Inteligencia Artificial
Act 9: Programando Regresión Lineal en Python

1 Introducción

La regresión lineal es un algoritmo de aprendizaje supervisado utilizado para predecir datos futuros en un conjunto de datos continuos. En estadística, la regresión lineal describe la relación entre la variable dependiente y una o más variables independientes, estimando los coeficientes de la ecuación lineal.

La regresión lineal ajusta una línea recta (y = mx + b) que minimiza las discrepancias entre los valores de salida previstos y reales, esta línea recta indicará la tendencia de un conjunto de datos continuos. Para hacerlo, minimizará el coste de una función de error cuadrático donde los coeficientes van a corresponder con la recta óptima.

Se utiliza en el campo científico y en el aprendizaje automático, el cual consiste en predecir un parámetro (Y) a partir de un parámetro conocido (X) donde el modelo va a encontrar la mejor relación entre los parámetros para predecir datos futuros.

2 Metodología

Para llevar a cabo la actividad, se siguieron las indicaciones del libro *Aprende Machine Learning*, en la página 28. Primero, se creó una carpeta llamada "Regresión Lineal" y se descargó el archivo de entrada CSV. En la carpeta se guardó el archivo CSV y se creó el archivo que va a contener el código Python, para realizar el código se hizo uso de Visual Studio Code.

Empezamos importando las librerías necesarias:

```
# Imports necesarios
import numpy as np
import pandas as pd
import seaborn as sb
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from matplotlib import cm
plt.rcParams['figure.figsize'] = (16, 9)
plt.style.use('ggplot')
from sklearn import linear_model
from sklearn.metrics import mean_squared_error, r2_score
```

Nota: Puede ocurrir el caso que se necesiten isntalar varias librerías con pip, para eso basta con abrir la terminal y escribir *pip install nombre de la librería*, por ejemplo: *pip install pandas*.

Leemos el archivo de entrada CSV y lo cargamos como un dataset de pandas.

```
# cargamos los datos de entrada
data = pd.read_csv("./articulos_ml.csv")
```

Al cargar el archivo, vemos varias cosas como la cantidad de dimensiones y registros que contiene, los primeros registros y las estadísticas de los datos, para eso se imprime:

```
print(data.shape) # dimensiones y registros
print("\n", data.head()) # primeros registros
print("\n", data.describe()) # estadisticas de los datos
```

Listing 1: Exploración de los datos

Visualizamos rápidamente las características de entrada, para hacerlo se eliminan tres columnas ('Title', 'url, y 'Elapsed days') que no son relevantes para el análisis futuro y generamos un histograma para cada una de las columnas restantes para posteriormente visualizar las gráficas.

```
# visualizacion en histograma
data.drop(['Title','url', 'Elapsed days'],axis=1).hist()
plt.show()
```

Listing 2: Visualización de datos

Ahora, procedemos a filtrar los datos de cantidad de palabras para conservar los registros con menos de 3,500 palabras y de igual manera con los datos de cantidad de compartidos para conservar aquellos que tengan menos de 80,000. Además, visualizaremos los datos filtrados con un diagrama de dispersión, pintando en azul los puntos por debajo de la media y con naranja los puntos por encima.

```
# filtrar los datos
filtered_data = data[(data['Word count'] <= 3500) & (data['# Shares'] <=
    80000)]

# pintar en colores los puntos por debajo y por encima de la media
colores=['orange','blue']
tamanios=[30,60]

f1 = filtered_data['Word count'].values
f2 = filtered_data['# Shares'].values

asignar = []
for index, row in filtered_data.iterrows():
    if(row['Word count']>1808):
        asignar.append(colores[0])
    else:
        asignar.append(colores[1])

plt.scatter(f1, f2, c=asignar, s=tamanios[0])
plt.show()
```

Listing 3: Filtrado y visualización de datos

Una vez preparados los datos, podemos seguir con el entrenamiento del modelo, por el momento usaremos sólo Word Count y # Shares. Empezamos etiquetando nuestra data y creando el objeto de regresión lineal para al final entrenar el modelo y hacer predicciones.

```
# etiquetar los datos para entrenamiento
dataX = filtered_data[["Word count"]]
X_train = np.array(dataX)
y_train = filtered_data['# Shares'].values

regr = linear_model.LinearRegression() # objeto de regresion linear
regr.fit(X_train, y_train) # entrenar el modelo
y_pred = regr.predict(X_train) # predicciones
```

