NIRF Rank Predictor

The goal of this machine learning problem is to build a predictive model that can accurately estimate the NIRF ranking of colleges and universities based on a set of relevant features and historical ranking data. By doing so, we aim to assist colleges and universities in assessing and enhancing their performance in various NIRF ranking parameters.

It is carried out in following steps:-

- Data Preprocessing and data cleaning
- Transforming raw data into features that can be used to create predictive models
- Exploratory Data Analysis
- Assessing various Machine Learning models
- Training and Testing ML models
- Finalizing the best model suited.

As per the methodology of NIRF, there are five ranking parameters which are as follows:

- Teaching, learning, and resources
- Research and professional practice
- Graduation outcomes
- Outreach and inclusivity
- Peer perception

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

Data of year 2016, 2017, 2018, 2019, 2020, 2021 is arranged from kaggle. The dataset contains:-

- Institute ID
- Institute Name
- City where it is located
- State
- Rank of various years
- TLR (Teaching, Learning and Resources)
- RPC (Research And Professional Practice)
- GO (Graduation Outcomes)
- OI (Outreach And Inclusivity)
- Perception

The dataset is then combined into one dataset.

In [2]:
```
data_2016 = pd.read_csv('EngineeringRanking_2016.csv')
```

In [3]:
```
data_2016.head()
```

Out[3]:

	Institute Id	Institute Name	City	State	Score	Rank	TLR	RPC	GO	OI	Perception
0	NIRF-ENGG-INF-77	Indian Institute Of Technology, Madras	Chennai	Tamil Nadu	89.41	1	88.26	94.02	81.81	86.11	98
1	NIRF-ENGG-INF-312	Indian Institute Of Technology, Bombay	Bombay	Maharashtra	87.66	2	85.93	94.14	84.97	74.84	99
2	NIRF-ENGG-INF-300	Indian Institute Of Technology, Kharagpur	Kharagpur	West Bengal	83.91	3	76.23	92.68	83.95	78.05	97
3	NIRF-ENGG-INF-79	Indian Institute Of Technology, Delhi	New Delhi	Delhi	82.02	4	80.27	91.62	74.72	66.17	98
4	NIRF-ENGG-INF-228	Indian Institute Of Technology, Kanpur	Kanpur	Uttar Pradesh	81.07	5	66.08	93.52	85.62	70.59	98

In [4]:
```
data_2017 = pd.read_csv('EngineeringRanking_2017.csv')
```

In [5]:
```
data_2017.head()
```

Out[5]:

	Institute Id	Institute Name	City	State	Score	Rank	TLR	RPC	GO	OI	Perception
0	IR17-ENGG-1-1-77	Indian Institute of Technology Madras	Chennai	Tamil Nadu	87.96	1	91.85	92.60	83.78	77.19	81.46
1	IR17-ENGG-2-18633	Indian Institute of Technology Bombay	Mumbai	Maharashtra	87.87	2	91.15	94.68	83.64	69.70	84.24
2	IR17-ENGG-2-18630	Indian Institute of Technology Kharagpur	Kharagpur	West Bengal	81.93	3	76.03	89.23	88.02	74.11	73.43
3	IR17-ENGG-2-1-79	Indian Institute of Technology Delhi	New Delhi	Delhi	81.08	4	79.63	89.47	77.45	71.41	77.24
4	IR17-ENGG-2-18248	Indian Institute of Technology Kanpur	Kanpur	Uttar Pradesh	76.83	5	84.28	77.28	74.29	61.35	73.59

In [6]:
```
data_2018 = pd.read_csv('EngineeringRanking_2018.csv')
```

In [7]:
```
data_2018.head()
```

Out[7]:

	Institute Id	Institute Name	City	State	Score	Rank	TLR	RPC	GO	OI	Perception
0	IR-2-E-OE-U-0456	Indian Institute of Technology Madras	Chennai	Tamil Nadu	88.95	1	93.83	91.44	84.91	63.88	100.00
1	IR-3-E-OEM-U-0306	Indian Institute of Technology Bombay	Mumbai	Maharashtra	84.82	2	89.61	96.04	76.53	44.71	93.48
2	IR-3-E-OEM-I-1074	Indian Institute of Technology Delhi	New Delhi	Delhi	82.18	3	80.83	89.35	81.47	59.72	88.60
3	IR-5-E-OEMAL-U-0573	Indian Institute of Technology Kharagpur	Kharagpur	West Bengal	77.78	4	73.73	84.26	85.65	53.99	78.51
4	IR-3-E-OEM-I-1075	Indian Institute of Technology Kanpur	Kanpur	Uttar Pradesh	75.24	5	78.51	77.15	78.99	41.46	85.89

In [8]:
```
data_2019 = pd.read_csv('EngineeringRanking_2019.csv')
```

In [9]:
```
data_2019.head()
```

Out[9]:

	Institute Id	Institute Name	City	State	Score	Rank	TLR	RPC	GO	OI	Perception
0	IR-E-U-0456	Indian Institute of Technology Madras	Chennai	Tamil Nadu	89.05	1	93.55	92.39	84.36	63.99	100.00
1	IR-E-I-1074	Indian Institute of Technology Delhi	New Delhi	Delhi	85.36	2	85.80	96.18	80.32	56.19	90.85
2	IR-E-U-0306	Indian Institute of Technology Bombay	Mumbai	Maharashtra	84.40	3	89.12	95.30	76.47	48.17	89.61
3	IR-E-U-0573	Indian Institute of Technology Kharagpur	Kharagpur	West Bengal	79.41	4	73.14	88.20	84.12	57.79	84.14
4	IR-E-I-1075	Indian Institute of Technology Kanpur	Kanpur	Uttar Pradesh	77.57	5	79.07	81.79	82.56	46.61	81.35

In [10]:
```
data_2020 = pd.read_csv('EngineeringRanking_2020.csv')
```

In [11]:
```
data_2020.head()
```

Out[11]:

	Institute Id	Institute Name	City	State	Score	Rank	TLR	RPC	GO	OI	Perception
0	IR-E-U-0456	Indian Institute of Technology Madras	Chennai	Tamil Nadu	89.93	1	95.42	94.64	83.90	61.31	100.00
1	IR-E-I-1074	Indian Institute of Technology Delhi	New Delhi	Delhi	88.08	2	90.79	96.15	80.36	64.81	94.46
2	IR-E-U-0306	Indian Institute of Technology Bombay	Mumbai	Maharashtra	85.08	3	91.00	93.37	77.60	49.99	92.51
3	IR-E-I-1075	Indian Institute of Technology Kanpur	Kanpur	Uttar Pradesh	82.18	4	86.22	82.08	88.44	54.21	85.78
4	IR-E-U-0573	Indian Institute of Technology Kharagpur	Kharagpur	West Bengal	80.56	5	77.32	87.11	83.21	56.62	89.31

In [12]:
```
data_2021 = pd.read_csv('EngineeringRanking_2021.csv')
```

In [13]:
```
data_2021.head()
```

Out[13]:

	Institute Id	Institute Name	City	State	Score	Rank	TLR	RPC	GO	OI	Perception
0	IR-E-U-0456	Indian Institute of Technology Madras	Chennai	Tamil Nadu	90.19	1	95.47	96.43	81.92	62.44	100.00
1	IR-E-I-1074	Indian Institute of Technology Delhi	New Delhi	Delhi	88.96	2	91.76	95.82	80.97	66.39	98.63
2	IR-E-U-0306	Indian Institute of Technology Bombay	Mumbai	Maharashtra	85.16	3	89.32	92.56	79.71	53.68	92.88
3	IR-E-I-1075	Indian Institute of Technology Kanpur	Kanpur	Uttar Pradesh	83.22	4	86.71	83.13	89.79	55.96	87.05
4	IR-E-U-0573	Indian Institute of Technology Kharagpur	Kharagpur	West Bengal	82.03	5	80.51	88.59	83.01	58.46	88.50

In [14]:
```
data_2016['year'],data_2017['year'],data_2018['year'],data_2019['year'],data_2020['year
```

In [15]:
```
df = [data_2016,data_2017,data_2018,data_2019,data_2020,data_2021]
df_combined = pd.concat(df , axis =0 , ignore_index = 'True')
```

In [16]:
```
# Combined dataset
df_combined.head()
```

Out[16]:

	Institute Id	Institute Name	City	State	Score	Rank	TLR	RPC	GO	OI	Perception
0	NIRF-ENGG-INF-77	Indian Institute Of Technology, Madras	Chennai	Tamil Nadu	89.41	1	88.26	94.02	81.81	86.11	98.0
1	NIRF-ENGG-INF-312	Indian Institute Of Technology, Bombay	Bombay	Maharashtra	87.66	2	85.93	94.14	84.97	74.84	99.0
2	NIRF-ENGG-INF-300	Indian Institute Of Technology, Kharagpur	Kharagpur	West Bengal	83.91	3	76.23	92.68	83.95	78.05	97.0
3	NIRF-ENGG-INF-79	Indian Institute Of Technology, Delhi	New Delhi	Delhi	82.02	4	80.27	91.62	74.72	66.17	98.0
4	NIRF-ENGG-INF-228	Indian Institute Of Technology, Kanpur	Kanpur	Uttar Pradesh	81.07	5	66.08	93.52	85.62	70.59	98.0

In [17]:
```
df_combined
```

Out[17]:

