Regresion Lineal Múltiple

Adrián Esteban Morales Rodriguez

April 1, 2025

1 Introducción

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

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

La regresión lineal se utiliza para generar conocimientos para los gráficos que contienen al menos dos campos continuos con uno identificado como el destino y el otro como un predictor. Además, se puede especificar un predictor categórico y dos campos continuos auxiliares en un gráfico y se pueden utilizar para generar un modelo de regresión adecuado.

2 Metodología

Esta actividad consta ahora de obtener la misma predicción que en la regresion lineal simple pero ahora agregando mas valores de entrada. Tendremos dos valores de entrada, el primero siendo la cantidad de palabras y el segundo siendo una suma de las columnas: cantidad de enlaces, comentarios y cantidad de imagenes.

- Paso 1

  Agregar una nueva variable que sera la suma de las tres columnas enlaces, comentarios e imagenes para asi tener una dimensión más:

```
suma = (filtered_data["# of Links"] + filtered_data['# of comments'].fillna(0) + filtered_data['# Images video'])

dataX2 =  pd.DataFrame()
dataX2["Word count"] = filtered_data["Word count"]
dataX2["suma"] = suma
XY_train = np.array(dataX2)
z_train = filtered_data['# Shares'].values
```

✓ 0.0s

+ Code + Markdown

- Paso 2

  Crear un nuevo objeto de regresion lineal pero el cual ahora tendra dos dimensiones que entrenar.

```
    regr2 = linear_model.LinearRegression()

    regr2.fit(XY_train, z_train)

    z_pred = regr2.predict(XY_train)

    print('Coefficients: \n', regr2.coef_)

    print("Mean squared error: %.2f" % mean_squared_error(z_train, z_pred))

    print('Variance score: %.2f' % r2_score(z_train, z_pred))
```

✓ 0.0s

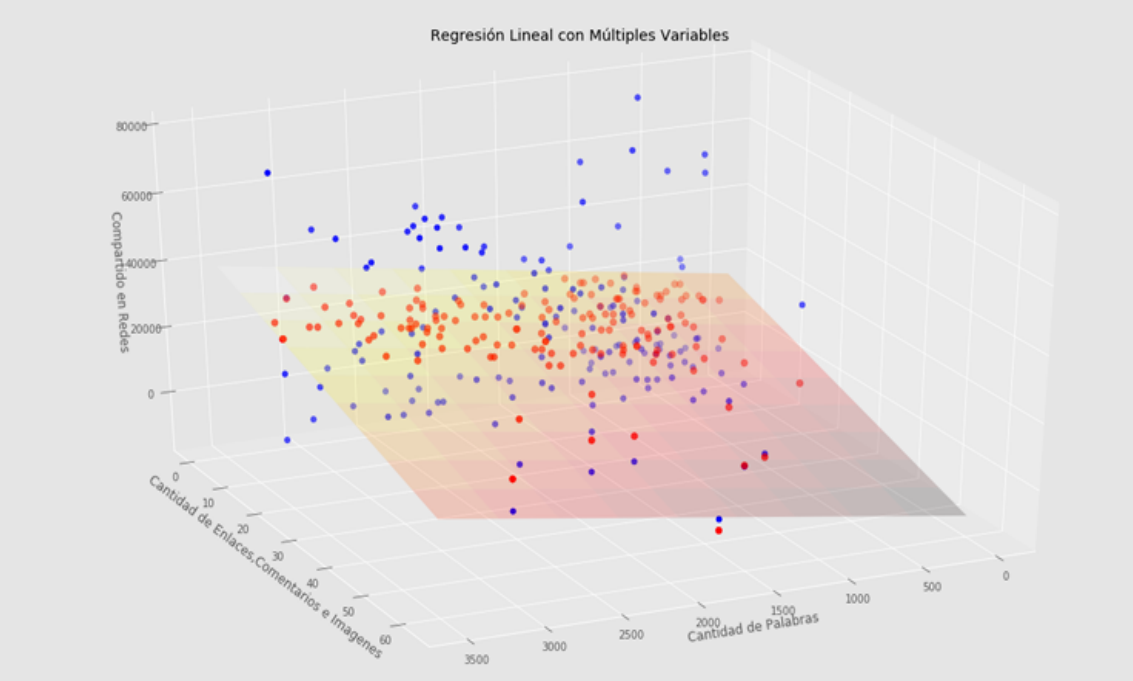
```
Coefficients:
 [   6.63216324 -483.40753769]
Mean squared error: 352122816.48
Variance score: 0.11
```

