Universidad Autónoma de Nuevo León

Facultad de Ciencias Físico Matemáticas

Minería de Datos

Ejercicios 1.

Profesora Mayra Cristina Berrones Reyes

Nombres	Matrículas
Evelyn Trejo Rodríguez	1811917
Magdaly Rodríguez Ortiz	1815330
Fernando González Castillo	1819011
Alfonso Llanos Morales	1887939
Alexis Hernández Morales	1887948

7° Semestre

Licenciatura en Actuaría

Ciudad San Nicolás de los Garza – 28 de Septiembre del 2020

EJERCICIO REGRESIÓN LINEAL.

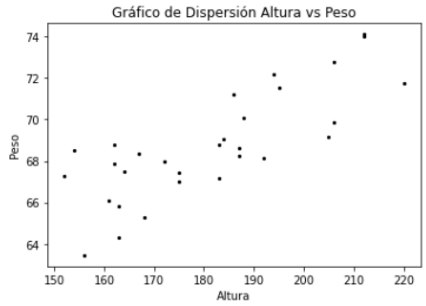
Tomando los datos de la siguiente tabla sobre los pesos y alturas de una población de 30 personas, crea una gráfica en donde el valor x represente la altura y el valor y represente el peso. Después traza una línea que se apegue lo mas posible a los datos que graficaste.

Peso	Altura	Peso	Altura	Peso	Altura
68.78	162	67.19	183	67.89	162
74.11	212	65.80	163	68.14	192
71.73	220	64.30	163	69.08	184
69.88	206	67.97	172	72.80	206
67.25	152	72.18	194	67.42	175
68.78	183	65.27	168	68.49	154
68.34	167	66.09	161	68.61	187
67.01	175	67.51	164	74.03	212
63.45	156	70.10	188	71.52	195
71.19	186	68.25	187	69.18	205

In [1]:
```
x_altura=[162,212,220,206,152,183,167,175,156,186,183,163,163,172,194,168,161,164,188,187,162,192,184,206,175,154,187,212,
          195,205]
y_peso=[68.78,74.11,71.73,69.88,67.25,68.78,68.34,67.01,63.45,71.19,67.19,65.80,64.3,67.97,72.18,65.27,66.09,67.51,70.10,
        68.25,67.89,68.14,69.08,72.80,67.42,68.49,68.61,74.03,71.52,69.18]
```

Gráfico de Dispersión

In [2]:
```
import matplotlib.pyplot as plt
plt.scatter(x=x_altura , y=y_peso, marker='o', c='black', s=5)
plt.title("Gráfico de Dispersión Altura vs Peso")
plt.xlabel("Altura")
plt.ylabel("Peso")
plt.show()
```

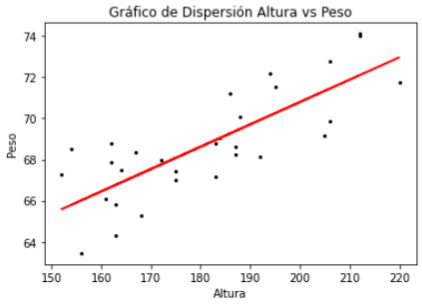


In [8]:
```
import numpy as np
array_x = np.array(x_altura)
array_y = np.array(y_peso)
n = len(array_x)
sum_x = sum(array_x)
sum_y = sum(array_y)
sum_xy = sum(array_x*array_y)
sum_xx = sum(array_x*array_x)
sum_yy = sum(array_y*array_y)
s_xy = sum_xy-(1/n)*sum_x*sum_y
s_xx = sum_xx-(1/n)*sum_x**2
beta_1 = s_xy / s_xx
beta_0 = (1/n)*sum_y - beta_1*(1/n)*sum_x
print("Los parámetros estimados para el modelo de regresión lineal son: ")
print("beta1: ",(beta_1))
print("beta0: ",(beta_0))
```

```
Los parámetros estimados para el modelo de regresión lineal son:
beta1:  0.1086107819535774
beta0:  49.07163369547534
```

Por lo tanto el modelo estimado de regresión lineal es: y=49.07163369547534 + 0.1086107819535774x

In [9]:
```
import matplotlib.pyplot as plt
plt.scatter(x=x_altura , y=y_peso, marker='o', c='black', s=5)
plt.plot(array_x, beta_0 + beta_1 * array_x, '-', c='red')
plt.title("Gráfico de Dispersión Altura vs Peso")
plt.xlabel("Altura")
plt.ylabel("Peso")
plt.show()
```



EJERCICIO REGLAS DE ASOCIACIÓN

Observa la tabla que se describe a continuación. Utilizando el algoritmo a priori, y la técnica de asociación, realiza la tabla de relaciones y resuelve cuál es el nivel **K** de soporte más alto al que podemos llegar con estos datos teniendo un umbral de 0.5.

