In [1]:

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

In [2]:

```
df1=pd.read_csv(r'C:\Users\user\Downloads\15_Horse Racing Results.csv')
df1
```

Out[2]:

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

27008 rows × 21 columns

In [3]:

```
df=df1.head(50)
df
```

Out[3]:

	Dato	Track	Race Number	Distance	Surface	Prize money	Starting position	Jockey	Jockey weight	C
0	03.09.2017	Sha Tin	10	1400	Gress	1310000	6	K C Leung	52	S
1	16.09.2017	Sha Tin	10	1400	Gress	1310000	14	C Y Ho	52	S
2	14.10.2017	Sha Tin	10	1400	Gress	1310000	8	C Y Ho	52	S
3	11.11.2017	Sha Tin	9	1600	Gress	1310000	13	Brett Prebble	54	S
4	26.11.2017	Sha Tin	9	1600	Gress	1310000	9	C Y Ho	52	S
5	10.12.2017	Sha Tin	1	1800	Gress	1310000	4	C Y Ho	52	S
6	01.01.2018	Sha Tin	9	1800	Gress	1310000	9	C Schofield	54	S
7	04.02.2018	Sha Tin	5	1800	Gress	1310000	6	Joao Moreira	57	S
8	03.03.2018	Sha Tin	8	1800	Gress	1310000	3	C Y Ho	56	S
9	11.03.2018	Sha Tin	10	1600	Gress	1310000	8	C Y Ho	57	S
10	28.03.2018	Happy Valley	8	1800	Gress	1310000	9	M F Poon	53	S
11	11.04.2018	Happy Valley	6	1650	Gress	1310000	11	W M Lai	55	S
12	25.04.2018	Happy Valley	3	2200	Gress	1310000	2	W M Lai	54	S
13	09.05.2018	Happy Valley	7	1650	Gress	1310000	3	W M Lai	54	S
14	22.09.2018	Sha Tin	4	1600	Gress	920000	11	C Y Ho	57	S
15	07.10.2018	Sha Tin	6	1600	Gress	920000	9	C Y Ho	56	S
16	02.12.2018	Sha Tin	3	1800	Dirt	920000	1	C Schofield	57	S
17	23.12.2018	Sha Tin	2	2000	Gress	920000	6	Silvestre De Sousa	59	S
18	17.02.2019	Sha Tin	1	2000	Gress	920000	4	C Wong	57	S
19	06.12.2017	Happy Valley	9	1800	Gress	1860000	5	Z Purton	55	Da
20	01.10.2017	Sha Tin	7	1000	Gress	3000000	8	Z Purton	60	
21	22.10.2017	Sha Tin	7	1200	Gress	4000000	2	M Chadwick	60	
22	19.11.2017	Sha Tin	7	1200	Gress	4000000	8	M Chadwick	56	

