train_test_split(X,y)

  rl.fit(X_train, y_train)
  print(rl.score(X_train, y_train))
  if rl.score(X_train, y_train) > 0.20:
    break

print("La predicción para el año 2022 es de: ", rl.predict([[2022]]))
print("La predicción para el año 2023 es de: ", rl.predict([[2023]]))

plt.figure(figsize=(16,8))
plt.scatter(X, y, color = "red")
plt.plot(X, rl.predict(X), color = "green", linewidth =3)
plt.title("Distribuión porcentual de Mujeres, Tasa de actividad según sexo Mujeres por nivel
educativo UTU completa")
plt.xlabel('años')
plt.ylabel('valor')
plt.legend(["Predicción del modelo según la evolución de los datos"])
plt.show()

"""# Tasa de actividad según sexo Mujeres por nivel educativo Magisterio o profesorado"""

selecciona1 = ['Mujeres']
selecciona2 = ['Magisterio o profesorado']
filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1) &
dataset['nivel_educa'].isin(selecciona2)]
filtered_df2

#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Distribuión porcentual de Mujeres según máximo nivel educativo que alcanzaron de
Magisterio o profesorado")
plt.legend(["Distribuión porcentual de Mujeres según máximo nivel educativo que alcanzaron
de Magisterio o profesorado"])
```

```
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")

from sklearn import linear_model

rl = linear_model.LinearRegression()

año =  filtered_df2["año"]
print(filtered_df2)

#como el linear model del sklearn espera que se le pase una matriz 2x2, hay que hacer un
reshape:
X = año[:, np.newaxis]

y = filtered_df2["valor"]
from sklearn import linear_model

rl = linear_model.LinearRegression()

from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_text = train_test_split(X,y)

while True:
  X_train, X_test, y_train, y_text = train_test_split(X,y)

  rl.fit(X_train, y_train)
  print(rl.score(X_train, y_train))
  if rl.score(X_train, y_train) > 0.60:
    break

print("La predicción para el año 2022 es de: ", rl.predict([[2022]]))
print("La predicción para el año 2023 es de: ", rl.predict([[2023]]))

plt.figure(figsize=(16,8))
plt.scatter(X, y, color = "red")
plt.plot(X, rl.predict(X), color = "green", linewidth =3)
plt.title("Distribuión porcentual de Mujeres según máximo nivel educativo que alcanzaron de
Magisterio o profesorado")
plt.xlabel('años')
plt.ylabel('valor')
plt.legend(["Predicción del modelo según la evolución de los datos"])
plt.show()

"""# Tasa de actividad según sexo Mujeres por nivel educativo Universidad o similar"""

selecciona1 = ['Mujeres']
selecciona2 = ['Universidad o similar']
filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1) &
dataset['nivel_educa'].isin(selecciona2)]
filtered_df2

#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Distribuión porcentual de Mujeres según máximo nivel educativo que alcanzaron de
Universidad o similar")
plt.legend(["Distribuión porcentual de Mujeres según máximo nivel educativo que alcanzaron
de Universidad o similar"])
```

```
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")

from sklearn import linear_model

rl = linear_model.LinearRegression()

año =  filtered_df2["año"]
print(filtered_df2)

#como el linear model del sklearn espera que se le pase una matriz 2x2, hay que hacer un
reshape:
X = año[:, np.newaxis]

y = filtered_df2["valor"]
from sklearn import linear_model

rl = linear_model.LinearRegression()

from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_text = train_test_split(X,y)

while True:
  X_train, X_test, y_train, y_text = train_test_split(X,y)

  rl.fit(X_train, y_train)
  print(rl.score(X_train, y_train))
  if rl.score(X_train, y_train) > 0.70:
    break

print("La predicción para el año 2022 es de: ", rl.predict([[2022]]))
print("La predicción para el año 2023 es de: ", rl.predict([[2023]]))

plt.figure(figsize=(16,8))
plt.scatter(X, y, color = "red")
plt.plot(X, rl.predict(X), color = "green", linewidth =3)
plt.title("Distribuión porcentual de Mujeres según máximo nivel educativo que alcanzaron de
Universidad o similar")
plt.xlabel('años')
plt.ylabel('valor')
plt.legend(["Predicción del modelo según la evolución de los datos"])
plt.show()


# -*- coding: utf-8 -*-
"""Tasa de actividad según sexo Femenino por nivel educativo, Perceptron Keras.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/1aAIqJS-vM0AduopxM7w1Mx1SJzC0nsux

# Proporción de Mujeres que se encuentran activos (trabajan o buscan trabajo) entre aquellas
de 14 y más años, según nivel educativo. Nos permite medir el grado de participación de las
mujeres  en el mercado de trabajo con dichas características.
"""