	Institute Id	Institute Name	City	State	Score	Rank	TLR	RPC	GO	OI	Perceptic
0	NIRF-ENGG-INF-77	Indian Institute Of Technology, Madras	Chennai	Tamil Nadu	89.41	1	88.26	94.02	81.81	86.11	98.(
1	NIRF-ENGG-INF-312	Indian Institute Of Technology, Bombay	Bombay	Maharashtra	87.66	2	85.93	94.14	84.97	74.84	99.(
2	NIRF-ENGG-INF-300	Indian Institute Of Technology, Kharagpur	Kharagpur	West Bengal	83.91	3	76.23	92.68	83.95	78.05	97.(
3	NIRF-ENGG-INF-79	Indian Institute Of Technology, Delhi	New Delhi	Delhi	82.02	4	80.27	91.62	74.72	66.17	98.(
4	NIRF-ENGG-INF-228	Indian Institute Of Technology, Kanpur	Kanpur	Uttar Pradesh	81.07	5	66.08	93.52	85.62	70.59	98.(
...	...	...	...	...	...	...	...	...	...	...	...
895	IR-E-C-1438	The National Institute of Engineering	Mysore	Karnataka	32.52	196	53.79	2.33	51.03	50.48	4.2
896	IR-E-C-33584	K. J. Somaiya College of Engineering	Mumbai	Maharashtra	32.48	197	52.22	3.33	58.94	38.08	2.1
897	IR-E-C-27400	Kakatiya Institute of Technology & Science	Warangal	Telangana	32.48	197	55.80	1.64	49.13	49.25	4.9
898	IR-E-C-11015	Walchand College of Engineering	Sangli	Maharashtra	32.46	199	48.25	4.54	56.11	47.93	6.1
899	IR-E-U-0037	Sri Venkateswara University	Tirupati	Andhra Pradesh	32.42	200	41.76	25.59	37.58	43.31	3.5

900 rows × 12 columns

In [18]:
```
df_combined.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 900 entries, 0 to 899
Data columns (total 12 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   Institute Id    900 non-null    object 
 1   Institute Name  900 non-null    object 
 2   City            900 non-null    object 
 3   State           900 non-null    object 
 4   Score           900 non-null    float64
 5   Rank            900 non-null    object 
 6   TLR             900 non-null    float64
 7   RPC             900 non-null    float64
 8   GO              900 non-null    float64
 9   OI              900 non-null    float64
 10  Perception      900 non-null    float64
 11  year            900 non-null    object 
dtypes: float64(6), object(6)
memory usage: 84.5+ KB
```

There are certain anomalies present in the data like in some fields Rank 21A , 26A is present. The letter is removed using lambda function and the datatype of rank is converted from string to float.

In [19]:
```
df_combined[219:228]
```

Out[19]:

	Institute Id	Institute Name	City	State	Score	Rank	TLR	RPC	GO	OI	Pe
219	IR-2-E-EM-I-1480	Thapar Institute of Engineering and Technology	Patiala	Punjab	56.14	20	69.77	45.75	73.11	56.55	
220	IR-2-E-OE-U-0237	National Institute of Technology Surathkal	Surathkal	Karnataka	53.16	21	60.34	38.36	71.27	51.10	
221	IR-2-E-OE-U-0584	Indian Institute of Engineering Science and Te...	Howrah	West Bengal	53.24	21A	67.15	45.47	61.09	40.27	
222	IR-2-E-OE-U-0378	Indian Institute of Technology Ropar	Rupnagar	Punjab	52.80	22	77.84	29.53	65.29	60.61	
223	IR-2-E-OE-U-0255	Indian Institute of Space Science and Technology	Thiruvananthapuram	Kerala	52.74	23	78.87	20.76	62.12	60.82	
224	IR-2-E-OE-U-0064	Indian Institute of Technology Patna	Patna	Bihar	52.37	24	74.43	35.03	65.64	49.30	
225	IR-2-E-OE-U-0025	National Institute of Technology Warangal	Warangal	Telangana	51.82	25	67.25	31.43	72.38	54.79	
226	IR-5-E-OEMAP-U-0202	Birla Institute of Technology	Ranchi	Jharkhand	51.12	26	71.22	36.61	59.88	51.69	
227	IR-2-E-OE-U-0184	Indian Institute of Technology Mandi	Mandi	Himachal Pradesh	51.28	26A	76.90	30.48	63.07	54.64	

In [20]:
```
df_combined['Rank'] = df_combined['Rank'].apply(lambda x: x if str(x).isdigit() else x[
```

In [21]:
```
df_combined['Rank'] = df_combined['Rank'].astype('float64')
```

In [22]:
```
df_combined[219:228]
```

Out[22]:

	Institute Id	Institute Name	City	State	Score	Rank	TLR	RPC	GO	OI	Pe
219	IR-2-E-EM-I-1480	Thapar Institute of Engineering and Technology	Patiala	Punjab	56.14	20.0	69.77	45.75	73.11	56.55	
220	IR-2-E-OE-U-0237	National Institute of Technology Surathkal	Surathkal	Karnataka	53.16	21.0	60.34	38.36	71.27	51.10	
221	IR-2-E-OE-U-0584	Indian Institute of Engineering Science and Te...	Howrah	West Bengal	53.24	21.0	67.15	45.47	61.09	40.27	
222	IR-2-E-OE-U-0378	Indian Institute of Technology Ropar	Rupnagar	Punjab	52.80	22.0	77.84	29.53	65.29	60.61	
223	IR-2-E-OE-U-0255	Indian Institute of Space Science and Technology	Thiruvananthapuram	Kerala	52.74	23.0	78.87	20.76	62.12	60.82	
224	IR-2-E-OE-U-0064	Indian Institute of Technology Patna	Patna	Bihar	52.37	24.0	74.43	35.03	65.64	49.30	
225	IR-2-E-OE-U-0025	National Institute of Technology Warangal	Warangal	Telangana	51.82	25.0	67.25	31.43	72.38	54.79	
226	IR-5-E-OEMAP-U-0202	Birla Institute of Technology	Ranchi	Jharkhand	51.12	26.0	71.22	36.61	59.88	51.69	
227	IR-2-E-OE-U-0184	Indian Institute of Technology Mandi	Mandi	Himachal Pradesh	51.28	26.0	76.90	30.48	63.07	54.64	

In [23]:
```
df_combined.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 900 entries, 0 to 899
Data columns (total 12 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   Institute Id    900 non-null    object 
 1   Institute Name  900 non-null    object 
 2   City            900 non-null    object 
 3   State           900 non-null    object 
 4   Score           900 non-null    float64
 5   Rank            900 non-null    float64
 6   TLR             900 non-null    float64
 7   RPC             900 non-null    float64
 8   GO              900 non-null    float64
 9   OI              900 non-null    float64
 10  Perception      900 non-null    float64
 11  year            900 non-null    object 
dtypes: float64(7), object(5)
memory usage: 84.5+ KB
```

The Institute name in 2016 is different from other year datasets so a operation is carried out to make it same.

In [24]:
```
data_2016.head(1)
```

Out[24]:

	Institute Id	Institute Name	City	State	Score	Rank	TLR	RPC	GO	OI	Perception	year
0	NIRF-ENGG-INF-77	Indian Institute Of Technology, Madras	Chennai	Tamil Nadu	89.41	1	88.26	94.02	81.81	86.11	98	2016

In [25]:
```
data_2017.head(1)
```

Out[25]:

	Institute Id	Institute Name	City	State	Score	Rank	TLR	RPC	GO	OI	Perception	year
0	IR17-ENGG-1-1-77	Indian Institute of Technology Madras	Chennai	Tamil Nadu	87.96	1	91.85	92.6	83.78	77.19	81.46	2017

In [26]:
```
data_2016['Institute Name'] = data_2016['Institute Name'].str.replace(',', '')
```

