Data collection

Importing libraries

In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

Importing dataset

In [2]:

```
data=pd.read_csv(r"C:\Users\user\Downloads\15_Horse Racing Results.CSV - 15_Horse Racin
data
```

Out[2]:

	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	...
...	...	...	...	...	...	...	...	...	...	...	...
27003	14.06.2020	Sha Tin	11	1200	Gress	1450000	6	A Hamelin	59	Australia	...
27004	21.06.2020	Sha Tin	2	1200	Gress	967000	7	K C Leung	57	Australia	...
27005	21.06.2020	Sha Tin	4	1200	Gress	967000	6	Blake Shinn	57	Australia	...
27006	21.06.2020	Sha Tin	5	1200	Gress	967000	14	Joao Moreira	57	New Zealand	...
27007	21.06.2020	Sha Tin	11	1200	Gress	1450000	7	C Schofield	55	New Zealand	...

27008 rows × 21 columns

head

In [3]:

```
# to display first 8 dataset values
da=data.head(8)
da
```

Out[3]:

	Dato	Track	Race Number	Distance	Surface	Prize money	Starting position	Jockey	Jockey weight	Country	...	Trai
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	10.12.2017	Sha Tin	1	1800	Gress	1310000	4	C Y Ho	52	Sverige	...	
6	01.01.2018	Sha Tin	9	1800	Gress	1310000	9	C Schofield	54	Sverige	...	
7	04.02.2018	Sha Tin	5	1800	Gress	1310000	6	Joao Moreira	57	Sverige	...	

8 rows × 21 columns

info

In [4]:

```
# to identify missing values
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 27008 entries, 0 to 27007
Data columns (total 21 columns):
 #   Column             Non-Null Count  Dtype 
---  ------             --------------  ----- 
 0   Dato               27008 non-null  object
 1   Track              27008 non-null  object
 2   Race Number        27008 non-null  int64 
 3   Distance           27008 non-null  int64 
 4   Surface            27008 non-null  object
 5   Prize money        27008 non-null  int64 
 6   Starting position  27008 non-null  int64 
 7   Jockey             27008 non-null  object
 8   Jockey weight      27008 non-null  int64 
 9   Country            27008 non-null  object
 10  Horse age          27008 non-null  int64 
 11  TrainerName        27008 non-null  object
 12  Race time          27008 non-null  object
 13  Path               27008 non-null  int64 
 14  Final place        27008 non-null  int64 
 15  FGrating           27008 non-null  int64 
 16  Odds               27008 non-null  object
 17  RaceType           27008 non-null  object
 18  HorseId            27008 non-null  int64 
 19  JockeyId           27008 non-null  int64 
 20  TrainerID          27008 non-null  int64 
dtypes: int64(12), object(9)
memory usage: 4.3+ MB
```

describe

In [5]:

```
# to display summary of the dataset
data.describe()
```

Out[5]:

	Race Number	Distance	Prize money	Starting position	Jockey weight	Horse age	Pat
count	27008.000000	27008.000000	2.700800e+04	27008.000000	27008.000000	27008.000000	27008.00000
mean	5.268624	1401.666173	1.479445e+06	6.741447	55.867373	5.246408	1.67802
std	2.780088	276.065045	2.162109e+06	3.691071	2.737006	1.519880	1.63178
min	1.000000	1000.000000	6.600000e+05	1.000000	47.000000	2.000000	0.00000
25%	3.000000	1200.000000	9.200000e+05	4.000000	54.000000	4.000000	0.00000
50%	5.000000	1400.000000	9.670000e+05	7.000000	56.000000	5.000000	1.00000
75%	8.000000	1650.000000	1.450000e+06	10.000000	58.000000	6.000000	3.00000
max	11.000000	2400.000000	2.800000e+07	14.000000	63.000000	12.000000	11.00000

columns

In [6]:

```
# to display headings of the dataset
data.columns
```

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]:

```
a=data.dropna(axis=1)
a
```

Out[7]:

	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	...
...	...	...	...	...	...	...	...	...	...	...	...
27003	14.06.2020	Sha Tin	11	1200	Gress	1450000	6	A Hamelin	59	Australia	...
27004	21.06.2020	Sha Tin	2	1200	Gress	967000	7	K C Leung	57	Australia	...
27005	21.06.2020	Sha Tin	4	1200	Gress	967000	6	Blake Shinn	57	Australia	...
27006	21.06.2020	Sha Tin	5	1200	Gress	967000	14	Joao Moreira	57	New Zealand	...
27007	21.06.2020	Sha Tin	11	1200	Gress	1450000	7	C Schofield	55	New Zealand	...