import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

```
from keras.models import Sequential
from keras.layers.core import Dense

import tensorflow as tf

dataset = pd.read_csv('/content/drive/MyDrive/PRIA Alfredo Poggio Manzor 24-08-2020 17:00
hrs/ProyectoFinal/DataSets/12559_tasa_de_actividad_segun_sexo_por_nivel_educativo-
_total_pais.csv',  encoding='latin-1')

dataset

"""# Tasa de actividad según sexo Mujeres por nivel educativo Sin instrucción"""

selecciona1 = ['Mujeres']
selecciona2 = ['Sin instrucción']
filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1) &
dataset['nivel_educa'].isin(selecciona2)]
filtered_df2
#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Tasa de actividad según sexo Mujeres por nivel educativo Sin instrucción")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Tasa de actividad según sexo Mujeres por nivel educativo Sin instrucción'])

"""# Preparo los conjuntos de entradas y salidas deseadas:"""

dataframeIII = filtered_df2["año"]

X = np.array(dataframeIII, dtype=float)

dataframeIV = filtered_df2["valor"]
Y = np.array(dataframeIV, dtype=float)

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,Y)

print(X_test)
print(y_test)

capa1 = tf.keras.layers.Dense(units = 2, input_shape = [1])
capa2 = tf.keras.layers.Dense(units=4)
capa3 = tf.keras.layers.Dense(units=2)
salida = tf.keras.layers.Dense(units=1)
modelo = tf.keras.Sequential([capa1, capa2, capa3,salida])

print(modelo.summary())

modelo.compile(
    optimizer = tf.keras.optimizers.Adam(0.01),
    loss = 'mean_squared_error'
)

historial = modelo.fit(X_train, y_train, epochs=60, verbose=False)
print("Modelo Entrenado")

import matplotlib.pyplot as plt
```

```
plt.xlabel('#Epoca')
plt.ylabel('Magnitud de pérdida')
plt.plot(historial.history['loss'])

print("Hagamos una Testeo")
resultado = modelo.predict(X_test)
print("Los valores calculados por la red neuronal son de: ")
print(str(resultado))
print("El valor esperado es de: ")
print(y_test)
print("Hagamos una predicción para el 2022: ")
resultado = modelo.predict([[2022]])
print(resultado)
print("Hagamos una predicción para el 2023: ")
resultado = modelo.predict([[2023]])
print(resultado)

"""# Tasa de actividad según sexo Mujeres por nivel educativo Primaria completa"""

selecciona1 = ['Mujeres']
selecciona2 = ['Primaria']
filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1) &
dataset['nivel_educa'].isin(selecciona2)]
filtered_df2
#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Tasa de actividad según sexo Mujeres por nivel educativo Primaria completa")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Tasa de actividad según sexo Mujeres por nivel educativo Primaria completa'])

dataframeIII = filtered_df2["año"]

X = np.array(dataframeIII, dtype=float)

dataframeIV = filtered_df2["valor"]
Y = np.array(dataframeIV, dtype=float)

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,Y)

print(X_test)
print(y_test)

capa1 = tf.keras.layers.Dense(units = 2, input_shape = [1])
capa2 = tf.keras.layers.Dense(units=4)
capa3 = tf.keras.layers.Dense(units=2)
salida = tf.keras.layers.Dense(units=1)
modelo = tf.keras.Sequential([capa1, capa2, capa3,salida])

print(modelo.summary())

modelo.compile(
    optimizer = tf.keras.optimizers.Adam(0.05),
    loss = 'mean_squared_error'
)

historial = modelo.fit(X_train, y_train, epochs=60, verbose=False)
```

```
print("Modelo Entrenado")

import matplotlib.pyplot as plt
plt.xlabel('#Epoca')
plt.ylabel('Magnitud de pérdida')
plt.plot(historial.history['loss'])

print("Hagamos una Testeo")
resultado = modelo.predict(X_test)
print("Los valores calculados por la red neuronal son de: ")
print(str(resultado))
print("El valor esperado es de: ")
print(y_test)
print("Hagamos una predicción para el 2022: ")
resultado = modelo.predict([[2022]])
print(resultado)
print("Hagamos una predicción para el 2023: ")
resultado = modelo.predict([[2023]])
print(resultado)

"""# Tasa de actividad según sexo Mujeres por nivel educativo Secundaria completa"""

selecciona1 = ['Mujeres']
selecciona2 = ['Secundaria']
filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1) &
dataset['nivel_educa'].isin(selecciona2)]
filtered_df2

#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Tasa de actividad según sexo Mujeres por nivel educativo Secundaria completa")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Tasa de actividad según sexo Mujeres por nivel educativo Secundaria completa'])

dataframeIII = filtered_df2["año"]

X = np.array(dataframeIII, dtype=float)

dataframeIV = filtered_df2["valor"]
Y = np.array(dataframeIV, dtype=float)

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,Y)

print(X_test)
print(y_test)

capa1 = tf.keras.layers.Dense(units = 2, input_shape = [1])
capa2 = tf.keras.layers.Dense(units=4)
capa3 = tf.keras.layers.Dense(units=2)
salida = tf.keras.layers.Dense(units=1)
modelo = tf.keras.Sequential([capa1, capa2, capa3,salida])

print(modelo.summary())

modelo.compile(
    optimizer = tf.keras.optimizers.Adam(0.01),
```

```
    loss = 'mean_squared_error'
)

historial = modelo.fit(X_train, y_train, epochs=60, verbose=False)
print("Modelo Entrenado")

import matplotlib.pyplot as plt
plt.xlabel('#Epoca')
plt.ylabel('Magnitud de pérdida')
plt.plot(historial.history['loss'])

print("Hagamos una Testeo")
resultado = modelo.predict(X_test)
print("Los valores calculados por la red neuronal son de: ")
print(str(resultado))
print("El valor esperado es de: ")
print(y_test)
print("Hagamos una predicción para el 2022: ")
resultado = modelo.predict([[2022]])
print(resultado)
print("Hagamos una predicción para el 2023: ")
resultado = modelo.predict([[2023]])
print(resultado)

"""# Tasa de actividad según sexo Mujeres por nivel educativo UTU completa"""

selecciona1 = ['Mujeres']
selecciona2 = ['UTU']
filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1) &
dataset['nivel_educa'].isin(selecciona2)]
filtered_df2

#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Tasa de actividad según sexo Mujeres por nivel educativo UTU completa")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Tasa de actividad según sexo Mujeres por nivel educativo UTU completa'])

dataframeIII = filtered_df2["año"]

X = np.array(dataframeIII, dtype=float)

dataframeIV = filtered_df2["valor"]
Y = np.array(dataframeIV, dtype=float)

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,Y)

print(X_test)
print(y_test)

capa1 = tf.keras.layers.Dense(units = 2, input_shape = [1])
capa2 = tf.keras.layers.Dense(units=4)
capa3 = tf.keras.layers.Dense(units=2)
salida = tf.keras.layers.Dense(units=1)
modelo = tf.keras.Sequential([capa1, capa2, capa3,salida])
```

```
print(modelo.summary())

modelo.compile(
    optimizer = tf.keras.optimizers.Adam(0.01),
    loss = 'mean_squared_error'
)

historial = modelo.fit(X_train, y_train, epochs=60, verbose=False)
print("Modelo Entrenado")

import matplotlib.pyplot as plt
plt.xlabel('#Epoca')
plt.ylabel('Magnitud de pérdida')
plt.plot(historial.history['loss'])

print("Hagamos una Testeo")
resultado = modelo.predict(X_test)
print("Los valores calculados por la red neuronal son de: ")
print(str(resultado))
print("El valor esperado es de: ")
print(y_test)
print("Hagamos una predicción para el 2022: ")
resultado = modelo.predict([[2022]])
print(resultado)
print("Hagamos una predicción para el 2023: ")
resultado = modelo.predict([[2023]])
print(resultado)

"""# Tasa de actividad según sexo Mujeres por nivel educativo Magisterio o profesorado"""

selecciona1 = ['Mujeres']
selecciona2 = ['Magisterio o profesorado']
filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1) &
dataset['nivel_educa'].isin(selecciona2)]
filtered_df2

#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Distribuión porcentual de Mujeres según máximo nivel educativo que alcanzaron de
Magisterio o profesorado")
plt.legend(["Distribuión porcentual de Mujeres según máximo nivel educativo que alcanzaron
de Magisterio o profesorado"])
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")

dataframeIII = filtered_df2["año"]

X = np.array(dataframeIII, dtype=float)

dataframeIV = filtered_df2["valor"]
Y = np.array(dataframeIV, dtype=float)

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,Y)

print(X_test)
print(y_test)
```

```
capa1 = tf.keras.layers.Dense(units = 2, input_shape = [1])
capa2 = tf.keras.layers.Dense(units=4)
capa3 = tf.keras.layers.Dense(units=2)
salida = tf.keras.layers.Dense(units=1)
modelo = tf.keras.Sequential([capa1, capa2, capa3,salida])

print(modelo.summary())

modelo.compile(
    optimizer = tf.keras.optimizers.Adam(0.01),
    loss = 'mean_squared_error'
)

historial = modelo.fit(X_train, y_train, epochs=60, verbose=False)
print("Modelo Entrenado")

import matplotlib.pyplot as plt
plt.xlabel('#Epoca')
plt.ylabel('Magnitud de pérdida')
plt.plot(historial.history['loss'])

print("Hagamos una Testeo")
resultado = modelo.predict(X_test)
print("Los valores calculados por la red neuronal son de: ")
print(str(resultado))
print("El valor esperado es de: ")
print(y_test)
print("Hagamos una predicción para el 2022: ")
resultado = modelo.predict([[2022]])
print(resultado)
print("Hagamos una predicción para el 2023: ")
resultado = modelo.predict([[2023]])
print(resultado)

"""# Tasa de actividad según sexo Mujeres por nivel educativo Universidad o similar"""

selecciona1 = ['Mujeres']
selecciona2 = ['Universidad o similar']
filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1) &
dataset['nivel_educa'].isin(selecciona2)]
filtered_df2

#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Distribuión porcentual de Mujeres según máximo nivel educativo que alcanzaron de
Universidad o similar")
plt.legend(["Distribuión porcentual de Mujeres según máximo nivel educativo que alcanzaron
de Universidad o similar"])
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")

dataframeIII = filtered_df2["año"]

X = np.array(dataframeIII, dtype=float)

dataframeIV = filtered_df2["valor"]
Y = np.array(dataframeIV, dtype=float)
```

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,Y)

print(X_test)
print(y_test)

capa1 = tf.keras.layers.Dense(units = 2, input_shape = [1])
capa2 = tf.keras.layers.Dense(units=4)
capa3 = tf.keras.layers.Dense(units=2)
salida = tf.keras.layers.Dense(units=1)
modelo = tf.keras.Sequential([capa1, capa2, capa3,salida])

print(modelo.summary())

modelo.compile(
    optimizer = tf.keras.optimizers.Adam(0.01),
    loss = 'mean_squared_error'
)

historial = modelo.fit(X_train, y_train, epochs=60, verbose=False)
print("Modelo Entrenado")

import matplotlib.pyplot as plt
plt.xlabel('#Epoca')
plt.ylabel('Magnitud de pérdida')
plt.plot(historial.history['loss'])

print("Hagamos una Testeo")
resultado = modelo.predict(X_test)
print("Los valores calculados por la red neuronal son de: ")
print(str(resultado))
print("El valor esperado es de: ")
print(y_test)
print("Hagamos una predicción para el 2022: ")
resultado = modelo.predict([[2022]])
print(resultado)
print("Hagamos una predicción para el 2023: ")
resultado = modelo.predict([[2023]])
print(resultado)


# -*- coding: utf-8 -*-
"""Tasa de actividad según sexo Masculino por nivel educativo, Perceptron Keras.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/1RIsOlbG4onqHa2Wn0hakEHyTS2TONAwn

