In [3]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [4]:
```
df=pd.read_csv(r"E:\154\15_Horse Racing Results.csv")
df
```

```
---------------------------------------------------------------------------
ParserError                               Traceback (most recent call last)
<ipython-input-4-dff0846d0835> in <module>
----> 1 df=pd.read_csv(r"E:\154\15_Horse Racing Results.csv")
      2 df

C:\ProgramData\Anaconda3\lib\site-packages\pandas\io\parsers.py in read_csv
(filepath_or_buffer, sep, delimiter, header, names, index_col, usecols, squ
eeze, prefix, mangle_dupe_cols, dtype, engine, converters, true_values, fal
se_values, skipinitialspace, skiprows, skipfooter, nrows, na_values, keep_d
efault_na, na_filter, verbose, skip_blank_lines, parse_dates, infer_datetim
e_format, keep_date_col, date_parser, dayfirst, cache_dates, iterator, chun
ksize, compression, thousands, decimal, lineterminator, quotechar, quoting,
doublequote, escapechar, comment, encoding, dialect, error_bad_lines, warn_
bad_lines, delim_whitespace, low_memory, memory_map, float_precision, stora
ge_options)
    608     kwds.update(kwds_defaults)
    609 
--> 610     return _read(filepath_or_buffer, kwds)
```

In [ ]:
```
df.head()
```

DATA CLEANING AND DATA PREPROCESSING

In [ ]:
```
df.info()
```

In [ ]:
```
df.describe()
```

In [ ]:
```
df.columns
```

In [ ]:
```
df1=df.dropna(axis=1)
df1
```

In [ ]:
```
df1.columns
```

In [ ]:
```
df1=df1[['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']]
```

EDA AND VISUALIZATION

In [ ]:
```
sns.pairplot(df1)
```

In [ ]:
```
sns.distplot(df1['Distance'])
```

In [ ]:
```
sns.heatmap(df1.corr())
```

TO TRAIN THE MODEL AND MODEL BULDING

In [ ]:
```
x=df[['Race Number', 'Prize money',
       'Starting position',  'Jockey weight',  'Horse age', 'Final place', 'FG
y=df['Distance']
```

In [ ]:
```
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 [ ]:
```
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
lr.fit(x_train,y_train)
```

In [ ]:
```
lr.intercept_
```

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

In [ ]:
```
prediction =lr.predict(x_test)
plt.scatter(y_test,prediction)
```

ACCURACY

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

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

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

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

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

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

ElasticNet

```
In [ ]: from sklearn.linear_model import ElasticNet
        en = ElasticNet()
        en.fit(x_train,y_train)
```

```
In [ ]: print(en.coef_)
```

```
In [ ]: print(en.intercept_)
```

```
In [ ]: print(en.predict(x_test))
```

```
In [ ]: print(en.score(x_test,y_test))
```

Evaluation Metrics

```
In [ ]: from sklearn import metrics
```

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

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

```
In [ ]: print("Root Mean Absolute Error:",np.sqrt(metrics.mean_squared_error(y_test,pr
```