In [ ]:

```
```

In [82]:

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

In [83]:

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

Out[83]:

	Dato	Track	Race Number	Distance	Surface	Prize money	Starting position	Jockey	Jockey weight	Country	...	Tra
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	...	P
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 [84]:

```
a=a.head(10)
a
```

Out[84]:

	Dato	Track	Race Number	Distance	Surface	Prize money	Starting position	Jockey	Jockey weight	Country	...	Trainerl
0	03.09.2017	Sha Tin	10	1400	Gress	1310000	6	K C Leung	52	Sverige	...	C
1	16.09.2017	Sha Tin	10	1400	Gress	1310000	14	C Y Ho	52	Sverige	...	C
2	14.10.2017	Sha Tin	10	1400	Gress	1310000	8	C Y Ho	52	Sverige	...	C
3	11.11.2017	Sha Tin	9	1600	Gress	1310000	13	Brett Prebble	54	Sverige	...	C
4	26.11.2017	Sha Tin	9	1600	Gress	1310000	9	C Y Ho	52	Sverige	...	C
5	10.12.2017	Sha Tin	1	1800	Gress	1310000	4	C Y Ho	52	Sverige	...	C
6	01.01.2018	Sha Tin	9	1800	Gress	1310000	9	C Schofield	54	Sverige	...	C
7	04.02.2018	Sha Tin	5	1800	Gress	1310000	6	Joao Moreira	57	Sverige	...	C
8	03.03.2018	Sha Tin	8	1800	Gress	1310000	3	C Y Ho	56	Sverige	...	C
9	11.03.2018	Sha Tin	10	1600	Gress	1310000	8	C Y Ho	57	Sverige	...	C

10 rows × 21 columns

In [85]:

```
# to find
a.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10 entries, 0 to 9
Data columns (total 21 columns):
 #   Column             Non-Null Count  Dtype 
---  ------             --------------  ----- 
 0   Dato               10 non-null     object
 1   Track              10 non-null     object
 2   Race Number        10 non-null     int64 
 3   Distance           10 non-null     int64 
 4   Surface            10 non-null     object
 5   Prize money        10 non-null     int64 
 6   Starting position  10 non-null     int64 
 7   Jockey             10 non-null     object
 8   Jockey weight      10 non-null     int64 
 9   Country            10 non-null     object
 10  Horse age          10 non-null     int64 
 11  TrainerName        10 non-null     object
 12  Race time          10 non-null     object
 13  Path               10 non-null     int64 
 14  Final place        10 non-null     int64 
 15  FGrating           10 non-null     int64 
 16  Odds               10 non-null     object
 17  RaceType           10 non-null     object
 18  HorseId            10 non-null     int64 
 19  JockeyId           10 non-null     int64 
 20  TrainerID          10 non-null     int64 
dtypes: int64(12), object(9)
memory usage: 1.8+ KB
```

In [86]:

```
# to display summary of statastic
a.describe()
```

Out[86]:

	Race Number	Distance	Prize money	Starting position	Jockey weight	Horse age	Path	Final place	FGratin
count	10.000000	10.000000	10.0	10.000000	10.000000	10.000000	10.000000	10.000000	10.00000
mean	8.100000	1620.000000	1310000.0	8.000000	53.800000	7.400000	1.500000	4.700000	120.10000
std	2.923088	175.119007	0.0	3.527668	2.149935	0.516398	1.581139	2.496664	6.47130
min	1.000000	1400.000000	1310000.0	3.000000	52.000000	7.000000	0.000000	1.000000	107.00000
25%	8.250000	1450.000000	1310000.0	6.000000	52.000000	7.000000	0.250000	3.000000	119.00000
50%	9.000000	1600.000000	1310000.0	8.000000	53.000000	7.000000	1.000000	4.000000	123.00000
75%	10.000000	1800.000000	1310000.0	9.000000	55.500000	8.000000	2.000000	6.000000	124.00000
max	10.000000	1800.000000	1310000.0	14.000000	57.000000	8.000000	5.000000	9.000000	125.00000

In [87]:

```
# to display colum heading
a.columns
```

Out[87]:

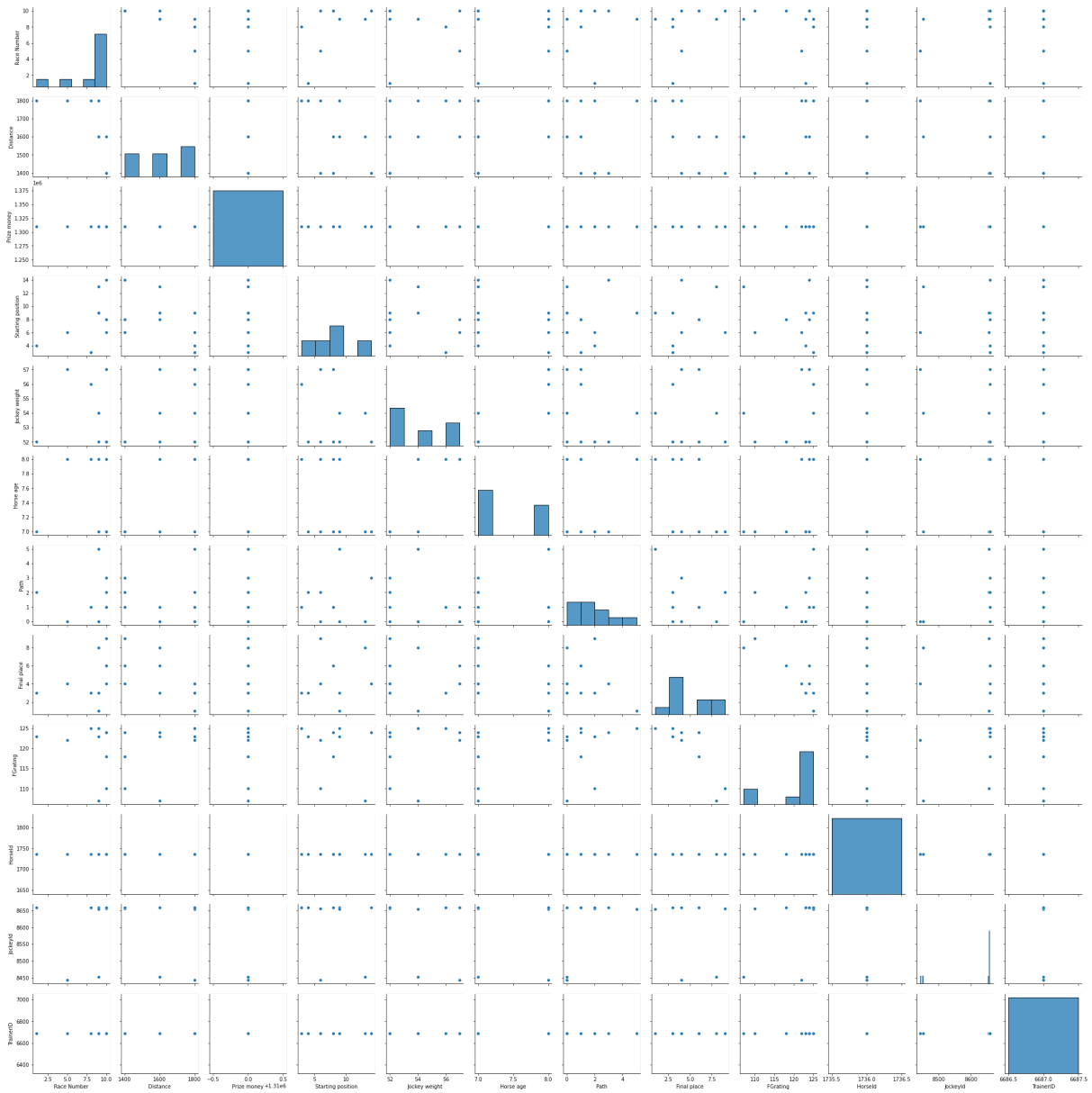
```
Index(['Dato', 'Track', 'Race Number', 'Distance', 'Surface', 'Prize money',
       'Starting position', 'Jockey', 'Jockey weight', 'Country', 'Horse age',
       'TrainerName', 'Race time', 'Path', 'Final place', 'FGrating', 'Odds',
       'RaceType', 'HorseId', 'JockeyId', 'TrainerID'],
      dtype='object')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

In [88]:

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

Out[88]:

```
<seaborn.axisgrid.PairGrid at 0x198ba310e20>
```

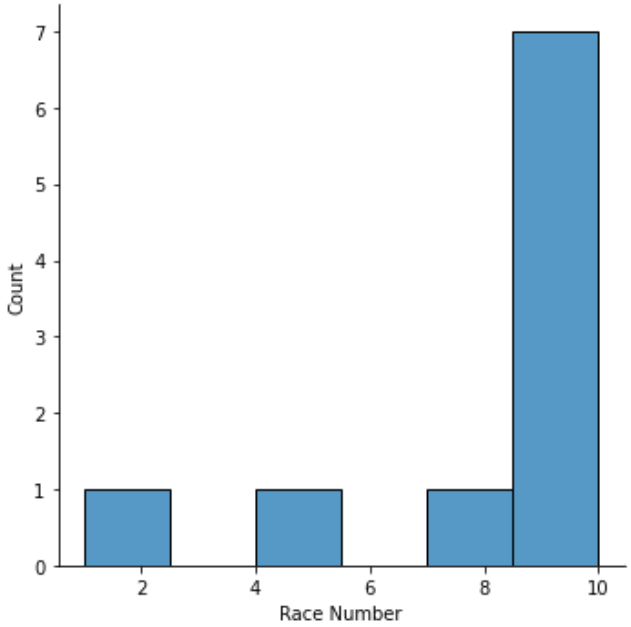


In [89]:

```
sns.displot(a["Race Number"])
```

Out[89]:

```
<seaborn.axisgrid.FacetGrid at 0x198bf882460>
```



In [90]:

```
b=a[['Track', 'Race Number', 'Distance', 'Surface', 'Prize money',
       'Starting position', 'Jockey' ]]
b
```

Out[90]:

	Track	Race Number	Distance	Surface	Prize money	Starting position	Jockey
0	Sha Tin	10	1400	Gress	1310000	6	K C Leung
1	Sha Tin	10	1400	Gress	1310000	14	C Y Ho
2	Sha Tin	10	1400	Gress	1310000	8	C Y Ho
3	Sha Tin	9	1600	Gress	1310000	13	Brett Prebble
4	Sha Tin	9	1600	Gress	1310000	9	C Y Ho
5	Sha Tin	1	1800	Gress	1310000	4	C Y Ho
6	Sha Tin	9	1800	Gress	1310000	9	C Schofield
7	Sha Tin	5	1800	Gress	1310000	6	Joao Moreira
8	Sha Tin	8	1800	Gress	1310000	3	C Y Ho
9	Sha Tin	10	1600	Gress	1310000	8	C Y Ho

In [91]:

```
sns.heatmap(b.corr())
```

Out[91]:

<AxesSubplot:>

In [99]:

```
x=a[['Race Number', 'Distance', 'Prize money',
       'Starting position']]
y=a['Race Number']
```

In [100]:

```
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 [101]:

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

Out[101]:

```
LinearRegression()
```

In [102]:

```
lr.intercept_
```

Out[102]:

```
-1.5987211554602254e-14
```

In [103]:

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

Out[103]:

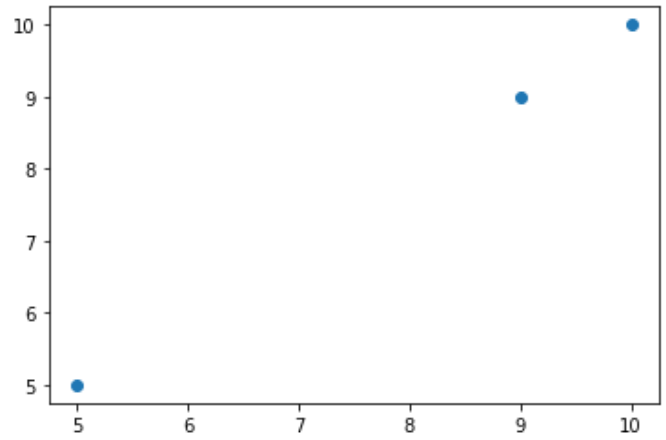
	Co-efficient
Race Number	1.000000e+00
Distance	1.012346e-17
Prize money	0.000000e+00
Starting position	-2.192399e-16

In [104]:

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

Out[104]:

```
<matplotlib.collections.PathCollection at 0x198c2934dc0>
```



In [105]:

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

Out[105]:

```
1.0
```

In [106]:

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

Out[106]:

```
1.0
```

In [107]:

```
from sklearn.linear_model import Ridge,Lasso
```

In [108]:

```
rr=Ridge(alpha=10)
rr.fit(x_test,y_test)
```

Out[108]:

```
Ridge(alpha=10)
```

In [109]:

```
rr.score(x_test,y_test)
```

Out[109]:

```
0.8871594645918177
```

In [110]:

```
la=Lasso(alpha=10)
la.fit(x_test,y_test)
```

Out[110]:

```
Lasso(alpha=10)
```

In [111]:

```
la.score(x_test,y_test)
```

Out[111]:

```
0.42616071428571434
```

In [112]:

```
from sklearn.linear_model import ElasticNet
en=ElasticNet()
en.fit(x_train,y_train)
```

Out[112]:

```
ElasticNet()
```

In [113]:

```
en.coef_
```

Out[113]:

```
array([ 0.78494747, -0.00282822,  0.        ,  0.        ])
```

In [114]:

```
en.intercept_
```

Out[114]:

```
6.195484221835903
```

In [115]:

```
prediction=en.predict(x_test)
prediction
```

Out[115]:

```
array([9.51981051, 5.02942959, 8.16921949])
```

In [116]:

```
en.score(x_test,y_test)
```

Out[116]:

```
0.9341682638724647
```

EVALUATION METRICS

In [117]:

```
from sklearn import metrics
```

In [118]:

```
print("Mean Absolute Error:",metrics.mean_absolute_error(y_test,prediction))
```

```
Mean Absolute Error: 0.4467998631727319
```

In [119]:

```
print("Mean Squared Error",metrics.mean_squared_error(y_test,prediction))
```

```
Mean Squared Error 0.30721476859516517
```

In [120]:

```
print("Root Mean Squared Error",np.sqrt(metrics.mean_squared_error(y_test,prediction)))
```

```
Root Mean Squared Error 0.5542695811562864
```

In [ ]:

```

```