# Proporción de varones que se encuentran activos (trabajan o buscan trabajo) entre aquellas de 14 y más años, según nivel educativo. Nos permite medir el grado de participación de las mujeres  en el mercado de trabajo con dichas características.
"""

import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

from keras.models import Sequential
from keras.layers.core import Dense
```

```
import tensorflow as tf

dataset = pd.read_csv('/content/drive/MyDrive/PRIA Alfredo Poggio Manzor 24-08-2020 17:00
hrs/ProyectoFinal/DataSets/12559_tasa_de_actividad_segun_sexo_por_nivel_educativo-
_total_pais.csv',  encoding='latin-1')

dataset

"""# Tasa de actividad según sexo Mujeres por nivel educativo Sin instrucción"""

selecciona1 = ['Varones']
selecciona2 = ['Sin instrucción']
filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1) &
dataset['nivel_educa'].isin(selecciona2)]
filtered_df2
#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Tasa de actividad según sexo Varones por nivel educativo Sin instrucción")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Tasa de actividad según sexo Varones por nivel educativo Sin instrucción'])

"""# Preparo los conjuntos de entradas y salidas deseadas:"""

dataframeIII = filtered_df2["año"]

X = np.array(dataframeIII, dtype=float)

dataframeIV = filtered_df2["valor"]
Y = np.array(dataframeIV, dtype=float)

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,Y)

print(X_test)
print(y_test)

capa1 = tf.keras.layers.Dense(units = 2, input_shape = [1])
capa2 = tf.keras.layers.Dense(units=4)
capa3 = tf.keras.layers.Dense(units=2)
salida = tf.keras.layers.Dense(units=1)
modelo = tf.keras.Sequential([capa1, capa2, capa3,salida])

print(modelo.summary())

modelo.compile(
    optimizer = tf.keras.optimizers.Adam(0.01),
    loss = 'mean_squared_error'
)

historial = modelo.fit(X_train, y_train, epochs=60, verbose=False)
print("Modelo Entrenado")

import matplotlib.pyplot as plt
plt.xlabel('#Epoca')
plt.ylabel('Magnitud de pérdida')
plt.plot(historial.history['loss'])
```

```
print("Hagamos una Testeo")
resultado = modelo.predict(X_test)
print("Los valores calculados por la red neuronal son de: ")
print(str(resultado))
print("El valor esperado es de: ")
print(y_test)
print("Hagamos una predicción para el 2022: ")
resultado = modelo.predict([[2022]])
print(resultado)
print("Hagamos una predicción para el 2023: ")
resultado = modelo.predict([[2023]])
print(resultado)

"""# Tasa de actividad según sexo Varones por nivel educativo Primaria completa"""

selecciona1 = ['Varones']
selecciona2 = ['Primaria']
filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1) &
dataset['nivel_educa'].isin(selecciona2)]
filtered_df2
#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Tasa de actividad según sexo Varones por nivel educativo Primaria completa")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Tasa de actividad según sexo Varones por nivel educativo Primaria completa'])

dataframeIII = filtered_df2["año"]

X = np.array(dataframeIII, dtype=float)

dataframeIV = filtered_df2["valor"]
Y = np.array(dataframeIV, dtype=float)

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,Y)

print(X_test)
print(y_test)

capa1 = tf.keras.layers.Dense(units = 2, input_shape = [1])
capa2 = tf.keras.layers.Dense(units=4)
capa3 = tf.keras.layers.Dense(units=2)
salida = tf.keras.layers.Dense(units=1)
modelo = tf.keras.Sequential([capa1, capa2, capa3,salida])

print(modelo.summary())

modelo.compile(
    optimizer = tf.keras.optimizers.Adam(0.05),
    loss = 'mean_squared_error'
)

historial = modelo.fit(X_train, y_train, epochs=60, verbose=False)
print("Modelo Entrenado")

import matplotlib.pyplot as plt
```

```
plt.xlabel('#Epoca')
plt.ylabel('Magnitud de pérdida')
plt.plot(historial.history['loss'])

print("Hagamos una Testeo")
resultado = modelo.predict(X_test)
print("Los valores calculados por la red neuronal son de: ")
print(str(resultado))
print("El valor esperado es de: ")
print(y_test)
print("Hagamos una predicción para el 2022: ")
resultado = modelo.predict([[2022]])
print(resultado)
print("Hagamos una predicción para el 2023: ")
resultado = modelo.predict([[2023]])
print(resultado)

"""# Tasa de actividad según sexo Varones por nivel educativo Secundaria completa"""

selecciona1 = ['Varones']
selecciona2 = ['Secundaria']
filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1) &
dataset['nivel_educa'].isin(selecciona2)]
filtered_df2

#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Tasa de actividad según sexo Varones por nivel educativo Secundaria completa")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Tasa de actividad según sexo Varones por nivel educativo Secundaria completa'])

dataframeIII = filtered_df2["año"]

X = np.array(dataframeIII, dtype=float)

dataframeIV = filtered_df2["valor"]
Y = np.array(dataframeIV, dtype=float)

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,Y)

print(X_test)
print(y_test)

capa1 = tf.keras.layers.Dense(units = 2, input_shape = [1])
capa2 = tf.keras.layers.Dense(units=4)
capa3 = tf.keras.layers.Dense(units=2)
salida = tf.keras.layers.Dense(units=1)
modelo = tf.keras.Sequential([capa1, capa2, capa3,salida])

print(modelo.summary())

modelo.compile(
    optimizer = tf.keras.optimizers.Adam(0.01),
    loss = 'mean_squared_error'
)
```

```
historial = modelo.fit(X_train, y_train, epochs=60, verbose=False)
print("Modelo Entrenado")

import matplotlib.pyplot as plt
plt.xlabel('#Epoca')
plt.ylabel('Magnitud de pérdida')
plt.plot(historial.history['loss'])

print("Hagamos una Testeo")
resultado = modelo.predict(X_test)
print("Los valores calculados por la red neuronal son de: ")
print(str(resultado))
print("El valor esperado es de: ")
print(y_test)
print("Hagamos una predicción para el 2022: ")
resultado = modelo.predict([[2022]])
print(resultado)
print("Hagamos una predicción para el 2023: ")
resultado = modelo.predict([[2023]])
print(resultado)

"""# Tasa de actividad según sexo Varones por nivel educativo UTU completa"""

selecciona1 = ['Varones']
selecciona2 = ['UTU']
filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1) &
dataset['nivel_educa'].isin(selecciona2)]
filtered_df2

#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Tasa de actividad según sexo Varones por nivel educativo UTU completa")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Tasa de actividad según sexo Varones por nivel educativo UTU completa'])

dataframeIII = filtered_df2["año"]

X = np.array(dataframeIII, dtype=float)

dataframeIV = filtered_df2["valor"]
Y = np.array(dataframeIV, dtype=float)

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,Y)

print(X_test)
print(y_test)

capa1 = tf.keras.layers.Dense(units = 2, input_shape = [1])
capa2 = tf.keras.layers.Dense(units=4)
capa3 = tf.keras.layers.Dense(units=2)
salida = tf.keras.layers.Dense(units=1)
modelo = tf.keras.Sequential([capa1, capa2, capa3,salida])

print(modelo.summary())

modelo.compile(
```

```
    optimizer = tf.keras.optimizers.Adam(0.01),
    loss = 'mean_squared_error'
)

historial = modelo.fit(X_train, y_train, epochs=60, verbose=False)
print("Modelo Entrenado")

import matplotlib.pyplot as plt
plt.xlabel('#Epoca')
plt.ylabel('Magnitud de pérdida')
plt.plot(historial.history['loss'])

print("Hagamos una Testeo")
resultado = modelo.predict(X_test)
print("Los valores calculados por la red neuronal son de: ")
print(str(resultado))
print("El valor esperado es de: ")
print(y_test)
print("Hagamos una predicción para el 2022: ")
resultado = modelo.predict([[2022]])
print(resultado)
print("Hagamos una predicción para el 2023: ")
resultado = modelo.predict([[2023]])
print(resultado)

"""# Tasa de actividad según sexo Varones por nivel educativo Magisterio o profesorado"""

selecciona1 = ['Varones']
selecciona2 = ['Magisterio o profesorado']
filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1) &
dataset['nivel_educa'].isin(selecciona2)]
filtered_df2

#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Distribuión porcentual de Varones según máximo nivel educativo que alcanzaron de
Magisterio o profesorado")
plt.legend(["Distribuión porcentual de Varones según máximo nivel educativo que alcanzaron
de Magisterio o profesorado"])
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")

dataframeIII = filtered_df2["año"]

X = np.array(dataframeIII, dtype=float)

dataframeIV = filtered_df2["valor"]
Y = np.array(dataframeIV, dtype=float)

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,Y)

print(X_test)
print(y_test)

capa1 = tf.keras.layers.Dense(units = 2, input_shape = [1])
capa2 = tf.keras.layers.Dense(units=4)
capa3 = tf.keras.layers.Dense(units=2)
```

