Deena 20104016

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as pp
```

Problem Statement

LINEAR REGRESSION

In [2]:
```
a = pd.read_csv("15_Horse Racing Results.csv - 15_Horse Racing Results.CSV.csv
```

Out[2]:

	Dato	Track	Race Number	Distance	Surface	Prize money	Starting position	Jockey	Jockey weight	Cour
0	03.09.2017	Sha Tin	10	1400	Gress	1310000	6	K C Leung	52	Sve
1	16.09.2017	Sha Tin	10	1400	Gress	1310000	14	C Y Ho	52	Sve
2	14.10.2017	Sha Tin	10	1400	Gress	1310000	8	C Y Ho	52	Sve
3	11.11.2017	Sha Tin	9	1600	Gress	1310000	13	Brett Prebble	54	Sve
4	26.11.2017	Sha Tin	9	1600	Gress	1310000	9	C Y Ho	52	Sve
...	...	...	...	...	...	...	...	...	...	
27003	14.06.2020	Sha Tin	11	1200	Gress	1450000	6	A Hamelin	59	Austr
27004	21.06.2020	Sha Tin	2	1200	Gress	967000	7	K C Leung	57	Austr
27005	21.06.2020	Sha Tin	4	1200	Gress	967000	6	Blake Shinn	57	Austr
27006	21.06.2020	Sha Tin	5	1200	Gress	967000	14	Joao Moreira	57	N Zeal
27007	21.06.2020	Sha Tin	11	1200	Gress	1450000	7	C Schofield	55	N Zeal

27008 rows × 21 columns

HEAD

In [3]:

Out[3]:

	Dato	Track	Race Number	Distance	Surface	Prize money	Starting position	Jockey	Jockey weight	Country	.
0	03.09.2017	Sha Tin	10	1400	Gress	1310000	6	K C Leung	52	Sverige	.
1	16.09.2017	Sha Tin	10	1400	Gress	1310000	14	C Y Ho	52	Sverige	.
2	14.10.2017	Sha Tin	10	1400	Gress	1310000	8	C Y Ho	52	Sverige	.
3	11.11.2017	Sha Tin	9	1600	Gress	1310000	13	Brett Prebble	54	Sverige	.
4	26.11.2017	Sha Tin	9	1600	Gress	1310000	9	C Y Ho	52	Sverige	.

5 rows × 21 columns

Data Cleaning and Preprocessing

In [4]:

Out[4]:

	Dato	Track	Race Number	Distance	Surface	Prize money	Starting position	Jockey	Jockey weight	Country	.
