Tarea de ayudantia compu-2

January 10, 2020

0.1 Ayudantía - Regresión Lineal

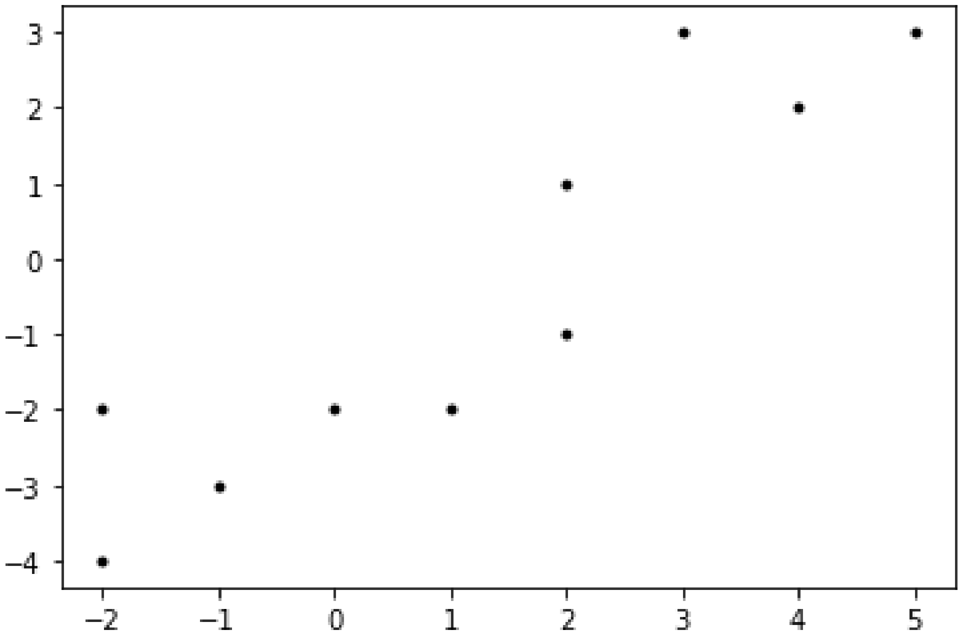
```
In [4]: import numpy as np
        import scipy as scp
        import matplotlib.pyplot as plt
        %matplotlib inline
```

0.1.1 Generación de los datos

```
In [5]: X = [-2,-2,-1,0,1,2,2,3,4,5]
        Y = [-4,-2,-3,-2,-2,-1,1,3,2,3]
```

```
In [6]: plt.plot(X,Y,'.k')
```

```
Out[6]: [<matplotlib.lines.Line2D at 0x114968fd0>]
```



0.1.2 Ajuste de un modelo lineal

```
In [7]: x_prom = np.mean(X)
        y_prom = np.mean(Y)
        x_var = np.var(X)
        covarianza = np.cov(X,Y, bias=True)[0][1] #Dividido por N
        print("Promedio X = ", x_prom)
        print("Promedio Y = ", y_prom)
        print("Varianza X = ", x_var)
        print("Covarianza  = ", covarianza)
```

```
Promedio X =  1.2
Promedio Y =  -0.5
Varianza X =  5.36
Covarianza  =  5.1000000000000005
```

```
In [8]: beta_1 = covarianza/x_var
        beta_0 = y_prom-beta_1*x_prom
        print("Beta_0 =", beta_0, "  Beta_1 = ",beta_1)
```

