UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

Facultad de Estudios Superiores
Acatlán

Practica 2:

Regresión lineal múltiple y Regresión polinómica.

Aprendizaje de maquina

Integrantes:

Gustavo Adolfo Alvarez Hernández

Profesor:

Eduardo Eloy Loza Pacheco

FES Acatlán, 12 de septiembre 2022

Objetivo

Aprender a discernir entre las variables que son significativas en el modelo de regresión lineal múltiple por medio del p-value y otros estadísticos relevantes. Además de identificar los casos donde el modelo lineal no se ajusta y si lo hace un modelo polinomial.

Materiales

- Google colab: Utilizando Sklearn, pandas y matplotlib.

Contexto

Suponga que se tiene una variable explicada y (lo que se busca predecir) y un conjunto de al menos una variable cuantitativa $x_i$que explica el comportamiento de y, entonces se plantea el modelo de regresión lineal múltiple (MLR model):

$$Y_i = x_{i,1}\beta_1 + \cdots + x_{i,p}\beta_p + e_i = X_i^t\beta + e_i$$

Siendo e_i el error de nuestra predicción, entonces podemos expresar el modelo de la forma matricial $Y = X\beta + e$, donde para n muestras y $p - 1$ variables explicativas se tiene:

$$\begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_n \end{bmatrix} = \begin{bmatrix} x_{1,1} & x_{1,2} & \dots & x_{1,p} \\ x_{2,1} & x_{2,2} & \dots & x_{2,p} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n,1} & x_{n,2} & \dots & x_{n,p} \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_p \end{bmatrix} + \begin{bmatrix} e_1 \\ e_2 \\ \vdots \\ e_n \end{bmatrix}.$$

Donde la primera columna de X se constituye de 1's y si se cumplen los siguientes supuestos:

1. Los errores son independientes e idénticamente distribuidos.
2. Los errores no dependen de las variables explicativas
3. Los errores se distribuyen como $N(0, \sigma^2)$

Entonces se puede estimar los coeficientes del MLR por medio de *Ordinal Least Squares* (OLS), que consiste en minimizar la suma de los errores al cuadrado y dando por estimación lo siguiente:

$$Q_{OLS}(\mathbf{b}) = \sum_{i=1}^{n} r_i^2(\mathbf{b}),$$

$$\text{and } \hat{\beta}_{OLS} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y}.$$

Ahora en el caso de que nuestros datos no tengan un comportamiento lineal, podemos darle un enfoque polinomial, el cual solo consiste en generar variables elevando hasta el grado deseado n y buscando el coeficiente adecuado por el método ya explicado. Esto se ve como:

$$Y = \beta_0 + \beta_1 x$$

$$Y = \beta_0 + \beta_1 x + \beta_2 x^2$$

Complicando el modelo, pero generando un mejor ajuste a los datos.

Desarrollo

1. Cargamos bibliotecas necesarias

```
import numpy as np
import pandas as pd
from sklearn.preprocessing import LabelEncoder,OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
import statsmodels.api as sm
from statsmodels.sandbox.regression.predstd import wls_prediction_std
np.set_printoptions(formatter={'float_kind':"{:.2f}".format})
```

2. Leemos los datos y separamos en variables explicativas y la variable explicada

```
df = pd.read_csv("/content/50_Startups.csv")
X= df.iloc[ : ,:-1].values
y= df.iloc[:,4].values
df.head()
```

	R&D Spend	Administration	Marketing Spend	State	Profit
0	165349.20	136897.80	471784.10	New York	192261.83
1	162597.70	151377.59	443898.53	California	191792.06
2	153441.51	101145.55	407934.54	Florida	191050.39

3. Al tener variables que no son numéricas, debemos convertir State en números, por lo cual haremos variables dummies.

Preprocesamiento

```
[10] ct= ColumnTransformer([('State',OneHotEncoder(categories='auto'),[3])], remainder='passthrough')
     X= ct.fit_transform(X).astype(float)

[11] X = X[:,1:]

[12] X

array([[0.00, 1.00, 165349.20, 136897.80, 471784.10],
       [0.00, 0.00, 162597.70, 151377.59, 443898.53],
       [1.00, 0.00, 153441.51, 101145.55, 407934.54],
       [0.00, 1.00, 144372.41, 118671.85, 383199.62],
       [1.00, 0.00, 142107.34, 91391.77, 366168.42],
```