27008 rows × 21 columns

In [8]:

```
a.columns
```

Out[8]:

```
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')
```

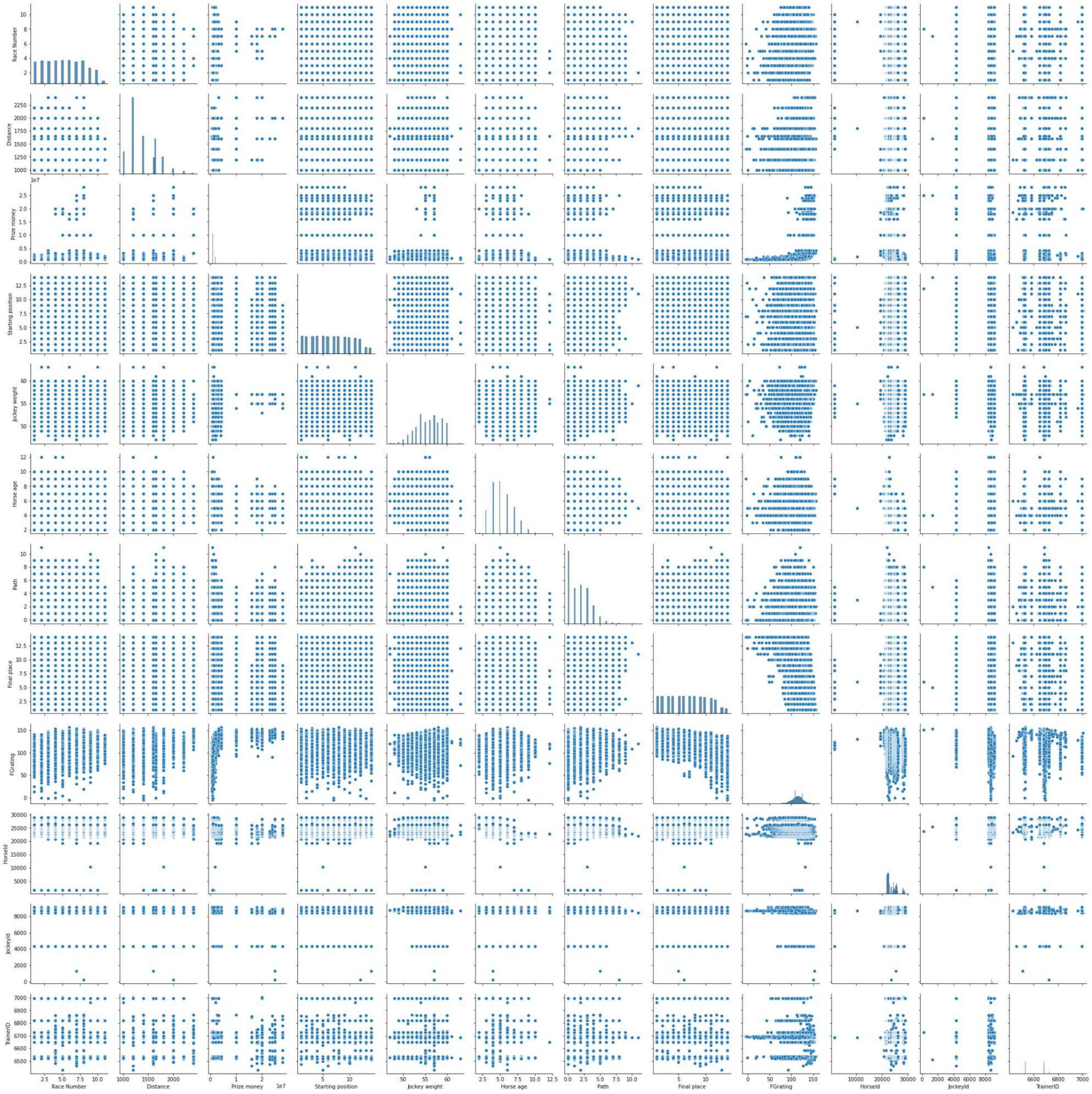
EDA and Visualization

In [9]:

```
sns.pairplot(a)
```

Out[9]:

```
<seaborn.axisgrid.PairGrid at 0x1af4c386130>
```



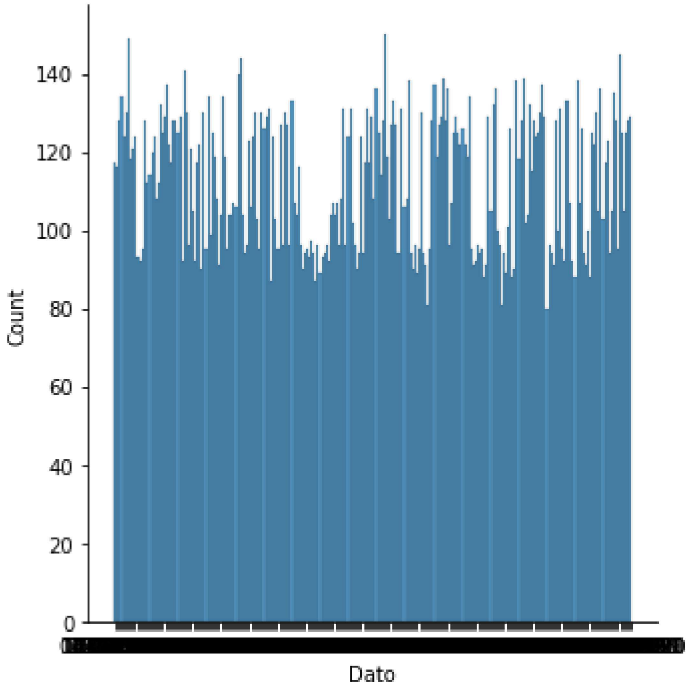
distribution plot

In [11]:

```python
sns.displot(a["Dato"])
```

Out[11]:

```
<seaborn.axisgrid.FacetGrid at 0x1af4c325fd0>
```



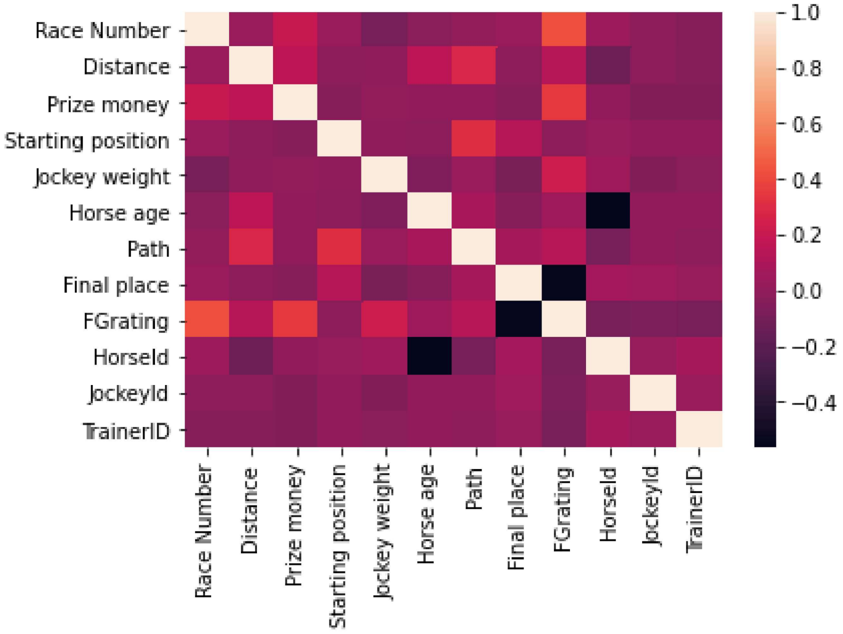
correlation

In [12]:

```
dat=data[['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']]
sns.heatmap(dat.corr())
```

Out[12]:

```
<AxesSubplot:>
```



To train the model-Model Building

In [16]:

```
x=a[['Race Number']]
y=a['Jockey weight']
```

In [17]:

```
# 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 [18]:

```
from sklearn.linear_model import LinearRegression
lr= LinearRegression()
lr.fit(x_train,y_train)
```

Out[18]:

```
LinearRegression()
```

In [19]:

```
print(lr.intercept_)