In [27]:
```
data_2016.head(1)
```

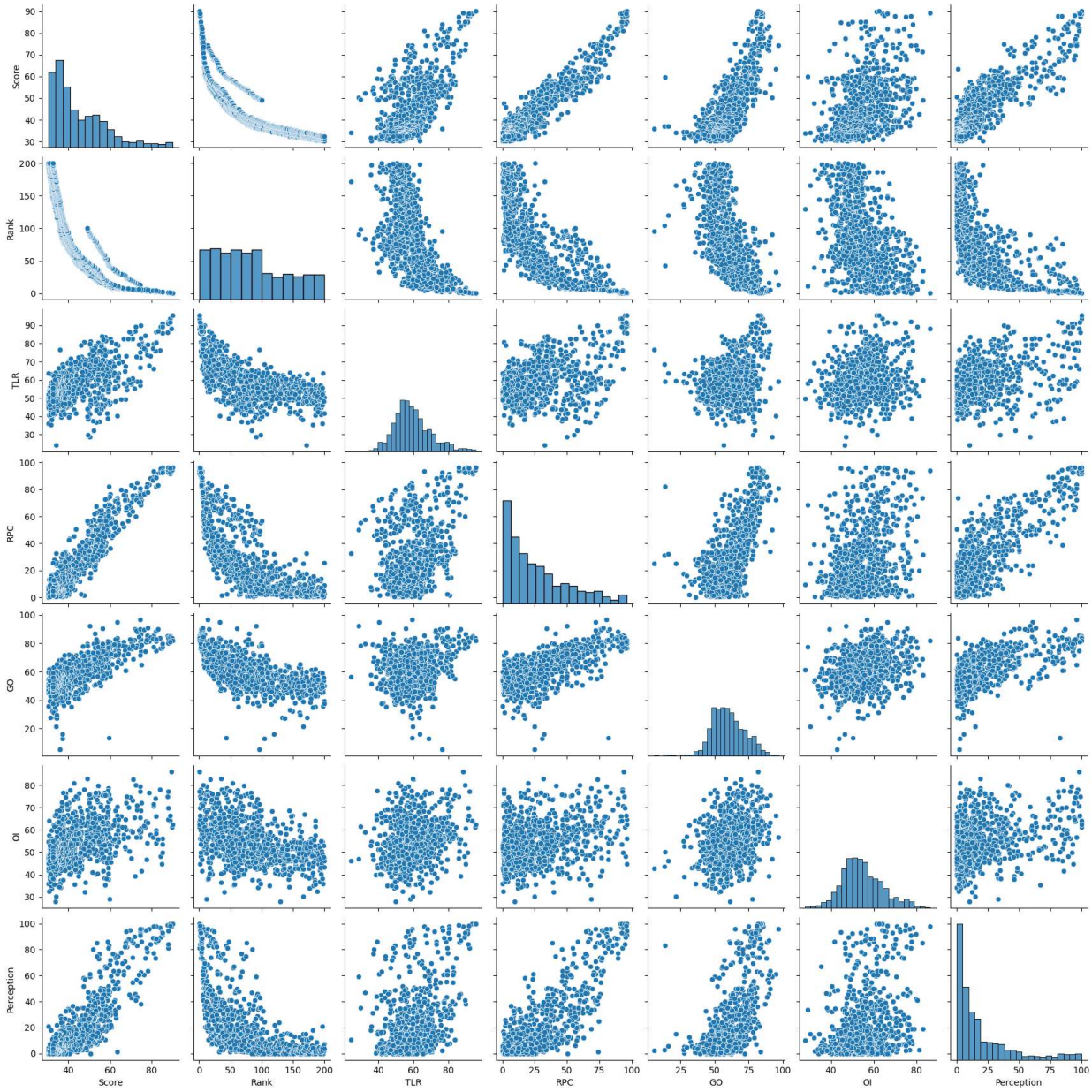
Out[27]:

	Institute Id	Institute Name	City	State	Score	Rank	TLR	RPC	GO	OI	Perception	year
0	NIRF-ENGG-INF-77	Indian Institute Of Technology Madras	Chennai	Tamil Nadu	89.41	1	88.26	94.02	81.81	86.11	98	2016

EXPLORATORY DATA ANALYSIS

In [28]:
```
sns.pairplot(df_combined)
```

Out[28]:
```
<seaborn.axisgrid.PairGrid at 0x1b7bd1835d0>
```



Mean RPC is low implies research sector is weak in clgs..

In [29]:
```
df_combined.describe()
```

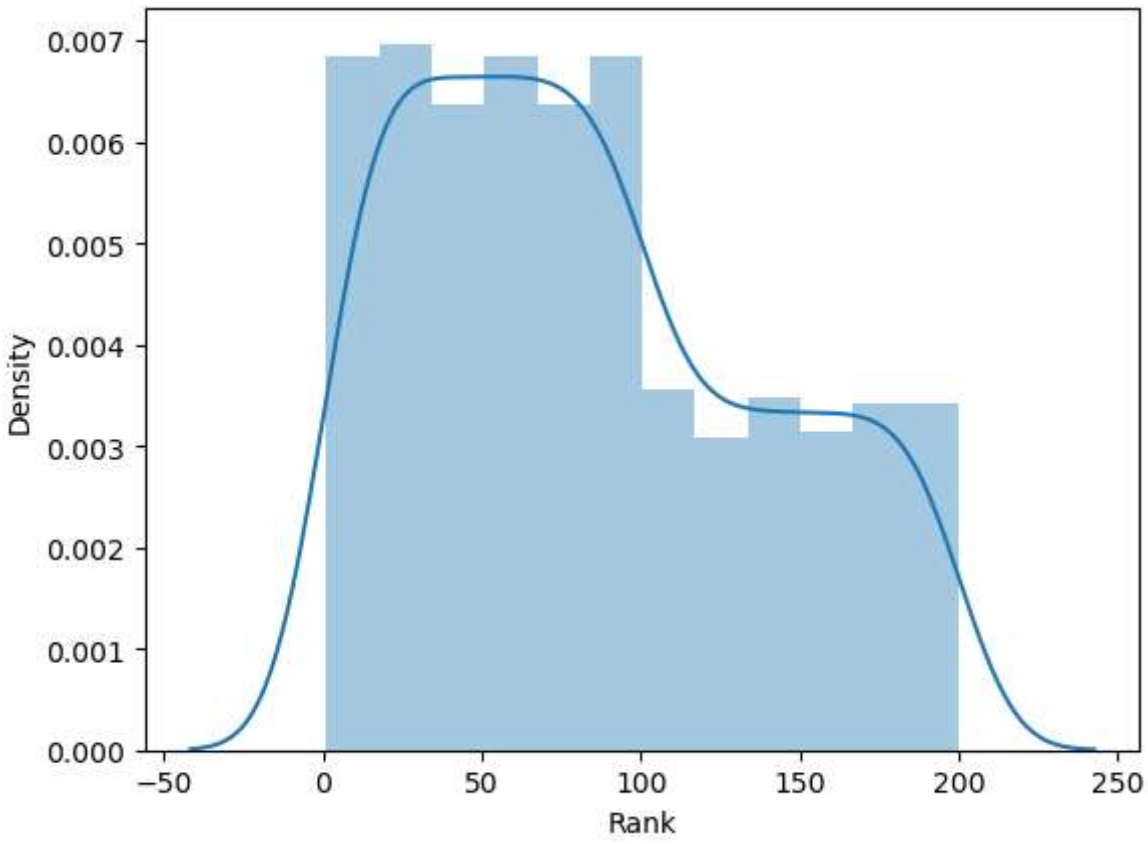
Out[29]:

	Score	Rank	TLR	RPC	GO	OI	Perception
count	900.000000	900.000000	900.000000	900.000000	900.000000	900.000000	900.000000
mean	45.122317	83.700000	59.289333	26.604467	59.949256	54.611733	20.129267
std	12.890893	55.335713	10.674734	23.832522	12.112018	9.368713	24.133590
min	30.310000	1.000000	24.310000	0.170000	5.460000	28.020000	0.000000
25%	35.465000	38.000000	52.340000	6.960000	51.617500	48.217500	3.560000
50%	40.495000	75.000000	57.820000	19.365000	58.995000	53.505000	10.320000
75%	52.755000	125.250000	64.847500	39.247500	67.667500	60.335000	27.402500
max	90.190000	200.000000	95.470000	96.430000	96.800000	86.110000	100.000000

In [30]:
```
sns.distplot(df_combined['Rank'])
```

```
C:\Users\abhyu\AppData\Local\Temp\ipykernel_364\3765976300.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df_combined['Rank'])
```

Out[30]:
```
<Axes: xlabel='Rank', ylabel='Density'>
```



In [31]:
```
df_combined.columns
```

Out[31]:
```
Index(['Institute Id', 'Institute Name', 'City', 'State', 'Score', 'Rank',
       'TLR', 'RPC', 'GO', 'OI', 'Perception', 'year'],
      dtype='object')
```

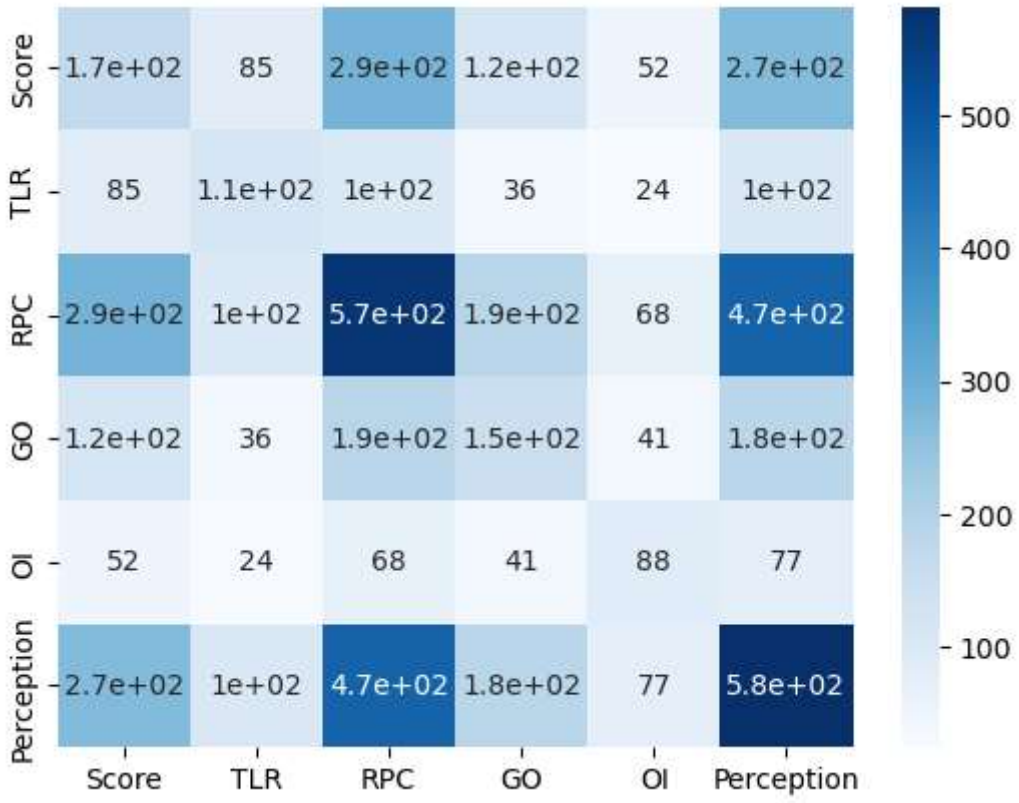
In [32]:
```
df_parameters = df_combined.drop(columns = ['Institute Id', 'Institute Name', 'City', 'S
```

