FIFA 18 – Data Analysis and Interpretation

IST 652: Scripting for Data Analysis

Professor David Myers

Group Members:

1. Pranay Lulla
2. Harsh Takrani

Project Proposal and Scope:

While deciding the topic for the final project and dataset, we decided to select FIFA-18 (a multi-player soccer game) dataset mainly because we are well acquainted with terminologies of the game as well as the dataset. This made the data exploration and data understanding process easier for the team. As the team understood the data very well, we were able to formulate several business questions and do in depth analysis of the dataset extracted. The team also aims to implement visualizations that help to understand the properties of the dataset to the new user easily. FIFA has been in the gaming industry for 25 years now, it is one of the best in the market for many years and holds a strong customer base. This intrigued us to know the feedback of its most recent version i.e. FIFA-18, for which our team aims to perform sentiment analysis using tweets extracted from Twitter. Our team also intends to apply machine learning algorithms such as Linear Regression and Support Vector Regression that would help us to determine the VALUE of a player based on OVERALL RATING, AGE and POTENTIAL parameters in the dataset.

Data Sources:

1. **www.sofifa.com :** This is the official website of EA SPORTS FIFA that allows users to download data depending on their requirements. The site also provides data of previous versions of the game till FIFA-07, which can be used for comparison purposes. The data can be extracted based on leagues, countries, qualities of players etc. For the project we decided to go with the complete dataset of FIFA-18 for analysis.
2. **www.twitter.com :** This dataset is used for sentiment analysis to understand the opinion/feedback of the game.

Project Goals:

- Collecting, preprocessing and loading complete dataset of players
- Clustering top players based on Overall Rating
- To identify best possible squad based on three formations
- Clustering young underrated players with high potential
- Sentiment analysis using Twitter to understand feedback of FIFA 18
- Applying Machine Learning Algorithm for prediction purposes
- Creating basic visualizations like Histograms and Line charts

Libraries Used:

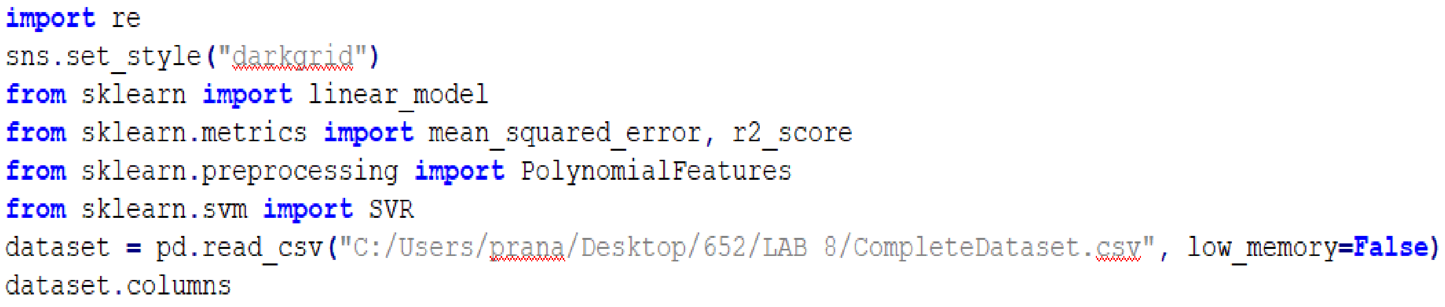
- Pandas (Data frame analysis and Loading data)
- Re (Preprocessing)
- Numpy (Preprocessing)
- Matplotlib (Visualization)
- Sklearn (Machine Learning)
- Tweepy (Sentiment Analysis)
- Streamhandler (Sentiment Analysis)
- Textblob (Sentiment Analysis)

1. Collecting, preprocessing and loading complete dataset of players

Initially, we loaded the csv file into a data frame using pandas. The dataset had null values and special characters in some columns, which had to be dealt with. For this purpose, we defined functions to deal with special characters. We also had to change the data types of certain columns which was necessary for analytical purposes.

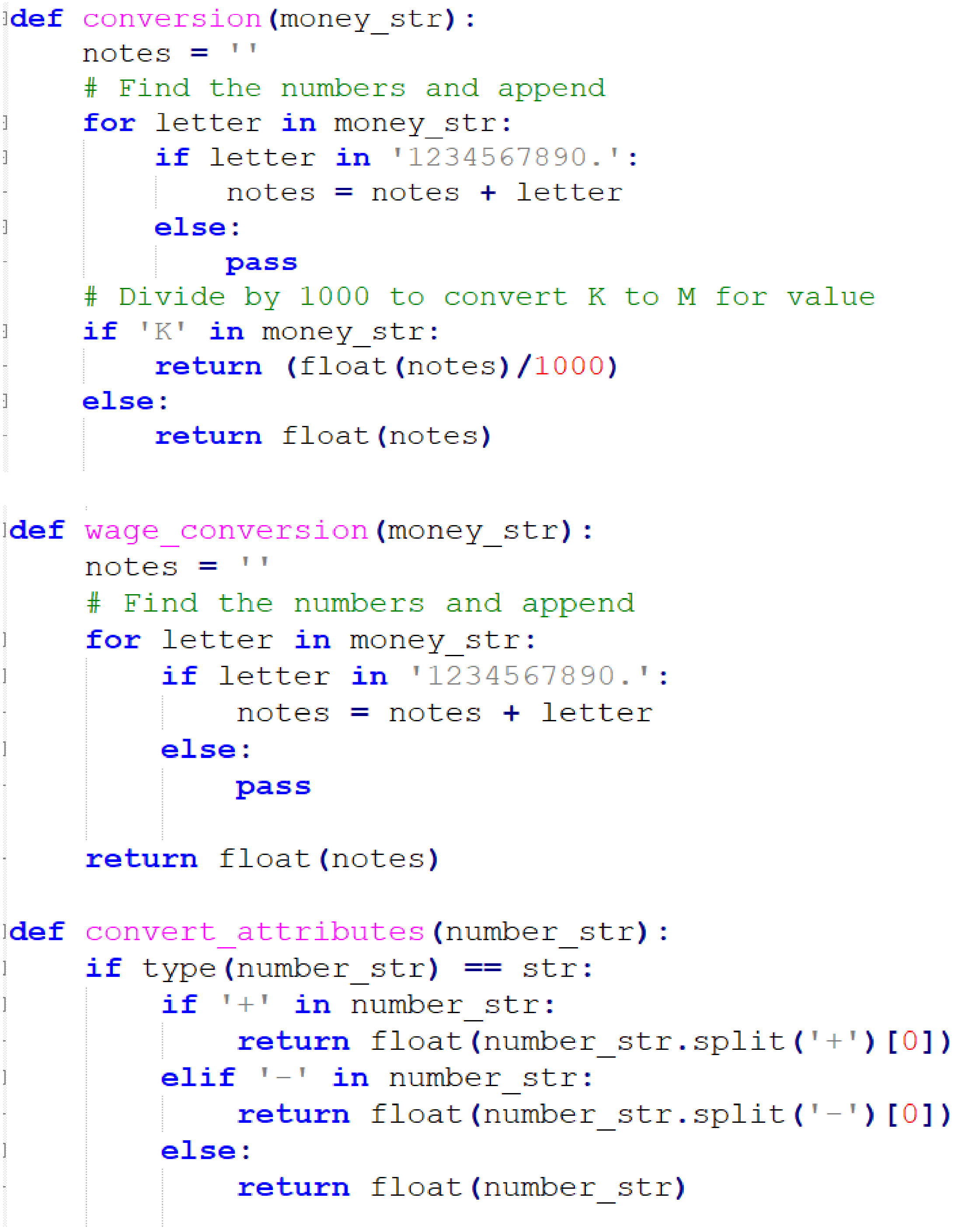
CODE SNIPPET:

Loading Data and importing packages for Visualizations and Machine Learning:



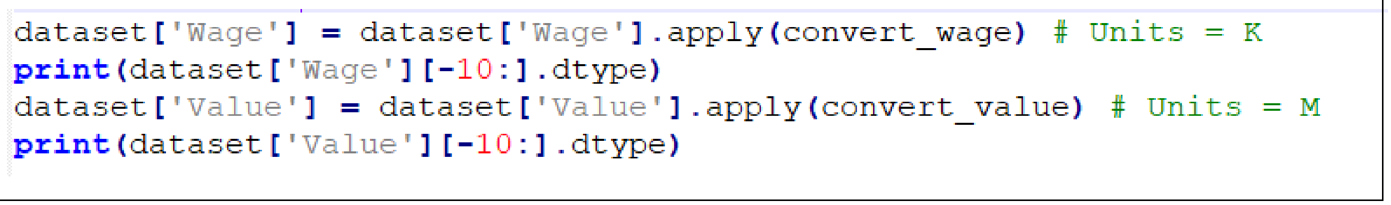
```
import re
sns.set_style("darkgrid")
from sklearn import linear_model
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.preprocessing import PolynomialFeatures
from sklearn.svm import SVR
dataset = pd.read_csv("C:/Users/prana/Desktop/652/LAB 8/CompleteDataset.csv", low_memory=False)
dataset.columns
```

1. Defining Functions to convert Value and Wage Columns in to Float data type
2. Defining function to convert string column type in to float:



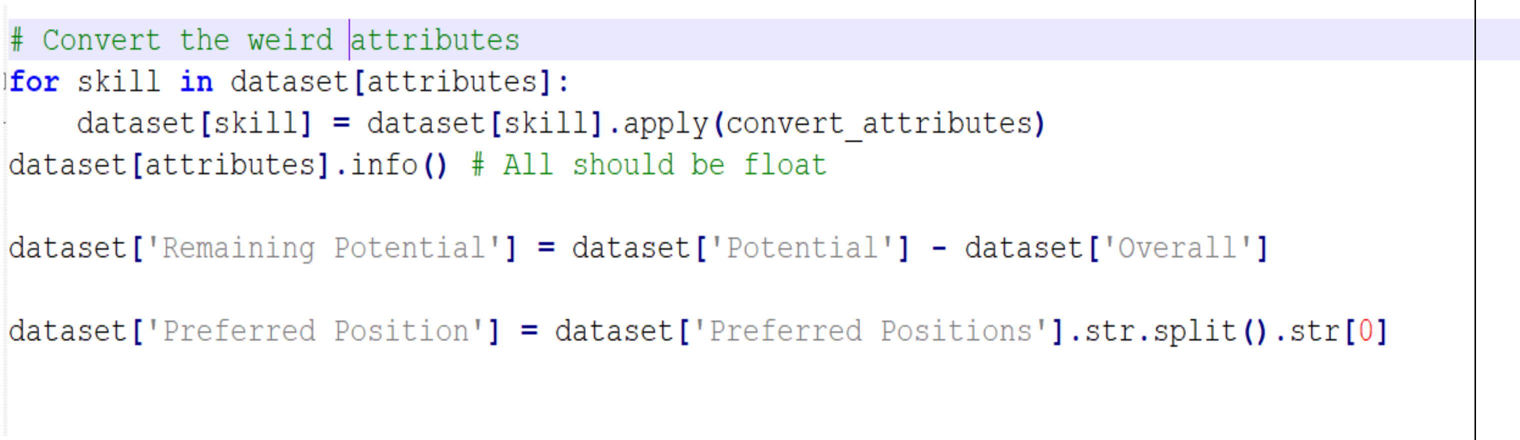
```
def conversion(money_str):
    notes = ''
    # Find the numbers and append
    for letter in money_str:
        if letter in '1234567890.':
            notes = notes + letter
        else:
            pass
    # Divide by 1000 to convert K to M for value
    if 'K' in money_str:
        return (float(notes)/1000)
    else:
        return float(notes)

def wage_conversion(money_str):
    notes = ''
    # Find the numbers and append
    for letter in money_str:
        if letter in '1234567890.':
            notes = notes + letter
        else:
            pass

    return float(notes)

def convert_attributes(number_str):
    if type(number_str) == str:
        if '+' in number_str:
            return float(number_str.split('+')[0])
        elif '-' in number_str:
            return float(number_str.split('-')[0])
        else:
            return float(number_str)
```

