Regresión Lineal Múltiple

Valeria Ybarra López
Matrícula: 2047880

1 Introducción

Un modelo de regresión lineal múltiple es un modelo estadístico versátil que evalúa las relaciones entre un destino continuo y los predictores.

Los predictores(o variables de entrada) pueden ser campos continuos, categóricos o derivados, de modo que las relaciones no lineales también estén soportadas. Es considerado un modelo lineal porque consiste en términos de aditivos en los que cada término es un predictor que se multiplica por un coeficiente estimado.

El tener más de una variable de entrada ayuda a obtener predicciones más complejas. La "ecuación de la recta", ahora pasa a ser: $Y = b + m1X1 + m2X2 + ... + m(n)X(n)$ y deja de ser una recta.

2 Metodología

Para la realización de esta activdad, se siguieron las instrucciones proporcionadas en la página 34 del libro "Aprenda Machine Learning".

2.1 Continuación de la actividad 9: Regresión Lineal

Se creó una carpeta con el nombre de "Regresión Lineal Múltiple" en donde se guardó el archivo .csv de entrada proporcionado anteriormente por el libro para poder realizar el código en python, en esa misma carpeta se creó un arvhico .py para realizar la actividad.

2.2 Código

Utilizando el mismo código de la actividad 9: Regresión Lineal, le agregaremos 2 "variables predicativas o de entrada" para poder realizar mejores predicciones y también el tener 2 variables nos permite gráficar en 3D.

La primera variable seguira siendo la **cantidad de palabras** y la segunda variable **la suma de 3 columnas de entrada**:

```
suma = (filtered_data["# of Links"] + filtered_data['# of comments'].fillna(0) +
dataX2 = pd.DataFrame()
dataX2["Word count"] = filtered_data["Word count"]
dataX2["suma"] = suma
XY_train = np.array(dataX2)
z_train = filtered_data['# Shares'].values
```

Tenemos las 2 variables de entrada en XY_train y nuestra variable de salida pasa de ser "Y" a ser el eje "Z", de esta manera se podra representar los datos en un gráfico 3D.

Creamos una instancia del modelo LinearRegression con SKLearn, pero esta véz tendrá dos dimensiones que entrenar: las que contiene XY_train. Utilizando el método ".fit()", entrenamos el modelo (XY_train como variables de entrada y z_train como variable objetivo). Imprimimos los coeficientes los cuales indicarán como fluyen las variables de entrada en la variable objetivo. (en este caso serán dos coeficientes,ya que hay dos dimensiones en los datos de entrada)y puntajes obtenidos cómo el error cuadrático medio calcula cuánto se desvían las predicciones (z_pred) de los valores reales (z_train); y la varianza que indica qué tan bien las variables de entrada explican la dispersión de la variable objetivo:

```
# Creamos un nuevo objeto de Regresi n Lineal
regr2 = linear_model.LinearRegression()
# Entrenamos el modelo, esta vez, con 2 dimensiones
# obtendremos 2 coeficientes, para graficar un plano
regr2.fit(XY_train, z_train)
 # Hacemos la predicci n con la que tendremos puntos sobre el plano hallado
z_pred = regr2.predict(XY_train)
# Los coeficientes
print('Coefficients: \n', regr2.coef_)
# Error cuadr tico medio
print("Mean squared error: %.2f" % mean_squared_error(z_train, z_pred))
# Evaluamos el puntaje de varianza (siendo 1.0 el mejor posible)
print('Variance score: %.2f' % r2_score(z_train, z_pred))
```

Visualizar un plano en 3 Dimensiones en Python

Se graficarán en 3D los puntos originales de las características de entrada en azul y los puntos proyectados en el plano en rojo. El eje Z de la gráfica representa la "altura," que es la cantidad de Shares.

En el siguiente código se crea una figura y un objeto 3D para visualizar los datos, después creamos una malla rectangular, definida por valores de entrada en el eje X (cantidad de palabras) y en el eje Y (cantidad de enlaces, comentarios e imágenes). Con los coeficientes de la regresión lineal múltiple, se calcula el valor correspondiente en Z (cantidad de Shares), este valor incluye la suma de las contribuciones de los ejes X e Y, además del punto de intercepción. Sobre la malla, se grafica una superficie semi-transparente para representar el plano de la regresión. Se dibujan los puntos originales en color azul y los puntos proyectados sobre el plano de regresión en color rojo. Por último, se ajusta la vista de la gráfica y añadimos las etiquetas y el título para facilitar la visualización de los datos.

```
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
# Creamos una malla, sobre la cual graficaremos el plano
xx, yy = np.meshgrid(np.linspace(0, 3500, num=10), np.linspace(0, 60, num=10))

# calculamos los valores del plano para los puntos x e y
nuevoX = (regr2.coef_[0] * xx)
nuevoY = (regr2.coef_[1] * yy)
```

```
# calculamos los valores para z. Debemos sumar el punto de intercepion
z = (nuevoX + nuevoY + regr2.intercept_)
# Graficamos el plano
ax.plot_surface(xx, yy, z, alpha=0.2, cmap='hot')
# Graficamos en azul los puntos en 3D
ax.scatter(XY_train[:, 0], XY_train[:, 1], z_train, c='blue',s=30)
 # Graficamos en rojo
ax.scatter(XY_train[:, 0], XY_train[:, 1], z_pred, c='red',s=40)

# con esto situamos la "camara" con la que visualizamos
ax.view_init(elev=30., azim=65)
ax.set_xlabel('Cantidad de Palabras')
ax.set_ylabel('Cantidad de Enlaces,Comentarios e Imagenes')
ax.set_zlabel('Compartido en Redes')
ax.set_title('Regresi n Lineal con Multiples Variables')
```