ID	Transacciones
1	A B C E
2	B E
3	C D E
4	A C D
5	A C E

In [1]:
```
pip install apyori
```

```
Requirement already satisfied: apyori in c:\users\admin\anaconda3\lib\site-packages (1.1.2)
Note: you may need to restart the kernel to use updated packages.
```

In [2]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from apyori import apriori
```

In [3]:
```
transactions = np.array((["A","B","C","E"], ["B","E"], ["C","D","E"],["A","C","D"],["A","C","E"]))
```

In [4]:
```
rules = apriori(transactions, min_support = 0.5, min_confidence = 0, min_lift = 0, min_length = 1)
```

In [5]:
```
results = list(rules)
```

In [6]:
```
results
```

Out[6]:
```
[RelationRecord(items=frozenset({'A'}), support=0.6, ordered_statistics=[OrderedStatistic(items_base=frozenset(), items_add=fro
zenset({'A'}), confidence=0.6, lift=1.0)]),
 RelationRecord(items=frozenset({'C'}), support=0.8, ordered_statistics=[OrderedStatistic(items_base=frozenset(), items_add=fro
zenset({'C'}), confidence=0.8, lift=1.0)]),
 RelationRecord(items=frozenset({'E'}), support=0.8, ordered_statistics=[OrderedStatistic(items_base=frozenset(), items_add=fro
zenset({'E'}), confidence=0.8, lift=1.0)]),
 RelationRecord(items=frozenset({'A', 'C'}), support=0.6, ordered_statistics=[OrderedStatistic(items_base=frozenset(), items_ad
d=frozenset({'A', 'C'}), confidence=0.6, lift=1.0), OrderedStatistic(items_base=frozenset({'A'}), items_add=frozenset({'C'}), c
onfidence=1.0, lift=1.25), OrderedStatistic(items_base=frozenset({'C'}), items_add=frozenset({'A'}), confidence=0.7499999999999
999, lift=1.2499999999999998)]),
 RelationRecord(items=frozenset({'C', 'E'}), support=0.6, ordered_statistics=[OrderedStatistic(items_base=frozenset(), items_ad
d=frozenset({'C', 'E'}), confidence=0.6, lift=1.0), OrderedStatistic(items_base=frozenset({'C'}), items_add=frozenset({'E'}), c
onfidence=0.7499999999999999, lift=0.9374999999999998), OrderedStatistic(items_base=frozenset({'E'}), items_add=frozenset
({'C'}), confidence=0.7499999999999999, lift=0.9374999999999998)])]
```

In [7]:
```
def inspect(results):
    rh          = [tuple(result[2][0][0]) for result in results]
    lh          = [tuple(result[2][0][1]) for result in results]
    supports    = [result[1] for result in results]
    confidences = [result[2][0][2] for result in results]
    lifts       = [result[2][0][3] for result in results]
    return list(zip(rh, lh, supports, confidences, lifts))
```

In [8]:
```
resultDataFrame = pd.DataFrame(inspect(results),
                columns=['rhs','lhs','support','confidence','lift'])
```

In [9]:
```
resultDataFrame
```

Out[9]:

	rhs	lhs	support	confidence	lift
0	()	(A,)	0.6	0.6	1.0
1	()	(C,)	0.8	0.8	1.0
2	()	(E,)	0.8	0.8	1.0
3	()	(A, C)	0.6	0.6	1.0
4	()	(C, E)	0.6	0.6	1.0

Conclusión

Los niveles de K de soporte mas alto al que podemos llegar con estos datos teniendo un soporte minimo de 0.5 es:

Cuando K=1

(A)
Support:0.6
Confidence:0.6
Lift:1.0
(C)
Support:0.8
Confidence:0.8
Lift:1.0
(E)
Support:0.8
Confidence:0.8
Lift:1.0

Cuando K=2

(C,A)
Support: 0.6
Confidence: 0.6
Lift: 1.0
_____
(A->C)
Support: 0.6
Confidence: 1.0
Lift: 1.25
(C->A)
Support: 0.6
Confidence: 0.7499999999999999
Lift: 1.2499999999999998

(E,C)
Support: 0.6
Confidence: 0.6
Lift: 1.0
_____
(E->C)
Support: 0.6
Confidence: 0.7499999999999999
Lift:0.9374999999999998
(C->E)
Support: 0.6
Confidence: 0.7499999999999999
Lift=0.9374999999999998