4. Dividimos los datos en entrenamiento y prueba para ver si generaliza nuestro modelo.

```
Train-test

[ ]  x_train,x_test,y_train,y_test = train_test_split(X,y,test_size=.2,random_state=0)
```

5. Entrenamos un modelo de regresión con todas las variables.

```
[15] regresion= LinearRegression()
     regresion.fit(x_train,y_train)

     LinearRegression()

[16] y_pred= regresion.predict(x_test)
```

6. Después validamos el impacto de las variables por medio de su p-value que no debe superar el umbral de 0.05 que nosotros definimos, todo esto usando OLS.

```
regresion_OLS = sm.OLS(endog=y,exog=X_opt).fit()
regresion_OLS.summary()
```

OLS Regression Results

Dep. Variable:	y	**R-squared:**	0.951
Model:	OLS	**Adj. R-squared:**	0.945
Method:	Least Squares	**F-statistic:**	169.9
Date:	Mon, 12 Sep 2022	**Prob (F-statistic):**	1.34e-27
Time:	23:42:08	**Log-Likelihood:**	-525.38
No. Observations:	50	**AIC:**	1063.
Df Residuals:	44	**BIC:**	1074.
Df Model:	5		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	5.013e+04	6884.820	7.281	0.000	3.62e+04	6.4e+04
x1	198.7888	3371.007	0.059	0.953	-6595.030	6992.607
x2	-41.8870	3256.039	-0.013	0.990	-6604.003	6520.229
x3	0.8060	0.046	17.369	0.000	0.712	0.900
x4	-0.0270	0.052	-0.517	0.608	-0.132	0.078
x5	0.0270	0.017	1.574	0.123	-0.008	0.062

   a. En este caso se descartaría X2 al ser el que tiene el valor más alto

7. Seguimos descartando variables siguiendo el criterio de si superan el umbral, se retiran y se vuelve a calcular, por lo cual al final de varias iteraciones tenemos que:

```
X_opt=X[:,[0,3,5]].tolist()
regresion_OLS=sm.OLS(endog=y,exog=X_opt).fit()
regresion_OLS.summary()
```

OLS Regression Results

Dep. Variable:	y	**R-squared:**	0.950
Model:	OLS	**Adj. R-squared:**	0.948
Method:	Least Squares	**F-statistic:**	450.8
Date:	Mon, 12 Sep 2022	**Prob (F-statistic):**	2.16e-31
Time:	17:07:24	**Log-Likelihood:**	-525.54
No. Observations:	50	**AIC:**	1057.
Df Residuals:	47	**BIC:**	1063.
Df Model:	2		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	4.698e+04	2689.933	17.464	0.000	4.16e+04	5.24e+04
x1	0.7966	0.041	19.266	0.000	0.713	0.880
x2	0.0299	0.016	1.927	0.060	-0.001	0.061

a. Resultando en que solo conservamos el termino independiente, el “R&D Spend” y “Marketing Spend”; puesto que son las variables que mas influyen en nuestro modelo.

8. Prosiguiendo con la práctica, leemos los datos de salarios que contiene el puesto y su salario.

Datos

```
[ ]  df = pd.read_csv("/content/Salarios.csv")
     X= df.iloc[ : ,1:2].values
     y= df.iloc[:,2].values
```

9. Generamos un modelo lineal y 3 multinomiales, estos últimos solo generando una columna extra por cada grado que se añada.