```
salida = tf.keras.layers.Dense(units=1)
modelo = tf.keras.Sequential([capa1, capa2, capa3,salida])

print(modelo.summary())

modelo.compile(
    optimizer = tf.keras.optimizers.Adam(0.01),
    loss = 'mean_squared_error'
)

historial = modelo.fit(X_train, y_train, epochs=60, verbose=False)
print("Modelo Entrenado")

import matplotlib.pyplot as plt
plt.xlabel('#Epoca')
plt.ylabel('Magnitud de pérdida')
plt.plot(historial.history['loss'])

print("Hagamos una Testeo")
resultado = modelo.predict(X_test)
print("Los valores calculados por la red neuronal son de: ")
print(str(resultado))
print("El valor esperado es de: ")
print(y_test)
print("Hagamos una predicción para el 2022: ")
resultado = modelo.predict([[2022]])
print(resultado)
print("Hagamos una predicción para el 2023: ")
resultado = modelo.predict([[2023]])
print(resultado)

"""# Tasa de actividad según sexo Varones por nivel educativo Universidad o similar"""

selecciona1 = ['Varones']
selecciona2 = ['Universidad o similar']
filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1) &
dataset['nivel_educa'].isin(selecciona2)]
filtered_df2

#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Distribuión porcentual de Varones según máximo nivel educativo que alcanzaron de
Universidad o similar")
plt.legend(["Distribuión porcentual de Varones según máximo nivel educativo que alcanzaron
de Universidad o similar"])
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")

dataframeIII = filtered_df2["año"]

X = np.array(dataframeIII, dtype=float)

dataframeIV = filtered_df2["valor"]
Y = np.array(dataframeIV, dtype=float)

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,Y)
```

```
print(X_test)
print(y_test)

capa1 = tf.keras.layers.Dense(units = 2, input_shape = [1])
capa2 = tf.keras.layers.Dense(units=4)
capa3 = tf.keras.layers.Dense(units=2)
salida = tf.keras.layers.Dense(units=1)
modelo = tf.keras.Sequential([capa1, capa2, capa3,salida])

print(modelo.summary())

modelo.compile(
    optimizer = tf.keras.optimizers.Adam(0.01),
    loss = 'mean_squared_error'
)

historial = modelo.fit(X_train, y_train, epochs=60, verbose=False)
print("Modelo Entrenado")

import matplotlib.pyplot as plt
plt.xlabel('#Epoca')
plt.ylabel('Magnitud de pérdida')
plt.plot(historial.history['loss'])

print("Hagamos una Testeo")
resultado = modelo.predict(X_test)
print("Los valores calculados por la red neuronal son de: ")
print(str(resultado))
print("El valor esperado es de: ")
print(y_test)
print("Hagamos una predicción para el 2022: ")
resultado = modelo.predict([[2022]])
print(resultado)
print("Hagamos una predicción para el 2023: ")
resultado = modelo.predict([[2023]])
print(resultado)
```

Tasa de desempleo según sexo por ascendencia afro. Total país

Código fuente:

```
# -*- coding: utf-8 -*-
"""Tasa de desempleo según sexo Masculino por ascendencia afro, Perceptron SKLearn.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/1p8Gwhd3nteSPoYqrJloMDyojT_3DiuZ9

# Proporción de varones y mujeres que buscan empleo y no tienen, en relación a la población económicamente activa, según ascendencia afro.
"""

import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import datetime
```

```
dataset = pd.read_csv('/content/drive/MyDrive/PRIA Alfredo Poggio Manzor 24-08-2020 17:00
hrs/ProyectoFinal/DataSets/10213_tasa_de_desempleo_segun_sexo_por_ascendencia_afro-
_total_pais.csv',  encoding='latin-1')

dataset

"""# Tasa de desempleo según sexo Varones no afrodescendientes

"""

selecciona1 = ['Varones']
selecciona2 = ['No Afrodescendiente']

filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1) & dataset['Ascendencia
Racial'].isin(selecciona2)]
filtered_df2
#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Distribuión porcentual de desempleo de varones según ascendencia no afro")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Distribuión porcentual de desempleo de varones según ascendencia no afro'])

from sklearn import linear_model

rl = linear_model.LinearRegression()

año =  filtered_df2["año"]
print(filtered_df2)
#como el linear model del sklearn espera que se le pase una matriz , hay que hacer un
reshape:
X = año[:, np.newaxis]

y = filtered_df2["valor"]
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_text = train_test_split(X,y)

while True:
  X_train, X_test, y_train, y_text = train_test_split(X,y)

  rl.fit(X_train, y_train)
  print(rl.score(X_train, y_train))
  if rl.score(X_train, y_train) > 0.60:
    break


print(rl.predict([[2022]]))
print(rl.predict([[2023]]))

plt.figure(figsize=(16,8))
plt.scatter(X, y, color = "red")
plt.plot(X, rl.predict(X), color = "green", linewidth =3)
plt.title("Predicción del modelo según la evolución de los datos")
plt.xlabel('años')
plt.ylabel('valor')
plt.legend(["Distribuión porcentual de desempleo de Varones por ascedencia no afro"])
plt.show()
```

```
"""# Tasa de desempleo según sexo Varones por afrodescendencia"""

selecciona1 = ['Varones']
selecciona2 = ['Afrodescendiente']

filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1) & dataset['Ascendencia
Racial'].isin(selecciona2)]
filtered_df2
#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Distribuión porcentual de desempleo de varones según ascendencia no afro")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Distribuión porcentual de desempleo de varones según ascendencia no afro'])

from sklearn import linear_model

rl = linear_model.LinearRegression()

año =  filtered_df2["año"]
print(filtered_df2)
#como el linear model del sklearn espera que se le pase una matriz , hay que hacer un
reshape:
X = año[:, np.newaxis]

y = filtered_df2["valor"]
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_text = train_test_split(X,y)

while True:
  X_train, X_test, y_train, y_text = train_test_split(X,y)

  rl.fit(X_train, y_train)
  print(rl.score(X_train, y_train))
  if rl.score(X_train, y_train) > 0.65:
    break


print(rl.predict([[2022]]))
print(rl.predict([[2023]]))

plt.figure(figsize=(16,8))
plt.scatter(X, y, color = "red")
plt.plot(X, rl.predict(X), color = "green", linewidth =3)
plt.title("Predicción del modelo según la evolución de los datos")
plt.xlabel('años')
plt.ylabel('valor')
plt.legend(["Distribuión porcentual de varones, Tasa de desempleo según sexo varones por
descendencia afro"])
plt.show()


# -*- coding: utf-8 -*-
"""Tasa de desempleo según sexo Masculino por ascendencia afro, Perceptron SKLearn.ipynb

Automatically generated by Colaboratory.
```

```
Original file is located at
    https://colab.research.google.com/drive/1p8Gwhd3nteSPoYqrJloMDyojT_3DiuZ9

# Proporción de varones y mujeres que buscan empleo y no tienen, en relación a la población
económicamente activa, según ascendencia afro.
"""

import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import datetime

dataset = pd.read_csv('/content/drive/MyDrive/PRIA Alfredo Poggio Manzor 24-08-2020 17:00
hrs/ProyectoFinal/DataSets/10213_tasa_de_desempleo_segun_sexo_por_ascendencia_afro-
_total_pais.csv',  encoding='latin-1')

dataset

"""# Tasa de desempleo según sexo Varones no afrodescendientes

"""

selecciona1 = ['Varones']
selecciona2 = ['No Afrodescendiente']

filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1) & dataset['Ascendencia
Racial'].isin(selecciona2)]
filtered_df2
#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Distribuión porcentual de desempleo de varones según ascendencia no afro")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Distribuión porcentual de desempleo de varones según ascendencia no afro'])

from sklearn import linear_model

rl = linear_model.LinearRegression()

año =  filtered_df2["año"]
print(filtered_df2)
#como el linear model del sklearn espera que se le pase una matriz , hay que hacer un
reshape:
X = año[:, np.newaxis]

y = filtered_df2["valor"]
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_text = train_test_split(X,y)

while True:
  X_train, X_test, y_train, y_text = train_test_split(X,y)

  rl.fit(X_train, y_train)
  print(rl.score(X_train, y_train))
  if rl.score(X_train, y_train) > 0.60:
    break
```

```
print(rl.predict([[2022]]))
print(rl.predict([[2023]]))

plt.figure(figsize=(16,8))
plt.scatter(X, y, color = "red")
plt.plot(X, rl.predict(X), color = "green", linewidth =3)
plt.title("Predicción del modelo según la evolución de los datos")
plt.xlabel('años')
plt.ylabel('valor')
plt.legend(["Distribuión porcentual de desempleo de Varones por ascedencia no afro"])
plt.show()