In [33]:
```
df_parameters.head()
```

Out[33]:

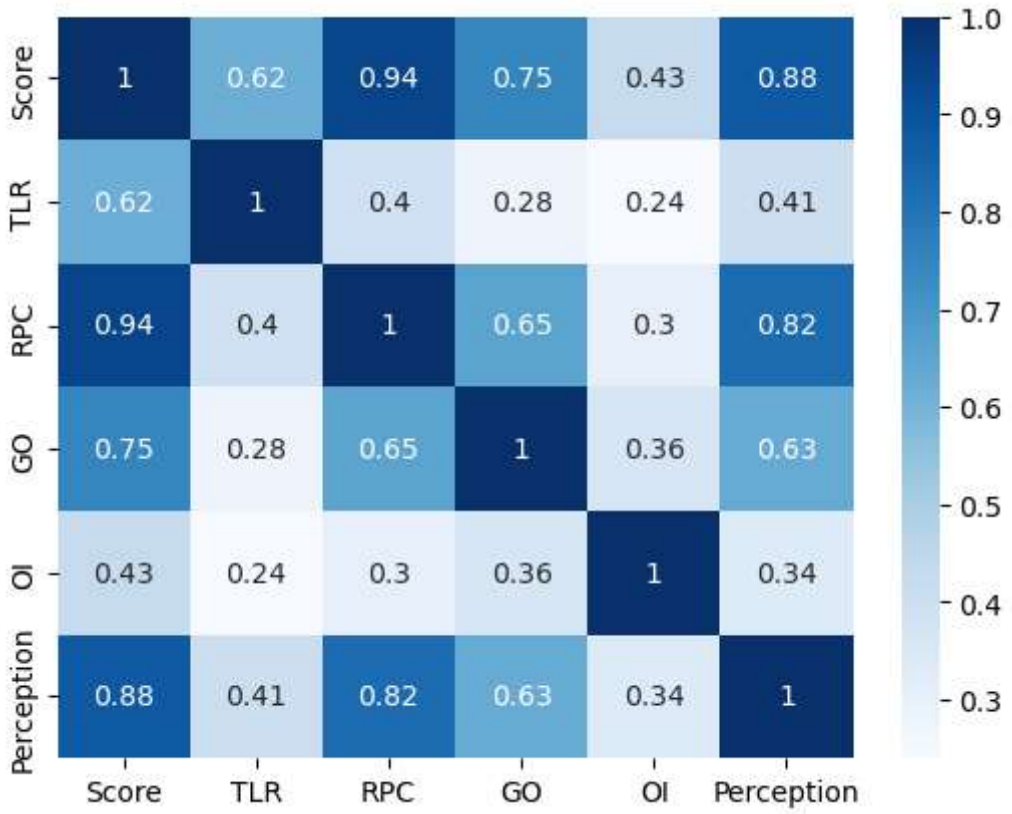
	Score	TLR	RPC	GO	OI	Perception
0	89.41	88.26	94.02	81.81	86.11	98.0
1	87.66	85.93	94.14	84.97	74.84	99.0
2	83.91	76.23	92.68	83.95	78.05	97.0
3	82.02	80.27	91.62	74.72	66.17	98.0
4	81.07	66.08	93.52	85.62	70.59	98.0

In [34]:
```
covmat = df_parameters.cov()
ax = sns.heatmap(covmat, annot = True , cmap = 'Blues')
plt.figure(figsize=(15,15))
plt.show()
```



<Figure size 1500x1500 with 0 Axes>

In [35]:
```
corrmat = df_parameters.corr()
ax = sns.heatmap(corrmat, annot = True , cmap = 'Blues')
plt.figure(figsize=(15,15))
plt.show()
```




```
<Figure size 1500x1500 with 0 Axes>
```

Here correlation of rpc with score is 0.94 it implies it greatly affects the overall score.

Liner Regressor

In [36]:
```
X = df_combined[['TLR' , 'RPC' , 'GO' , 'OI' , 'Perception']]
y = df_combined['Score']
```

In [37]:
```
from sklearn.model_selection import train_test_split
```

In [38]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=10
```

In [39]:
```
from sklearn.linear_model import LinearRegression
```

In [40]:
```
lm = LinearRegression()
```

In [41]:
```
lm.fit(X_train,y_train)
```

Out[41]:
```
▾ LinearRegression
LinearRegression()
```

In [42]:
```
# print the intercept
print(lm.intercept_)
```

```
-0.3774396671306164
```

In [43]:
```
coeff_df = pd.DataFrame(lm.coef_,X.columns,columns=['Coefficient'])
coeff_df
```

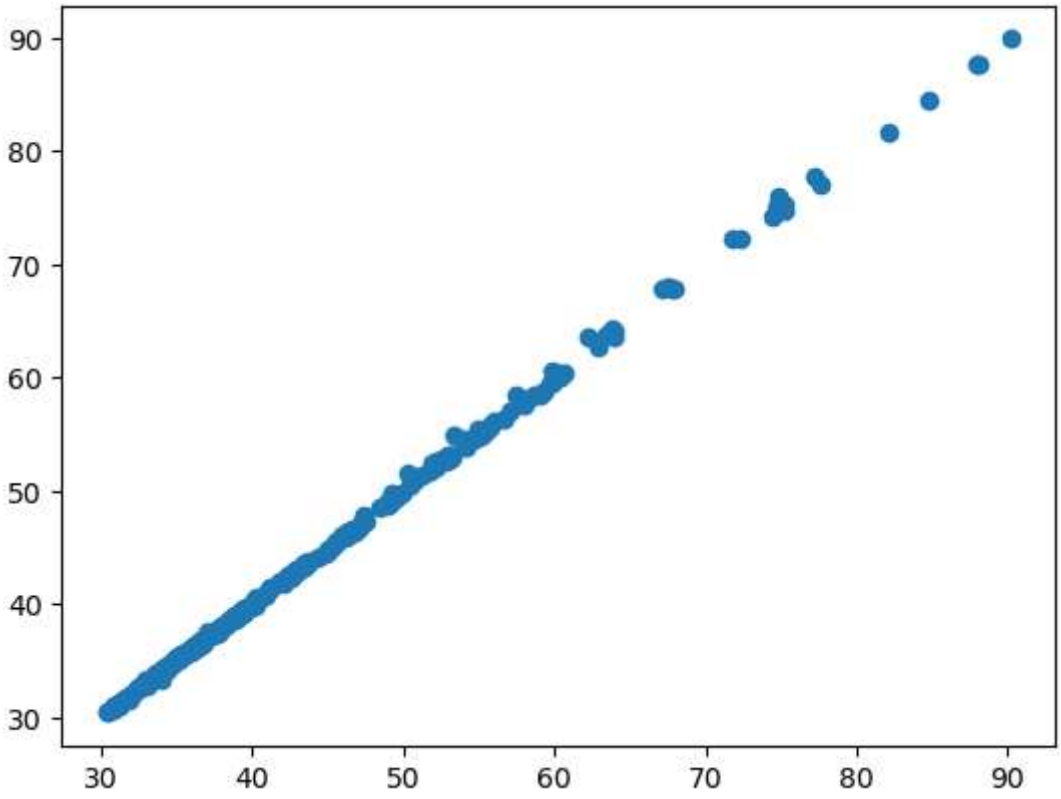
Out[43]:

	Coefficient
TLR	0.314024
RPC	0.294388
GO	0.191477
OI	0.102426
Perception	0.098394

In [44]:
```
predictions = lm.predict(X_test)
```

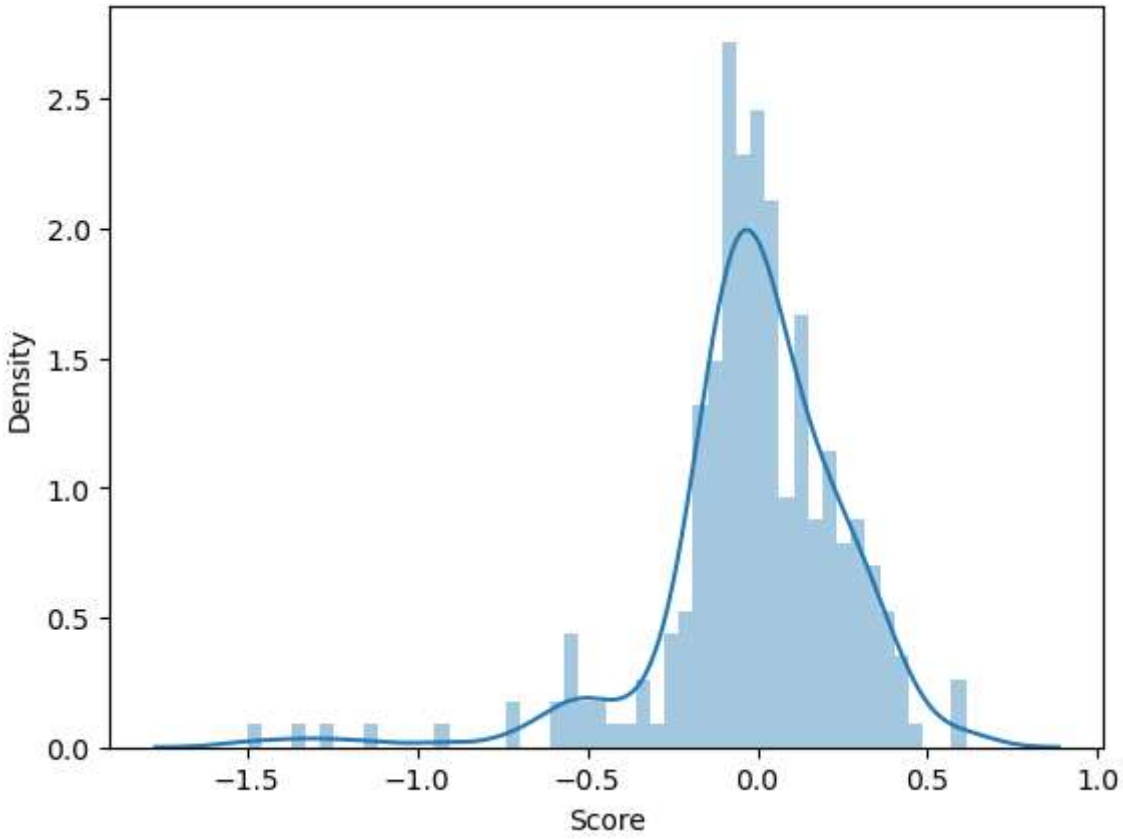
In [45]:
```
plt.scatter(y_test,predictions)
```

Out[45]:
```
<matplotlib.collections.PathCollection at 0x1b7c7edfc90>
```



In [46]:
```
sns.distplot((y_test-predictions),bins=50);
```