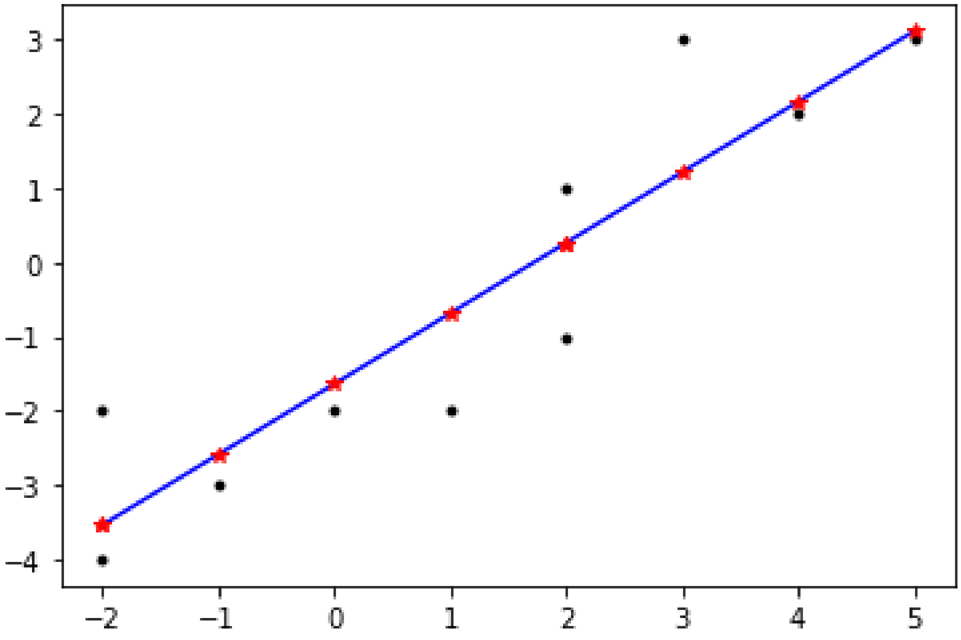
```
Beta_0 = -1.6417910447761195   Beta_1 =  0.9514925373134329
```

```
In [9]: y_pred = beta_1*np.array(X)+beta_0
```

```
In [10]: e_i = Y-y_pred  # residuos
```

```
In [11]: plt.plot(X,Y,'.k')
         plt.plot(X,y_pred,'-b')
         plt.plot(X,y_pred,'*r')
```

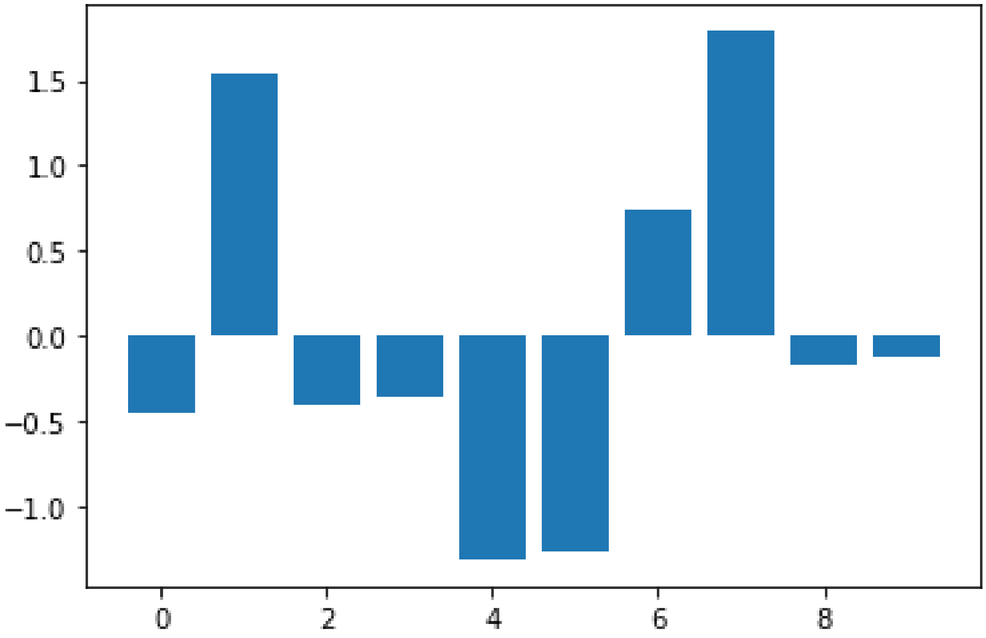
```
Out[11]: [<matplotlib.lines.Line2D at 0x114aa0110>]
```



0.1.3 Gráfico de los residuos

```
In [12]: plt.bar(range(10),e_i) # Gráfico de los residuos
```

```
Out[12]: <BarContainer object of 10 artists>
```



```
In [13]: rho=np.corrcoef(X,Y)[0][1] # coefciente de correlación
         print("Coeficiente de correlación de Pearson = ",rho)
```

```
Coeficiente de correlación de Pearson =  0.9107723725393417
```