0	03.09.2017	Sha Tin	10	1400	Gress	1310000	6	K C Leung	52	Sverige	.
1	16.09.2017	Sha Tin	10	1400	Gress	1310000	14	C Y Ho	52	Sverige	.
2	14.10.2017	Sha Tin	10	1400	Gress	1310000	8	C Y Ho	52	Sverige	.
3	11.11.2017	Sha Tin	9	1600	Gress	1310000	13	Brett Prebble	54	Sverige	.
4	26.11.2017	Sha Tin	9	1600	Gress	1310000	9	C Y Ho	52	Sverige	.

5 rows × 21 columns

In [5]:

Out[5]:

	Race Number	Distance	Prize money	Starting position	Jockey weight	Horse age	
count	27008.000000	27008.000000	2.700800e+04	27008.000000	27008.000000	27008.000000	270
mean	5.268624	1401.666173	1.479445e+06	6.741447	55.867373	5.246408	
std	2.780088	276.065045	2.162109e+06	3.691071	2.737006	1.519880	
min	1.000000	1000.000000	6.600000e+05	1.000000	47.000000	2.000000	
25%	3.000000	1200.000000	9.200000e+05	4.000000	54.000000	4.000000	
50%	5.000000	1400.000000	9.670000e+05	7.000000	56.000000	5.000000	
75%	8.000000	1650.000000	1.450000e+06	10.000000	58.000000	6.000000	
max	11.000000	2400.000000	2.800000e+07	14.000000	63.000000	12.000000	

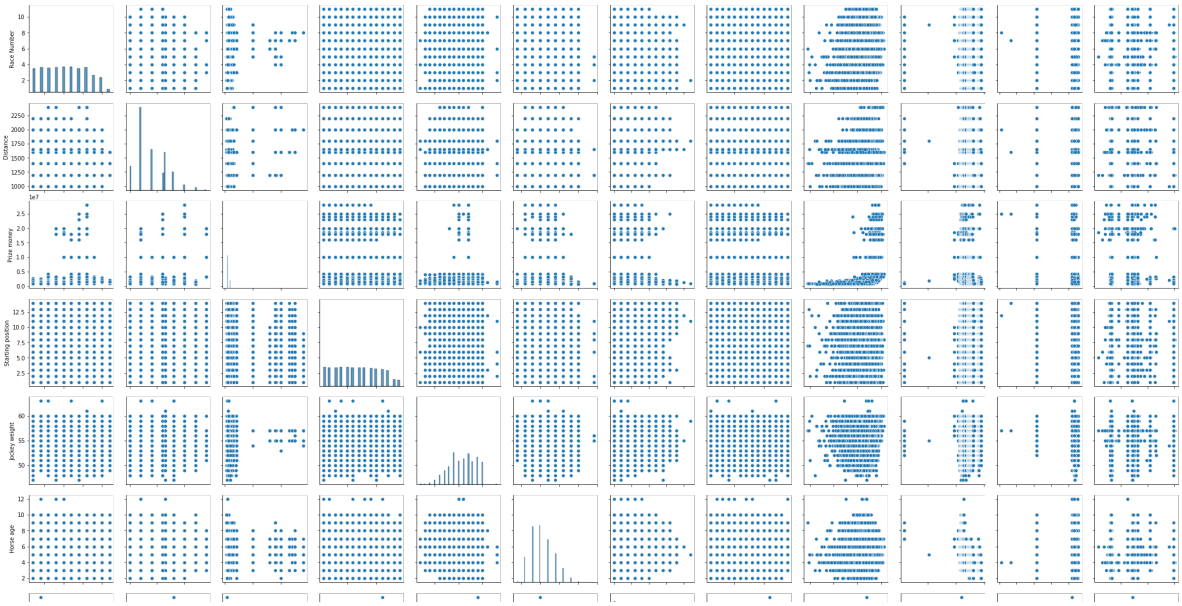
To display heading

In [6]:

Out[6]:
```
Index(['Dato', 'Track', 'Race Number', 'Distance', 'Surface', 'Prize money',
       'Starting position', 'Jockey', 'Jockey weight', 'Country', 'Horse age
',
       'TrainerName', 'Race time', 'Path', 'Final place', 'FGrating', 'Odds',
       'RaceType', 'HorseId', 'JockeyId', 'TrainerID'],
      dtype='object')
```

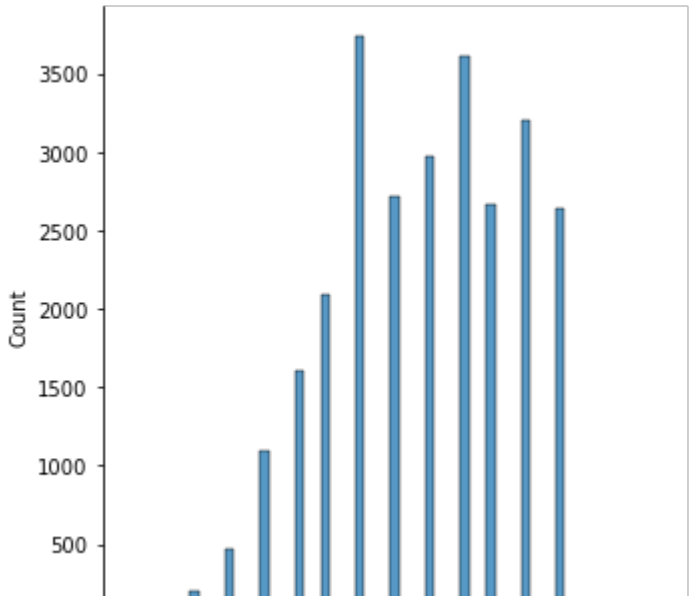
In [7]:

Out[7]: <seaborn.axisgrid.PairGrid at 0x2c83ac3b340>



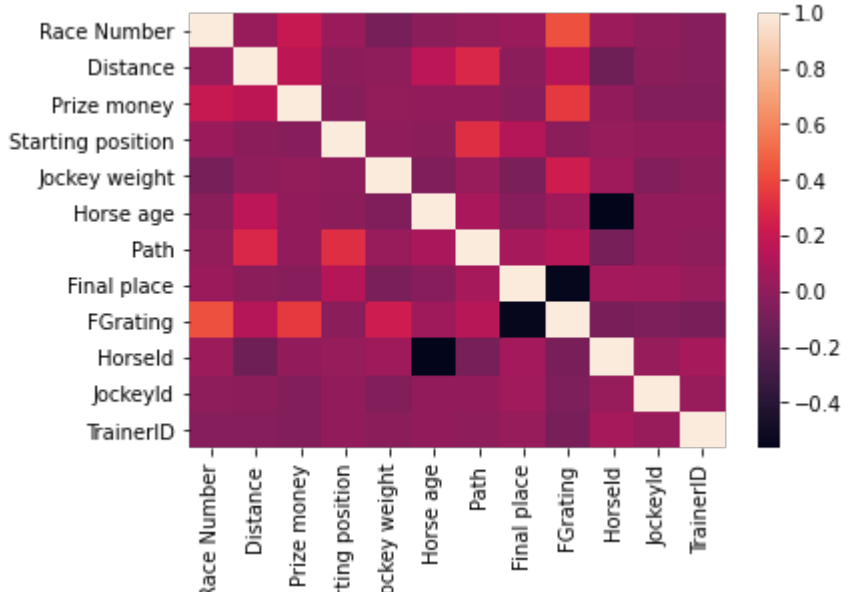
In [8]:

Out[8]: <seaborn.axisgrid.FacetGrid at 0x2c836e93670>



In [9]:

Out[9]: <AxesSubplot:>