```
C:\Users\abhyu\AppData\Local\Temp\ipykernel_364\1326397652.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot((y_test-predictions),bins=50);
```



In [47]:
```
from sklearn import metrics
```

In [48]:
```
print('MAE:', metrics.mean_absolute_error(y_test, predictions))
print('MSE:', metrics.mean_squared_error(y_test, predictions))
print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, predictions)))
```

```
MAE: 0.18383221952155138
MSE: 0.07726423877763
RMSE: 0.2779644559608836
```

Decision Tree Regressor

In [49]:
```
from sklearn.tree import DecisionTreeRegressor
```

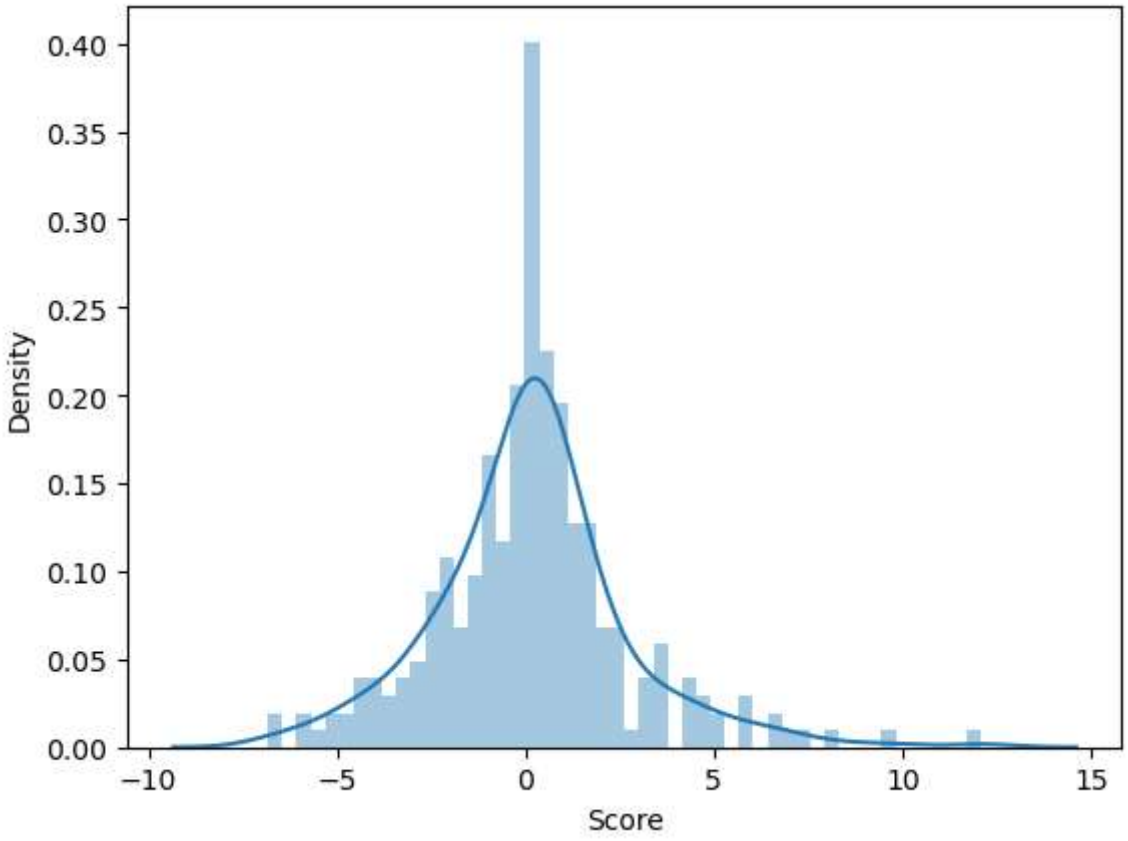
In [50]:
```
dtree = DecisionTreeRegressor()
```

In [51]:
```
dtree.fit(X_train,y_train)
```

Out[51]:
```
▾ DecisionTreeRegressor
DecisionTreeRegressor()
```

In [54]:
```
sns.distplot((y_test-pred1),bins=50);
```

```
C:\Users\abhyu\AppData\Local\Temp\ipykernel_364\172980259.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot((y_test-pred1),bins=50);
```



In [55]:
```
pred1 = dtree.predict(X_test)
```

In [56]:
```
print('MAE:', metrics.mean_absolute_error(y_test, pred1))
print('MSE:', metrics.mean_squared_error(y_test, pred1))
print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, pred1)))
```

```
MAE: 1.7935185185185187
MSE: 6.6213411111111125
RMSE: 2.5731966716734096
```

Random Forest Regressor

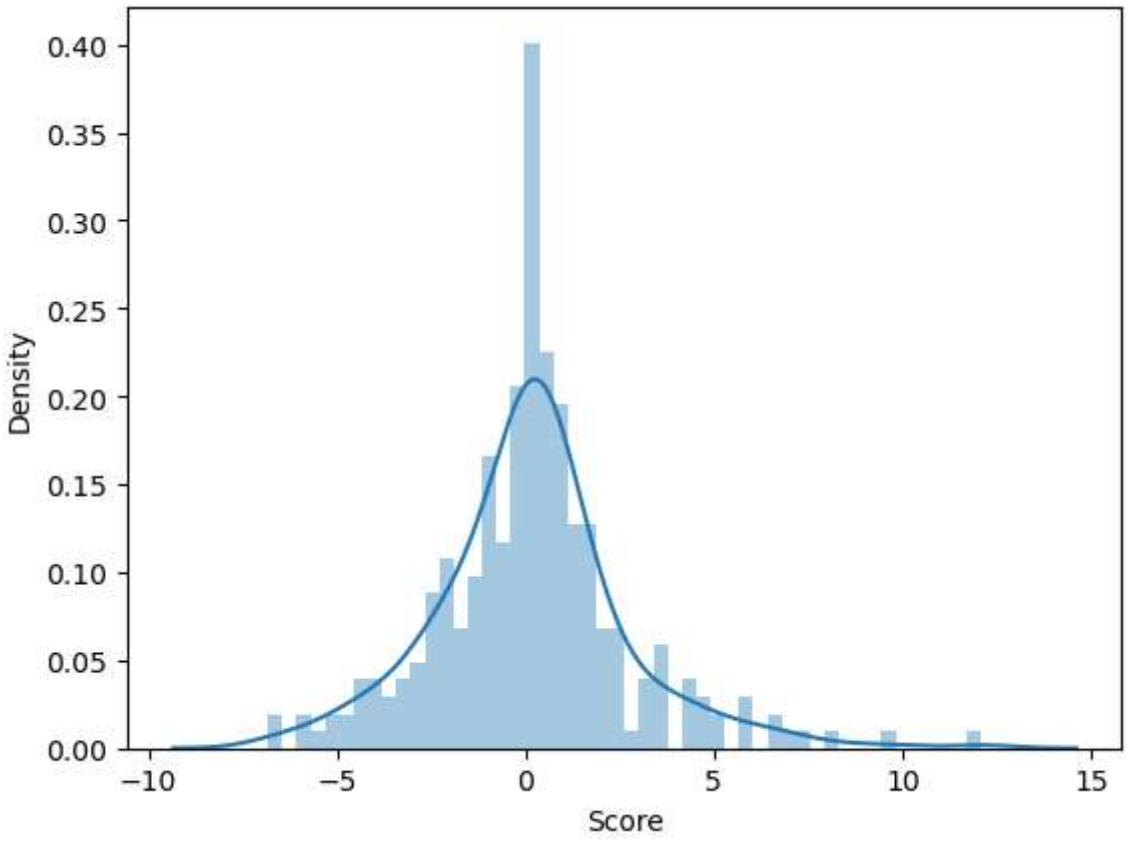
In [57]:
```
from sklearn.ensemble import RandomForestRegressor
rfc = RandomForestRegressor(n_estimators=100)
rfc.fit(X_train, y_train)
```

Out[57]:
```
RandomForestRegressor
RandomForestRegressor()
```

In [58]:
```
rfc_pred = rfc.predict(X_test)
```

In [59]:
```
sns.distplot((y_test-pred1),bins=50);
```