0.2 Regresión lineal usando STATMODELS

https://www.statsmodels.org/stable/index.html

```
In [14]: import statsmodels.api as sm
```

```
In [15]: X = sm.add_constant(X)
```

```
In [16]: modelo_lineal = sm.OLS(Y,X).fit()
```

```
In [17]: modelo_lineal.summary()
```

```
/Users/constanzapardo/opt/anaconda3/lib/python3.7/site-packages/scipy/stats/stats.py:1450: Use
  "anyway, n=%i" % int(n))
```

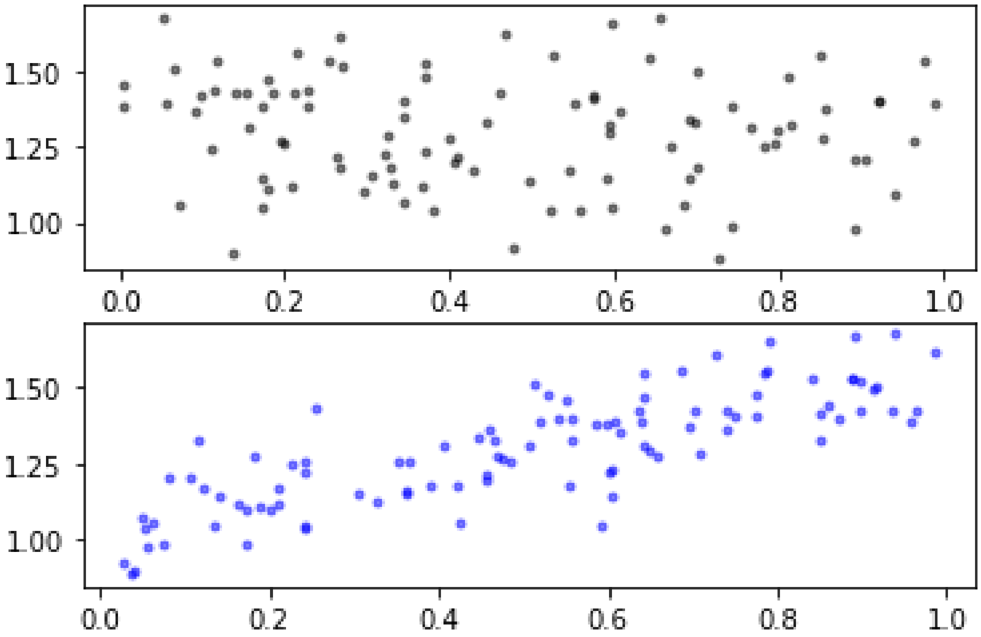
```
Out[17]: <class 'statsmodels.iolib.summary.Summary'>
         """
                                     OLS Regression Results
         ==============================================================================
         Dep. Variable:                      y   R-squared:                       0.830
         Model:                            OLS   Adj. R-squared:                  0.808
         Method:                 Least Squares   F-statistic:                     38.92
         Date:                Thu, 09 Jan 2020   Prob (F-statistic):           0.000249
         Time:                        19:31:00   Log-Likelihood:                -14.176
         No. Observations:                  10   AIC:                             32.35
         Df Residuals:                       8   BIC:                             32.96
         Df Model:                           1
         Covariance Type:            nonrobust
         ==============================================================================
                          coef    std err          t      P>|t|      [0.025      0.975]
         ------------------------------------------------------------------------------
         const         -1.6418      0.398     -4.128      0.003      -2.559      -0.725
         x1             0.9515      0.153      6.239      0.000       0.600       1.303
         ==============================================================================
         Omnibus:                        0.942   Durbin-Watson:                   1.767
         Prob(Omnibus):                  0.624   Jarque-Bera (JB):                0.753
         Skew:                           0.535   Prob(JB):                        0.686
         Kurtosis:                       2.187   Cond. No.                         3.04
         ==============================================================================

         Warnings:
         [1] Standard Errors assume that the covariance matrix of the errors is correctly spec
         """
```