	Dato	Track	Race Number	Distance	Surface	Prize money	Starting position	Jockey	Jockey weight	C
23	10.12.2017	Sha Tin	5	1200	Gress	18500000	9	M Chadwick	57	
24	01.01.2018	Sha Tin	10	1400	Gress	3000000	10	N Rawiller	58	
25	28.01.2018	Sha Tin	7	1200	Gress	10000000	3	Brett Prebble	57	
26	25.02.2018	Sha Tin	9	1400	Gress	10000000	2	Brett Prebble	57	
27	11.03.2018	Sha Tin	7	1200	Gress	2500000	4	N Callan	56	
28	08.04.2018	Sha Tin	7	1200	Gress	4000000	6	N Callan	56	
29	29.04.2018	Sha Tin	7	1200	Gress	16000000	2	N Callan	57	
30	01.10.2017	Sha Tin	7	1000	Gress	3000000	4	Tommy Berry	59	Au
31	22.10.2017	Sha Tin	7	1200	Gress	4000000	4	Tommy Berry	59	Au
32	19.11.2017	Sha Tin	7	1200	Gress	4000000	9	Tommy Berry	56	Au
33	10.12.2017	Sha Tin	5	1200	Gress	18500000	10	Tommy Berry	57	Au
34	07.01.2018	Sha Tin	7	1000	Gress	3000000	6	Tommy Berry	60	Au
35	28.01.2018	Sha Tin	7	1200	Gress	10000000	1	Tommy Berry	57	Au
36	11.03.2018	Sha Tin	7	1200	Gress	2500000	2	M F Poon	55	Au
37	08.04.2018	Sha Tin	7	1200	Gress	4000000	5	S Clipperton	56	Au
38	29.04.2018	Sha Tin	4	1400	Gress	2500000	10	Brett Prebble	60	Au
39	01.10.2018	Sha Tin	7	1000	Gress	3250000	7	C Y Ho	51	Au
40	21.10.2018	Sha Tin	7	1200	Gress	4250000	4	C Y Ho	52	Au
41	25.11.2018	Sha Tin	3	1000	Gress	1950000	9	Silvestre De Sousa	60	Au
42	19.12.2018	Sha Tin	7	1200	Dirt	1950000	8	Silvestre De Sousa	59	Au
43	10.12.2017	Sha Tin	4	2400	Gress	18000000	8	Ryan Moore	57	
44	10.12.2017	Sha Tin	3	1400	Gress	2500000	3	N Callan	57	
45	01.01.2018	Sha Tin	10	1400	Gress	3000000	12	C Schofield	53	
46	18.02.2018	Sha Tin	8	1400	Gress	2500000	4	C Schofield	59	
47	11.03.2018	Sha Tin	7	1200	Gress	2500000	11	C Schofield	55	

	Dato	Track	Race Number	Distance	Surface	Prize money	Starting position	Jockey	Jockey weight	C
48	08.04.2018	Sha Tin	7	1200	Gress	4000000	3	C Schofield	56	
49	29.04.2018	Sha Tin	4	1400	Gress	2500000	8	Z Purton	59	

50 rows × 21 columns

In [4]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 50 entries, 0 to 49
Data columns (total 21 columns):
 #   Column             Non-Null Count  Dtype 
---  ------             --------------  ----- 
 0   Dato               50 non-null     object
 1   Track              50 non-null     object
 2   Race Number        50 non-null     int64 
 3   Distance           50 non-null     int64 
 4   Surface            50 non-null     object
 5   Prize money        50 non-null     int64 
 6   Starting position  50 non-null     int64 
 7   Jockey             50 non-null     object
 8   Jockey weight      50 non-null     int64 
 9   Country            50 non-null     object
 10  Horse age          50 non-null     int64 
 11  TrainerName        50 non-null     object
 12  Race time          50 non-null     object
 13  Path               50 non-null     int64 
 14  Final place        50 non-null     int64 
 15  FGrating           50 non-null     int64 
 16  Odds               50 non-null     object
 17  RaceType           50 non-null     object
 18  HorseId            50 non-null     int64 
 19  JockeyId           50 non-null     int64 
 20  TrainerID          50 non-null     int64 
dtypes: int64(12), object(9)
memory usage: 8.3+ KB
```

In [5]:

```
df.describe()
```

Out[5]:

	Race Number	Distance	Prize money	Starting position	Jockey weight	Horse age	Path	I
count	50.000000	50.000000	5.000000e+01	50.000000	50.000000	50.000000	50.000000	50.
mean	6.560000	1438.000000	3.954000e+06	6.460000	56.120000	7.400000	1.460000	6.
std	2.383275	326.165102	4.632386e+06	3.375845	2.512337	0.832993	1.501156	2.
min	1.000000	1000.000000	9.200000e+05	1.000000	51.000000	5.000000	0.000000	1.
25%	5.000000	1200.000000	1.310000e+06	4.000000	54.250000	7.000000	0.000000	4.
50%	7.000000	1400.000000	2.500000e+06	6.000000	56.500000	8.000000	1.000000	6.
75%	8.000000	1637.500000	4.000000e+06	9.000000	57.000000	8.000000	2.000000	8.
max	10.000000	2400.000000	1.850000e+07	14.000000	60.000000	9.000000	5.000000	11.

In [6]:

```
df.columns
```

Out[6]:

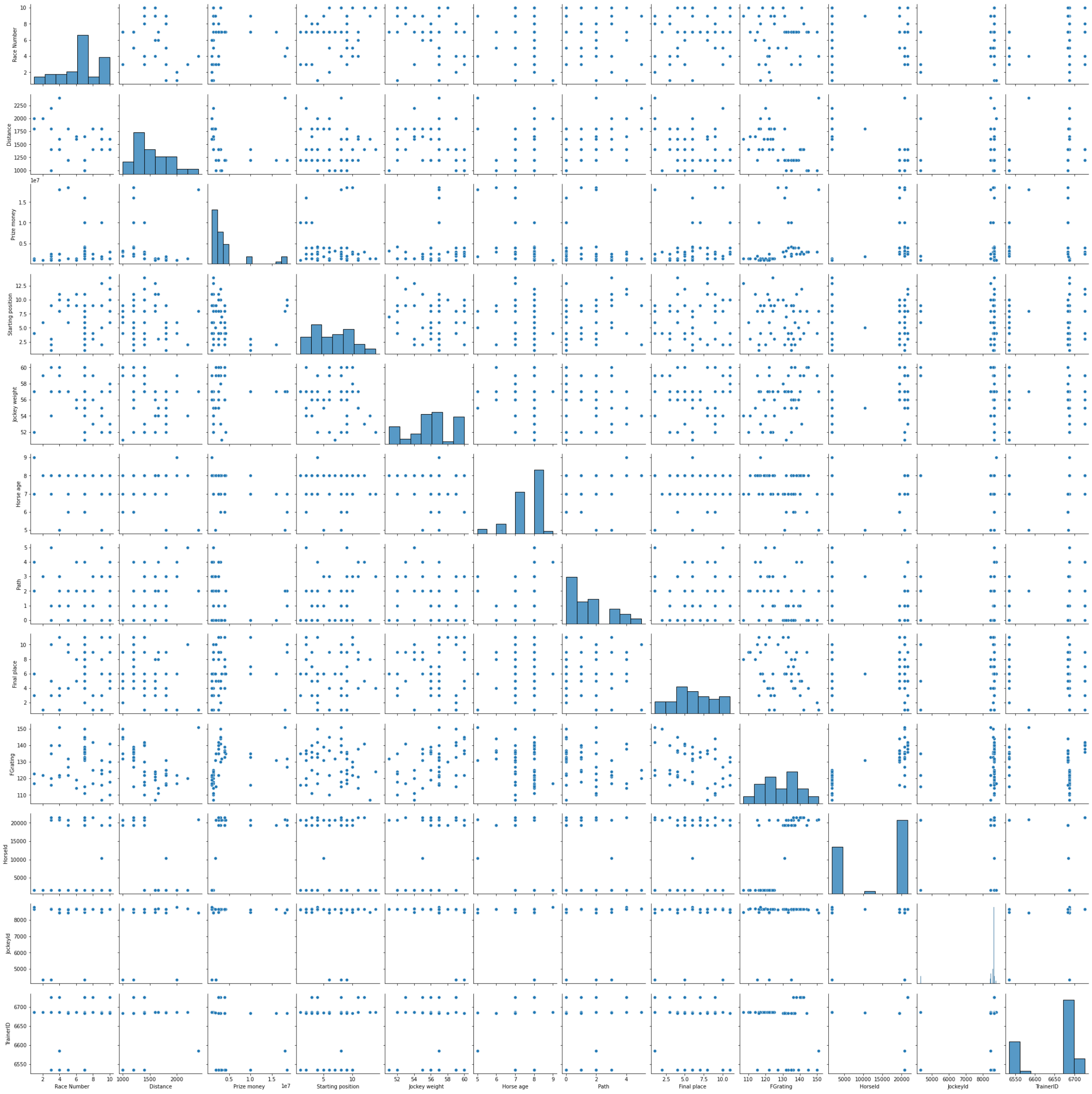
```
Index(['Dato', 'Track', 'Race Number', 'Distance', 'Surface', 'Prize mone
y',
       'Starting position', 'Jockey', 'Jockey weight', 'Country', 'Horse a
ge',
       'TrainerName', 'Race time', 'Path', 'Final place', 'FGrating', 'Odd
s',
       'RaceType', 'HorseId', 'JockeyId', 'TrainerID'],
      dtype='object')
```

In [7]:

```
sns.pairplot(df)
```

Out[7]:

```
<seaborn.axisgrid.PairGrid at 0x11b82dd5c40>
```



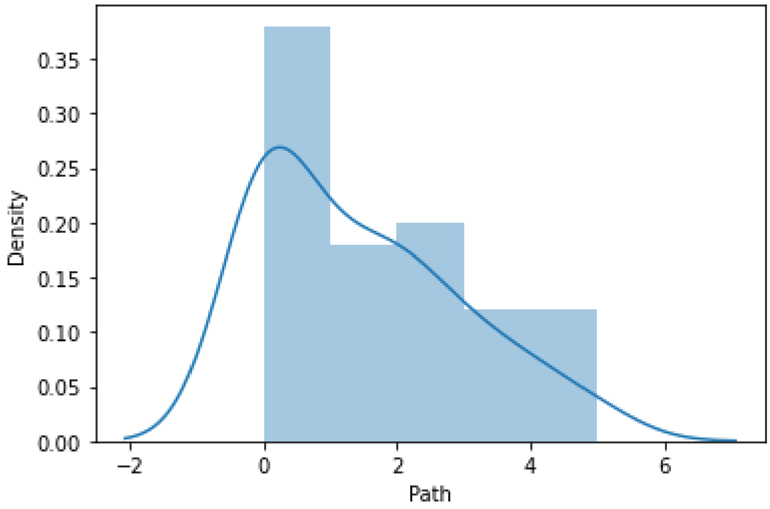
In [8]:

```
sns.distplot(df['Path'])
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2557:
FutureWarning: `distplot` is a deprecated function and will be removed in
a future version. Please adapt your code to use either `displot` (a figure
-level function with similar flexibility) or `histplot` (an axes-level fun
ction for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[8]:

```
<AxesSubplot:xlabel='Path', ylabel='Density'>
```

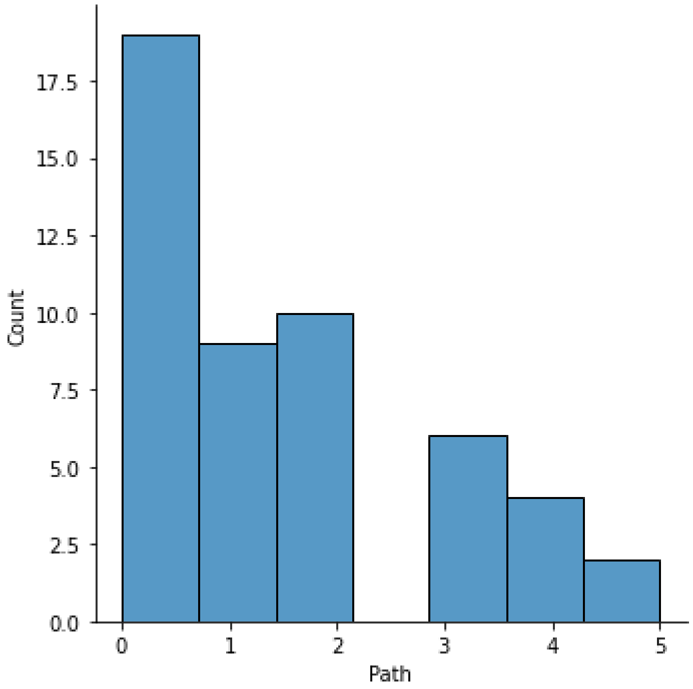


In [9]:

```
sns.displot(df["Path"])
```

Out[9]:

```
<seaborn.axisgrid.FacetGrid at 0x11b888093d0>
```



In [10]:

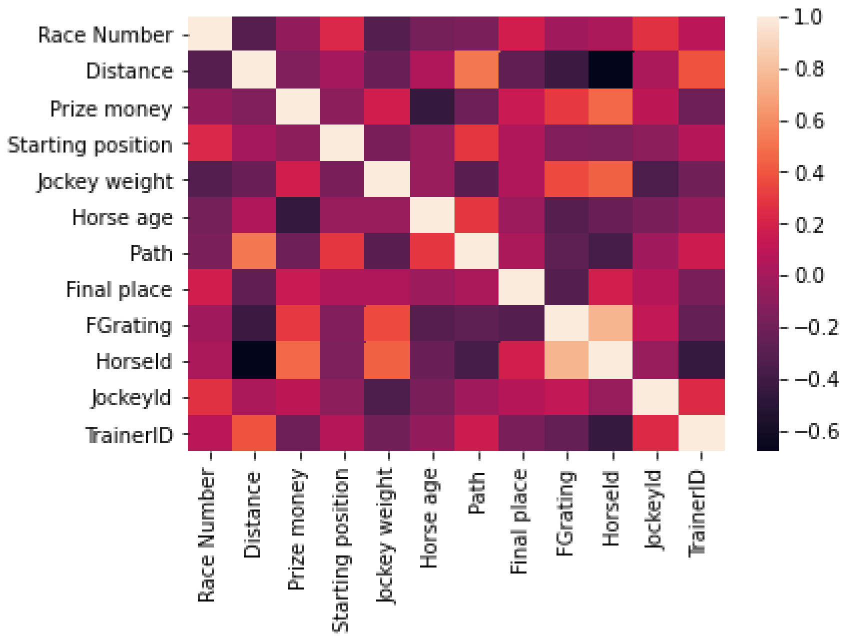
```
df1=df[['Dato', 'Track', 'Race Number', 'Distance', 'Surface', 'Prize money',
       'Starting position', 'Jockey', 'Jockey weight', 'Country', 'Horse age',
       'TrainerName', 'Race time', 'Path', 'Final place', 'FGrating', 'Odds',
       'RaceType', 'HorseId', 'JockeyId', 'TrainerID']]
```

In [11]:

```
sns.heatmap(df1.corr())
```

Out[11]:

```
<AxesSubplot:>
```



In [12]:

```
df2=df.dropna(axis=1)
df2
```

Out[12]:

	Dato	Track	Race Number	Distance	Surface	Prize money	Starting position	Jockey	Jockey weight	C
0	03.09.2017	Sha Tin	10	1400	Gress	1310000	6	K C Leung	52	S
1	16.09.2017	Sha Tin	10	1400	Gress	1310000	14	C Y Ho	52	S
2	14.10.2017	Sha Tin	10	1400	Gress	1310000	8	C Y Ho	52	S
3	11.11.2017	Sha Tin	9	1600	Gress	1310000	13	Brett Prebble	54	S
4	26.11.2017	Sha Tin	9	1600	Gress	1310000	9	C Y Ho	52	S
5	10.12.2017	Sha Tin	1	1800	Gress	1310000	4	C Y Ho	52	S
6	01.01.2018	Sha Tin	9	1800	Gress	1310000	9	C Schofield	54	S
7	04.02.2018	Sha Tin	5	1800	Gress	1310000	6	Joao Moreira	57	S
8	03.03.2018	Sha Tin	8	1800	Gress	1310000	3	C Y Ho	56	S
9	11.03.2018	Sha Tin	10	1600	Gress	1310000	8	C Y Ho	57	S
10	28.03.2018	Happy Valley	8	1800	Gress	1310000	9	M F Poon	53	S
11	11.04.2018	Happy Valley	6	1650	Gress	1310000	11	W M Lai	55	S
12	25.04.2018	Happy Valley	3	2200	Gress	1310000	2	W M Lai	54	S
13	09.05.2018	Happy Valley	7	1650	Gress	1310000	3	W M Lai	54	S
14	22.09.2018	Sha Tin	4	1600	Gress	920000	11	C Y Ho	57	S
15	07.10.2018	Sha Tin	6	1600	Gress	920000	9	C Y Ho	56	S
16	02.12.2018	Sha Tin	3	1800	Dirt	920000	1	C Schofield	57	S
17	23.12.2018	Sha Tin	2	2000	Gress	920000	6	Silvestre De Sousa	59	S
18	17.02.2019	Sha Tin	1	2000	Gress	920000	4	C Wong	57	S
19	06.12.2017	Happy Valley	9	1800	Gress	1860000	5	Z Purton	55	Da
20	01.10.2017	Sha Tin	7	1000	Gress	3000000	8	Z Purton	60	
21	22.10.2017	Sha Tin	7	1200	Gress	4000000	2	M Chadwick	60	
22	19.11.2017	Sha Tin	7	1200	Gress	4000000	8	M Chadwick	56	

