In [1]:
```
##### Import necessary libraries

import numpy as np

import matplotlib.pyplot as plt
import pandas as pd

import seaborn as sns

from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error, mean_squared_error,r2_score


from sklearn.preprocessing import MinMaxScaler


import joblib
```

Read fifa data

In [2]:
```
fifa = pd.read_excel("C:\Users\user\Desktop\Desktop\Machine Learning\fifa_data-2.xlsx", skiprows = 1)
```

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

Out[3]:

	Column1	ID	Name	Age	Photo	Nationality	Flag	Overall	Potential	Club	...	Composure	Marking	StandingTackle	SlidingTackle	GKDiving	GKHandling	GKKicking	GKP
0	0	158023	L. Messi	31	https://cdn.sofifa.org/players/4/19/158023.png	Argentina	https://cdn.sofifa.org/flags/52.png	94	94	FC Barcelona	...	96.0	33.0	28.0	26.0	6.0	11.0	15.0	
1	1	20801	Cristiano Ronaldo	33	https://cdn.sofifa.org/players/4/19/20801.png	Portugal	https://cdn.sofifa.org/flags/38.png	94	94	Juventus	...	95.0	28.0	31.0	23.0	7.0	11.0	15.0	
2	2	190871	Neymar Jr	26	https://cdn.sofifa.org/players/4/19/190871.png	Brazil	https://cdn.sofifa.org/flags/54.png	92	93	Paris Saint-Germain	...	94.0	27.0	24.0	33.0	9.0	9.0	15.0	
3	3	193080	De Gea	27	https://cdn.sofifa.org/players/4/19/193080.png	Spain	https://cdn.sofifa.org/flags/45.png	91	93	Manchester United	...	68.0	15.0	21.0	13.0	90.0	85.0	87.0	
4	4	192985	K. De Bruyne	27	https://cdn.sofifa.org/players/4/19/192985.png	Belgium	https://cdn.sofifa.org/flags/7.png	91	92	Manchester City	...	88.0	68.0	58.0	51.0	15.0	13.0	5.0	

5 rows × 62 columns

In [4]:
```
fifa.columns
```

Out[4]:
```
Index(['Column1', 'ID', 'Name', 'Age', 'Photo', 'Nationality', 'Flag',
       'Overall', 'Potential', 'Club', 'Club Logo', 'Value', 'Special',
       'Preferred Foot', 'International Reputation', 'Weak Foot',
       'Skill Moves', 'Work Rate', 'Body Type', 'Real Face', 'Position',
       'Jersey Number', 'Joined', 'Loaned From', 'Contract Valid Until',
       'Height', 'Weight', 'Crossing', 'Finishing', 'HeadingAccuracy',
       'ShortPassing', 'Volleys', 'Dribbling', 'Curve', 'FKAccuracy',
       'LongPassing', 'BallControl', 'Acceleration', 'SprintSpeed', 'Agility',
       'Reactions', 'Balance', 'ShotPower', 'Jumping', 'Stamina', 'Strength',
       'LongShots', 'Aggression', 'Interceptions', 'Positioning', 'Vision',
       'Penalties', 'Composure', 'Marking', 'StandingTackle', 'SlidingTackle',
       'GKDiving', 'GKHandling', 'GKKicking', 'GKPositioning', 'GKReflexes',
       'Release Clause'],
      dtype='object')
```

In [5]:
```
fifa_data = fifa[["Name","Age","Overall","Potential","Wage"]]
```

In [6]:
```
fifa_data.head()
```

Out[6]:

	Name	Age	Overall	Potential	Wage
0	L. Messi	31	94	94	565K
1	Cristiano Ronaldo	33	94	94	405K
2	Neymar Jr	26	92	93	290K
3	De Gea	27	91	93	260K
4	K. De Bruyne	27	91	92	355K

In [7]:
```
fifa_data['Wage'] = fifa_data['Wage'].str.replace('K','000').astype(float)
```

```
C:\Users\user\AppData\Local\Temp\ipykernel_9764\4059960986.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  fifa_data['Wage'] = fifa_data['Wage'].str.replace('K','000').astype(float)
```

In [8]:
```
fifa_data.head()
```

Out[8]:

	Name	Age	Overall	Potential	Wage
0	L. Messi	31	94	94	565000.0
1	Cristiano Ronaldo	33	94	94	405000.0
2	Neymar Jr	26	92	93	290000.0
3	De Gea	27	91	93	260000.0
4	K. De Bruyne	27	91	92	355000.0

In [9]:
```
fifa_data.info()
```

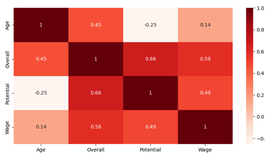
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 18206 entries, 0 to 18205
Data columns (total 5 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Name       18206 non-null  object
 1   Age        18206 non-null  int64
 2   Overall    18206 non-null  int64
 3   Potential  18206 non-null  int64
 4   Wage       17945 non-null  float64
dtypes: float64(1), int64(3), object(1)
memory usage: 711.3+ KB
```

In [10]:
```
fifa_correlation = fifa_data.corr()
```