0.3 Ejemplo 2

```
In [18]: n_obs = 100
         X = np.random.random((n_obs,2))
         X = sm.add_constant(X)
         beta = [1, 0.1, 0.5]
         e = np.random.randn(n_obs)*0.1
         Y = np.dot(X,beta) + e
         plt.subplot(211)
         plt.plot(X[:,1],Y,'.k',alpha=0.5)
         plt.subplot(212)
         plt.plot(X[:,2],Y,'.b',alpha=0.5)
```

```
Out[18]: [<matplotlib.lines.Line2D at 0x1c1d0fad10>]
```



```
In [19]: modelo = sm.OLS(Y,X)
         modelo = modelo.fit()

In [20]: modelo.summary()

Out[20]: <class 'statsmodels.iolib.summary.Summary'>
         """
                                     OLS Regression Results
         ==============================================================================
         Dep. Variable:                      y   R-squared:                       0.668
         Model:                            OLS   Adj. R-squared:                  0.661
         Method:                 Least Squares   F-statistic:                     97.41
         Date:                Thu, 09 Jan 2020   Prob (F-statistic):           6.32e-24
         Time:                        19:32:28   Log-Likelihood:                 83.238
         No. Observations:                 100   AIC:                            -160.5
         Df Residuals:                      97   BIC:                            -152.7
         Df Model:                           2
         Covariance Type:            nonrobust
         ==============================================================================
                          coef    std err          t      P>|t|      [0.025      0.975]
         ------------------------------------------------------------------------------
         const          1.0138      0.031     32.667      0.000       0.952       1.075
         x1             0.0238      0.040      0.600      0.550      -0.055       0.103
         x2             0.5410      0.039     13.850      0.000       0.463       0.619
         ==============================================================================
         Omnibus:                        0.417   Durbin-Watson:                   1.886
```

```
Prob(Omnibus):                  0.812   Jarque-Bera (JB):                0.248
Skew:                           0.121   Prob(JB):                        0.883
Kurtosis:                       3.025   Cond. No.                         5.93
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly spec
"""
```

```
In [21]: modelo.params
```

```
Out[21]: array([1.01381804, 0.02383884, 0.54100075])
```

```
In [22]: print("R2 (coef determinacion) = ", modelo.rsquared)
         print("Correlacion Pearson = ", np.sqrt(modelo.rsquared))
```

```
R2 (coef determinacion) =  0.6675950278510947
Correlacion Pearson =  0.8170648859491483
```

```
In [ ]:
```

0.4 Descripción de la Tarea

En este trabajo se espera que el estudiante realice un análisis de regresión con bases de datos. Deberá entregar el archivo Jupyter y un archivo pdf con las tablas resultantes con una discusión de los resultados.

Para realizar el estudio experimental deberá utilizar uno de los siguientes conjuntos de datos extraídos de la UCI Machine Learning Repository.
https://archive.ics.uci.edu/ml/index.php
Bank Marketing Data Set: https://archive.ics.uci.edu/ml/datasets/Bank+Marketing
Student Performance Data Set: https://archive.ics.uci.edu/ml/datasets/Student+Performance
Census Income Data Set: https://archive.ics.uci.edu/ml/datasets/Census+Income
Heart Disease Data Set: https://archive.ics.uci.edu/ml/datasets/Heart+Disease
Breast Cancer Wisconsin (Diagnostic) Data Set: https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+W

0.4.1 Para la presente tarea se deberá realizar:

1. Realizar un estudio de estadística descriptiva a dos variables numéricas. Hacer una tabla con los resultados.

2. Visualizar al menos 3 variables con los gráficos a elección

3. Tomar dos variables de interés y realizar una regresión lineal

4. De la regresión lineal, realizar un grafico y comentar los resultados.

5. Con variables numéricas, sacar promedio, varianza y covarianza.

6. Crear un dataFrame con los datos.

0.4.2 Consideraciones:

ůDeberá entregar un informe utilizando Jupyter.

ůFecha y hora de entrega: 23/01/2020, 23:55 hrs utilizando la plataforma https://classroom.google.com

ůCada día de atraso será penalizado con 1 décima

ůSe evaluará complejidad e interpretación de los datos

0.4.3 Referencias

1. ScyPy.org,StatisticalFunctions(scipy.stats).http://docs.scipy.org/doc/scipy/reference/stats.html
2. Pandas, Python Data Analysis Library. http://pandas.pydata.org/
3. Seaborn, Seaborn: statistical data visualization. https://stanford.edu/~mwaskom/software/seaborn/
4. Numpy, Fundamental package for scientific computing with Python. http://www.numpy.org/
5. Matplotlib, Biblioteca gráfica para python. http://matplotlib.org
6. Scipy Lecture Notes. http://www.scipy-lectures.org
7. STATMODELS. https://www.statsmodels.org/stable/index.html