```
C:\Users\abhyu\AppData\Local\Temp\ipykernel_364\172980259.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot((y_test-pred1),bins=50);
```



In [60]:
```
print('MAE:', metrics.mean_absolute_error(y_test, rfc_pred))
print('MSE:', metrics.mean_squared_error(y_test, rfc_pred))
print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, rfc_pred)))
```

```
MAE: 1.0584419629629649
MSE: 2.43276812119593
RMSE: 1.5597333493889043
```

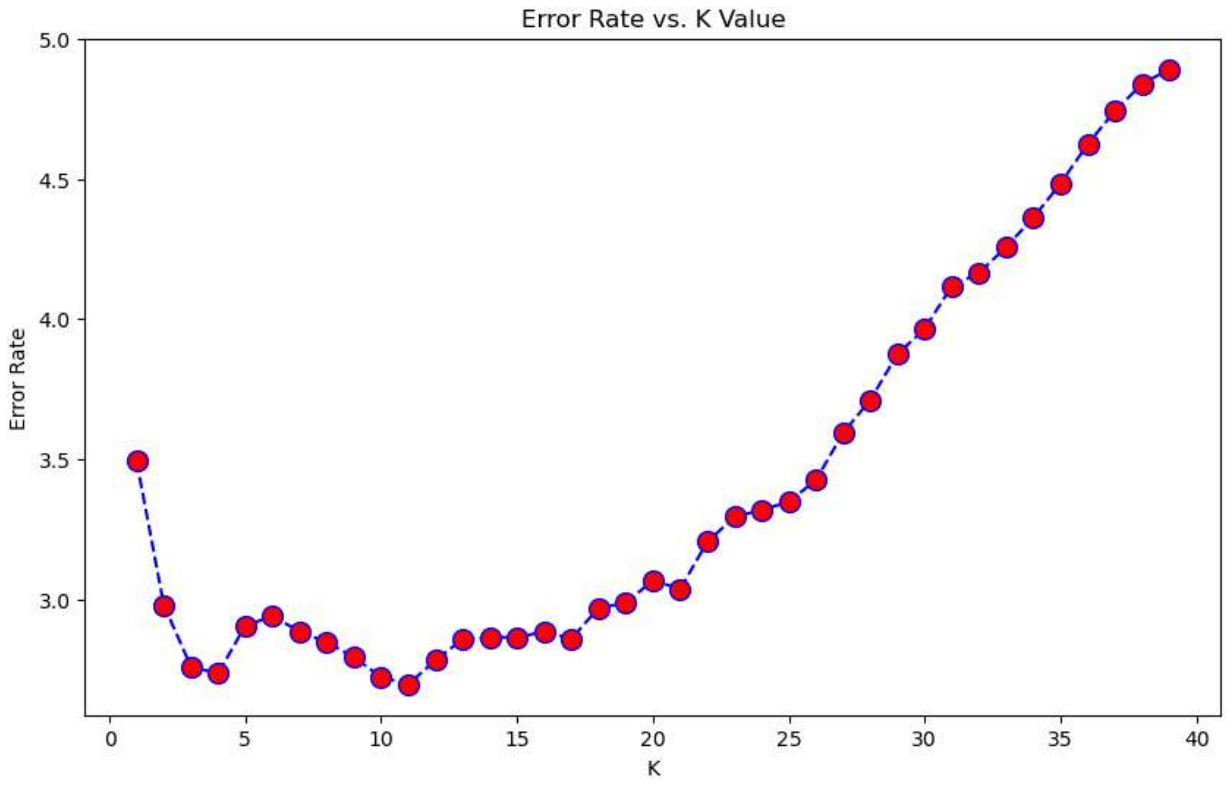
KNN Regressor

In [61]:
```
from sklearn.neighbors import KNeighborsRegressor
```

In [62]:
```
error_rate = []

for i in range(1,40):

    knn = KNeighborsRegressor(n_neighbors=i)
    knn.fit(X_train,y_train)
    pred_i = knn.predict(X_test)
    exp_i = list(y_test)
    error_rate.append((np.square(np.subtract(pred_i, exp_i))).mean())
```

In [63]:
```
plt.figure(figsize=(10,6))
plt.plot(range(1,40),error_rate,color='blue', linestyle='dashed', marker='o',
         markerfacecolor='red', markersize=10)
plt.title('Error Rate vs. K Value')
plt.xlabel('K')
plt.ylabel('Error Rate')
```

Out[63]:
```
Text(0, 0.5, 'Error Rate')
```



In [195]:
```
knn_model= KNeighborsRegressor(n_neighbors=11)

knn_model.fit(X_train,y_train)
pred_new_KNN = knn_model.predict(X_test)

print('WITH K=11')
print('\n')
print('MAE:', metrics.mean_absolute_error(y_test, pred_new_KNN))
```

```
print('MSE:', metrics.mean_squared_error(y_test, pred_new_KNN))
print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, pred_new_KNN)))
```

```
WITH K=11


MAE: 1.0961016835016835
MSE: 2.6959305383532297
RMSE: 1.641928907825558
```

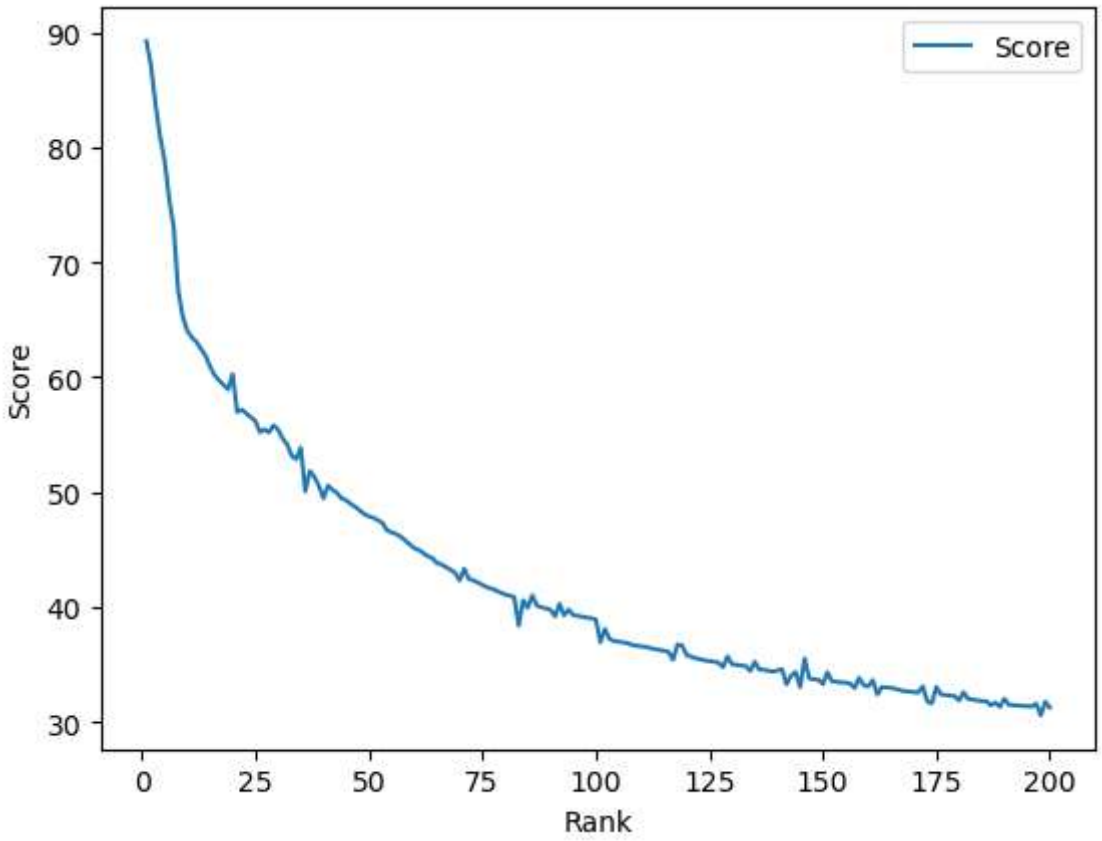
In [64]:
```
# for k =19
knn = KNeighborsRegressor(n_neighbors=19)

knn.fit(X_train,y_train)
pred2 = knn.predict(X_test)

print('WITH K=19')
print('\n')
print('MAE:', metrics.mean_absolute_error(y_test, pred2))
print('MSE:', metrics.mean_squared_error(y_test, pred2))
print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, pred2)))
```