- Paso 3

  Visualizar el gráfico en 3D

```
fig = plt.figure()
ax = Axes3D(fig)

xx, yy = np.meshgrid(np.linspace(0, 3500, num=10), np.linspace(0, 60, num=10))

nuevoX = (regr2.coef_[0] * xx)
nuevoY = (regr2.coef_[1] * yy)

z = (nuevoX + nuevoY + regr2.intercept_)

ax.plot_surface(xx, yy, z, alpha=0.2, cmap='hot')

ax.scatter(XY_train[:, 0], XY_train[:, 1], z_train, c='blue',s=30)

ax.scatter(XY_train[:, 0], XY_train[:, 1], z_pred, c='red',s=40)

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('Regresión Lineal con Múltiples Variables')
```

✓ 0.0s

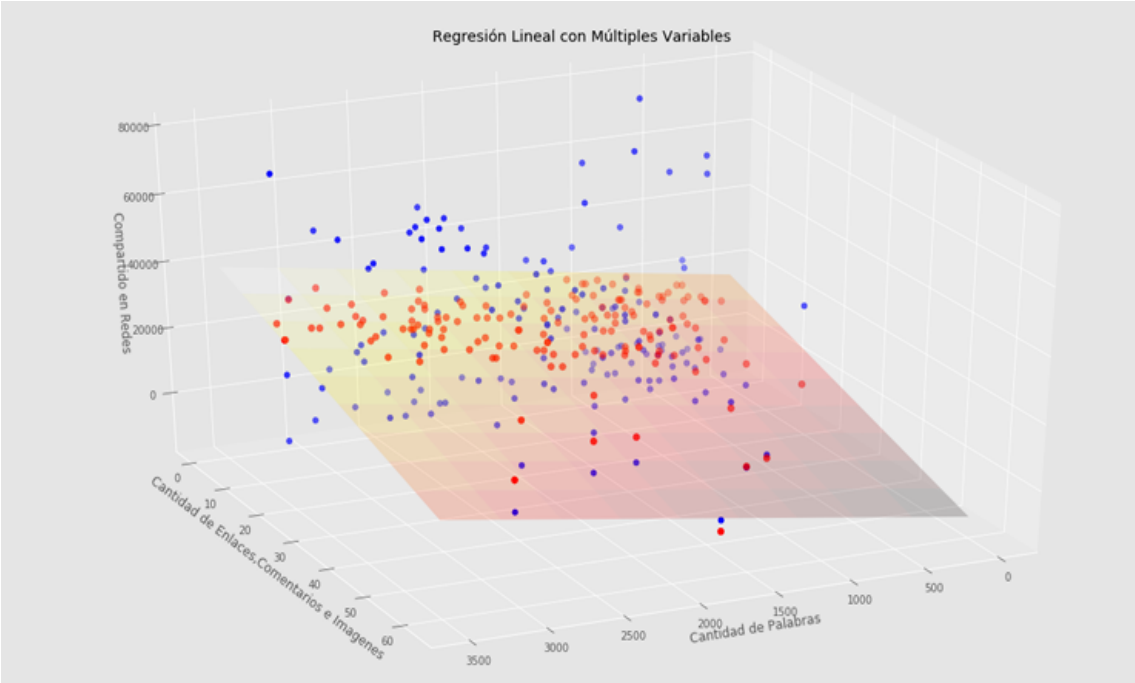


- Paso 4

  Filtrar los datos de cantidad de palabras para quedarnos con los registros

con menos de 3,500 palabras y también con los que tengan Cantidad de compartidos menos a 80,000. Pintando en azul los puntos con menos de 1808 palabras (la media) y en naranja los que tengan más.

```
fig = plt.figure()
ax = Axes3D(fig)

xx, yy = np.meshgrid(np.linspace(0, 3500, num=10), np.linspace(0, 60, num=10))

nuevoX = (regr2.coef_[0] * xx)
nuevoY = (regr2.coef_[1] * yy)

z = (nuevoX + nuevoY + regr2.intercept_)

ax.plot_surface(xx, yy, z, alpha=0.2, cmap='hot')

ax.scatter(XY_train[:, 0], XY_train[:, 1], z_train, c='blue',s=30)

ax.scatter(XY_train[:, 0], XY_train[:, 1], z_pred, c='red',s=40)

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('Regresión Lineal con Múltiples Variables')


✓ 0.0s
```



3 Resultados

Finalmente obtuve la siguiente predicción de la posible cantidad de compartidos en redes sociales con un articulo de 2000 palabras, 10 enlaces, 4 comentarios y 6 imagenes.

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

✓ 0.0s

```
20518
C:\Users\adria\AppData\Local\Temp\ipykernel_24
```

4 Conclusión

Tras la realización de esta actividad solo tuve algunos incovennientes con la creación del grafico en 3d el cual no se podía visualizar. Viendo los resultados y comparando el método de regresión lineal simple y multiple, veo que son menos share en la multiple pero siento que es un resultado mas real ya que toma mas parámetros y eso influye mucho con los resultados.