Predicción con el modeo de Múltiples Variables

El modelo predictivo toma como entrada los valores de las características y realiza una estimación sumando las cantidades relevantes (en este caso, enlaces, comentarios e imágenes) y realiza la predicción con el método ".predict()" . El código predice cuantos "Shares" obtendría un artículo con 2000 palabras, 10 enlaces, 4 comentarios y 6 imagenes.

```
z_Dosmil = regr2.predict([[2000, 10+4+6]])
print('Prediccion:',int(z_Dosmil[0]))
```

3 Resultados

Coeficientes, error y varianza:

```
Coefficients:
[   6.63216324 -483.40753769]
Mean squared error: 352122816.48
Variance score: 0.11
```

Figure 1: Coeficientes y puntajes obtenidos del modelo.

Grafica 3D:

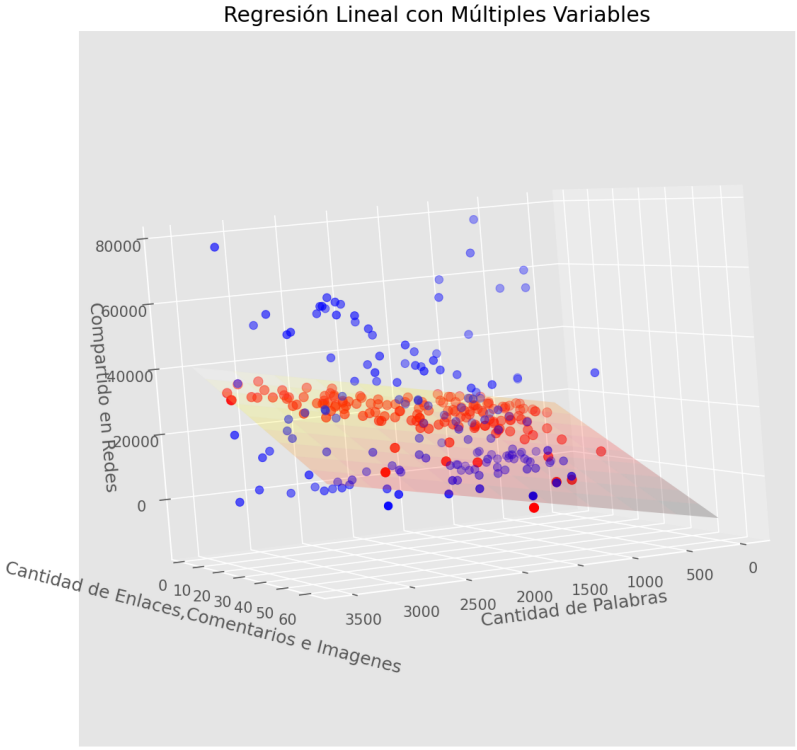

Figure 2: La gráfica permite visualizar cómo el modelo ajusta los datos reales (puntos azules) al plano de regresión (puntos rojos) y muestra las tendencias aprendidas en el espacio 3D. Si los puntos azules y rojos están cerca, el modelo tiene un buen ajuste.

Predicción utilizando el modelo de regresión lineal múltiple:

Figure 3: Esta predicción nos da 20518, un poco mejor que nuestra predicción anterior en la cual solo se usaba 1 variable.

4 Conclusión

Utilizamos SKLearn en Python para construir modelos de regresión lineal con 1 o múltiples variables, aunque las predicciones obtenidas no fueron precisas debido al alto margen de error. Este ejercicio permite ver el funcionamiento de un modelo con múltiples variables, y nos deja a entender que para obtener un modelo más preciso, debemos de utilizar más dimensiones y encontrar mejores datos de entrada.

5 Referencias

Bagnato, J. (2020). Aprende Machine Learning en Español.

IBM.(2024). Regresión Lineal Múltiple. https://www.ibm.com/docs/es/cognos-analytics/11.1.0?topic=test multiple-linear-regression