	Dato	Track	Race Number	Distance	Surface	Prize money	Starting position	Jockey	Jockey weight	C
23	10.12.2017	Sha Tin	5	1200	Gress	18500000	9	M Chadwick	57	
24	01.01.2018	Sha Tin	10	1400	Gress	3000000	10	N Rawiller	58	
25	28.01.2018	Sha Tin	7	1200	Gress	10000000	3	Brett Prebble	57	
26	25.02.2018	Sha Tin	9	1400	Gress	10000000	2	Brett Prebble	57	
27	11.03.2018	Sha Tin	7	1200	Gress	2500000	4	N Callan	56	
28	08.04.2018	Sha Tin	7	1200	Gress	4000000	6	N Callan	56	
29	29.04.2018	Sha Tin	7	1200	Gress	16000000	2	N Callan	57	
30	01.10.2017	Sha Tin	7	1000	Gress	3000000	4	Tommy Berry	59	A
31	22.10.2017	Sha Tin	7	1200	Gress	4000000	4	Tommy Berry	59	A
32	19.11.2017	Sha Tin	7	1200	Gress	4000000	9	Tommy Berry	56	A
33	10.12.2017	Sha Tin	5	1200	Gress	18500000	10	Tommy Berry	57	A
34	07.01.2018	Sha Tin	7	1000	Gress	3000000	6	Tommy Berry	60	A
35	28.01.2018	Sha Tin	7	1200	Gress	10000000	1	Tommy Berry	57	A
36	11.03.2018	Sha Tin	7	1200	Gress	2500000	2	M F Poon	55	A
37	08.04.2018	Sha Tin	7	1200	Gress	4000000	5	S Clipperton	56	A
38	29.04.2018	Sha Tin	4	1400	Gress	2500000	10	Brett Prebble	60	A
39	01.10.2018	Sha Tin	7	1000	Gress	3250000	7	C Y Ho	51	A
40	21.10.2018	Sha Tin	7	1200	Gress	4250000	4	C Y Ho	52	A
41	25.11.2018	Sha Tin	3	1000	Gress	1950000	9	Silvestre De Sousa	60	A
42	19.12.2018	Sha Tin	7	1200	Dirt	1950000	8	Silvestre De Sousa	59	A
43	10.12.2017	Sha Tin	4	2400	Gress	18000000	8	Ryan Moore	57	
44	10.12.2017	Sha Tin	3	1400	Gress	2500000	3	N Callan	57	
45	01.01.2018	Sha Tin	10	1400	Gress	3000000	12	C Schofield	53	
46	18.02.2018	Sha Tin	8	1400	Gress	2500000	4	C Schofield	59	
47	11.03.2018	Sha Tin	7	1200	Gress	2500000	11	C Schofield	55	

	Dato	Track	Race Number	Distance	Surface	Prize money	Starting position	Jockey	Jockey weight	C
48	08.04.2018	Sha Tin	7	1200	Gress	4000000	3	C Schofield	56	
49	29.04.2018	Sha Tin	4	1400	Gress	2500000	8	Z Purton	59	

