In [56]:
```
import pandas as pd
import numpy as np
teams = pd.read_csv("teams.csv")
teams
```

Out[56]:

	team	year	athletes	events	age	height	weight	prev_medals	medals
0	AFG	1964	8	8	22.0	161.0	64.2	0.0	0
1	AFG	1968	5	5	23.2	170.2	70.0	0.0	0
2	AFG	1972	8	8	29.0	168.3	63.8	0.0	0
3	AFG	1980	11	11	23.6	168.4	63.2	0.0	0
4	AFG	2004	5	5	18.6	170.8	64.8	0.0	0
...	...	...	...	...	...	...	...	...	...
2009	ZIM	2000	26	19	25.0	179.0	71.1	0.0	0
2010	ZIM	2004	14	11	25.1	177.8	70.5	0.0	3
2011	ZIM	2008	16	15	26.1	171.9	63.7	3.0	4
2012	ZIM	2012	9	8	27.3	174.4	65.2	4.0	0
2013	ZIM	2016	31	13	27.5	167.8	62.2	0.0	0

2014 rows × 9 columns

In [57]:
```
teams.shape
```

Out[57]:
```
(2014, 9)
```

In [58]:
```
X = teams[["athletes", "prev_medals"]].copy()
Y = teams[["medals"]].copy()
```

In [59]:
```
X
```

Out[59]:

	athletes	prev_medals
0	8	0.0
1	5	0.0
2	8	0.0
3	11	0.0
4	5	0.0
...	...	...
2009	26	0.0
2010	14	0.0
2011	16	3.0
2012	9	4.0
2013	31	0.0

2014 rows × 2 columns

In [60]:
```
Y
```

Out[60]:

	medals
0	0
1	0
2	0
3	0
4	0
...	...
2009	0
2010	3
2011	4
2012	0
2013	0

2014 rows × 1 columns

In [61]:
```
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)

print(X_train.shape)
print(X_test.shape)
print(Y_train.shape)
print(Y_test.shape)
```

```
(1409, 2)
(605, 2)
(1409, 1)
(605, 1)
```

In [62]:
```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error

lin_model = LinearRegression()
lin_model.fit(X_train, Y_train)
```

Out[62]:
```
▾ LinearRegression
LinearRegression()
```

In [63]:
```
from sklearn.metrics import r2_score
y_test_predict = lin_model.predict(X_test)
rmse = (np.sqrt(mean_squared_error(Y_test, y_test_predict)))
r2 = r2_score(Y_test, y_test_predict)

print('RMSE is {}'.format(rmse))
print('R2 score is {}'.format(r2))
```

```
RMSE is 12.587528953356765
R2 score is 0.880162117186997
```

In [64]:
```
import pandas as pd
import numpy as np
import warnings
warnings.filterwarnings('ignore')

from sklearn.model_selection import cross_val_score
from sklearn.model_selection import RepeatedKFold
from sklearn.linear_model import Lasso

data = pd.read_csv('teams.csv')
X = data.iloc[:,1:8]
Y = data['medals']
```

In [65]:
```
X
```

Out[65]:

	year	athletes	events	age	height	weight	prev_medals
0	1964	8	8	22.0	161.0	64.2	0.0
1	1968	5	5	23.2	170.2	70.0	0.0
2	1972	8	8	29.0	168.3	63.8	0.0
3	1980	11	11	23.6	168.4	63.2	0.0
4	2004	5	5	18.6	170.8	64.8	0.0
...	...	...	...	...	...	...	...
2009	2000	26	19	25.0	179.0	71.1	0.0
2010	2004	14	11	25.1	177.8	70.5	0.0
2011	2008	16	15	26.1	171.9	63.7	3.0
2012	2012	9	8	27.3	174.4	65.2	4.0
2013	2016	31	13	27.5	167.8	62.2	0.0

2014 rows × 7 columns

In [66]:
```
Y
```

Out[66]:
```
0       0
1       0
2       0
3       0
4       0
       ..
2009    0
2010    3
2011    4
2012    0
2013    0
Name: medals, Length: 2014, dtype: int64
```

LINEAR REGRESSION

In [67]:
```
from sklearn.linear_model import Lasso
from sklearn.linear_model import LinearRegression

lr_model = LinearRegression()

cv = RepeatedKFold(n_splits=10, n_repeats=3, random_state=1)

scores = cross_val_score(lr_model, X, Y, scoring='neg_mean_absolute_error', cv=cv, n_jobs=-1)

scores = np.absolute(scores)
print('Mean MAE: %.3f (%.3f)' % (np.mean(scores), np.std(scores)))
```

```
Mean MAE: 4.730 (0.597)
```

LASSO

In [68]:
```
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import RepeatedKFold
from sklearn.linear_model import Lasso

lasso_model = Lasso(alpha=0.01)

cv = RepeatedKFold(n_splits=10, n_repeats=3, random_state=1)

scores = cross_val_score(lasso_model, X, Y, scoring='neg_mean_absolute_error', cv=cv, n_jobs=-1)

scores = np.absolute(scores)
print('Mean MAE: %.3f (%.3f)' % (np.mean(scores), np.std(scores)))
```

```
Mean MAE: 4.729 (0.597)
```

RIDGE

In [69]:
```
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import RepeatedKFold
from sklearn.linear_model import Ridge
import pandas as pd
import numpy as np

ridge_model = Ridge(alpha=0.5)

cv = RepeatedKFold(n_splits=10, n_repeats=3, random_state=1)

scores = cross_val_score(ridge_model, X, Y, scoring='neg_mean_absolute_error', cv=cv, n_jobs=-1)

scores = np.absolute(scores)
print('Mean MAE: %.3f (%.3f)' % (np.mean(scores), np.std(scores)))
```

```
Mean MAE: 4.730 (0.597)
```

In [ ]:
```

```