"""# Tasa de desempleo según sexo Varones por afrodescendencia"""

selecciona1 = ['Varones']
selecciona2 = ['Afrodescendiente']

filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1) & dataset['Ascendencia
Racial'].isin(selecciona2)]
filtered_df2
#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Distribuión porcentual de desempleo de varones según ascendencia no afro")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Distribuión porcentual de desempleo de varones según ascendencia no afro'])

from sklearn import linear_model

rl = linear_model.LinearRegression()

año =  filtered_df2["año"]
print(filtered_df2)
#como el linear model del sklearn espera que se le pase una matriz , hay que hacer un
reshape:
X = año[:, np.newaxis]

y = filtered_df2["valor"]
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_text = train_test_split(X,y)

while True:
  X_train, X_test, y_train, y_text = train_test_split(X,y)

  rl.fit(X_train, y_train)
  print(rl.score(X_train, y_train))
  if rl.score(X_train, y_train) > 0.65:
    break


print(rl.predict([[2022]]))
print(rl.predict([[2023]]))

plt.figure(figsize=(16,8))
plt.scatter(X, y, color = "red")
```

```
plt.plot(X, rl.predict(X), color = "green", linewidth =3)
plt.title("Predicción del modelo según la evolución de los datos")
plt.xlabel('años')
plt.ylabel('valor')
plt.legend(["Distribuión porcentual de varones, Tasa de desempleo según sexo varones por
descendencia afro"])
plt.show()


# -*- coding: utf-8 -*-
"""Tasa de desempleo según sexo Femenino, por ascendencia afro, Perceptron Keras.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/1ZAYtNWZbFvpdlpGAOhi-Z6ertNm1W9P6

# Proporción de varones y mujeres que buscan empleo y no tienen, en relación a la población
económicamente activa, según ascendencia afro..
"""

import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

from keras.models import Sequential
from keras.layers.core import Dense

import tensorflow as tf

dataset = pd.read_csv('/content/drive/MyDrive/PRIA Alfredo Poggio Manzor 24-08-2020 17:00
hrs/ProyectoFinal/DataSets/10213_tasa_de_desempleo_segun_sexo_por_ascendencia_afro-
_total_pais.csv',  encoding='latin-1')

dataset

"""# Tasa de desempleo según sexo Mujeres por no afrodescendencia"""

selecciona1 = ['Mujeres']
selecciona2 = ['No Afrodescendiente']
filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1) & dataset['Ascendencia
Racial'].isin(selecciona2)]
filtered_df2
#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Tasa de desempleo según sexo Mujeres por No Afrodescendiente")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Tasa de desempleo según sexo Mujeres por No Afrodescendiente'])

"""# Preparo los conjuntos de entradas y salidas deseadas:"""

dataframeIII = filtered_df2["año"]

X = np.array(dataframeIII, dtype=float)

dataframeIV = filtered_df2["valor"]
Y = np.array(dataframeIV, dtype=float)
```

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,Y)

print(X_test)
print(y_test)

capa1 = tf.keras.layers.Dense(units = 2, input_shape = [1])
capa2 = tf.keras.layers.Dense(units=4)
capa3 = tf.keras.layers.Dense(units=2)
salida = tf.keras.layers.Dense(units=1)
modelo = tf.keras.Sequential([capa1, capa2, capa3,salida])

print(modelo.summary())

modelo.compile(
    optimizer = tf.keras.optimizers.Adam(0.01),
    loss = 'mean_squared_error'
)

historial = modelo.fit(X_train, y_train, epochs=60, verbose=False)
print("Modelo Entrenado")

import matplotlib.pyplot as plt
plt.xlabel('#Epoca')
plt.ylabel('Magnitud de pérdida')
plt.plot(historial.history['loss'])

print("Hagamos una Testeo")
resultado = modelo.predict(X_test)
print("Los valores calculados por la red neuronal son de: ")
print(str(resultado))
print("El valor esperado es de: ")
print(y_test)
print("Hagamos una predicción para el 2022: ")
resultado = modelo.predict([[2022]])
print(resultado)
print("Hagamos una predicción para el 2023: ")
resultado = modelo.predict([[2023]])
print(resultado)

"""# Tasa de desempleo según sexo Mujeres por ascendencia afro"""

selecciona1 = ['Mujeres']
selecciona2 = ['Afrodescendiente']
filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1) & dataset['Ascendencia
Racial'].isin(selecciona2)]
filtered_df2
#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Tasa de desempleo según sexo Mujeres por Afrodescendiente")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Tasa de desempleo según sexo Mujeres por Afrodescendiente'])

dataframeIII = filtered_df2["año"]

X = np.array(dataframeIII, dtype=float)
```

```
dataframeIV = filtered_df2["valor"]
Y = np.array(dataframeIV, dtype=float)

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,Y)

print(X_test)
print(y_test)

capa1 = tf.keras.layers.Dense(units = 2, input_shape = [1])
capa2 = tf.keras.layers.Dense(units=4)
capa3 = tf.keras.layers.Dense(units=2)
salida = tf.keras.layers.Dense(units=1)
modelo = tf.keras.Sequential([capa1, capa2, capa3,salida])

print(modelo.summary())

modelo.compile(
    optimizer = tf.keras.optimizers.Adam(0.01),
    loss = 'mean_squared_error'
)

historial = modelo.fit(X_train, y_train, epochs=60, verbose=False)
print("Modelo Entrenado")

import matplotlib.pyplot as plt
plt.xlabel('#Epoca')
plt.ylabel('Magnitud de pérdida')
plt.plot(historial.history['loss'])

print("Hagamos una Testeo")
resultado = modelo.predict(X_test)
print("Los valores calculados por la red neuronal son de: ")
print(str(resultado))
print("El valor esperado es de: ")
print(y_test)
print("Hagamos una predicción para el 2022: ")
resultado = modelo.predict([[2022]])
print(resultado)
print("Hagamos una predicción para el 2023: ")
resultado = modelo.predict([[2023]])
print(resultado)


# -*- coding: utf-8 -*-
"""Tasa de desempleo según sexo Masculino, por ascendencia afro, Perceptron Keras.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/1fS_Om0BXjL1hpjNsf-p-VUwij9sKmpRW

# Proporción de varones y mujeres que buscan empleo y no tienen, en relación a la población económicamente activa, según ascendencia afro..
"""

import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

```
from keras.models import Sequential
from keras.layers.core import Dense

import tensorflow as tf

dataset = pd.read_csv('/content/drive/MyDrive/PRIA Alfredo Poggio Manzor 24-08-2020 17:00
hrs/ProyectoFinal/DataSets/10213_tasa_de_desempleo_segun_sexo_por_ascendencia_afro-
_total_pais.csv',  encoding='latin-1')

dataset

"""# Tasa de desempleo según sexo Varones por no afrodescendencia"""

selecciona1 = ['Varones']
selecciona2 = ['No Afrodescendiente']
filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1) & dataset['Ascendencia
Racial'].isin(selecciona2)]
filtered_df2
#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Tasa de desempleo según sexo Varones por No Afrodescendiente")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Tasa de desempleo según sexo Varones por No Afrodescendiente'])

"""# Preparo los conjuntos de entradas y salidas deseadas:"""

dataframeIII = filtered_df2["año"]

X = np.array(dataframeIII, dtype=float)

dataframeIV = filtered_df2["valor"]
Y = np.array(dataframeIV, dtype=float)

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,Y)

print(X_test)
print(y_test)

capa1 = tf.keras.layers.Dense(units = 2, input_shape = [1])
capa2 = tf.keras.layers.Dense(units=4)
capa3 = tf.keras.layers.Dense(units=2)
salida = tf.keras.layers.Dense(units=1)
modelo = tf.keras.Sequential([capa1, capa2, capa3,salida])

print(modelo.summary())

modelo.compile(
    optimizer = tf.keras.optimizers.Adam(0.01),
    loss = 'mean_squared_error'
)

historial = modelo.fit(X_train, y_train, epochs=60, verbose=False)
print("Modelo Entrenado")

import matplotlib.pyplot as plt
```

```
plt.xlabel('#Epoca')
plt.ylabel('Magnitud de pérdida')
plt.plot(historial.history['loss'])

print("Hagamos una Testeo")
resultado = modelo.predict(X_test)
print("Los valores calculados por la red neuronal son de: ")
print(str(resultado))
print("El valor esperado es de: ")
print(y_test)
print("Hagamos una predicción para el 2022: ")
resultado = modelo.predict([[2022]])
print(resultado)
print("Hagamos una predicción para el 2023: ")
resultado = modelo.predict([[2023]])
print(resultado)