```
WITH K=19


MAE: 1.1800072124756336
MSE: 2.9860027697240166
RMSE: 1.7280054310458681
```

NOW WITH K=29 knn = KNeighborsRegressor(n_neighbors=3) knn.fit(X_train,y_train) pred3 = knn.predict(X_test) print('WITH K=29') print('\n') print('MAE:', metrics.mean_absolute_error(y_test, pred3)) print('MSE:', metrics.mean_squared_error(y_test, pred3)) print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, pred3)))

In [65]:
```
df_score_rank = df_combined[['Score' , 'Rank']]
df_rel_score_rank = df_score_rank.groupby('Rank').mean()
df_rel_score_rank.plot(kind = 'line' , ylabel = 'Score')
```

Out[65]:
```
<Axes: xlabel='Rank', ylabel='Score'>
```



Here it is shown the relation between Overall Score and the Rank of the institute.

Comparison of Regression Models

In [66]:
```
test_values = [83.50, 89.80, 78.00, 65.50, 94.0]
```

In [67]:
```
arr = np.asarray(test_values)
y_predict = lm.predict(arr.reshape(1,-1))
print('Predicted score: ', y_predict[0])
rank_score = df_rel_score_rank.values     # extract the ranks and scores from the datafr

pos, value = min(enumerate(rank_score), key=lambda x: y_predict[0]<=x[1]) # to predict t

print('Predicted rank: ',pos+1)
```

```
Predicted score:  83.17274823359438
Predicted rank:  4
C:\Users\abhyu\anaconda3\Lib\site-packages\sklearn\base.py:464: UserWarning: X does not
have valid feature names, but LinearRegression was fitted with feature names
  warnings.warn(
```

In [68]:
```
arr = np.asarray(test_values)
y_predict = dtree.predict(arr.reshape(1,-1))
print('Predicted score: ', y_predict[0])

pos, value = min(enumerate(rank_score), key=lambda x: y_predict[0]<=x[1]) # to predict t

print('Predicted rank: ',pos+1)
```

```
Predicted score:  87.66
Predicted rank:  2
```

```
C:\Users\abhyu\anaconda3\Lib\site-packages\sklearn\base.py:464: UserWarning: X does not have valid feature names, but DecisionTreeRegressor was fitted with feature names
  warnings.warn(
```

In [69]:
```
arr = np.asarray(test_values)
y_predict = rfc.predict(arr.reshape(1,-1))
print('Predicted score: ', y_predict[0])

pos, value = min(enumerate(rank_score), key=lambda x: y_predict[0]<=x[1]) # to predict t

print('Predicted rank: ',pos+1)
```

```
Predicted score:  84.77059999999994
Predicted rank:  3
```

```
C:\Users\abhyu\anaconda3\Lib\site-packages\sklearn\base.py:464: UserWarning: X does not have valid feature names, but RandomForestRegressor was fitted with feature names
  warnings.warn(
```

In [70]:
```
arr = np.asarray(test_values)
y_predict = knn.predict(arr.reshape(1,-1))
print('Predicted score: ', y_predict[0])

pos, value = min(enumerate(rank_score), key=lambda x: y_predict[0]<=x[1]) # to predict t

print('Predicted rank: ',pos+1)
```

```
Predicted score:  84.28684210526318
Predicted rank:  3
```

```
C:\Users\abhyu\anaconda3\Lib\site-packages\sklearn\base.py:464: UserWarning: X does not have valid feature names, but KNeighborsRegressor was fitted with feature names
  warnings.warn(
```

As from above we can comprehend that the Mean absolute error(MAE) and Root mean squared error(RMSE) of Linear Regressor is least , so it is best suited model for prediction of rank.

Final NIRF Rank predictor model

In [71]:
```
test_values = [83.50, 89.80, 78.00, 65.50, 94.0]
arr = np.asarray(test_values)
y_predict = lm.predict(arr.reshape(1,-1))
print('Predicted score: ', y_predict[0])
rank_score = df_rel_score_rank.values     # extract the ranks and scores from the datafr

pos, value = min(enumerate(rank_score), key=lambda x: y_predict[0]<=x[1]) # to predict t