```
C:\Users\user\AppData\Local\Temp\ipykernel_9764\1364431056.py:1: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only v
alid columns or specify the value of numeric_only to silence this warning.
  fifa_correlation = fifa_data.corr()
```

In [11]:
```
plt.figure(figsize = (15,5))
sns.heatmap(fifa_correlation, annot = True, cmap = 'Reds')
```

Out[11]:
```
<Axes: >
```



In [12]:
```
fifa_data.isnull().sum()
```

Out[12]:
```
Name          0
Age           0
Overall       0
Potential     0
Wage        261
dtype: int64
```

In [13]:
```
average = fifa_data['Wage'].mean()
```

In [14]:
```
fifa_data['Wage'] = fifa_data['Wage'].fillna(average)
```

```
C:\Users\user\AppData\Local\Temp\ipykernel_9764\3504888883.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  fifa_data['Wage'] = fifa_data['Wage'].fillna(average)
```

In [15]:
```
fifa_data.isnull().sum()
```

Out[15]:
```
Name         0
Age          0
Overall      0
Potential    0
Wage         0
dtype: int64
```

In [16]:
```
y = fifa_data["Potential"]
```

In [17]:
```
X = fifa_data[["Overall","Wage"]]
```

In [18]:
```
y
```

Out[18]:
```
0        94
1        94
2        93
3        93
4        92
         ..
18201    65
18202    63
18203    67
18204    66
18205    66
Name: Potential, Length: 18206, dtype: int64
```

In [19]:
```
X
```

Out[19]:

	Overall	Wage
0	94	565000.0
1	94	405000.0
2	92	290000.0
3	91	260000.0
4	91	355000.0
...	...	...
18201	47	1000.0
18202	47	1000.0
18203	47	1000.0
18204	47	1000.0
18205	46	1000.0

18206 rows × 2 columns

In [20]:
```
X_train,X_test,y_train,y_test = train_test_split(X, y, test_size=0.2, random_state = 42)
```

In [21]:
```
X_train
```

Out[21]:

	Overall	Wage
17471	54	1000.0
5970	69	13000.0
10263	65	4000.0
176	83	64000.0
3750	72	15000.0
...	...	...
11284	64	1000.0
11964	64	1000.0
5390	70	8000.0
860	78	53000.0
15795	59	2000.0

14564 rows × 2 columns

In [22]:
```
X_test
```

Out[22]:

	Overall	Wage
6847	69	8000.0
14632	61	1000.0
15999	58	3000.0
12350	63	2000.0
1989	75	10000.0
...	...	...
9623	66	10000.0
1604	75	12000.0
10098	65	3000.0
17725	53	1000.0
3733	72	16000.0

3642 rows × 2 columns

In [23]:
```
y_train
```

Out[23]:
```
17471    66
5970     80
10263    65
176      91
3750     75
         ..
11284    69
11964    70
5390     70
860      78
15795    59
Name: Potential, Length: 14564, dtype: int64
```

In [24]:
```
y_test
```

Out[24]:
```
6847     77
14632    67
15999    78
12350    68
1989     80
         ..
9623     74
1604     75
10098    66
17725    53
3733     74
Name: Potential, Length: 3642, dtype: int64
```

In [25]:
```
model = LinearRegression()
model.fit(X_train,y_train)
```

Out[25]:
```
LinearRegression
LinearRegression()
```

In [26]:
```
y_pred = model.predict(X_test)
```

In [27]:
```
y_pred
```

Out[27]:
```
array([72.60054429, 68.23619021, 66.81535517, ..., 70.35027068,
       64.19024677, 74.48423283])
```

In [28]:
```
results = pd.DataFrame({'Actual':y_test, 'Predicted':y_pred})
```

In [29]:
```
results
```

Out[29]:

	Actual	Predicted
6847	77	72.600544
14632	67	68.236190
15999	78	66.815355
12350	68	69.294230
1989	80	75.722601
...	...	...
9623	74	71.177709
1604	75	75.814729
10098	66	70.350271
17725	53	64.190287
3733	72	74.484233

3642 rows × 2 columns

In [30]:
```
#Evaluate the model
mse = mean_squared_error(y_test,y_pred)
rmse = np.sqrt(mse)
```

In [31]:
```
rmse
```

Out[31]:
```
4.508146188872435
```

In [32]:
```
r2_score(y_test,y_pred)
```

Out[32]:
```
0.4499385671052788
```

In [33]:
```
y_test.shape
```

Out[33]:
```
(3642,)
```

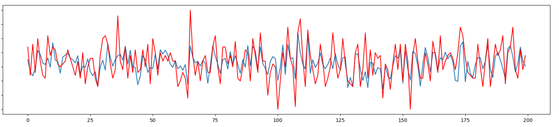
In [34]:
```
y_test = y_test.values.reshape(-1,1)
```

In [35]:
```
y_test.shape
```

Out[35]:
```
(3642, 1)
```

In [36]:
```
plt.figure(figsize=(25,6))
plt.plot(y_pred[:200])
plt.plot(y_test[:200], '*')
```

Out[37]:
```
[<matplotlib.lines.Line2D at 0x23a4557fa30>]
```


90
85
80
75
70
65
60
55
0
25
50
75
100
125
150
175
200