"""# Tasa de desempleo según sexo Varones por ascendencia afro"""

selecciona1 = ['Varones']
selecciona2 = ['Afrodescendiente']
filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1) & dataset['Ascendencia
Racial'].isin(selecciona2)]
filtered_df2
#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Tasa de desempleo según sexo Varones por Afrodescendiente")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Tasa de desempleo según sexo Varones por Afrodescendiente'])

dataframeIII = filtered_df2["año"]

X = np.array(dataframeIII, dtype=float)

dataframeIV = filtered_df2["valor"]
Y = np.array(dataframeIV, dtype=float)

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,Y)

print(X_test)
print(y_test)

capa1 = tf.keras.layers.Dense(units = 2, input_shape = [1])
capa2 = tf.keras.layers.Dense(units=4)
capa3 = tf.keras.layers.Dense(units=2)
salida = tf.keras.layers.Dense(units=1)
modelo = tf.keras.Sequential([capa1, capa2, capa3,salida])

print(modelo.summary())

modelo.compile(
    optimizer = tf.keras.optimizers.Adam(0.01),
    loss = 'mean_squared_error'
)

historial = modelo.fit(X_train, y_train, epochs=60, verbose=False)
```

```
print("Modelo Entrenado")

import matplotlib.pyplot as plt
plt.xlabel('#Epoca')
plt.ylabel('Magnitud de pérdida')
plt.plot(historial.history['loss'])

print("Hagamos una Testeo")
resultado = modelo.predict(X_test)
print("Los valores calculados por la red neuronal son de: ")
print(str(resultado))
print("El valor esperado es de: ")
print(y_test)
print("Hagamos una predicción para el 2022: ")
resultado = modelo.predict([[2022]])
print(resultado)
print("Hagamos una predicción para el 2023: ")
resultado = modelo.predict([[2023]])
print(resultado)
```

Porcentaje de personas en situación de pobreza según sexo por ascendencia afro. Total país

Código fuente:

```
# -*- coding: utf-8 -*-
"""Porcentaje de personas en situación de pobreza según sexo Masculino, por ascendencia
afro, Perceptron Keras.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/1rLyaLwVaEiRzwgUrFHkzymekBaWJLC2N

# Porcentaje de varones y mujeres que habitan en hogares cuyo ingreso per cápita es inferior
a la línea de pobreza (metodología INE 2006), según ascendencia afro.
"""

import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

from keras.models import Sequential
from keras.layers.core import Dense

import tensorflow as tf

dataset = pd.read_csv('/content/drive/MyDrive/PRIA Alfredo Poggio Manzor 24-08-2020 17:00
hrs/ProyectoFinal/DataSets/10885_porcentaje_de_personas_en_situacion_de_pobreza_segun_sexo_p
or_ascendencia_afro-_total_pais.csv',  encoding='latin-1')

dataset

"""# Porcentaje de personas en situación de pobreza según sexo Varones por no
afrodescendencia"""
```

```
selecciona1 = ['Varones']
selecciona2 = ['No Afrodescendiente']
filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1) & dataset['Ascendencia
Racial'].isin(selecciona2)]
filtered_df2
#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Porcentaje de personas en situación de pobreza  según sexo Varones por No
Afrodescendiente")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Porcentaje de personas en situación de pobreza  según sexo Varones por No
Afrodescendiente'])

"""# Preparo los conjuntos de entradas y salidas deseadas:"""

dataframeIII = filtered_df2["año"]

X = np.array(dataframeIII, dtype=float)

dataframeIV = filtered_df2["valor"]
Y = np.array(dataframeIV, dtype=float)

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,Y)

print(X_test)
print(y_test)

capa1 = tf.keras.layers.Dense(units = 2, input_shape = [1])
capa2 = tf.keras.layers.Dense(units=4)
capa3 = tf.keras.layers.Dense(units=2)
salida = tf.keras.layers.Dense(units=1)
modelo = tf.keras.Sequential([capa1, capa2, capa3,salida])

print(modelo.summary())

modelo.compile(
    optimizer = tf.keras.optimizers.Adam(0.01),
    loss = 'mean_squared_error'
)

historial = modelo.fit(X_train, y_train, epochs=60, verbose=False)
print("Modelo Entrenado")

import matplotlib.pyplot as plt
plt.xlabel('#Epoca')
plt.ylabel('Magnitud de pérdida')
plt.plot(historial.history['loss'])

print("Hagamos una Testeo")
resultado = modelo.predict(X_test)
print("Los valores calculados por la red neuronal son de: ")
print(str(resultado))
print("El valor esperado es de: ")
print(y_test)
print("Hagamos una predicción para el 2022: ")
resultado = modelo.predict([[2022]])
```

```
print(resultado)
print("Hagamos una predicción para el 2023: ")
resultado = modelo.predict([[2023]])
print(resultado)

"""# Porcentaje de personas en situación de pobreza según sexo Varones por ascendencia
afro"""

selecciona1 = ['Varones']
selecciona2 = ['Afrodescendiente']
filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1) & dataset['Ascendencia
Racial'].isin(selecciona2)]
filtered_df2
#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Porcentaje de personas en situación de pobreza según sexo Varones por
Afrodescendiente")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Porcentaje de personas en situación de pobreza según sexo Varones por
Afrodescendiente'])

dataframeIII = filtered_df2["año"]

X = np.array(dataframeIII, dtype=float)

dataframeIV = filtered_df2["valor"]
Y = np.array(dataframeIV, dtype=float)

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,Y)

print(X_test)
print(y_test)

capa1 = tf.keras.layers.Dense(units = 2, input_shape = [1])
capa2 = tf.keras.layers.Dense(units=4)
capa3 = tf.keras.layers.Dense(units=2)
salida = tf.keras.layers.Dense(units=1)
modelo = tf.keras.Sequential([capa1, capa2, capa3,salida])

print(modelo.summary())

modelo.compile(
    optimizer = tf.keras.optimizers.Adam(0.01),
    loss = 'mean_squared_error'
)

historial = modelo.fit(X_train, y_train, epochs=60, verbose=False)
print("Modelo Entrenado")

import matplotlib.pyplot as plt
plt.xlabel('#Epoca')
plt.ylabel('Magnitud de pérdida')
plt.plot(historial.history['loss'])

print("Hagamos una Testeo")
resultado = modelo.predict(X_test)
```

```
print("Los valores calculados por la red neuronal son de: ")
print(str(resultado))
print("El valor esperado es de: ")
print(y_test)
print("Hagamos una predicción para el 2022: ")
resultado = modelo.predict([[2022]])
print(resultado)
print("Hagamos una predicción para el 2023: ")
resultado = modelo.predict([[2023]])
print(resultado)

# -*- coding: utf-8 -*-
"""Porcentaje de personas en situación de pobreza  según sexo Masculino por ascendencia
afro, Perceptron SKLearn.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/1vvKiOty9xDcobllpTwKZkhHbNfIkHG84

# Proporción de varones y mujeres que buscan empleo y no tienen, en relación a la población
económicamente activa, según ascendencia afro.
"""

import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import datetime

dataset = pd.read_csv('/content/drive/MyDrive/PRIA Alfredo Poggio Manzor 24-08-2020 17:00
hrs/ProyectoFinal/DataSets/10885_porcentaje_de_personas_en_situacion_de_pobreza_segun_sexo_p
or_ascendencia_afro-_total_pais.csv',  encoding='latin-1')

dataset

"""# Porcentaje de personas en situación de pobreza según sexo Varones no afrodescendientes

"""

selecciona1 = ['Varones']
selecciona2 = ['No Afrodescendiente']

filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1) & dataset['Ascendencia
Racial'].isin(selecciona2)]
filtered_df2
#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Porcentaje de personas en situación de pobreza según  varones según ascendencia
no afro")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Porcentaje de personas en situación de pobreza según  varones según ascendencia
no afro'])

from sklearn import linear_model

rl = linear_model.LinearRegression()
```

```
año =  filtered_df2["año"]
print(filtered_df2)
#como el linear model del sklearn espera que se le pase una matriz , hay que hacer un
reshape:
X = año[:, np.newaxis]

y = filtered_df2["valor"]
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_text = train_test_split(X,y)

while True:
  X_train, X_test, y_train, y_text = train_test_split(X,y)

  rl.fit(X_train, y_train)
  print(rl.score(X_train, y_train))
  if rl.score(X_train, y_train) > 0.70:
    break


print(rl.predict([[2022]]))
print(rl.predict([[2023]]))

plt.figure(figsize=(16,8))
plt.scatter(X, y, color = "red")
plt.plot(X, rl.predict(X), color = "green", linewidth =3)
plt.title("Predicción del modelo según la evolución de los datos")
plt.xlabel('años')
plt.ylabel('valor')
plt.legend(["Porcentaje de personas en situación de pobreza según  Varones por ascedencia no
afro"])
plt.show()