print('Predicted rank: ',pos+1)
```

```
Predicted score:  83.17274823359438
Predicted rank:  4
```

```
C:\Users\abhyu\anaconda3\Lib\site-packages\sklearn\base.py:464: UserWarning: X does not have valid feature names, but LinearRegression was fitted with feature names
  warnings.warn(
```

Linear Regression Accuracy

In [78]:
```
arr_test = y_test.values
```

In [117]:
```
(arr_test)
```

Out[117]:
```
array([37.44, 53.01, 88.08, 58.7 , 33.98, 56.  , 41.02, 35.19, 39.46,
       54.94, 37.1 , 35.28, 59.89, 35.95, 38.29, 34.34, 34.14, 60.63,
       39.36, 38.33, 49.24, 45.03, 35.41, 35.94, 32.71, 59.67, 58.09,
       31.38, 31.24, 34.09, 60.35, 36.62, 36.82, 75.2 , 50.33, 36.06,
       35.84, 33.98, 36.71, 42.31, 46.19, 33.85, 47.49, 71.8 , 47.21,
       31.6 , 54.17, 38.38, 34.72, 49.55, 46.57, 46.18, 52.85, 37.22,
       53.13, 32.46, 38.95, 36.42, 41.83, 32.53, 72.3 , 38.58, 36.36,
       53.91, 39.13, 47.47, 90.19, 63.94, 36.32, 31.23, 33.8 , 50.39,
       42.86, 39.67, 35.87, 36.82, 40.35, 62.26, 49.57, 46.45, 39.09,
       40.48, 31.72, 40.98, 46.57, 77.22, 48.94, 60.24, 50.6 , 36.66,
       52.14, 35.93, 33.46, 50.6 , 84.82, 63.97, 67.12, 42.63, 74.88,
       53.09, 35.  , 46.4 , 36.93, 35.42, 39.82, 43.09, 59.91, 49.03,
       40.45, 38.06, 38.81, 31.87, 41.13, 48.49, 37.74, 36.13, 43.69,
       77.57, 57.69, 54.91, 50.45, 33.07, 37.45, 87.96, 52.87, 36.67,
       60.31, 36.28, 34.99, 35.49, 55.25, 74.68, 45.59, 33.79, 34.42,
       38.94, 31.15, 54.17, 34.79, 32.69, 63.4 , 31.36, 39.21, 38.56,
       34.97, 45.52, 74.4 , 45.34, 57.14, 54.73, 55.48, 43.53, 82.18,
       37.31, 33.24, 57.52, 37.19, 56.62, 36.7 , 55.71, 33.29, 62.88,
       67.94, 37.8 , 48.95, 49.2 , 33.57, 45.94, 44.85, 54.02, 40.1 ,
       53.37, 33.07, 41.36, 37.46, 36.49, 49.15, 35.65, 34.87, 42.56,
       31.59, 43.64, 32.61, 37.26, 37.27, 59.32, 46.87, 52.58, 55.74,
       58.31, 31.36, 53.21, 33.98, 44.33, 54.42, 37.84, 53.22, 35.95,
       31.14, 49.97, 30.45, 54.32, 43.49, 57.97, 37.11, 34.48, 67.51,
       59.23, 33.65, 35.86, 30.79, 32.95, 36.42, 39.88, 36.8 , 57.7 ,
       36.46, 47.12, 33.58, 40.28, 37.93, 39.64, 46.77, 36.11, 39.87,
       32.1 , 30.91, 34.18, 37.45, 40.28, 37.88, 50.77, 49.49, 51.28,
       59.09, 58.02, 36.6 , 43.42, 33.58, 42.06, 40.24, 31.45, 32.05,
       75.24, 46.07, 63.77, 30.48, 33.64, 34.17, 36.59, 32.48, 51.82,
       41.98, 40.54, 34.52, 32.64, 51.92, 67.8 , 35.69, 45.02, 38.74,
       34.85, 33.07, 35.66, 50.55, 34.48, 36.51, 52.17, 42.68, 33.63])
```

In [79]:
```
sum_test =0
```

In [80]:
```
for i in arr_test:
    sum_test = sum_test + i
```

In [81]:
```
sum_test
```

Out[81]:
```
12090.179999999997
```

In [82]:
```
predictions = lm.predict(X_test)
```

In [83]:
```
predictions
```

Out[83]:
```
array([37.36863882, 53.1600812 , 87.75781592, 58.4042728 , 33.91889982,
       56.08311269, 41.0317241 , 35.10496841, 39.23132045, 54.73235694,
       37.49978251, 35.43866701, 59.60982411, 35.96077923, 38.19267526,
       34.42493066, 34.19090731, 60.39093689, 39.48095564, 38.06515594,
       49.76922022, 44.73372923, 35.57440213, 35.87470323, 32.78785553,
       59.51748683, 58.18233871, 31.45674157, 31.03252791, 34.18228022,
       60.08060384, 36.69519267, 36.77519486, 75.32308394, 51.60213606,
       36.18653108, 35.88718758, 33.38693202, 36.66345832, 42.44680557,
       45.89396579, 33.90976489, 47.36726227, 72.23492277, 47.27299385,
       31.50263709, 53.92000052, 38.31764814, 34.67741548, 49.44074311,
       46.58764194, 46.31167531, 53.00122674, 37.26496484, 52.88174025,
       32.45003129, 38.56969815, 36.3703659 , 41.98152971, 32.52748458,
       72.31278807, 38.59242222, 36.32992419, 54.62055395, 39.16272013,
       47.80225851, 89.910932  , 64.14079237, 36.40238914, 30.99858068,
       33.59223088, 50.42984574, 43.02466792, 39.72173166, 35.82160979,
       36.85966676, 40.39185014, 63.62558676, 49.43910034, 46.47499134,
       38.88767281, 40.53582703, 31.70574756, 40.74260185, 46.26966755,
       77.71156139, 48.92990082, 60.34357851, 50.89322469, 36.51753615,
       52.14547782, 36.00886675, 33.42388317, 50.82558918, 84.46633067,
       63.58497048, 67.81777089, 42.74413757, 76.02793536, 53.13057809,
       35.07807377, 46.50060481, 37.01590286, 35.62530364, 39.68674044,
       42.97295358, 60.5143599 , 49.11367322, 40.33118771, 38.13288088,
       38.81975096, 31.54679949, 41.38480793, 48.61052914, 37.36382093,
       36.0655067 , 43.78902775, 77.11719449, 58.16739063, 55.4603012 ,
       50.54857334, 33.21092974, 37.61619411, 87.68937273, 52.65857124,
       36.82150802, 60.12920523, 36.40257233, 35.19354041, 35.50436525,
       54.87955655, 75.17512796, 45.43530975, 33.81870277, 34.23962283,
       39.01800729, 31.11206506, 54.12589551, 34.66998135, 32.83311187,
       63.71792927, 31.32524263, 39.14687516, 38.61467943, 35.08188225,
       45.49073202, 74.31156271, 45.04451554, 56.97465206, 54.61269636,
       55.35933476, 43.39018185, 81.74290427, 37.26128865, 33.24824145,
       58.4520912 , 37.18602013, 56.27705677, 36.65585999, 55.69703333,
       33.12166652, 62.79486027, 67.9332202 , 37.67496566, 48.80744801,
       49.1509662 , 33.52422933, 45.98853399, 44.5519261 , 54.06204928,
       40.17842465, 54.86808117, 32.84477005, 41.41790323, 37.44526012,
       36.39952901, 48.8422497 , 35.64837118, 34.91391403, 42.33092851,
       31.68334389, 43.61134118, 32.64934863, 37.274065  , 37.46742704,
       58.90781256, 46.43928077, 52.73249064, 55.77913666, 58.09632242,
       31.45822194, 53.06737039, 33.96298929, 44.05081647, 54.38600126,
       37.80805049, 53.16903834, 35.82713105, 30.97370104, 49.74174127,
       30.431458  , 54.51748336, 43.25549438, 57.78815866, 37.22903463,
       34.462161  , 67.970335  , 58.61856623, 33.80099449, 35.96258473,
       30.59362365, 33.30133263, 36.07073605, 39.65914053, 36.54329237,
       57.88384148, 36.33599111, 46.74982358, 33.52687691, 39.94936022,
       37.60553711, 39.68495732, 46.64329957, 36.35480041, 39.87674703,
       31.99247901, 30.9638811 , 34.26339335, 37.56411491, 40.26806742,
       37.98277786, 50.85158793, 49.50594721, 51.39696828, 58.47682021,
       57.66024542, 36.56813699, 43.53045331, 33.68868602, 41.89280364,
       40.50261767, 31.48244413, 32.06511241, 74.81102137, 46.0016186 ,
       64.33649526, 30.40285261, 33.71245502, 34.31907986, 36.64755373,
       32.56178228, 51.68704299, 42.05882584, 40.59508004, 34.75334806,
       32.74398157, 52.5153541 , 67.90721204, 35.87004822, 44.83984598,
       38.91067367, 34.84897723, 33.14003624, 35.68028143, 51.08595579,
       34.47753013, 36.66380472, 52.71195703, 42.29529135, 33.87059056])
```

In [96]:
```
len(predictions)
```

Out[96]:
```
270
```

In [84]:
```
sum_predictions =0
```

In [97]:
```
for i in predictions:
    sum_predictions = sum_predictions + i
```

In [98]:
```
sum_predictions
```

Out[98]:
```
12363.533024501943
```

In [101]:
```
bias= (sum_test - sum_predictions) / (sum_test)
```

In [102]:
```
bias
```

Out[102]:
```
-0.022609508253966987
```

In [103]:
```
100 -(-1 *bias)
```

Out[103]:
```
99.97739049174604
```

In [105]:
```
df_combined.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 900 entries, 0 to 899
Data columns (total 12 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   Institute Id    900 non-null    object 
 1   Institute Name  900 non-null    object 
 2   City            900 non-null    object 
 3   State           900 non-null    object 
 4   Score           900 non-null    float64
 5   Rank            900 non-null    float64
 6   TLR             900 non-null    float64
 7   RPC             900 non-null    float64
 8   GO              900 non-null    float64
 9   OI              900 non-null    float64
 10  Perception      900 non-null    float64
 11  year            900 non-null    object 
dtypes: float64(7), object(5)
memory usage: 84.5+ KB
```

In [138]:
```
new_predicitions = ((abs(y_test - predictions))/y_test)
```

In [139]:
```
new_predicitions
```

Out[139]:
```
184    0.001906
532    0.002831
501    0.003658
517    0.005038
871    0.001798
       ...
416    0.000072
196    0.004213
83     0.010388
258    0.009014
432    0.007154
Name: Score, Length: 270, dtype: float64
```

In [140]:
```
new_predicitions =new_predicitions*100
```

In [141]:
```
new_predicitions
```

Out[141]:
```
184    0.190601
532    0.283119
501    0.365786
517    0.503794
871    0.179812
       ...
416    0.007163
196    0.421267
83     1.038829
258    0.901379
432    0.715405
Name: Score, Length: 270, dtype: float64
```

In [112]:
```
sum_linear_regression=0
```

In [114]:
```
for i in new_predicitions:
    sum_linear_regression = sum_linear_regression + i
```

Mean Absolute Percentage Error Linear Regression

In [132]:
```
mape = (sum_linear_regression/270)
```

In [155]:
```
accuracy_linear_regression = 100-mape
accuracy_linear_regression
```

Out[155]:
```
99.92273576122237
```

Decision Tree Regressor Accuracy

In [160]:
```
predicitions_Decision_tree = dtree.predict(X_test)
```

In [161]:
```
new_predicitions_decision_tree = ((abs(y_test - predicitions_Decision_tree))/y_test)*100
```

In [165]:
```
sum_decision_tree = 0
for i in new_predicitions_decision_tree:
    sum_decision_tree = sum_decision_tree + i
mape_decision_tree = sum_decision_tree/270
mape_decision_tree
```

Out[165]:
```
3.970571228230969
```

In [167]:
```
accuracy_decision_tree = 100-mape_decision_tree
accuracy_decision_tree
```

Out[167]:
```
96.02942877176903
```

Random Forest Accuracy

In [168]:
```
rfc_pred = rfc.predict(X_test)
```

In [169]:
```
new_predicitions_random_forest = ((abs(y_test -rfc_pred))/y_test)*100
```

In [170]:
```
sum_random_forest = 0
for i in new_predicitions_random_forest:
    sum_random_forest = sum_random_forest + i
mape_random_forest = sum_random_forest/270
mape_random_forest
```

Out[170]:
```
2.342182000634199
```

In [172]:
```
accuracy_random_forest = 100-mape_random_forest
accuracy_random_forest
```

Out[172]:
```
97.6578179993658
```

KNN Accuracy

knn_model= KNeighborsRegressor(n_neighbors=11)

knn_model.fit(X_train,y_train) pred_new_KNN = knn_model.predict(X_test)

print('WITH K=11') print('\n') print('MAE:', metrics.mean_absolute_error(y_test, pred_new_KNN)) print('MSE:', metrics.mean_squared_error(y_test, pred_new_KNN)) print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, pred_new_KNN)))

In [196]:
```
Predicitons_KNN = knn_model.predict(X_test)
```

In [197]:
```
new_predicitions_KNN = ((abs(y_test -Predicitons_KNN))/y_test)*100
```

In [198]:
```
sum_KNN= 0
for i in new_predicitions_KNN:
    sum_KNN = sum_KNN + i
```

```
mape_KNN = sum_KNN/270
mape_KNN
```

Out[198]:
```
2.356150441672377
```

In [199]:
```
accuracy_KNN = 100-mape_KNN
accuracy_KNN
```

Out[199]:
```
97.64384955832762
```

Overall Accuracy of the models

In [201]:
```
print("The Accuracy of the models are ")
print("Linear_Regression:", accuracy_linear_regression)
print("Decision Tree:", accuracy_decision_tree)
print("Random Forest:", accuracy_random_forest )
print("KNN:",accuracy_KNN )
```

```
The Accuracy of the models are
Linear_Regression: 99.92273576122237
Decision Tree: 96.02942877176903
Random Forest: 97.6578179993658
KNN: 97.64384955832762
```