50 rows × 21 columns

In [13]:

```
x=df2[['Race Number', 'Distance', 'Prize money',
       'Starting position', 'Jockey weight', 'Horse age', 'Final place',
       'FGrating', 'HorseId', 'JockeyId', 'TrainerID']]
y=df2[['Path']]
```

In [14]:

```
from sklearn.model_selection import train_test_split
```

In [15]:

```
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.3)
```

In [16]:

```
from sklearn.linear_model import LinearRegression

lr=LinearRegression()
lr.fit(x_train,y_train)#ValueError: Input contains NaN, infinity or a value too large for
```

Out[16]:

```
LinearRegression()
```

In [17]:

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

```
[-4.43076772]
```

In [18]:

```
coef= pd.DataFrame(lr.coef_)
coef
```

Out[18]:

	0	1	2	3	4	5	6	7	
0	0.027056	0.004633	-5.097714e-09	0.116894	-0.139447	0.387862	0.274889	0.083757	0.00001

In [19]:

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

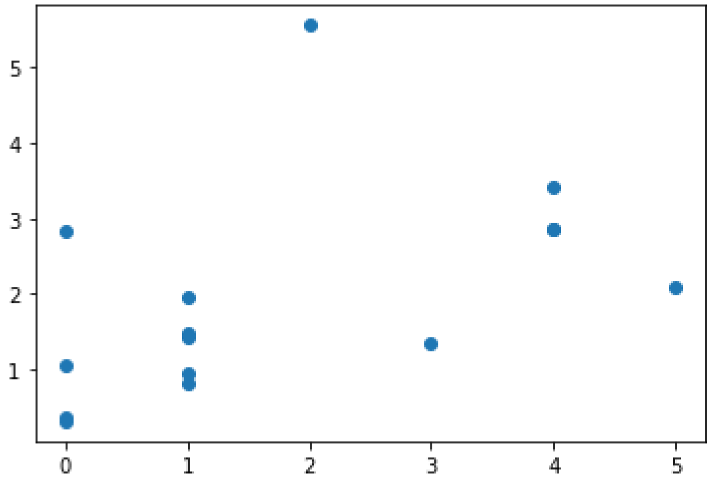
```
0.11437222962263349
```

In [20]:

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

Out[20]:

```
<matplotlib.collections.PathCollection at 0x11b8b5240d0>
```



In [21]:

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

Out[21]:

```
0.11437222962263349
```

In [22]:

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

Out[22]:

```
0.6310916282779255
```

In [23]:

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

In [24]:

```
rr=Ridge(alpha=10)
rr.fit(x_train,y_train)
```

Out[24]:

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

In [25]:

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

Out[25]:

```
0.0562680371456048
```

In [26]:

```
la=Lasso(alpha=10)
la.fit(x_train,y_train)
```

Out[26]:

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

In [27]:

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

Out[27]:

```
-0.07056762841163078
```

Elastic Net

In [28]:

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

Out[28]:

```
ElasticNet()
```

In [29]:

```
print(en.coef_)
```

```
[-0.00000000e+00  4.12360438e-03 -3.58102151e-08  3.03117206e-02
 -0.00000000e+00  0.00000000e+00  4.59841677e-02 -0.00000000e+00
  3.04407360e-05  3.48396088e-05 -5.75737035e-03]
```

In [30]:

```
print(en.intercept_)
```

```
[32.7155427]
```

In [31]:

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

```
[1.25827949 0.16575641 2.12819583 1.65271569 1.59906819 0.12055434
 0.09732249 1.45635567 5.27143452 3.18595029 1.20712465 1.63978288
 0.55393383 2.26475528 0.81506201]
```

In [32]:

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

```
-0.1556318161876067
```

Evaluation Metrics

In [33]:

```
from sklearn import metrics
```

In [34]:

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

```
Mean Absolute Error: 1.3908375404256357
```

In [35]:

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

```
Mean Squared Error: 3.2665859337569687
```

In [36]:

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

```
Root Mean Squared Error: 1.8073698940053662
```