```

```
56.37917600269962
```

In [20]:

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

Out[20]:

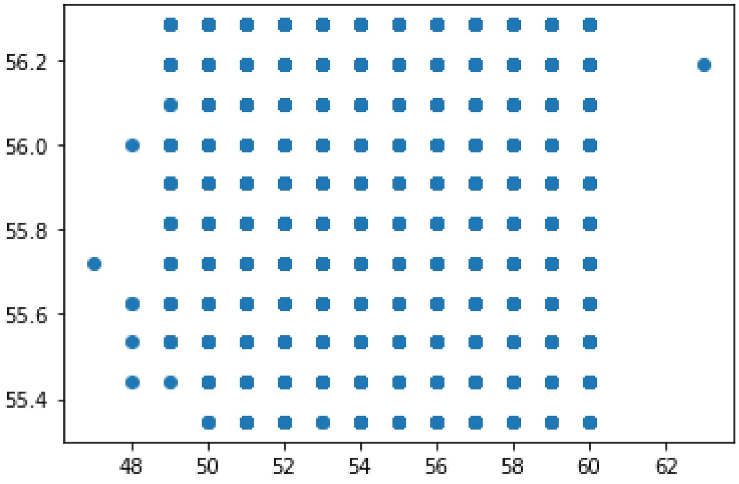
	Co-efficient
Race Number	-0.093948

In [21]:

```
prediction=lr.predict(x_test)
plt.scatter(y_test,prediction)
```

Out[21]:

```
<matplotlib.collections.PathCollection at 0x1af64a2d040>
```



In [22]:

```
print(lr.score(x_test,y_test))
```

```
0.006732476871311843
```

In [23]:

```
lr.score(x_train,y_train)
```

```
Out[23]: 0.00910212961827328
```

Ridge regression

```
In [24]: from sklearn.linear_model import Ridge,Lasso
```

```
In [25]: rr=Ridge(alpha=10)
         rr.fit(x_train,y_train)
         rr.score(x_test,y_test)

Out[25]: 0.006732614394663439
```

```
In [26]: rr.score(x_train,y_train)

Out[26]: 0.009102129575765061
```

Lasso regression

```
In [27]: la=Lasso(alpha=10)
         la.fit(x_train,y_train)
         la.score(x_train,y_train)

Out[27]: 0.0
```

```
In [28]: la.score(x_test,y_test)

Out[28]: -0.0003735320215088045
```

```
In [ ]:
```