Al terminar de entrenar el modelo, imprimimos los coeficientes obtenidos, los términos independientes, el error cuadrado medio y el puntaje de varianza, esto para interpretar el modelo y poder evaluar el rendimiento de nuestro modelo de regresión lineal.

```
print('\nCoeficientes: ', regr.coef_)
print('Terminos independientes: ', regr.intercept_)
print("Error cuadrado medio: %.2f" % mean_squared_error(y_train, y_pred))
print("Puntaje de Varianza: %.2f" % r2_score(y_train, y_pred))
```

Listing 4: Resultados del modelo

Para terminar, vamos a probar el algoritmo que acabamos de crear, suponiendo que quisiéramos predecir cuántos “compartir” obtendrá un articulo sobre ML de 2000 palabras.

```
# Prediccion usando el algoritmo
y_Dosmil = regr.predict([[2000]])
print("Prediccion: ", int(y_Dosmil[0]))
```

Listing 5: Prediccion del algoritmo

3 Resultados

A continuación, se van a presentar los resultados obtenidos a partir del código mostrado.

3.1 Exploración de Datos

En esta parte (Listing 1) se realizó una exploración del dataset. Los resultados obtenidos fueron los siguientes:

- La dimensión y la cantidad de registros del dataset es:

```
(161, 8)
```

- Los primeros 5 registros del dataset son:

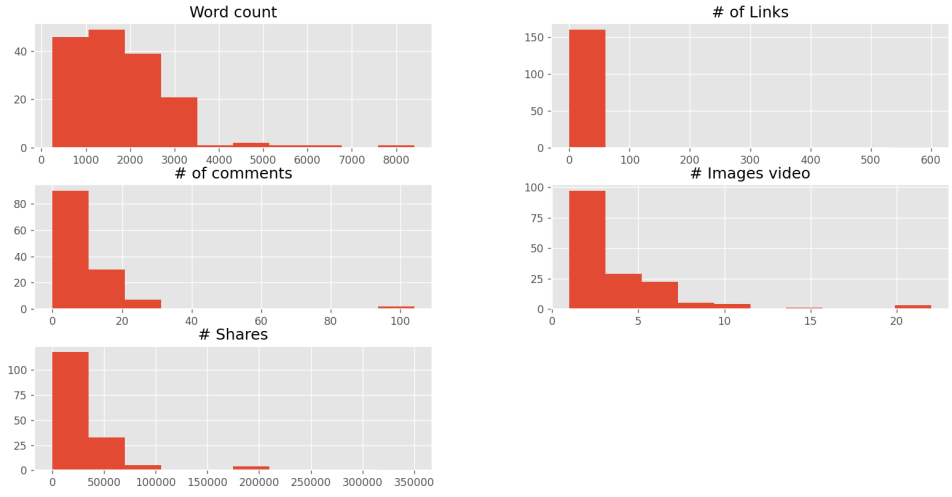
	Title	url	Word count	# of Links	# of comments	# Images video	Elapsed days	# Shares
0	What is Machine Learning and how do we use it ...	https://blog.signals.network/what-is-machine-l...	1888	1	2.0	2	34	200000
1	10 Companies Using Machine Learning in Cool Ways	NaN	1742	9	NaN	9	5	25000
2	How Artificial Intelligence Is Revolutionizing...	NaN	962	6	0.0	1	10	42000
3	Dbrain and the Blockchain of Artificial Intell...	NaN	1221	3	NaN	2	68	200000
4	Nasa finds entire solar system filled with eig...	NaN	2039	1	104.0	4	131	200000

- Las estadísticas de los datos son:

	Word count	# of Links	# of comments	# Images video	Elapsed days	# Shares
count	161.000000	161.000000	129.000000	161.000000	161.000000	161.000000
mean	1808.260870	9.739130	8.782946	3.670807	98.124224	27948.347826
std	1141.919385	47.271625	13.142822	3.418290	114.337535	43408.006839
min	250.000000	0.000000	0.000000	1.000000	1.000000	0.000000
25%	990.000000	3.000000	2.000000	1.000000	31.000000	2800.000000
50%	1674.000000	5.000000	6.000000	3.000000	62.000000	16458.000000
75%	2369.000000	7.000000	12.000000	5.000000	124.000000	35691.000000
max	8401.000000	600.000000	104.000000	22.000000	1002.000000	350000.000000

3.2 Visualización de Datos

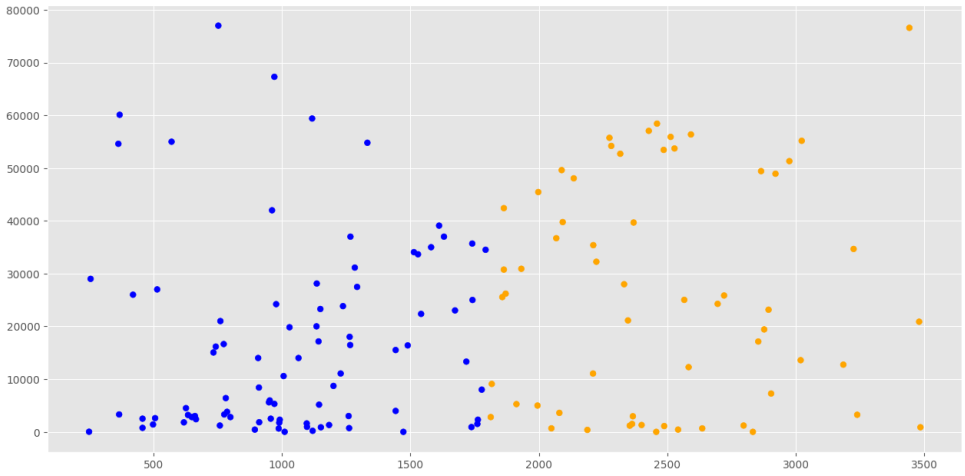
Los histogramas generados en el Listing 2 son los siguientes:



En las gráficas se puede apreciar entre qué valores se concentran la mayoría de registros.

3.3 Filtrado y visualización de Datos

El resultado de filtrar los datos según la cantidad de palabras y la cantidad de compartidos es el siguiente: (ver Listing 3)



3.4 Resultados del modelo

En el Listing 4 se evaluó el modelo, obteniendo los siguientes resultados:

```
Coeficientes:  [5.69765366]
Terminos independientes:  11200.30322307416
Error cuadrado medio: 372888728.34
Puntaje de Varianza: 0.06
```

Si nos basamos con la ecuación de la recta y = mX + b, "m" será 5.69 mientras que "b" será 11200. Viendo el error cuadrático medio, es un número muy grande, esto nos dice que nuestro modelo no será muy eficiente, de igual manera con la varianza, nuestra varianza debería acercarse a 1, pero obtuvimos un 0.06.

3.5 Predicción del algoritmo

Finalmente, para el Listing 5 se probo el algoritmo para predecir cuantos "Shares" se obtendrán para un artículo de 2000 palabras, el resultado fue el siguiente:

```
Predicción:  22595
```

Entonces, dado nuestro algoritmo, se obtendrán 22,595 Shares para un artículo de 2000 palabras.

4 Conclusión

En esta actividad, utilizamos un archivo CSV para entrenar un modelo de regresión lineal simple. Primero se leyó y se cargó el CSV a un dataframe de pandas para despues examinar un poco sus características y visualizar los datos en histogramas. Luego se filtraron los datos según la cantidad de palabras y la cantidad de compartidos para despues visualizar la distribución por medio de un diagrama de dispersión, que nos ayudo a ver los datos que estan por encima y por debajo de la media. Finalmente, el modelo fue entrenado y evluado con métricas de rendimiento, donde se logró notar que dado el error cuadrado medio, el modelo no será muy eficiente, sin embargo el algoritmo hecho nos proporciona una base para alguna mejora futura.