Regresion lineal simple

```
[ ]  lin_reg= LinearRegression()
     lin_reg.fit(X,y)
```

```
LinearRegression()
```

Regresiòn polinomica

Cuadràtico

```
[ ]  poly_reg= PolynomialFeatures(degree=2)
     X_poly= poly_reg.fit_transform(X)
     lin_reg2 = LinearRegression()
     lin_reg2.fit(X_poly,y)
```

```
LinearRegression()
```

▾ Cubico

```
[ ]  poly_reg3= PolynomialFeatures(degree=3)
     X_poly3= poly_reg3.fit_transform(X)
     lin_reg3 = LinearRegression()
     lin_reg3.fit(X_poly3,y)
```

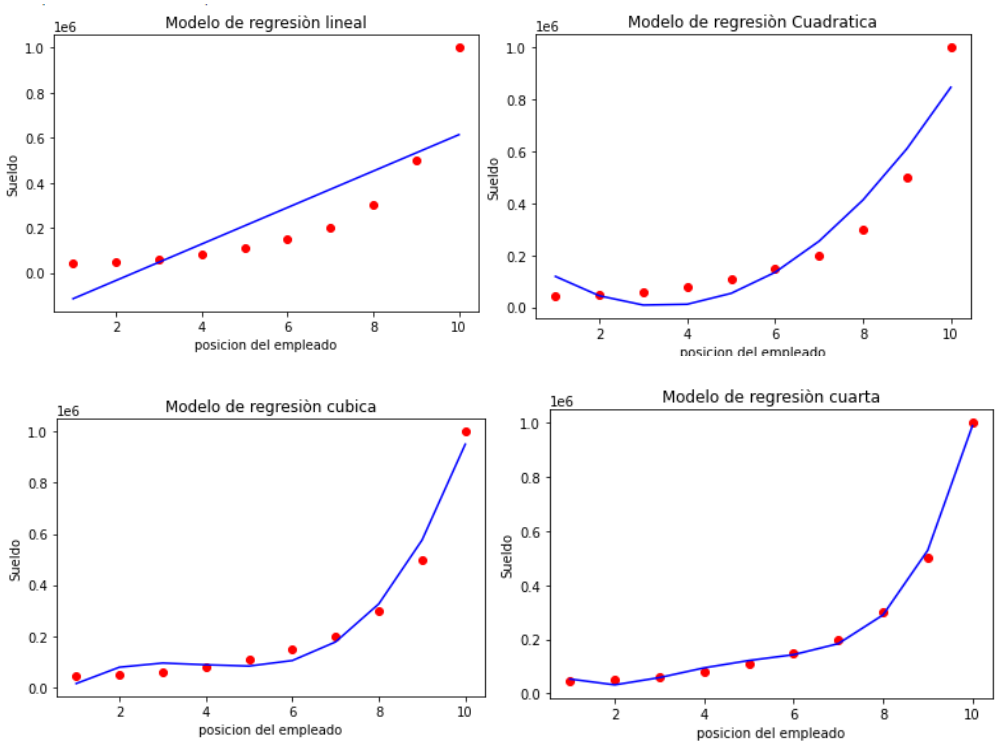
```
LinearRegression()
```

▾ Cuarta

```
[ ]  poly_reg4= PolynomialFeatures(degree=4)
     X_poly4= poly_reg4.fit_transform(X)
     lin_reg4 = LinearRegression()
     lin_reg4.fit(X_poly4,y)
```

```
LinearRegression()
```

10. Por último, graficamos su predicción con los datos que tenemos y observamos su desempeño.



Conclusiones

En cuanto a la selección de variables para el modelo, cabe resaltar la importancia de las pruebas estadísticas para ver que tan importante es para el modelo. Otro punto para tomar en cuenta es si los intervalos de confianza que se tienen contienen el cero, pues de serlo así es posible que un coeficiente sea 0 y esto nos habla de que la variable no es representativa para el modelo.

Por su parte, podemos observar que hay datos que no son modelables con una recta, pero podemos emplear modelos más complejos para aproximarnos mejor al comportamiento, además que, debido a mayor grado, mejor se ajustan; esto no quiere decir que sea lo propicio, puesto que habrá un sobre ajuste en los datos y nuestro modelo no generalizará, lo cual es su misión principal.

Referencias

[1] D. Olive, «Multiple Linear Regression,» de *Linear Regression*, Cham, Springer, 2017, pp. 17-83.

[2] A. Agarwal, «Towards Data Science,» 8 Oct 2018. [En línea]. Available: https://towardsdatascience.com/polynomial-regression-bbe8b9d97491. [Último acceso: 12 Sep 2022].