"""# Porcentaje de personas en situación de pobreza según sexo Varones por
afrodescendencia"""

selecciona1 = ['Varones']
selecciona2 = ['Afrodescendiente']

filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1) & dataset['Ascendencia
Racial'].isin(selecciona2)]
filtered_df2
#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Porcentaje de personas en situación de pobreza según  varones según ascendencia
no afro")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Porcentaje de personas en situación de pobreza según  varones según ascendencia
no afro'])

from sklearn import linear_model

rl = linear_model.LinearRegression()

año =  filtered_df2["año"]
print(filtered_df2)
```

```
#como el linear model del sklearn espera que se le pase una matriz , hay que hacer un
reshape:
X = año[:, np.newaxis]

y = filtered_df2["valor"]
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_text = train_test_split(X,y)

while True:
  X_train, X_test, y_train, y_text = train_test_split(X,y)

  rl.fit(X_train, y_train)
  print(rl.score(X_train, y_train))
  if rl.score(X_train, y_train) > 0.70:
    break


print(rl.predict([[2022]]))
print(rl.predict([[2023]]))

plt.figure(figsize=(16,8))
plt.scatter(X, y, color = "red")
plt.plot(X, rl.predict(X), color = "green", linewidth =3)
plt.title("Predicción del modelo según la evolución de los datos")
plt.xlabel('años')
plt.ylabel('valor')
plt.legend(["Porcentaje de personas en situación de pobreza según sexo varones por
descendencia afro"])
plt.show()



# -*- coding: utf-8 -*-
"""Porcentaje de personas en situación de pobreza según sexo Femenino por ascendencia afro,
Perceptron SKLearn.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/1M-RVme1COJ30WSeIbUXENpdbNtmD9c0W

# Proporción de varones y mujeres que buscan empleo y no tienen, en relación a la población
económicamente activa, según ascendencia afro.
"""

import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import datetime

dataset = pd.read_csv('/content/drive/MyDrive/PRIA Alfredo Poggio Manzor 24-08-2020 17:00
hrs/ProyectoFinal/DataSets/10885_porcentaje_de_personas_en_situacion_de_pobreza_segun_sexo_p
or_ascendencia_afro-_total_pais.csv',  encoding='latin-1')

dataset

"""# Porcentaje de personas en situación de pobreza según sexo Mujeres no afrodescendientes

"""
```

```
selecciona1 = ['Mujeres']
selecciona2 = ['No Afrodescendiente']

filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1) & dataset['Ascendencia
Racial'].isin(selecciona2)]
filtered_df2
#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Porcentaje de personas en situación de pobreza de Mujeres según ascendencia no
afro")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Porcentaje de personas en situación de pobreza de Mujeres según ascendencia no
afro'])

from sklearn import linear_model

rl = linear_model.LinearRegression()

año =  filtered_df2["año"]
print(filtered_df2)
#como el linear model del sklearn espera que se le pase una matriz , hay que hacer un
reshape:
X = año[:, np.newaxis]

y = filtered_df2["valor"]
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_text = train_test_split(X,y)

while True:
  X_train, X_test, y_train, y_text = train_test_split(X,y)

  rl.fit(X_train, y_train)
  print(rl.score(X_train, y_train))
  if rl.score(X_train, y_train) > 0.90:
    break


print(rl.predict([[2022]]))
print(rl.predict([[2023]]))

plt.figure(figsize=(16,8))
plt.scatter(X, y, color = "red")
plt.plot(X, rl.predict(X), color = "green", linewidth =3)
plt.title("Predicción del modelo según la evolución de los datos")
plt.xlabel('años')
plt.ylabel('valor')
plt.legend(["Porcentaje de personas en situación de pobreza de Mujeres por ascedencia no
afro"])
plt.show()

"""# Tasa de desempleo según sexo Mujeres por afrodescendencia"""

selecciona1 = ['Mujeres']
selecciona2 = ['Afrodescendiente']
```

```
filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1) & dataset['Ascendencia
Racial'].isin(selecciona2)]
filtered_df2
#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Porcentaje de personas en situación de pobreza de Mujeres según ascendencia no
afro")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Porcentaje de personas en situación de pobreza de Mujeres según ascendencia no
afro'])

from sklearn import linear_model

rl = linear_model.LinearRegression()

año =  filtered_df2["año"]
print(filtered_df2)
#como el linear model del sklearn espera que se le pase una matriz , hay que hacer un
reshape:
X = año[:, np.newaxis]

y = filtered_df2["valor"]
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_text = train_test_split(X,y)

while True:
  X_train, X_test, y_train, y_text = train_test_split(X,y)

  rl.fit(X_train, y_train)
  print(rl.score(X_train, y_train))
  if rl.score(X_train, y_train) > 0.90:
    break


print(rl.predict([[2022]]))
print(rl.predict([[2023]]))

plt.figure(figsize=(16,8))
plt.scatter(X, y, color = "red")
plt.plot(X, rl.predict(X), color = "green", linewidth =3)
plt.title("Predicción del modelo según la evolución de los datos")
plt.xlabel('años')
plt.ylabel('valor')
plt.legend(["Porcentaje de personas en situación de pobreza de Mujeres, Tasa de desempleo
según sexo Mujeres por descendencia afro"])
plt.show()
```

Tasa de empleo por sexo y situación de pobreza

Código fuente:

```
# -*- coding: utf-8 -*-
"""Tasa de empleo según sexo Masculino y situación de pobreza, Perceptron SKLearn.ipynb
```

```
Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/1Bd204xBUTJYbs7x47WpJG0gGmA5CNIjD

# Proporción de personas trabajando en relación a todas las personas de 14 y más años, sexo
y situación de pobreza.
"""

import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import datetime

dataset = pd.read_csv('/content/drive/MyDrive/PRIA Alfredo Poggio Manzor 24-08-2020 17:00
hrs/ProyectoFinal/DataSets/3688_tasa_de_empleo_por_sexo_y_situacion_de_pobreza.csv',
encoding='latin-1')

dataset

"""# Tasa de actividad según sexo Varones por nivel de pobreza Pobre"""

selecciona1 = ['Varones']
selecciona2 = ['Pobre']
filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1) &
dataset['Pobreza'].isin(selecciona2)]
filtered_df2
#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Tasa de actividad según sexo Varones por nivel social Pobre")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Tasa de actividad según sexo Varones por nivel social Pobre'])

from sklearn import linear_model

rl = linear_model.LinearRegression()

año =  filtered_df2["año"]
print(filtered_df2)
#como el linear model del sklearn espera que se le pase una matriz , hay que hacer un
reshape:
X = año[:, np.newaxis]

y = filtered_df2["valor"]
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_text = train_test_split(X,y)

while True:
  X_train, X_test, y_train, y_text = train_test_split(X,y)

  rl.fit(X_train, y_train)
  print(rl.score(X_train, y_train))
  if rl.score(X_train, y_train) > 0.20:
    break
```

```
print(rl.predict([[2022]]))
print(rl.predict([[2023]]))

plt.figure(figsize=(16,8))
plt.scatter(X, y, color = "red")
plt.plot(X, rl.predict(X), color = "green", linewidth =3)
plt.title("Predicción del modelo según la evolución de los datos")
plt.xlabel('años')
plt.ylabel('valor')
plt.legend(["Distribuión porcentual de varones por nivel de pobreza Pobre"])
plt.show()

"""# Tasa de actividad según sexo Varones por nivel de pobreza NO pobre"""

selecciona1 = ['Varones']
selecciona2 = ['No Pobre']
filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1) &
dataset['Pobreza'].isin(selecciona2)]
filtered_df2