3. Converting columns using above created functions:



```
dataset['Wage'] = dataset['Wage'].apply(convert_wage) # Units = K
print(dataset['Wage'][-10:].dtype)
dataset['Value'] = dataset['Value'].apply(convert_value) # Units = M
print(dataset['Value'][-10:].dtype)
```

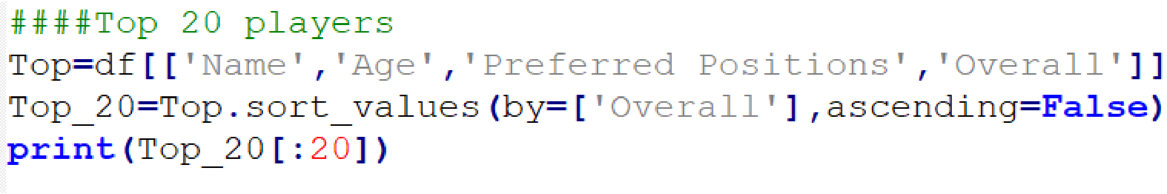
4. Converting weird attributes in the dataset like 72 + 3 in the skill column etc.:



```
# Convert the weird attributes
for skill in dataset[attributes]:
    dataset[skill] = dataset[skill].apply(convert_attributes)
dataset[attributes].info() # All should be float

dataset['Remaining Potential'] = dataset['Potential'] - dataset['Overall']

dataset['Preferred Position'] = dataset['Preferred Positions'].str.split().str[0]
```

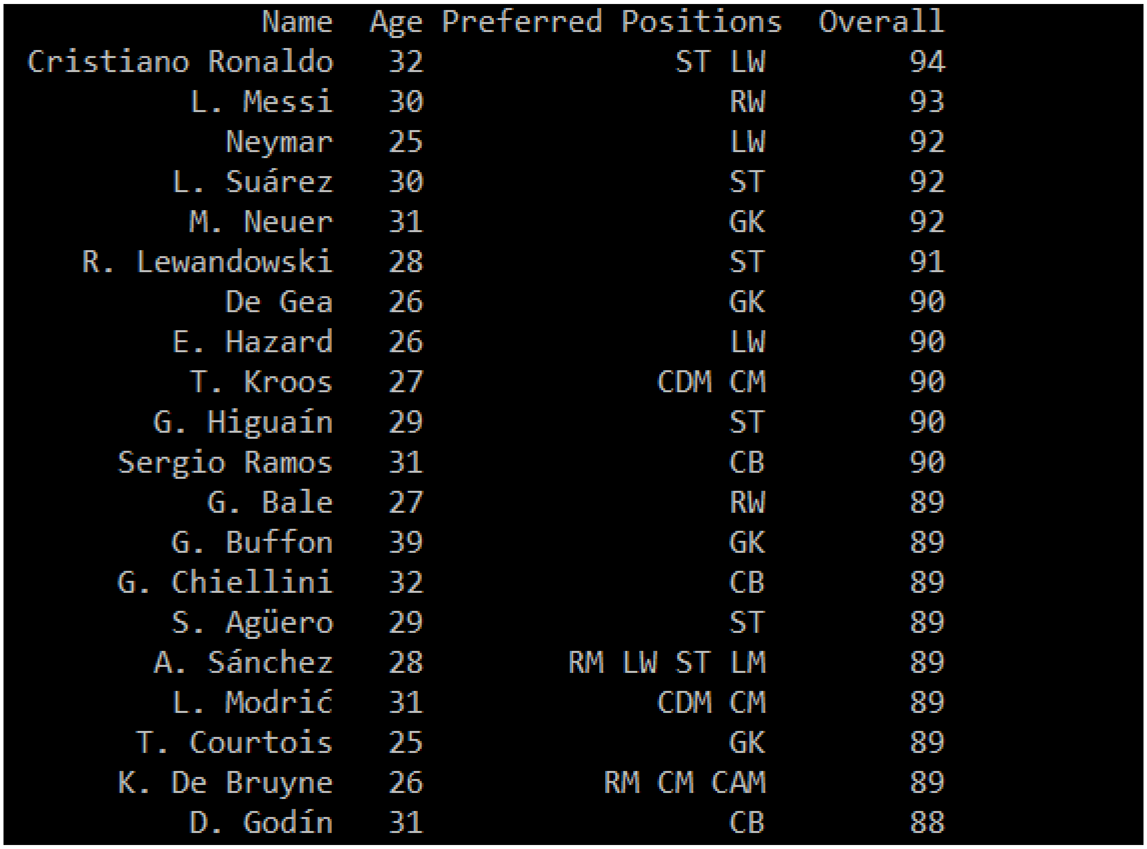
2. Clustering top players based on Overall Rating

CODE SNIPPET:



```
####Top 20 players
Top=df[['Name','Age','Preferred Positions','Overall']]
Top_20=Top.sort_values(by=['Overall'],ascending=False)
print(Top_20[:20])
```

OUTPUT:

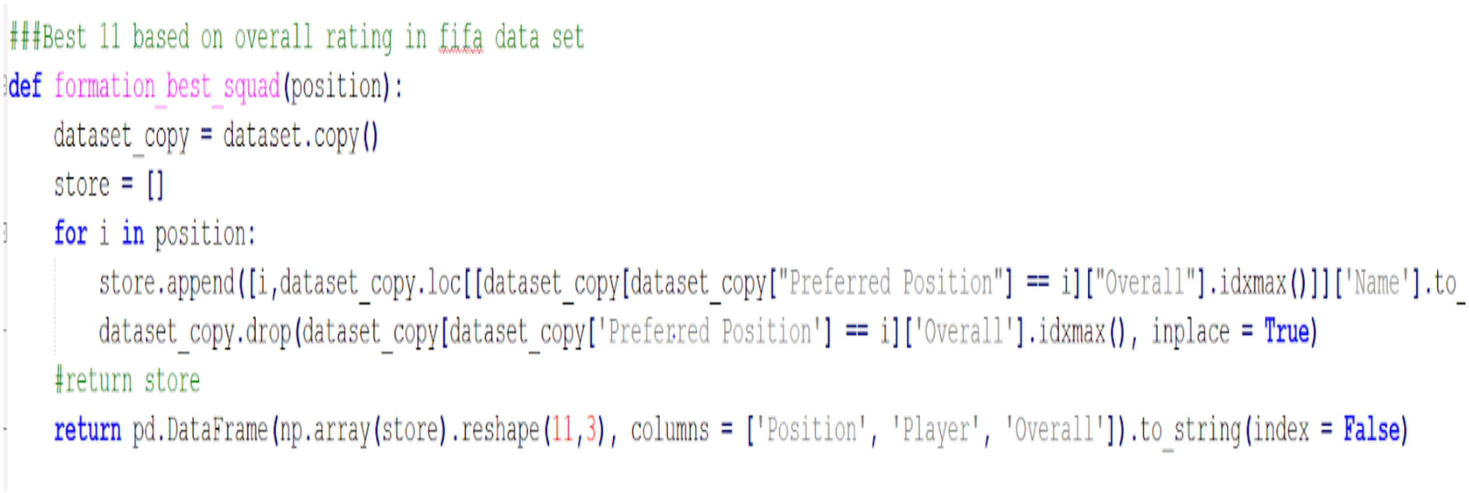


```
                Name  Age Preferred Positions  Overall
   Cristiano Ronaldo   32              ST LW        94
            L. Messi   30                 RW        93
              Neymar   25                 LW        92
           L. Suárez   30                 ST        92
            M. Neuer   31                 GK        92
      R. Lewandowski   28                 ST        91
             De Gea   26                 GK        90
           E. Hazard   26                 LW        90
            T. Kroos   27             CDM CM        90
          G. Higuaín   29                 ST        90
        Sergio Ramos   31                 CB        90
             G. Bale   27                 RW        89
           G. Buffon   39                 GK        89
        G. Chiellini   32                 CB        89
           S. Agüero   29                 ST        89
          A. Sánchez   28        RM LW ST LM        89
           L. Modrić   31             CDM CM        89
         T. Courtois   25                 GK        89
        K. De Bruyne   26          RM CM CAM        89
            D. Godín   31                 CB        88
```

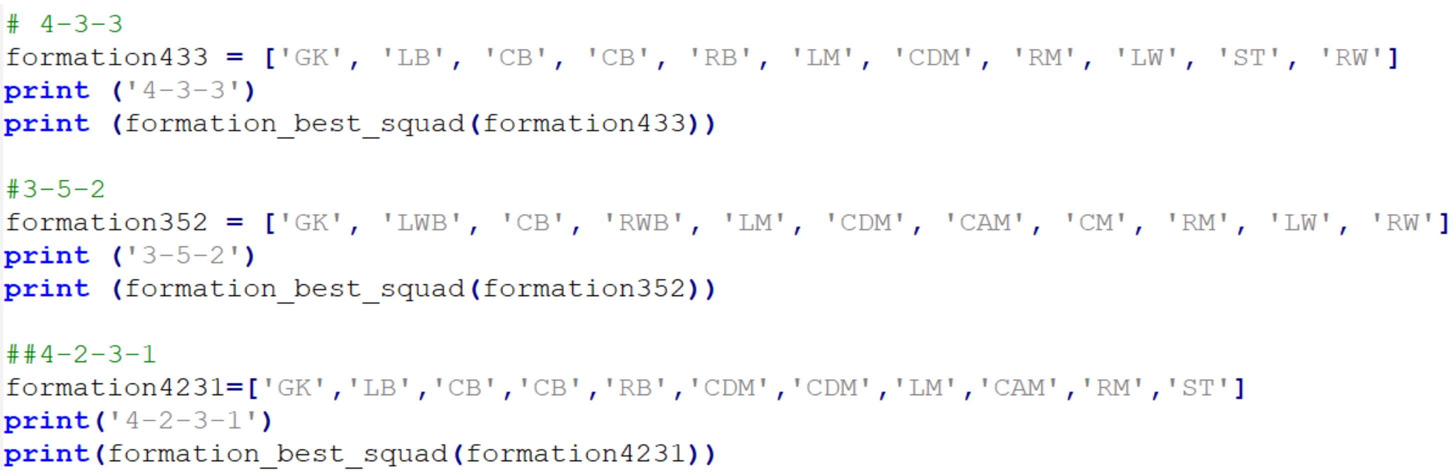
3. To identify best possible squad based on three formations

For this, first we have created a function that identifies position from preferred positions of a player and gives the player with highest overall value for that position. We did this process for three formations which are widely used by pro players throughout the world, although the function can return a value for any formation. The formations used are 4-3-3, 3-5-2 and 4-2-3-1.