TO TRAIN THE MODEL - MODEL BUILDING

In [10]:
```
x = a[['Jockey weight']]
```

In [11]:
```
# to split my dataset into training and test data
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.3)
```

In [12]:
```
from sklearn.linear_model import LinearRegression
lr = LinearRegression()
```

Out[12]:
```
LinearRegression()
```

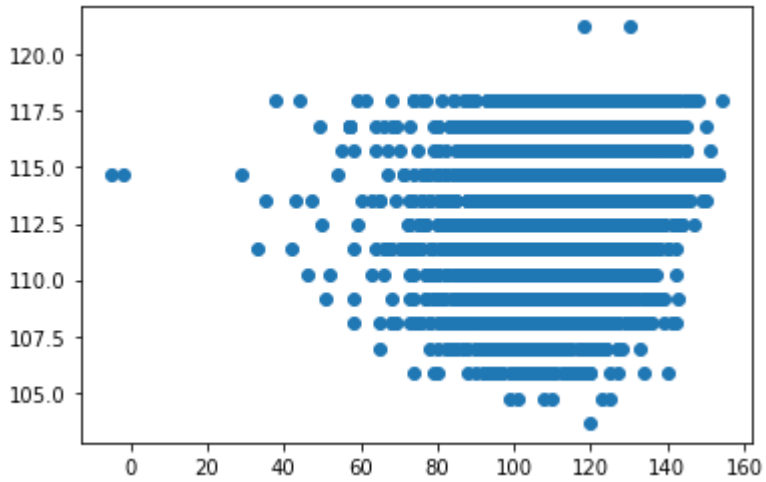
In [13]:
```
coeff = pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
```

Out[13]:

	Co-efficient
Jockey weight	1.100466

In [14]:
```
prediction= lr.predict(x_test)
```

Out[14]:
```
<matplotlib.collections.PathCollection at 0x2c8533ab100>
```



In [15]:

Out[15]:
```
0.04985076417769496
```

RIDGE & LASSO

In [16]:
```
from sklearn.linear_model import Ridge,Lasso
rr=Ridge(alpha=10)
```

Out[16]:
```
Ridge(alpha=10)
```

In [17]:

Out[17]:
```
0.04985092108294287
```

In [18]:
```
la=Lasso(alpha=10)
```

Out[18]:
```
Lasso(alpha=10)
```

In [19]:

Out[19]:
```
-2.4706552035125284e-05
```

In [20]:
```
from sklearn.linear_model import ElasticNet
a=ElasticNet()
```

Out[20]:
```
ElasticNet()
```

In [21]:
```
print(a.coef_)
print(a.intercept_)
print(a.score(x_test,y_test))
```

```
[0.96950061]
59.2422851267223
0.04937861951629163
[116.44282117 115.47332055 108.68681628 ... 115.47332055 111.59531811
 115.47332055]
```

In [22]:
```
from sklearn import metrics
print(" Mean Absolute Error :",metrics.mean_absolute_error(y_test,prediction))
print(" Mean Squared Error :",metrics.mean_squared_error(y_test,prediction))
```

```
 Mean Absolute Error : 9.58826923859465
 Mean Squared Error : 163.9561810540656
 Root Mean Absolute Error : 3.0964930548274525
```