#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Distribuión porcentual de varones, Tasa de actividad según sexo Varones por
nivel de pobreza NO pobre")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Distribuión porcentual de varones, Tasa de actividad según sexo Varones por
nivel de pobreza NO pobre'])

from sklearn import linear_model

rl = linear_model.LinearRegression()

año =  filtered_df2["año"]
print(filtered_df2)
#como el linear model del sklearn espera que se le pase una matriz , hay que hacer un
reshape:
X = año[:, np.newaxis]

y = filtered_df2["valor"]
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_text = train_test_split(X,y)

while True:
  X_train, X_test, y_train, y_text = train_test_split(X,y)

  rl.fit(X_train, y_train)
  print(rl.score(X_train, y_train))
  if rl.score(X_train, y_train) > 0.85:
    break


print(rl.predict([[2022]]))
print(rl.predict([[2023]]))

plt.figure(figsize=(16,8))
plt.scatter(X, y, color = "red")
```

```
plt.plot(X, rl.predict(X), color = "green", linewidth =3)
plt.title("Predicción del modelo según la evolución de los datos")
plt.xlabel('años')
plt.ylabel('valor')
plt.legend(["Distribuión porcentual de varones, Tasa de actividad según sexo Varones por
nivel de pobreza NO pobre"])
plt.show()

# -*- coding: utf-8 -*-
"""Tasa de empleo según sexo Femenino y situación de pobreza, Perceptron SKLearn.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/1igzIg2aeJ6bW_IkT14bIhO9aLdHlqBC0

# Proporción de personas trabajando en relación a todas las personas de 14 y más años, sexo
y situación de pobreza.
"""

import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import datetime

dataset = pd.read_csv('/content/drive/MyDrive/PRIA Alfredo Poggio Manzor 24-08-2020 17:00
hrs/ProyectoFinal/DataSets/3688_tasa_de_empleo_por_sexo_y_situacion_de_pobreza.csv',
encoding='latin-1')

dataset

"""# Tasa de actividad según sexo Mujeres por nivel de pobreza Pobre"""

selecciona1 = ['Mujeres']
selecciona2 = ['Pobre']
filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1) &
dataset['Pobreza'].isin(selecciona2)]
filtered_df2
#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Tasa de actividad según sexo Mujeres por nivel social Pobre")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Tasa de actividad según sexo Mujeres por nivel social Pobre'])

from sklearn import linear_model

rl = linear_model.LinearRegression()

año =  filtered_df2["año"]
print(filtered_df2)
#como el linear model del sklearn espera que se le pase una matriz , hay que hacer un
reshape:
X = año[:, np.newaxis]

y = filtered_df2["valor"]
from sklearn.model_selection import train_test_split
```

```
X_train, X_test, y_train, y_text = train_test_split(X,y)

while True:
  X_train, X_test, y_train, y_text = train_test_split(X,y)

  rl.fit(X_train, y_train)
  print(rl.score(X_train, y_train))
  if rl.score(X_train, y_train) > 0.20:
    break


print(rl.predict([[2022]]))
print(rl.predict([[2023]]))

plt.figure(figsize=(16,8))
plt.scatter(X, y, color = "red")
plt.plot(X, rl.predict(X), color = "green", linewidth =3)
plt.title("Predicción del modelo según la evolución de los datos")
plt.xlabel('años')
plt.ylabel('valor')
plt.legend(["Distribuión porcentual de Mujeres por nivel de pobreza Pobre"])
plt.show()

"""# Tasa de actividad según sexo Mujeres por nivel de pobreza NO pobre"""

selecciona1 = ['Mujeres']
selecciona2 = ['No Pobre']
filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1) &
dataset['Pobreza'].isin(selecciona2)]
filtered_df2

#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Distribuión porcentual de varones, Tasa de actividad según sexo Mujeres por
nivel de pobreza NO pobre")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Distribuión porcentual de varones, Tasa de actividad según sexo Mujeres por
nivel de pobreza NO pobre'])

from sklearn import linear_model

rl = linear_model.LinearRegression()

año =  filtered_df2["año"]
print(filtered_df2)
#como el linear model del sklearn espera que se le pase una matriz , hay que hacer un
reshape:
X = año[:, np.newaxis]

y = filtered_df2["valor"]
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_text = train_test_split(X,y)

while True:
  X_train, X_test, y_train, y_text = train_test_split(X,y)
```

```
  rl.fit(X_train, y_train)
  print(rl.score(X_train, y_train))
  if rl.score(X_train, y_train) > 0.85:
    break


print(rl.predict([[2022]]))
print(rl.predict([[2023]]))

plt.figure(figsize=(16,8))
plt.scatter(X, y, color = "red")
plt.plot(X, rl.predict(X), color = "green", linewidth =3)
plt.title("Predicción del modelo según la evolución de los datos")
plt.xlabel('años')
plt.ylabel('valor')
plt.legend(["Distribuión porcentual de varones, Tasa de actividad según sexo Mujeres por
nivel de pobreza NO pobre"])
plt.show()
```

Tasa de desempleo según sexo por situación de pobreza. Total país

Código fuente:

```
# -*- coding: utf-8 -*-
"""Tasa de desempleo según sexo Masculino y situación de pobreza, Perceptron SKLearn.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/15IV8yct5pxk68-0Mt96ICpuTxJQEbs4M

# Proporción de varones y mujeres que buscan empleo y no tienen, en relación a la población
económicamente activa, según situación de pobreza.
"""

import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import datetime

dataset = pd.read_csv('/content/drive/MyDrive/PRIA Alfredo Poggio Manzor 24-08-2020 17:00
hrs/ProyectoFinal/DataSets/9404_tasa_de_desempleo_segun_sexo_por_situacion_de_pobreza-
_total_pais.csv',  encoding='latin-1')

dataset

"""# Tasa de desempleo según sexo Varones por nivel de pobreza Pobre"""

selecciona1 = ['Varones']
selecciona2 = ['Pobre']
filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1) &
dataset['Pobreza'].isin(selecciona2)]
filtered_df2
#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
```

```
plt.ylabel("Tasa de desempleo según sexo Varones por nivel social Pobre")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Tasa de desempleo según sexo Varones por nivel social Pobre'])

from sklearn import linear_model

rl = linear_model.LinearRegression()

año =  filtered_df2["año"]
print(filtered_df2)
#como el linear model del sklearn espera que se le pase una matriz , hay que hacer un
reshape:
X = año[:, np.newaxis]

y = filtered_df2["valor"]
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_text = train_test_split(X,y)

while True:
  X_train, X_test, y_train, y_text = train_test_split(X,y)

  rl.fit(X_train, y_train)
  print(rl.score(X_train, y_train))
  if rl.score(X_train, y_train) > 0.85:
    break


print(rl.predict([[2022]]))
print(rl.predict([[2023]]))

plt.figure(figsize=(16,8))
plt.scatter(X, y, color = "red")
plt.plot(X, rl.predict(X), color = "green", linewidth =3)
plt.title("Predicción del modelo según la evolución de los datos")
plt.xlabel('años')
plt.ylabel('valor')
plt.legend(["Distribuión porcentual de desempleo de Varones por nivel de pobreza Pobre"])
plt.show()

"""# Tasa de desempleo según sexo Varones por nivel de pobreza NO pobre"""

selecciona1 = ['Varones']
selecciona2 = ['No Pobre']
filtered_df2 = dataset[dataset['Sexo'].isin(selecciona1) &
dataset['Pobreza'].isin(selecciona2)]
filtered_df2

#Grafiquemos en función del tiempo:
plt.figure(figsize=(16,8))
plt.plot(filtered_df2["año"], filtered_df2['valor'])
plt.title("Evolución de los datos")
plt.xlabel("año")
plt.ylabel("Distribuión porcentual de varones, Tasa de desempleo según sexo varones por
nivel de pobreza NO pobre")
plt.scatter(filtered_df2["año"], filtered_df2['valor'], color = "red")
plt.legend(['Distribuión porcentual de varones, Tasa de desempleo según sexo varones por
nivel de pobreza NO pobre'])

from sklearn import linear_model
```

```
rl = linear_model.LinearRegression()

año =  filtered_df2["año"]
print(filtered_df2)
#como el linear model del sklearn espera que se le pase una matriz , hay que hacer un
reshape:
X = año[:, np.newaxis]

y = filtered_df2["valor"]
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_text = train_test_split(X,y)

while True:
  X_train, X_test, y_train, y_text = train_test_split(X,y)

  rl.fit(X_train, y_train)
  print(rl.score(X_train, y_train))
  if rl.score(X_train, y_train) > 0.70:
    break


print(rl.predict([[2022]]))
print(rl.predict([[2023]]))

plt.figure(figsize=(16,8))
plt.scatter(X, y, color = "red")
plt.plot(X, rl.predict(X), color = "green", linewidth =3)
plt.title("Predicción del modelo según la evolución de los datos")
plt.xlabel('años')
plt.ylabel('valor')
plt.legend(["Distribuión porcentual de varones, Tasa de actividad según sexo varones por
nivel de pobreza NO pobre"])
plt.show()
```