CODE SNIPPET:

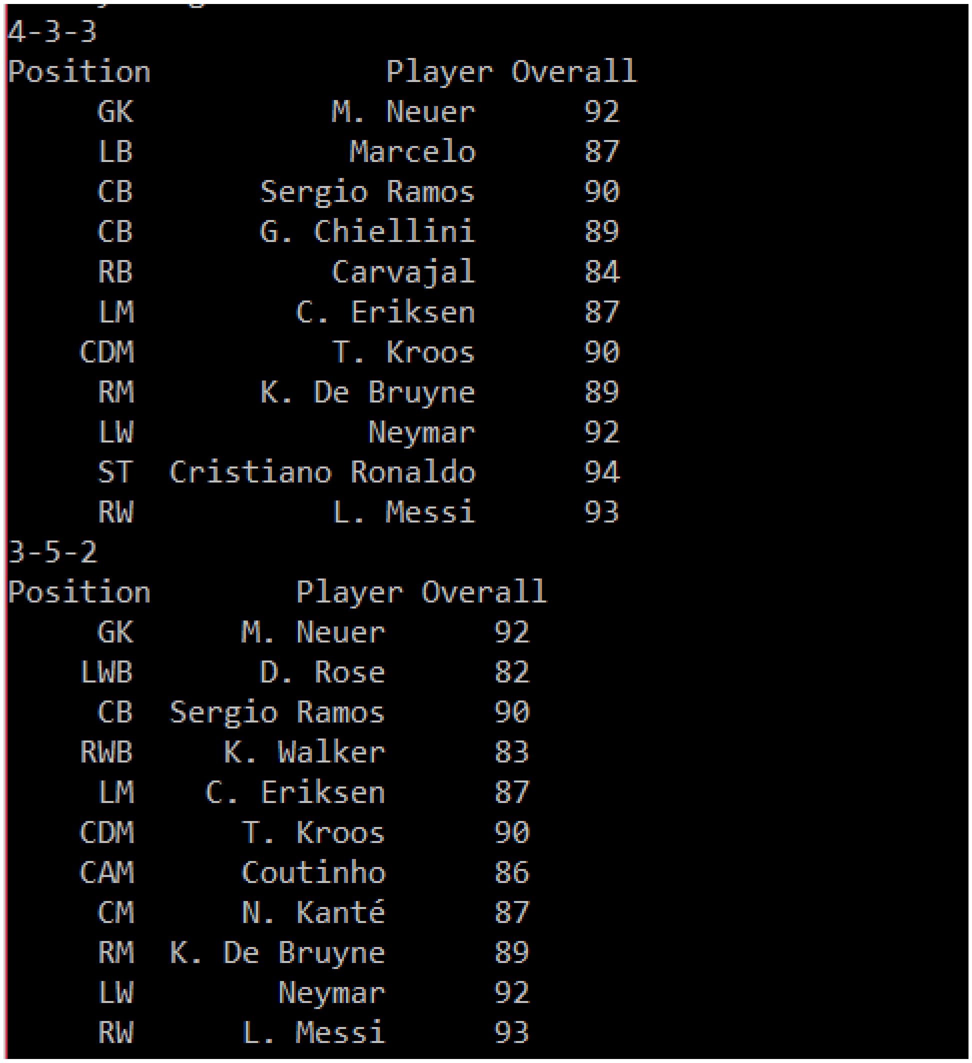


```
###Best 11 based on overall rating in fifa data set
def formation_best_squad(position):
    dataset_copy = dataset.copy()
    store = []
    for i in position:
        store.append([i,dataset_copy.loc[[dataset_copy[dataset_copy["Preferred Position"] == i]["Overall"].idxmax()]]['Name'].to_
        dataset_copy.drop(dataset_copy[dataset_copy['Preferred Position'] == i]['Overall'].idxmax(), inplace = True)
    #return store
    return pd.DataFrame(np.array(store).reshape(11,3), columns = ['Position', 'Player', 'Overall']).to_string(index = False)
```

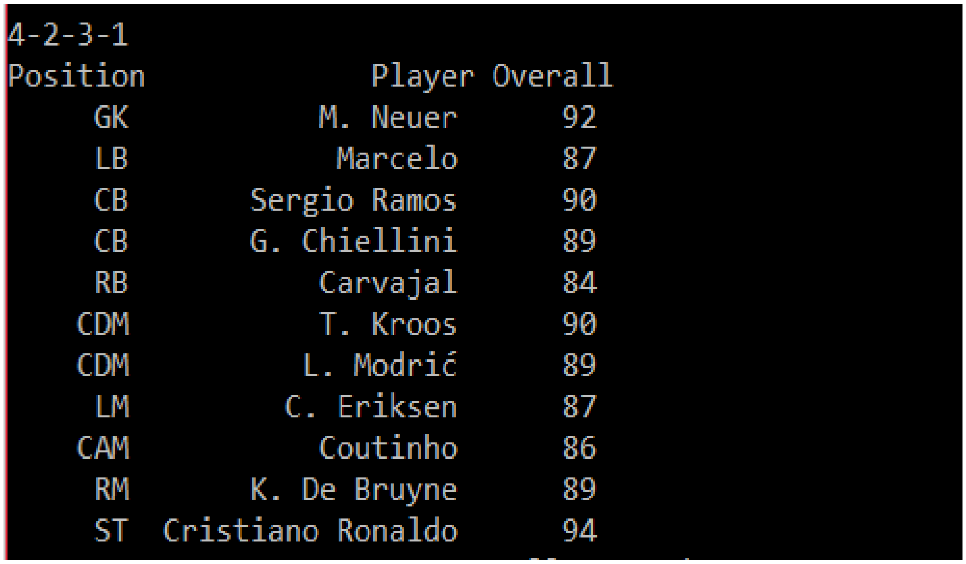


```
# 4-3-3
formation433 = ['GK', 'LB', 'CB', 'CB', 'RB', 'LM', 'CDM', 'RM', 'LW', 'ST', 'RW']
print ('4-3-3')
print (formation_best_squad(formation433))

#3-5-2
formation352 = ['GK', 'LWB', 'CB', 'RWB', 'LM', 'CDM', 'CAM', 'CM', 'RM', 'LW', 'RW']
print ('3-5-2')
print (formation_best_squad(formation352))

##4-2-3-1
formation4231=['GK','LB','CB','CB','RB','CDM','CDM','LM','CAM','RM','ST']
print('4-2-3-1')
print(formation_best_squad(formation4231))
```

OUTPUT:



```
4-3-3
Position             Player Overall
     GK           M. Neuer      92
     LB            Marcelo      87
     CB       Sergio Ramos      90
     CB       G. Chiellini      89
     RB           Carvajal      84
     LM         C. Eriksen      87
    CDM           T. Kroos      90
     RM       K. De Bruyne      89
     LW             Neymar      92
     ST  Cristiano Ronaldo      94
     RW           L. Messi      93
3-5-2
Position        Player Overall
     GK      M. Neuer      92
    LWB       D. Rose      82
     CB  Sergio Ramos      90
    RWB     K. Walker      83
     LM    C. Eriksen      87
    CDM      T. Kroos      90
    CAM      Coutinho      86
     CM      N. Kanté      87
     RM  K. De Bruyne      89
     LW        Neymar      92
     RW      L. Messi      93
```

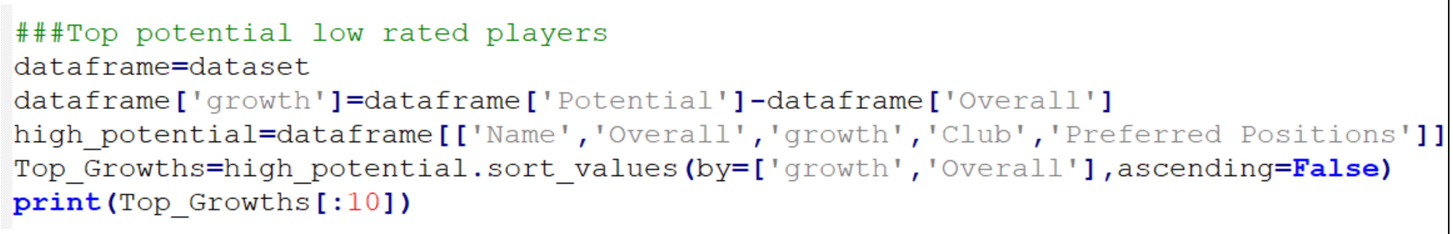


```
4-2-3-1
Position             Player Overall
     GK           M. Neuer      92
     LB            Marcelo      87
     CB       Sergio Ramos      90
     CB       G. Chiellini      89
     RB           Carvajal      84
    CDM           T. Kroos      90
    CDM          L. Modrić      89
     LM         C. Eriksen      87
    CAM           Coutinho      86
     RM       K. De Bruyne      89
     ST  Cristiano Ronaldo      94
```

4. Clustering young underrated players with high potential

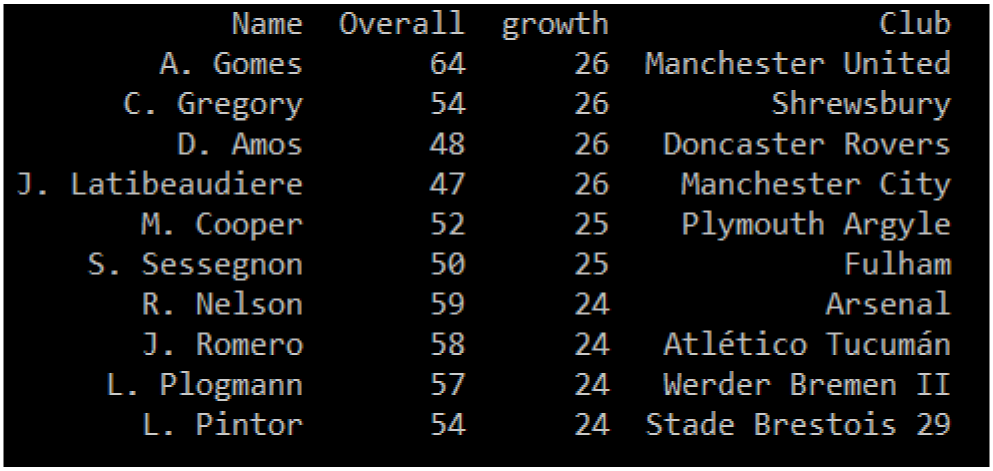
This analysis is used to identified young players with high potential, so that gamers focusing to build a team can do that cheaply as young underrated players can be bought for less coins. For this, we created a Growth column as Growth = Potential-Overall

CODE SNIPPET:



```
###Top potential low rated players
dataframe=dataset
dataframe['growth']=dataframe['Potential']-dataframe['Overall']
high_potential=dataframe[['Name','Overall','growth','Club','Preferred Positions']]
Top_Growths=high_potential.sort_values(by=['growth','Overall'],ascending=False)
print(Top_Growths[:10])
```

OUTPUT:



```
              Name  Overall  growth               Club
          A. Gomes       64      26  Manchester United
        C. Gregory       54      26         Shrewsbury
           D. Amos       48      26    Doncaster Rovers
J. Latibeaudiere       47      26     Manchester City
         M. Cooper       52      25    Plymouth Argyle
      S. Sessegnon       50      25             Fulham
         R. Nelson       59      24            Arsenal
         J. Romero       58      24   Atlético Tucumán
       L. Plogmann       57      24    Werder Bremen II
         L. Pintor       54      24  Stade Brestois 29
```

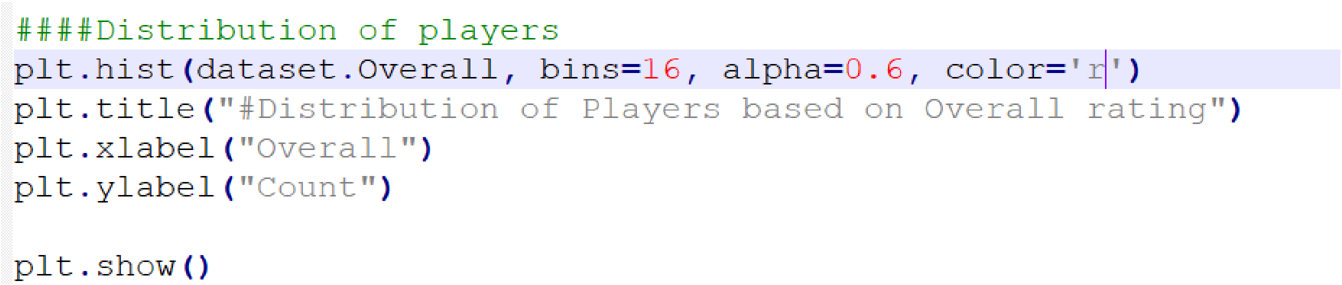
DEDUCTION:

As we can see here, A. Gomes is the player with a decent overall rating and a high growth margin. That means he can go until 64+26=90 overall rating in future.

5. Creating basic visualizations like Histograms and Line charts

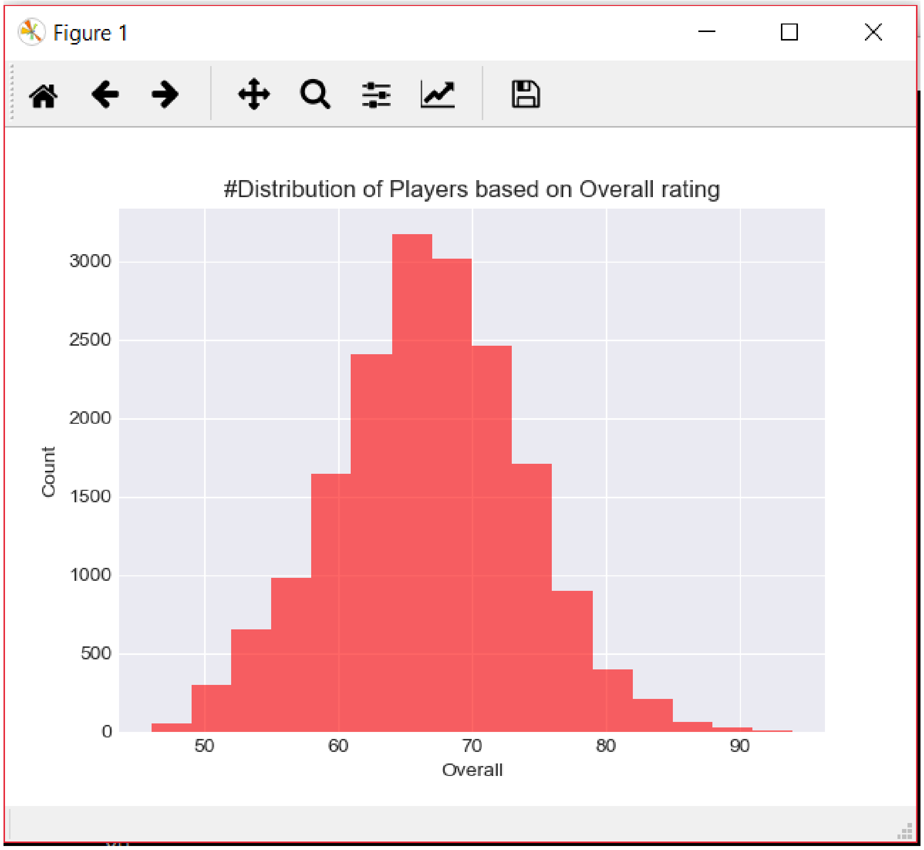
1. Distribution of players based on overall rating

CODE SNIPPET:



```
####Distribution of players
plt.hist(dataset.Overall, bins=16, alpha=0.6, color='r')
plt.title("#Distribution of Players based on Overall rating")
plt.xlabel("Overall")
plt.ylabel("Count")

plt.show()
```

OUTPUT:

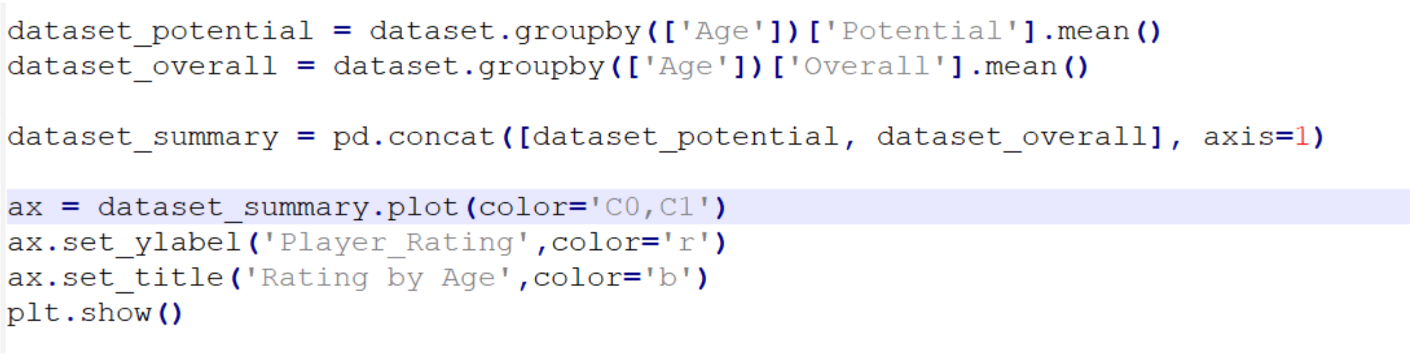


DEDUCTION:

As we can see here, the players are normally distributed around overall rating of 67 (mean rating as seen from the graph above)

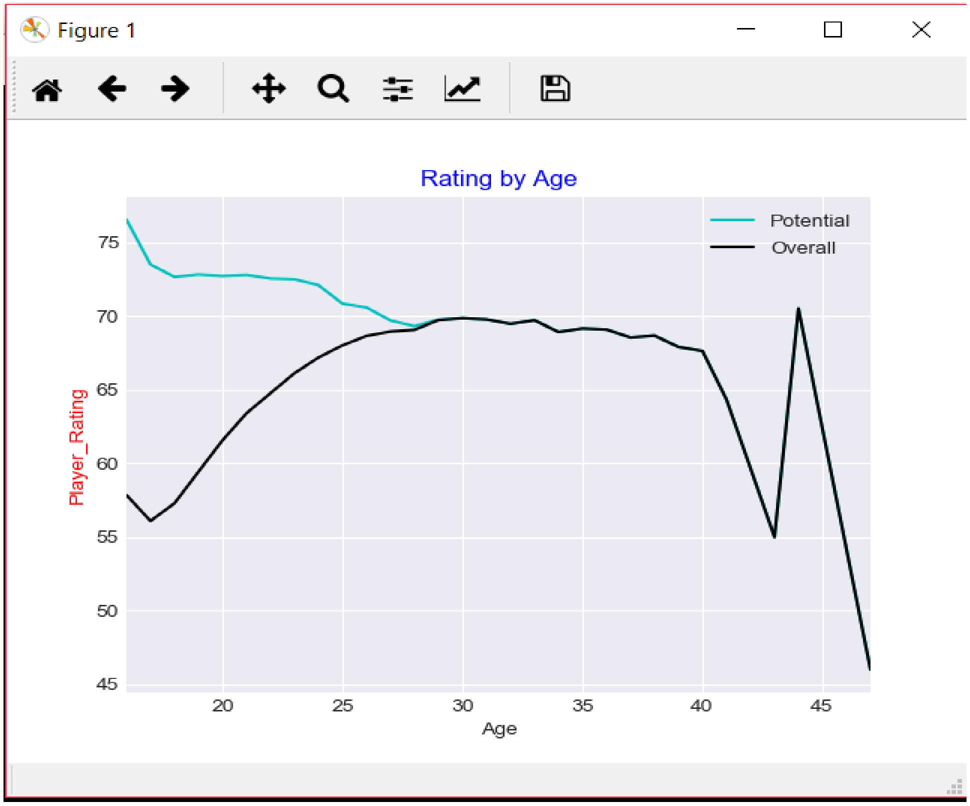
2. Plotting age against Player rating to get the peak age of player

   CODE SNIPPET:



```
dataset_potential = dataset.groupby(['Age'])['Potential'].mean()
dataset_overall = dataset.groupby(['Age'])['Overall'].mean()

dataset_summary = pd.concat([dataset_potential, dataset_overall], axis=1)

ax = dataset_summary.plot(color='C0,C1')
ax.set_ylabel('Player_Rating',color='r')
ax.set_title('Rating by Age',color='b')
plt.show()
```

OUTPUT:

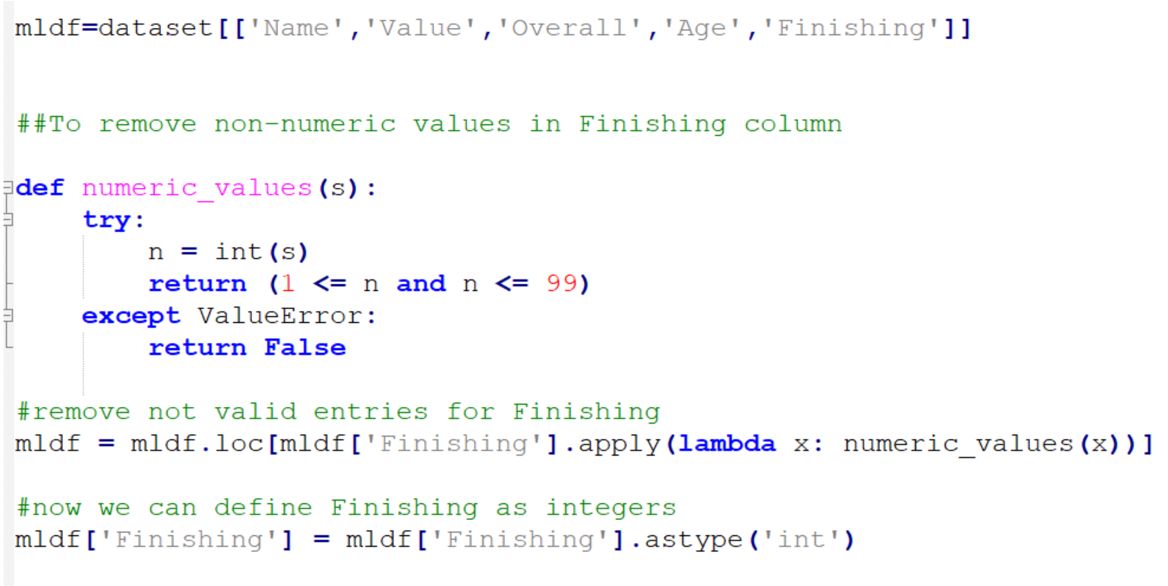


DEDUCTION: As we can see here, that a player hits his peak value around 27. Before that, potential of a player is high but the actual (Overall) rating is low.

6. Applying Machine Learning Algorithm for prediction purposes

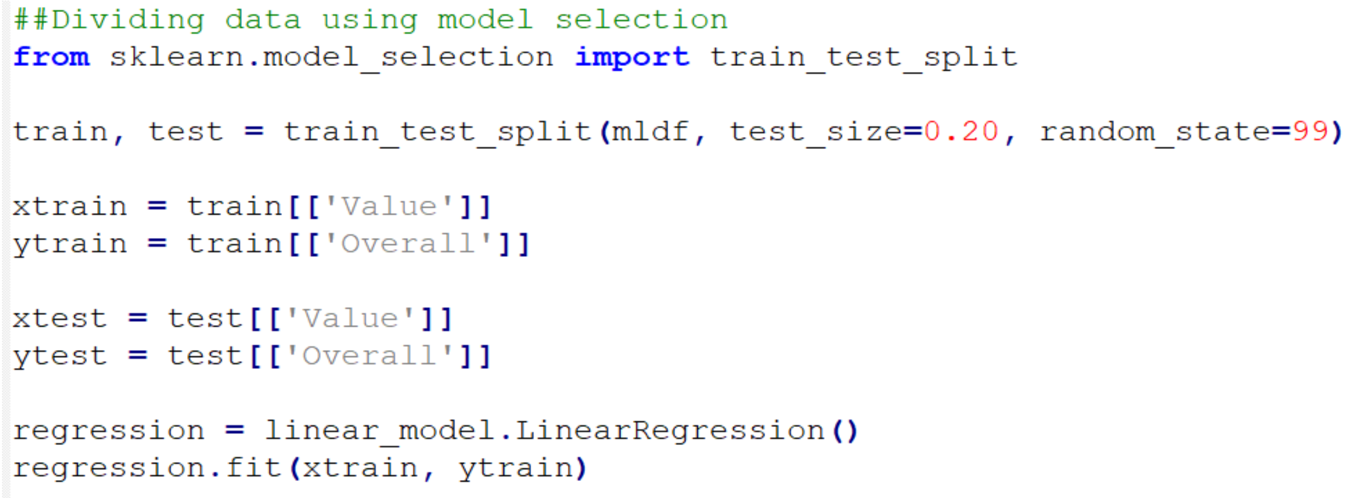
Here, we predict value of a player using Overall rating. For this we split the data into test and training set. We have used sklearn library to do this. Here, we have implemented two algorithms for this purpose and the algorithm that gives low mean squared error is considered.

First, we define the data frame for which we will used machine learning algorithms and then clean it. The finishing column has values that are not numeric. For this, we have created a function:



```
mldf=dataset[['Name','Value','Overall','Age','Finishing']]


##To remove non-numeric values in Finishing column

def numeric_values(s):
    try:
        n = int(s)
        return (1 <= n and n <= 99)
    except ValueError:
        return False

#remove not valid entries for Finishing
mldf = mldf.loc[mldf['Finishing'].apply(lambda x: numeric_values(x))]

#now we can define Finishing as integers
mldf['Finishing'] = mldf['Finishing'].astype('int')
```

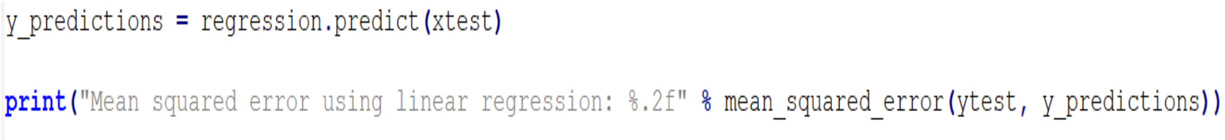
Creating training and test data sets and applying Linear Regression:



```
##Dividing data using model selection
from sklearn.model_selection import train_test_split

train, test = train_test_split(mldf, test_size=0.20, random_state=99)

xtrain = train[['Value']]
ytrain = train[['Overall']]

xtest = test[['Value']]
ytest = test[['Overall']]

regression = linear_model.LinearRegression()
regression.fit(xtrain, ytrain)
```

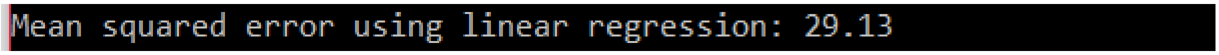
6.1. Linear Regression: This algorithm is used to predict values by training the model using fit function.

CODE SNIPPET:



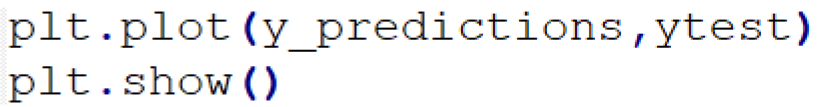
```
y_predictions = regression.predict(xtest)

print("Mean squared error using linear regression: %.2f" % mean_squared_error(ytest, y_predictions))
```

OUTPUT:

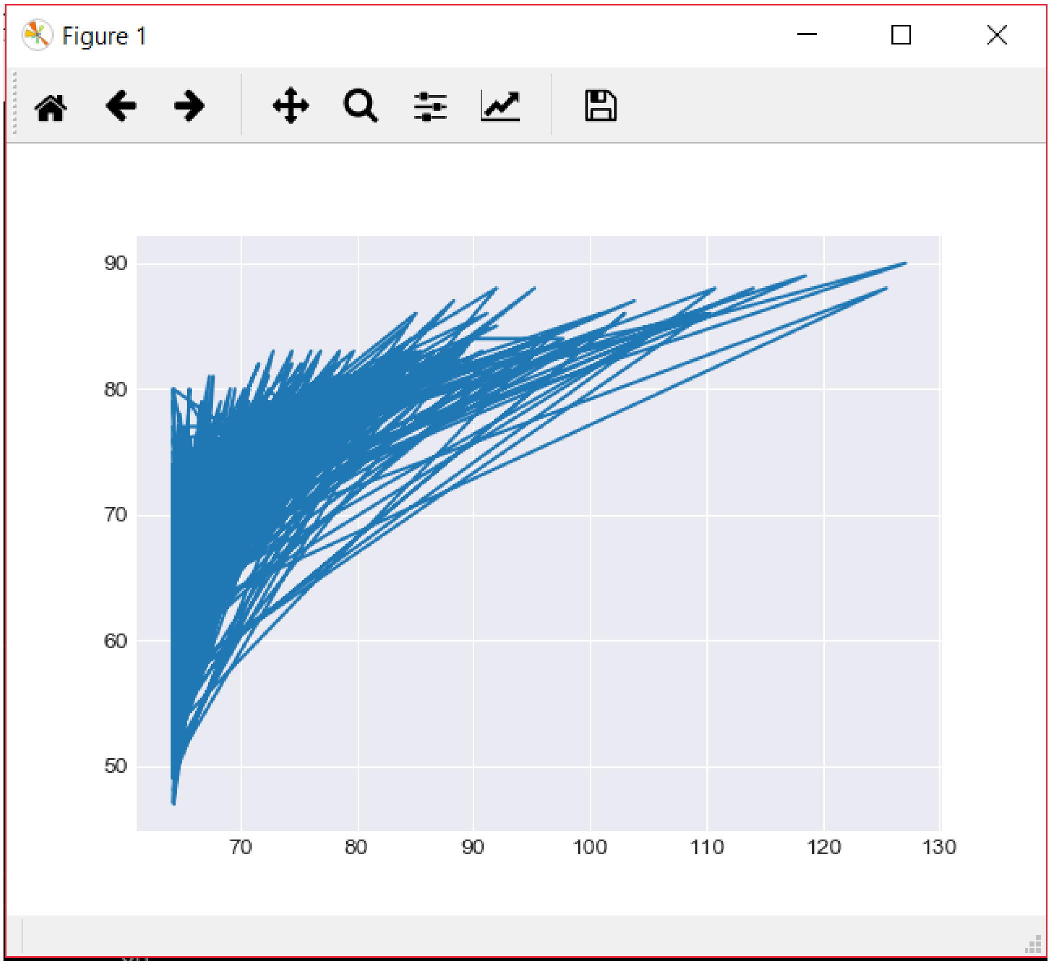


```
Mean squared error using linear regression: 29.13
```

LINEAR REGRESSION PLOT:



```
plt.plot(y_predictions,ytest)
plt.show()
```

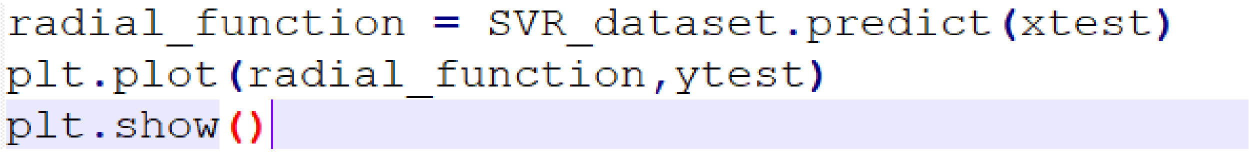


DEDUCTION:

This plot of linear regression shows that, although the model is not good, but it can be used to get a gist of how accurately the values are predicted. Given y-axis as original dataset and x-axis as predicted values.

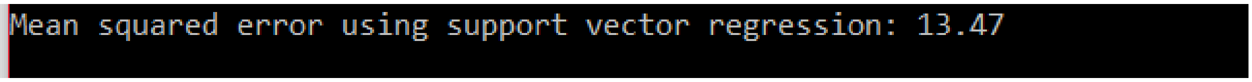
6.2. SUPPORT VECTOR REGRESSION: It is another machine learning technique to predict the values and the kernel used is radial basis function which is especially used for nonlinear problems.

CODE SNIPPET:



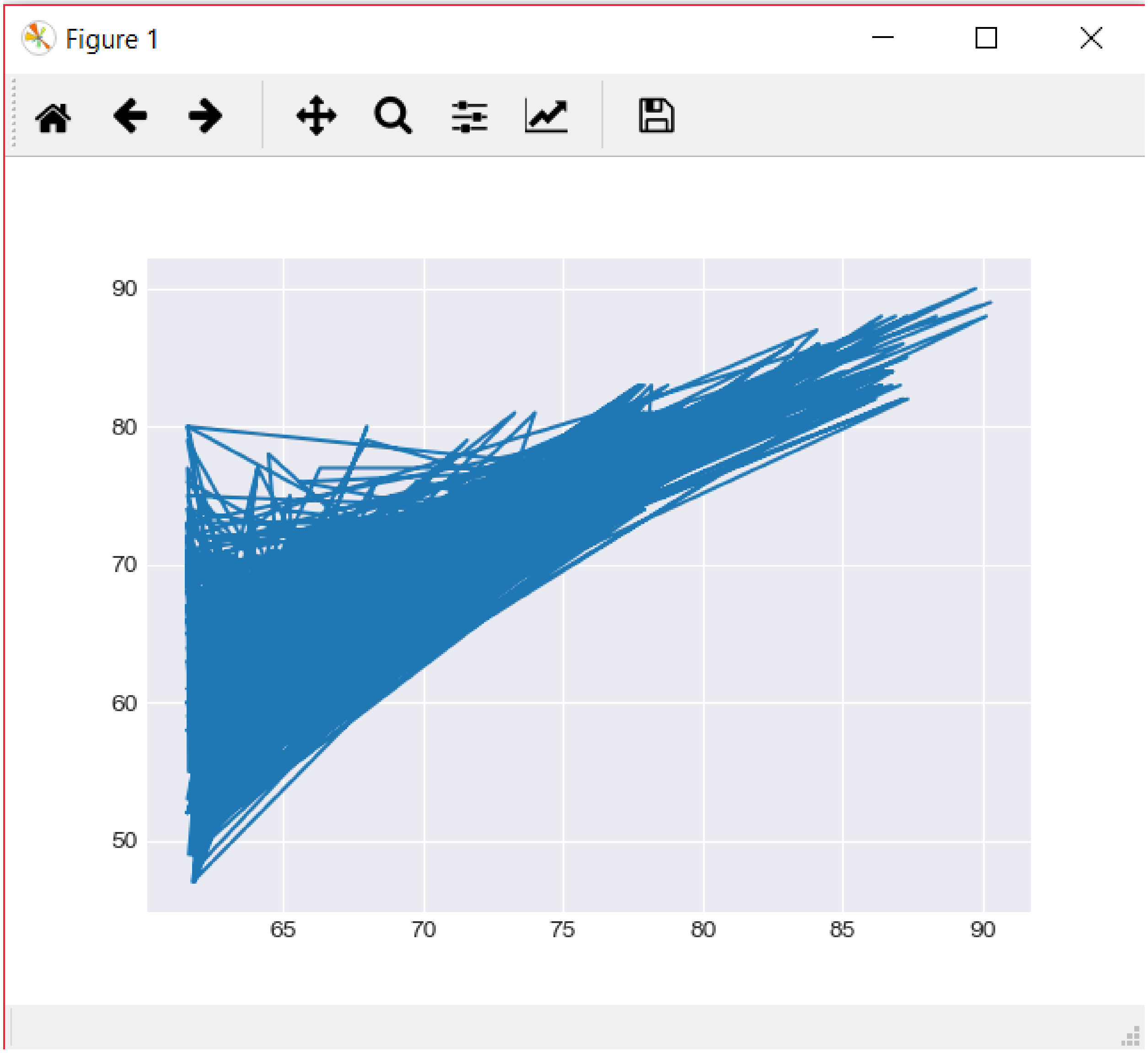
```
radial_function = SVR_dataset.predict(xtest)
plt.plot(radial_function,ytest)
plt.show()
```

OUTPUT:



```
Mean squared error using support vector regression: 13.47
```

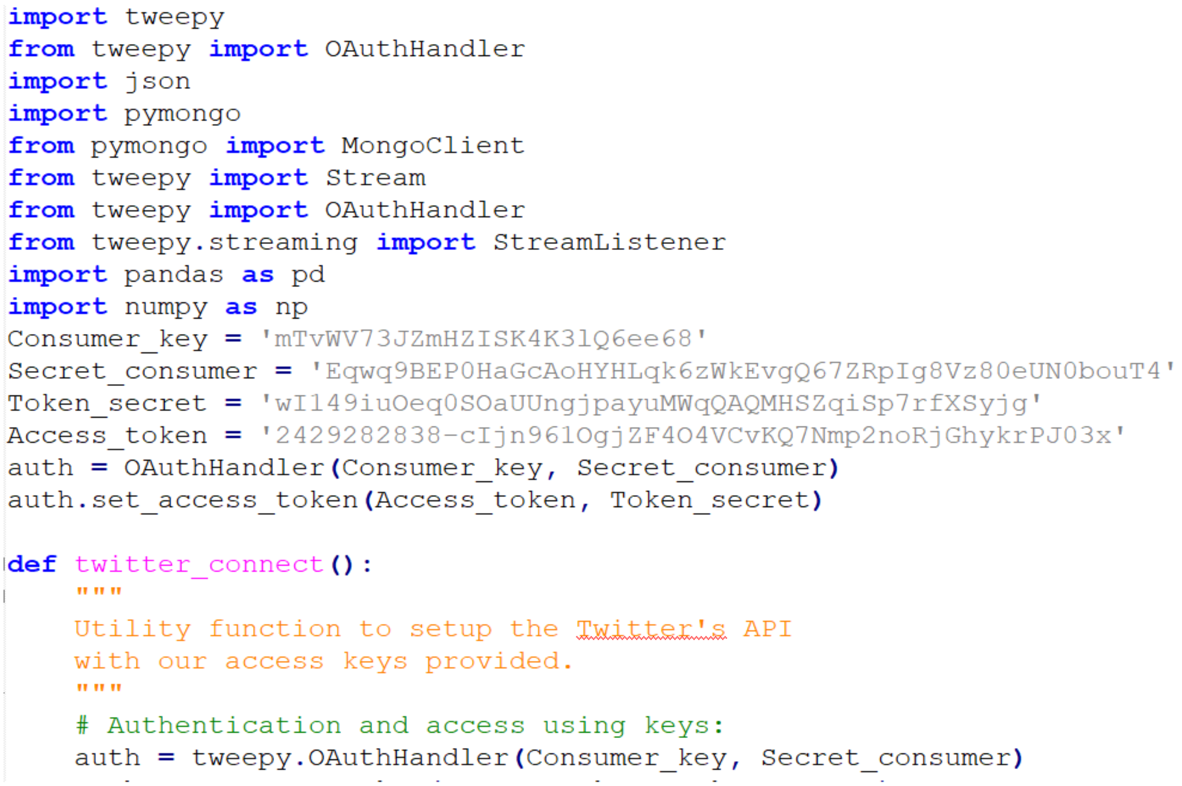
PLOT:



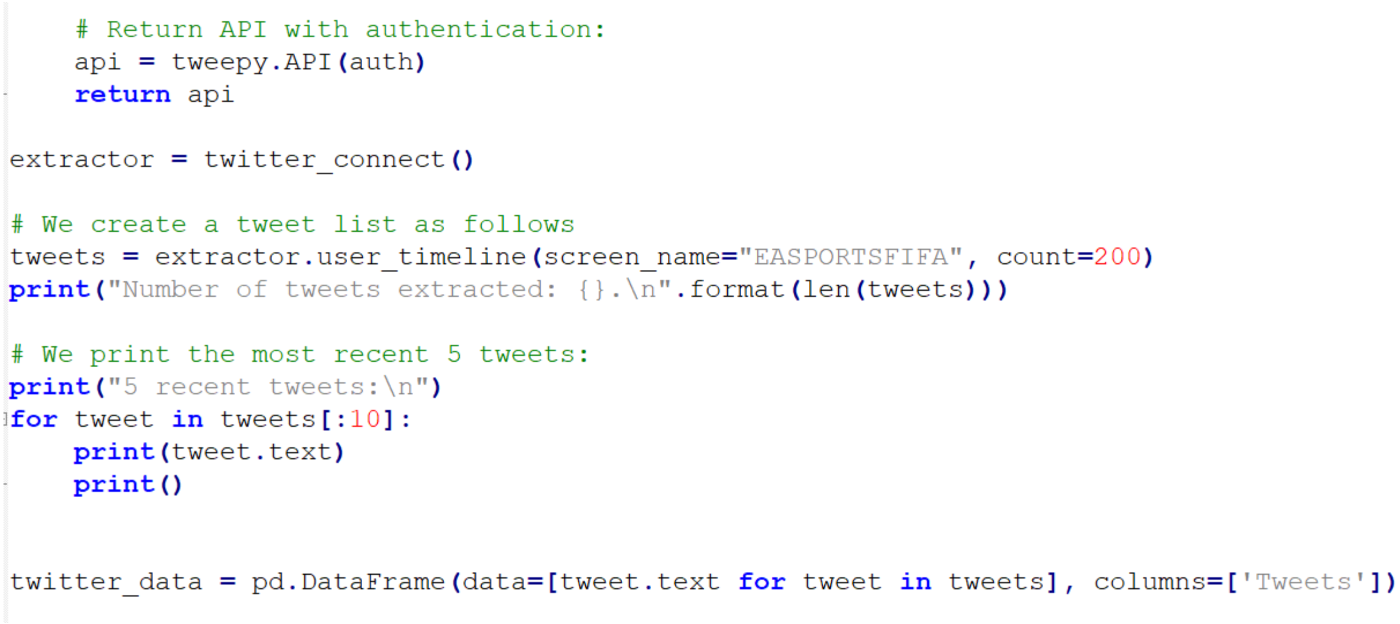
DEDUCTION: Plot of SVR is more descriptive and accurate than the plot of linear regression

7. Sentiment analysis using Twitter data

CODE SNIPPET:

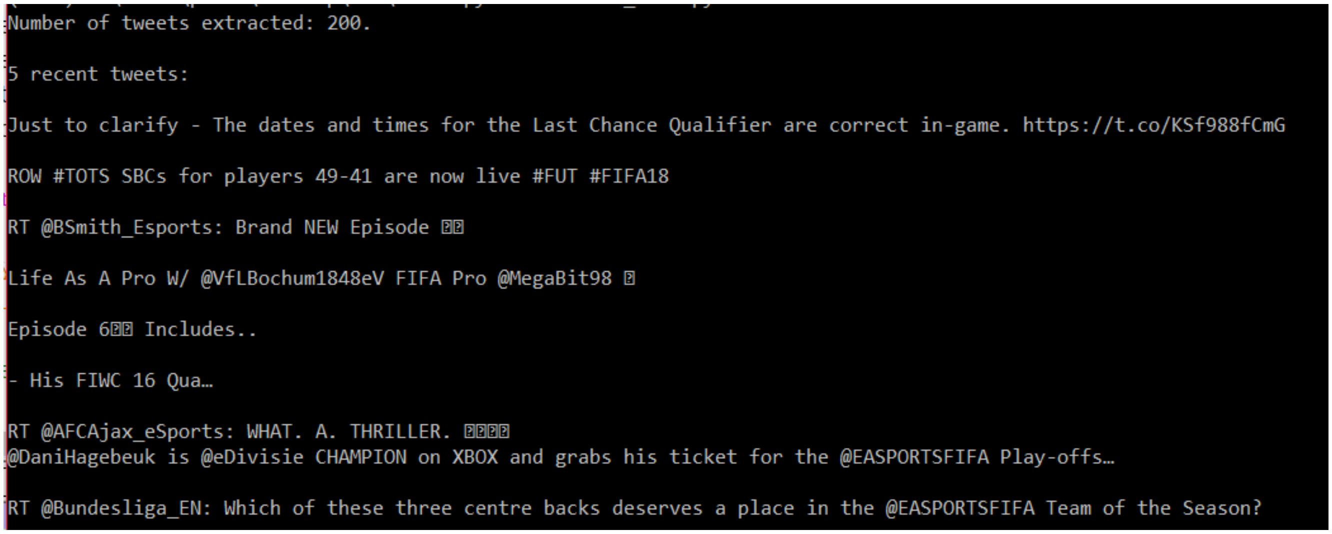


```
import tweepy
from tweepy import OAuthHandler
import json
import pymongo
from pymongo import MongoClient
from tweepy import Stream
from tweepy import OAuthHandler
from tweepy.streaming import StreamListener
import pandas as pd
import numpy as np
Consumer_key = 'mTvWV73JZmHZISK4K3lQ6ee68'
Secret_consumer = 'Eqwq9BEP0HaGcAoHYHLqk6zWkEvgQ67ZRpIg8Vz80eUN0bouT4'
Token_secret = 'wI149iuOeq0SOaUUngjpayuMWqQAQMHSZqiSp7rfXSyjg'
Access_token = '2429282838-cIjn961OgjZF4O4VCvKQ7Nmp2noRjGhykrPJ03x'
auth = OAuthHandler(Consumer_key, Secret_consumer)
auth.set_access_token(Access_token, Token_secret)

def twitter_connect():
    """
    Utility function to setup the Twitter's API
    with our access keys provided.
    """
    # Authentication and access using keys:
    auth = tweepy.OAuthHandler(Consumer_key, Secret_consumer)
```



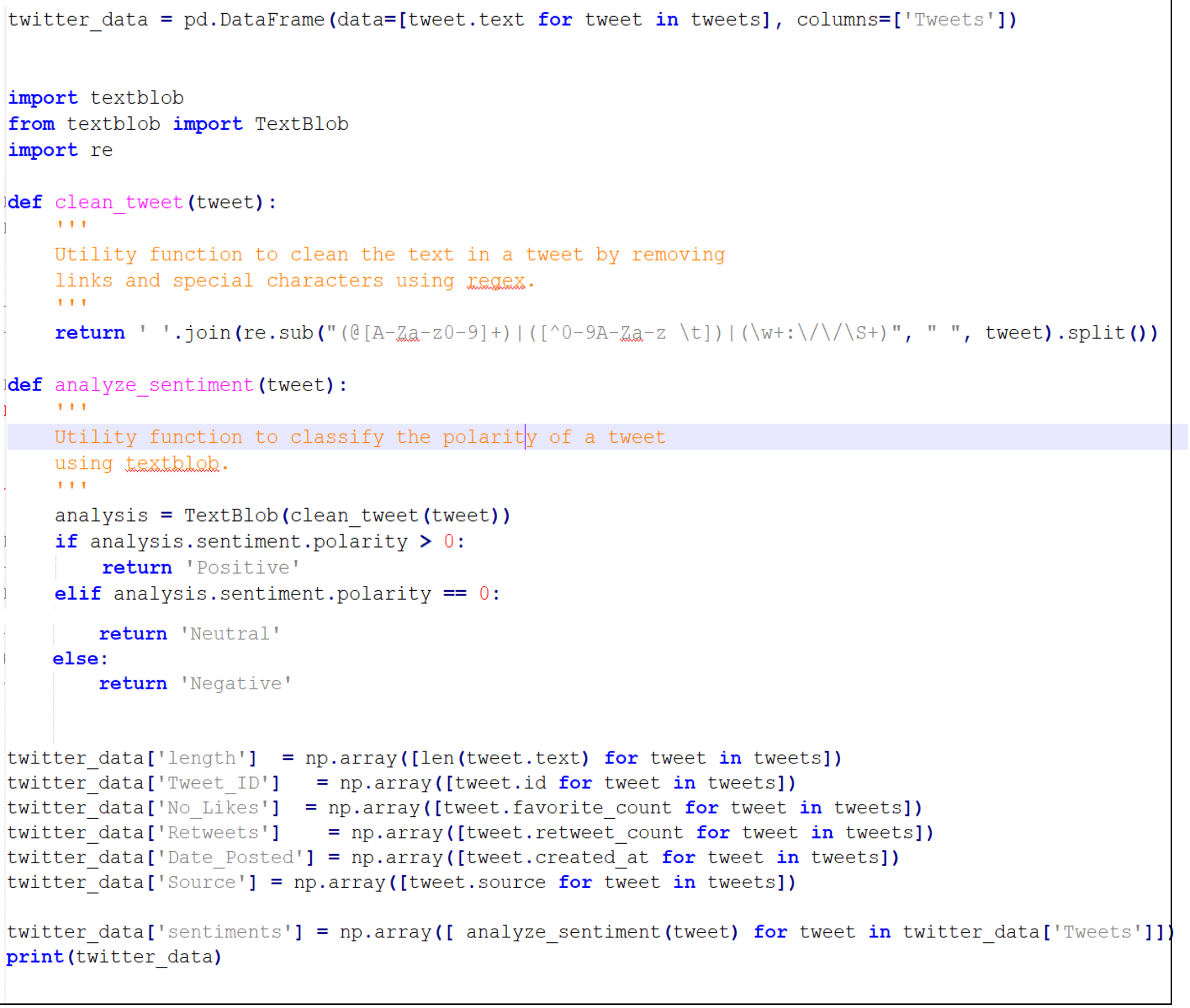
```
    # Return API with authentication:
    api = tweepy.API(auth)
    return api

extractor = twitter_connect()

# We create a tweet list as follows
tweets = extractor.user_timeline(screen_name="EASPORTSFIFA", count=200)
print("Number of tweets extracted: {}.\n".format(len(tweets)))

# We print the most recent 5 tweets:
print("5 recent tweets:\n")
for tweet in tweets[:10]:
    print(tweet.text)
    print()


twitter_data = pd.DataFrame(data=[tweet.text for tweet in tweets], columns=['Tweets'])
```

RECENT TWEETS:



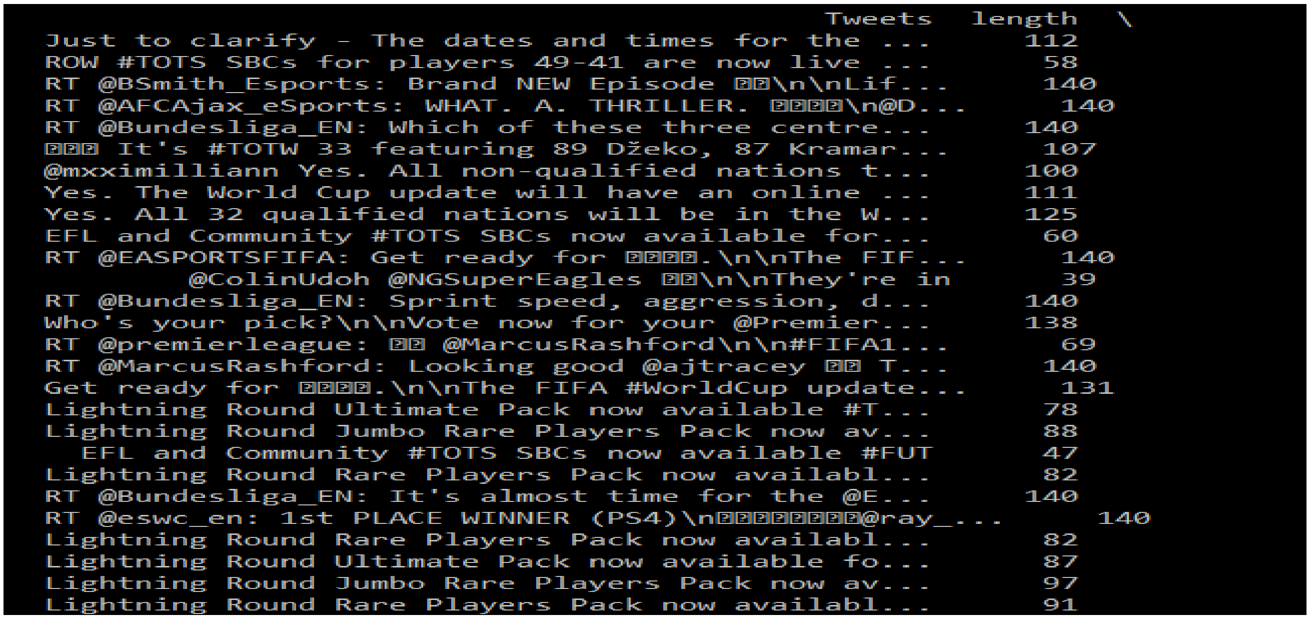
```
Number of tweets extracted: 200.

5 recent tweets:

Just to clarify - The dates and times for the Last Chance Qualifier are correct in-game. https://t.co/KSf988fCmG

ROW #TOTS SBCs for players 49-41 are now live #FUT #FIFA18

RT @BSmith_Esports: Brand NEW Episode □□

Life As A Pro W/ @VfLBochum1848eV FIFA Pro @MegaBit98 □

Episode 6□□ Includes..

- His FIWC 16 Qua…

RT @AFCAjax_eSports: WHAT. A. THRILLER. □□□□
@DaniHagebeuk is @eDivisie CHAMPION on XBOX and grabs his ticket for the @EASPORTSFIFA Play-offs…

RT @Bundesliga_EN: Which of these three centre backs deserves a place in the @EASPORTSFIFA Team of the Season?
```

USING TEXTBLOB FOR SENTIMENT ANALYSIS:



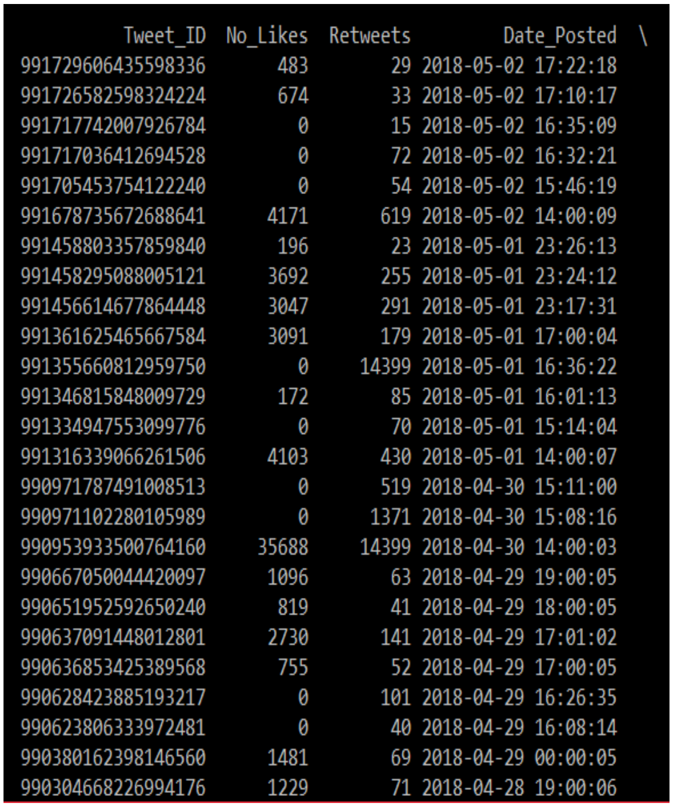
```python
twitter_data = pd.DataFrame(data=[tweet.text for tweet in tweets], columns=['Tweets'])


import textblob
from textblob import TextBlob
import re

def clean_tweet(tweet):
    '''
    Utility function to clean the text in a tweet by removing
    links and special characters using regex.
    '''
    return ' '.join(re.sub("(@[A-Za-z0-9]+)|([^0-9A-Za-z \t])|(\w+:\/\/\S+)", " ", tweet).split())

def analyze_sentiment(tweet):
    '''
    Utility function to classify the polarity of a tweet
    using textblob.
    '''
    analysis = TextBlob(clean_tweet(tweet))
    if analysis.sentiment.polarity > 0:
        return 'Positive'
    elif analysis.sentiment.polarity == 0:

        return 'Neutral'
    else:
        return 'Negative'


twitter_data['length']  = np.array([len(tweet.text) for tweet in tweets])
twitter_data['Tweet_ID']   = np.array([tweet.id for tweet in tweets])
twitter_data['No_Likes']  = np.array([tweet.favorite_count for tweet in tweets])
twitter_data['Retweets']    = np.array([tweet.retweet_count for tweet in tweets])
twitter_data['Date_Posted'] = np.array([tweet.created_at for tweet in tweets])
twitter_data['Source'] = np.array([tweet.source for tweet in tweets])

twitter_data['sentiments'] = np.array([ analyze_sentiment(tweet) for tweet in twitter_data['Tweets']])
print(twitter_data)
```

TWEETS EXTRACTED:

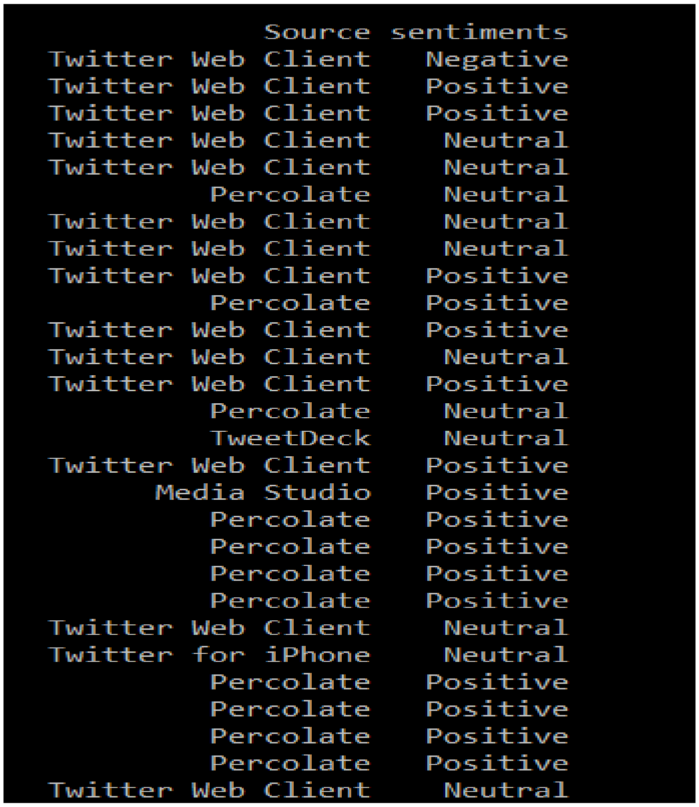


```
                                              Tweets  length  \
  Just to clarify - The dates and times for the ...     112
  ROW #TOTS SBCs for players 49-41 are now live ...      58
  RT @BSmith_Esports: Brand NEW Episode ⍰⍰\n\nLif...     140
  RT @AFCAjax_eSports: WHAT. A. THRILLER. ⍰⍰⍰⍰\n@D...     140
  RT @Bundesliga_EN: Which of these three centre...     140
  ⍰⍰⍰ It's #TOTW 33 featuring 89 Džeko, 87 Kramar...     107
  @mxximilliann Yes. All non-qualified nations t...     100
  Yes. The World Cup update will have an online ...     111
  Yes. All 32 qualified nations will be in the W...     125
  EFL and Community #TOTS SBCs now available for...      60
  RT @EASPORTSFIFA: Get ready for ⍰⍰⍰⍰.\n\nThe FIF...     140
          @ColinUdoh @NGSuperEagles ⍰⍰\n\nThey're in      39
  RT @Bundesliga_EN: Sprint speed, aggression, d...     140
  Who's your pick?\n\nVote now for your @Premier...     138
  RT @premierleague: ⍰⍰ @MarcusRashford\n\n#FIFA1...      69
  RT @MarcusRashford: Looking good @ajtracey ⍰⍰ T...     140
  Get ready for ⍰⍰⍰⍰.\n\nThe FIFA #WorldCup update...     131
  Lightning Round Ultimate Pack now available #T...      78
  Lightning Round Jumbo Rare Players Pack now av...      88
    EFL and Community #TOTS SBCs now available #FUT      47
  Lightning Round Rare Players Pack now availabl...      82
  RT @Bundesliga_EN: It's almost time for the @E...     140
  RT @eswc_en: 1st PLACE WINNER (PS4)\n⍰⍰⍰⍰⍰⍰⍰⍰@ray_...      140
  Lightning Round Rare Players Pack now availabl...      82
  Lightning Round Ultimate Pack now available fo...      87
  Lightning Round Jumbo Rare Players Pack now av...      97
  Lightning Round Rare Players Pack now availabl...      91
```

TWEETS AND THEIR NUMBER OF LIKES, RETWEETS AND SENTIMENTS:



```
           Tweet_ID  No_Likes  Retweets         Date_Posted  \
 991729606435598336       483        29 2018-05-02 17:22:18
 991726582598324224       674        33 2018-05-02 17:10:17
 991717742007926784         0        15 2018-05-02 16:35:09
 991717036412694528         0        72 2018-05-02 16:32:21
 991705453754122240         0        54 2018-05-02 15:46:19
 991678735672688641      4171       619 2018-05-02 14:00:09
 991458803357859840       196        23 2018-05-01 23:26:13
 991458295088005121      3692       255 2018-05-01 23:24:12
 991456614677864448      3047       291 2018-05-01 23:17:31
 991361625465667584      3091       179 2018-05-01 17:00:04
 991355660812959750         0     14399 2018-05-01 16:36:22
 991346815848009729       172        85 2018-05-01 16:01:13
 991334947553099776         0        70 2018-05-01 15:14:04
 991316339066261506      4103       430 2018-05-01 14:00:07
 990971787491008513         0       519 2018-04-30 15:11:00
 990971102280105989         0      1371 2018-04-30 15:08:16
 990953933500764160     35688     14399 2018-04-30 14:00:03
 990667050044420097      1096        63 2018-04-29 19:00:05
 990651952592650240       819        41 2018-04-29 18:00:05
 990637091448012801      2730       141 2018-04-29 17:01:02
 990636853425389568       755        52 2018-04-29 17:00:05
 990628423885193217         0       101 2018-04-29 16:26:35
 990623806333972481         0        40 2018-04-29 16:08:14
 990380162398146560      1481        69 2018-04-29 00:00:05
 990304668226994176      1229        71 2018-04-28 19:00:06
```



```
                Source sentiments
    Twitter Web Client   Negative
    Twitter Web Client   Positive
    Twitter Web Client   Positive
    Twitter Web Client    Neutral
    Twitter Web Client    Neutral
             Percolate    Neutral
    Twitter Web Client    Neutral
    Twitter Web Client    Neutral
    Twitter Web Client   Positive
             Percolate   Positive
    Twitter Web Client   Positive
    Twitter Web Client    Neutral
    Twitter Web Client   Positive
             Percolate    Neutral
             TweetDeck    Neutral
    Twitter Web Client   Positive
          Media Studio   Positive
             Percolate   Positive
             Percolate   Positive
             Percolate   Positive
             Percolate   Positive
    Twitter Web Client    Neutral
    Twitter for iPhone    Neutral
             Percolate   Positive
             Percolate   Positive
             Percolate   Positive
             Percolate   Positive
    Twitter Web Client    Neutral
```

DEDUCTION:

As we can see from the sentiments, most of the tweets are positive or neutral, which means there is not a lot of negative feedback for the game and EASPORTSFIFA in general.

8. Contribution

CONTRIBUTIONS	
HARSH TAKRANI	PRANAY LULLA
1. Data extraction and preprocessing	1. Clustering top players based on overall rating
2. Finding top potential young underrated players (young players <= 23 age)	2. Debugging the whole code
3. Machine Learning Algorithms	3. Finding best team based on three formations
4. Reporting	4. Creating visualizations like histograms and line charts
	5. Sentiment analysis using Twitter data
FINAL PROJECT PRESENTATION (TEAM)	

9. Conclusion:

The part of pre-processing was challenging. We had to figure out the useful columns and to modify the columns according to the project requirements. We came across certain unexpected players which can be considered for buying as the high potential players. The list of top players was expected, since it was based on overall ratings of the players.

Furthermore, for machine learning algorithms we had to run different type of models using different attributes to reduce the mean squared error and figure out the kernel for support vector regression. Finally, extracting data from twitter just for FIFA 18 was challenging and we found an uncomplicated way to figure out the sentiments of tweets using Text Blob package in python. Overall, the project was interesting, and the trends identified using the visualizations were useful in answering the business questions.