In [2]:
```
# This Python 3 environment comes with many helpful analytics libraries installed
# It is defined by the kaggle/python Docker image: https://github.com/kaggle/docker-python
# For example, here's several helpful packages to load

import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

# Input data files are available in the read-only "../input/" directory
# For example, running this (by clicking run or pressing Shift+Enter) will list all files under the input direc

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))

# You can write up to 20GB to the current directory (/kaggle/working/) that gets preserved as output when you c
# You can also write temporary files to /kaggle/temp/, but they won't be saved outside of the current session
```

```
/kaggle/input/pythonfile/teams.csv
```

In [5]:
```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression, Ridge, Lasso
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.model_selection import cross_val_score
from sklearn.preprocessing import StandardScaler
```

In [6]:
```
teams_data = pd.read_csv("/kaggle/input/pythonfile/teams.csv")
teams_data
```

Out[6]:

	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 [12]:
```
X = teams_data[['year', 'athletes', 'events', 'age', 'height', 'weight', 'prev_medals']]
Y = teams_data['medals']

print(X)
print(Y)
```

```
      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 x 7 columns]
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
```

In [14]:
```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=42)
```

```
print(X_train, X_test, Y_train, Y_test)
```

```
      year  athletes  events   age  height  weight  prev_medals
261   1996       251      93  25.3   181.0    75.7         14.0
746   2000        18      14  25.6   171.8    66.0          0.0
1380  2008         1       1  21.0   175.0   148.0          0.0
240   2004        11      11  20.5   162.0    54.4          0.0
1644  1980        16      10  28.8   174.7    72.6          0.0
...    ...       ...     ...   ...     ...     ...          ...
1130  1980         4       4  30.0   175.3    95.3          0.0
1294  1964        12      11  26.0   167.7    61.7          0.0
860   1988        26      21  22.8   171.4    66.3          2.0
1459  1984        32      28  23.4   168.4    59.2          0.0
1126  1964        27      18  27.5   168.9    65.8          0.0

[1611 rows x 7 columns]       year  athletes  events   age  height  weight  prev_medals
1198  1968       446     146  22.5   170.8    65.0          1.0
526   2016        30      22  25.7   172.5    63.8          2.0
393   1976         4       1  23.8   174.0    68.5          0.0
1407  1972        26      12  24.2   173.5    72.8         13.0
433   2016       157      85  25.8   169.7    64.6          8.0
...    ...       ...     ...   ...     ...     ...          ...
1793  1992        64      47  22.2   166.5    58.8          1.0
1111  1984         8       7  26.6   172.8    72.6          0.0
693   2004         2       2  20.5   171.5    65.5          0.0
1492  1992       263     136  24.3   178.1    73.9         21.0
921   1972        16       9  32.3   181.7    78.7          0.0

[403 rows x 7 columns] 261     63
746      0
1380     0
240      0
1644     0
        ..
1130     0
1294     0
860      1
1459     0
1126     0
Name: medals, Length: 1611, dtype: int64 1198     9
526      1
393      0
1407    15
433      8
        ..
1793     1
1111     0
693      0
1492    46
921      0
Name: medals, Length: 403, dtype: int64
```

In [23]:
```
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

linear_reg = LinearRegression()
linear_reg.fit(X_train, Y_train)
linear_reg_predictions = linear_reg.predict(X_test)
linear_reg_rmse = mean_squared_error(Y_test, linear_reg_predictions, squared=False)
linear_reg_r2 = r2_score(Y_test, linear_reg_predictions)
linear_reg_cv_score = cross_val_score(linear_reg, X, Y, cv=5)

print("Linear Regression:")
print("RMSE:", linear_reg_rmse)
print("R2 Score:", linear_reg_r2)
print("Cross Validation Score:", linear_reg_cv_score.mean())
```

```
Linear Regression:
RMSE: 10.972282972268285
R2 Score: 0.8382725954900433
Cross Validation Score: 0.8474694167427048
```

In [27]:
```
ridge_reg = Ridge(alpha=0.5)
ridge_reg.fit(X_train, Y_train)
ridge_reg_predictions = ridge_reg.predict(X_test)
ridge_reg_rmse = mean_squared_error(Y_test, ridge_reg_predictions, squared=False)
ridge_reg_r2 = r2_score(Y_test, ridge_reg_predictions)
ridge_reg_cv_score = cross_val_score(ridge_reg, X, Y, cv=5)

print("Ridge Regression:")
print("RMSE:", ridge_reg_rmse)
print("R2 Score:", ridge_reg_r2)
print("Cross Validation Score:", ridge_reg_cv_score.mean())
```

```
Ridge Regression:
RMSE: 10.945714580679113
R2 Score: 0.8390548638335263
Cross Validation Score: 0.847469440098424
```

In [25]:
```
lasso_reg = Lasso(alpha=0.1)
lasso_reg.fit(X_train, Y_train)
lasso_reg_predictions = lasso_reg.predict(X_test)
lasso_reg_rmse = mean_squared_error(Y_test, lasso_reg_predictions, squared=False)
lasso_reg_r2 = r2_score(Y_test, lasso_reg_predictions)
lasso_reg_cv_score = cross_val_score(lasso_reg, X, Y, cv=5)

print("Lasso Regression:")
print("RMSE:", lasso_reg_rmse)
print("R2 Score:", lasso_reg_r2)
print("Cross Validation Score:", lasso_reg_cv_score.mean())
```

```
Lasso Regression:
RMSE: 10.716253244294444
R2 Score: 0.8457321047043443
Cross Validation Score: 0.8475554841274153
```

In [ ]: