Code the Game – Stage 1 Report

Team ID: CO1051

Complete Code and Other Files:

1. **Python-3 Codes:**

commentary_extract.py

```python
import re
import pandas as pd

file = open('test.txt','r')

data = []
data = file.readlines(-1)
#print(data)
pattern = r"([0-9]\.[0-9][0-9]pm)"
time = []
commentary = []

#pdb.set_trace()
for i in range(len(data)):
    ext = re.split(pattern,data[i])
    for j in range(len(ext)):
        if(re.match(pattern,ext[j])):
            time.append(ext[j])
            commentary.append(ext[j+1])

df = pd.DataFrame(data = {"Time":time,"Commentary":commentary})
df.to_csv("./Commentary.csv", sep=',',index=False)

```

commentary_extract.py | end_of_over.py | Final_PdfT

```
import pandas as pd
import re

file = open('test.txt','r')
data = []
data = file.readlines(-1)
end = []
for i in range(21):
    j=0
    small_data = data[i]
    #print(i)
    #print(data[i])
    while(j<100):
        pattern = r"(MCID=\""+str(j)+r"\">)(.*?)(</P>)"
        ext_data = re.search(pattern, small_data)
        if ext_data is None:
            break;
        #print(ext_data.group(2))
        if('|' in ext_data.group(2)):
            end.append(ext_data.group(2))
        j = j+1

end1=[]
j=0
i=0
while i<(len(end)):
    if(len(end[i])<=10):
        end1.append(end[i]+end[i+1])
        i=i+1
    else:
        end1.append(end[i])

    j=j+1
    i=i+1
```

commentary_extract.py | end_of_over.py | Final_PdfToCsv.py | Final_with_sentim

```
            end1.append(end[i]+end[i+1])
            i=i+1
        else:
            end1.append(end[i])

        j=j+1
        i=i+1

    # for i in range(len(end1)):
    #    print(end1[i])

    df = pd.DataFrame(data = {"Over":[] , "Runs Scored":[] ,"Wickets": [], "Total":[]})
    # print(df.columns)

    for i in range(20):
        string = ""
        count = 0
        str2 = ""
        x=0
        string = end1[i]
        for j in string:
            # print(j)
            if(j == '|' and count == 0):
                df["Over"][i] = (str2)
                str2 = ""
                count = count+1
            elif(j == 'R' and count == 1 ):
                df["Runs Scored"][i] = (str2)
                str2 = ""
                count = count+1
            elif(j =="s" and count ==2):
                str2 = ""
                # count = count+1
            elif( j == "W" and count ==2):
                df["Wickets"][i] = str2
                str2 = ""
                x = 1
            elif(j =="|" and count == 2):
```

```
    string = end1[i]
    for j in string:
        # print(j)
        if(j == '|' and count == 0):
            df["Over"][i] = (str2)
            str2 = ""
            count = count+1
        elif(j == 'R' and count == 1 ):
            df["Runs Scored"][i] = (str2)
            str2 = ""
            count = count+1
        elif(j =="s" and count ==2):
            str2 = ""
            # count = count+1
        elif( j == "W" and count ==2):
            df["Wickets"][i] = str2
            str2 = ""
            x = 1
        elif(j =="|" and count == 2):
            # if(x==0):
            #    df["Wickets"][i] = 0
            str2 = ""
            count = count+1
        elif(j ==":" and count ==3):
            str2 = ""
            count = count+1
        else:
            str2 = str2+j
    df["Total"][i] = (str2)

df1 = pd.DataFrame(data = {"Overs" : df.Over , "Runs": df["Runs Scored"],"Wickets" : df.Wickets , "Total" : df.Total})
#dataframe.to_csv("end_of_over_scores.csv", sep = ',' , index = False)
df1.Wickets.fillna(0 , inplace = True)
df1.to_csv("./EOO_Summary.csv", sep=',',index=False)
```

commentary_extract.py | end_of_over.py | Final_PdfToCsv.py | Final_with_sentiment.py

```
import re
import pandas as pd

file = open('test.txt','r')
data = []
data = file.readlines(-1)

over = []
score = []
commentry = []
string = []
flag=1
var=0
overcount=0
commOnprevPage=0
test = ""

for i in range(21):
    j=0
    small_data = data[i]
    while(j<100):
        pattern = r"(MCID=\""+str(j)+r"\">)(.*?)(</P>)"
        ext_data = re.search(pattern, small_data)
        string = []
        if ext_data is None:
            break;
        else:
            var=0
            if(len(ext_data.group(2)) >5 and overcount>0):
                if(flag==1):
                    while(1):
                        if(len(ext_data.group(2)) <=5 and ext_data.group(2)!=""):
                            if j==0 and commOnprevPage==1 and len(ext_data.group(2))<=5:
                                commentry.append([test])
                                print("111111============",test)
                                test=""
                                commOnprevPage=0
                            elif commOnprevPage==1 and len(ext_data.group(2))<=5:
                                commentry.append([test])
                                print("33333333333============",test)
                                test=""
                                commOnprevPage=0
```

commentary_extract.py | end_of_over.py | Final_PdfToCsv.py | Final_with_sentiment.py | score_vs_

```
                                print("33333333333============",test)
                                test=""
                                commOnprevPage=0
                            else:
                                commentry.append(string)
                            if("E" not in ext_data.group(2)):
                                #j = j-1
                                flag=1
                                var=1
                                break
                            else:
                                #j = j-1
                                flag=0
                                var=1
                                break
                        else:
                            if j==0 and commOnprevPage==1 and len(ext_data.group(2))>5:
                                commOnprevPage=0
                                test += "".join(ext_data.group(2))
                                print("5555555==========",test)
                                string.append(test)
                                test=""
                            else:
                                string.append(ext_data.group(2))
                                print(string)
                                print("commOnprevPage =  ",commOnprevPage)
                            if("|" in "".join(string)):
                                var=1
                                break
                            else:
                                None
                        if var==1:
                            break
                        j = j+1
                        pattern = r"(MCID=\""+str(j)+r"\">)(.*?)(</P>)"
                        ext_data = re.search(pattern, small_data)
                        if ext_data is None:
                            if(string):
                                commOnprevPage=1
                                test = "".join(string)
                                print("22222================",test)
                            else:
```

commentary_extract.py | end_of_over.py | Final_PdfToCsv.py | Final_with_sentiment.py | score_vs_ball.py

```
                                test = "".join(string)
                                print("22222================",test)
                            else:
                                commOnprevPage=0
                                #commentry.append(string)
                            break

                    while(1):
                        if ext_data is None:
                            break;
                        #print(ext_data.group(2))
                        if("." not in ext_data.group(2) and len(ext_data.group(2)) >5):
                            None
                        elif("." in ext_data.group(2) and len(ext_data.group(2))<=5):
                            flag=1
                            #j = j-1
                            break
                        j = j+1
                        pattern = r"(MCID=\""+str(j)+r"\">)(.*?)(</P>)"
                        ext_data = re.search(pattern, small_data)
                        if ext_data is None:
                            break;
                    #commentry.append(ext_data.group(2))
                if(ext_data is None):
                    break
                if(len(ext_data.group(2))<=5 and ext_data.group(2)!="" and ext_data.group(2)!=" "):
                    if(j==0 or j==1 and commOnprevPage==1):
                        commOnprevPage=0
                        commentry.append([test])
                    #   print("44444444============",test)
                        test=""
                    if("." not in ext_data.group(2)):
                        if(len(ext_data.group(2))==1 and "D" not in ext_data.group(2) and "O" not in ext_data.group(2)):
                            score.append(ext_data.group(2))
                        elif("lb" not in ext_data.group(2) and "w" not in ext_data.group(2) and "nb" not in ext_data.group(2)):
                            None
                        else:
                            score.append(ext_data.group(2))
                    else:
                        x = len(ext_data.group(2))
                        overcount=1
                        over.append((ext_data.group(2))[:x-1])
```

commentary_extract.py | end_of_over.py | Final_PdfToCsv.py | Final_with_sentiment.py | score_vs_ball.py

```
                        None
                    elif("." in ext_data.group(2) and len(ext_data.group(2))<=5):
                        flag=1
                        #j = j-1
                        break
                    j = j+1
                    pattern = r"(MCID=\""+str(j)+r"\">)(.*?)(</P>)"
                    ext_data = re.search(pattern, small_data)
                    if ext_data is None:
                        break;
                #commentry.append(ext_data.group(2))
            if(ext_data is None):
                break
            if(len(ext_data.group(2))<=5 and ext_data.group(2)!="" and ext_data.group(2)!=" "):
                if(j==0 or j==1 and commOnprevPage==1):
                    commOnprevPage=0
                    commentry.append([test])
                #   print("44444444============",test)
                    test=""
                if("." not in ext_data.group(2)):
                    if(len(ext_data.group(2))==1 and "D" not in ext_data.group(2) and "O" not in ext_data.group(2)):
                        score.append(ext_data.group(2))
                    elif("lb" not in ext_data.group(2) and "w" not in ext_data.group(2) and "nb" not in ext_data.group(2)):
                        None
                    else:
                        score.append(ext_data.group(2))
                else:
                    x = len(ext_data.group(2))
                    overcount=1
                    over.append((ext_data.group(2))[:x-1])
            j=j+1

print(len(score))
print(len((over)))
print((len(commentry))
for i in range(len(commentry)):
    x = "".join(commentry[i])
    commentry[i] = x
df = pd.DataFrame(data={"Overs": over, "ScoreBoard": score,"Commentry": commentry})
df.to_csv("./Final_CSV.csv", sep=',',index=False)
```

commentary_extract.py | end_of_over.py | Final_PdfToCsv.py | Final_with_sentiment.py

```python
import json
import pandas as pd
from watson_developer_cloud import NaturalLanguageUnderstandingV1
import watson_developer_cloud.natural_language_understanding.features.v1 \
  as Features

sentiment = []
score = []
df = pd.read_csv('Final_CSV.csv')
over = df.Overs
ScoreBoard = df.ScoreBoard
Commentry = df.Commentry
Score = df.ScoreBoard
Overs = df.Overs

ScoreNew = []
CommentryNew = []
OversNew = []
for i in range(len(df)):
  if(Score[i] == 'W' or Score[i]=='6' or Score[i]=='4'):
    ScoreNew.append(Score[i])
    OversNew.append(Overs[i])
    CommentryNew.append(Commentry[i])
  if('nb' in Score[i]):
    if('w' in Score[i+1] or '4' in Score[i+1] or '6' in Score[i+1] or Score[i] >= '2'):
      ScoreNew.append(Score[i])
      OversNew.append(Overs[i])
      CommentryNew.append(Commentry[i])
  if(len(ScoreNew)>=50):
    break
if(len(ScoreNew)>=50):
    exit()
#print(len(OversNew) , len(ScoreNew) , len(CommentryNew))
natural_language_understanding = NaturalLanguageUnderstandingV1(
  username="**************",
  password="************",
  version="2017-02-27")

for i in Commentry:
  response = natural_language_understanding.analyze(
    text = i,
    features=[
```

commentary_extract.py | end_of_over.py | Final_PdfToCsv.py | Final_with_sentiment.py | score_vs_ball.py

```python
natural_language_understanding = NaturalLanguageUnderstandingV1(
  username="**************",
  password="************",
  version="2017-02-27")

for i in Commentry:
  response = natural_language_understanding.analyze(
    text = i,
    features=[
      Features.Sentiment(
        )
    ]
  )

  res = response['sentiment']['document']['score']
  if res <0:
    res = -1*res
  score.append(res)
sover = []
scomm = []
sscore = []
ssenti = []
df1 = pd.DataFrame(data = {'Over' : Overs,'Score' : Score, 'Commentry' : Commentry, 'Sentiment' : score})
df1 = df1.sort_values(['Sentiment'],ascending = [0])
df1.to_csv("./Senti_Score.csv", sep=',',index=False)
# print(len(Overs) , len(Score) , len(Commentry))
# print(len(df1))
# print(df1.head())
remain = 50 - len(OversNew)
count = 0
df1 = pd.read_csv('Senti_Score.csv')
# print(len(df1.Over))
# print(df1.Over[0] , df1.Score[0] , df1.Commentry[0])
for i in range(len(df1)):
  if count>=remain:
    break
  if df1.Commentry[i] not in CommentryNew:
    # print(df1.Over[i])
    sover.append(df1.Over[i])
    scomm.append(df1.Commentry[i])
    sscore.append(df1.Score[i])
    ssenti.append(df1.Sentiment[i])
```

commentary_extract.py | end_of_over.py | Final_PdfToCsv.py | Final_with_sentiment.py | score_vs_ball.p

```
    )

    res = response['sentiment']['document']['score']
    if res <0:
      res = -1*res
    score.append(res)
  sover = []
  scomm = []
  sscore = []
  ssenti = []
  df1 = pd.DataFrame(data = {'Over' : Overs,'Score' : Score, 'Commentry' : Commentry, 'Sentiment' : score})
  df1 = df1.sort_values(['Sentiment'],ascending = [0])
  df1.to_csv("./Senti_Score.csv", sep=',',index=False)
  # print(len(Overs) , len(Score) , len(Commentry))
  # print(len(df1))
  # print(df1.head())
  remain = 50 - len(OversNew)
  count = 0
  df1 = pd.read_csv('Senti_Score.csv')
  # print(len(df1.Over))
  # print(df1.Over[0] , df1.Score[0] , df1.Commentry[0])
  for i in range(len(df1)):
    if count>=remain:
      break
    if df1.Commentry[i] not in CommentryNew:
      # print(df1.Over[i])
      sover.append(df1.Over[i])
      scomm.append(df1.Commentry[i])
      sscore.append(df1.Score[i])
      ssenti.append(df1.Sentiment[i])
      count = count+1
  OversNew = OversNew + sover
  CommentryNew = CommentryNew +scomm
  ScoreNew = ScoreNew + sscore

  # print(len(OversNew) , len(ScoreNew) , len(CommentryNew))
  df = pd.DataFrame(data = {'Over' : OversNew,'Score' : ScoreNew , 'Commentry' : CommentryNew})
  df = df.sort_values(['Over'])
  df.to_csv("./Top50Events.csv", sep=',',index=False)

```

commentary_extract.py | end_of_over.py | Final_PdfToCsv.py | Final_with_sentiment.py | sco

```
import pandas as pd
df = pd.read_csv('Final_CSV.csv')

x = df['Overs'].values
y = df['ScoreBoard'].values

for i in range(len(y)):
    y[i]=y[i][0]
    if(y[i]=='W'):
        y[i]=-1
    else:
        y[i]=int(y[i])


import matplotlib.pyplot as plt
ind = np.arange(min(x)-0.1, max(x)+1,1)
ind = ind.reshape((len(ind), 1))
get_ipython().magic('matplotlib inline')
plt.figure(figsize=(20,10))
plt.plot(x,y,color='blue')
plt.xticks(ind)
plt.title('Score v/s Ball Analysis',fontsize = 40)
plt.xlabel('Balls in Overs 1-20',fontsize = 30)
plt.ylabel('Runs',fontsize = 30)
plt.show()

df1 = pd.read_csv('EOO_Summary.csv')
x = df1['Overs'].values
y = df1['Runs'].values

ind = np.arange(min(x), max(x)+1, 1)
ind = ind.reshape((len(ind), 1))

width = 0.5

fig, ax = plt.subplots(figsize=(20,10))
rects1 = ax.bar(ind, y, width, color='r')
plt.xticks(ind)
plt.title('Overs vs Runs Analysis',fontsize = 40)
plt.xlabel('Over',fontsize = 30)
plt.ylabel('Runs',fontsize = 30)

```

2). Python-2 Codes:

PdfMinerInstaller.py

```
1  ##########################STEPS TO INSTALL PDFMINER PACKAGE FOR PDF TO TXT CONVERTOR##############
2  # 1). First unzip the tar file pdfminer-20140328.tar
3  # 2). Then use "cd pdfminer-20140328" command to go inside the pdfminer-20140328 directory
4  # 3). Then run this given script it will install the PdfMiner Package on to your system
5
6  import os
7  os.system(python setup.py install)
```

PdfMinerInstaller.py | PdfToTxt.py

```
1  import os
2  os.system(pdf2txt.py -o test.txt -t tag MIvsKKR.pdf)
```

3). R Codes:

PdfMinerInstaller.py | PdfToTxt.py | Word_Cloud.r

```r
library("tm")
library("SnowballC")
library("wordcloud2")
library("RColorBrewer")
df = read.csv("Final_CSV.csv")
df1 = df['Commentry']
for (i in df1)
{
  i <- Corpus(VectorSource(i))
  i <- tm_map(i, removeWords, stopwords("english"))
  i <- tm_map(i, stemDocument)
  dtm <- TermDocumentMatrix(i)
  m <- as.matrix(dtm)
  v <- sort(rowSums(m),decreasing=TRUE)
  d <- data.frame(word = names(v),freq=v)
}
wordcloud2(data = d)
```

Aspects taken into account for selecting top events –

We have followed the data driven approach for this natural language processing task. We have used quantitative methods to discover relations. After converting the data to structured form, we have gone through methods like word frequency counting, ranking by means of the Term Frequency – Inverse Document Frequency metric, n-grams, and clustering. We effectively selected the events corresponding to balls which makes the game biased towards change like balls with sixes, fours, wickets, no-balls etc. After that we evaluated the sentiment score of commentary corresponding to each ball and selected the balls which gave us the highest score. We have chosen this strategy so that we may be able to analyze any special events that have occurred with a particular delivery.

We used IBM Watson Natural Language Understanding API for different tasks like categorization of content into five-level classification hierarchy. (However there is no need of such categorization but to finally provide the output to user (who will use the GUI) with full perfection.) , identifying high level concepts that are not directly referenced in the text, identified the entities mentioned in the content, this includes people, places, events and other types of entities and also done the emotional analysis of each entity. We searched the relevant keywords from the document so that particular analysis can be done on those keywords. Identification of semantic roles is done by parsing the text into subject-object and action form, and identified entities and keywords that are subjects and objects for an action.

We are looking forward to work on –

We have selected the top 50 events till now. Now we have the challenging task to make the best team. For this we will go through the corresponding commentary of top events and we will apply named entity recognition on that to extract the name of players from those events. After that we will focus on the keywords around that entity. (We will use regex parsing for that and will prepare keywords on our own in a separate file which will include “bat”, ”bowl”, “sixes”,” fours”, “field”, ”keeping” ,”stumping”,etc. because ultimately we need the finalized team which have best players from every segment. After that we will evaluate the parameters like average strike rate, bowling economy, catches taken, stumping made, run out, and for bowlers we will consider the parameters like no-ball, Yorker, short pitch, good-length, long pitch. This will give use features corresponding to each player. After that we will calculate the score by allocating weights to each parameter and players with best score will be selected. We will use Watson API for drawing insights from the data and applying Natural Language processing concepts.

For doing this task we will create the GUI using flask in python which have the option to select multiple pdf’s file, after that we will convert those pdf into text. Then different parsing methods will be applied that will produce three different csv files and covert unstructured data into structured form. One will contain the commentary data, another will contain the runs/wicket on each ball along with commentary corresponding to that ball, and one will contain the end of over summary. After that we will integrate the second csv file with IBM Watson natural language tool kit API, Watson discovery API to again select the top 50 events and follow the strategy as discussed above, the second and third csv file will be used to plot the visualization and directly analyzer can get aware with the insights of the data.

Google Drive Link Containing All the Data:

https://drive.google.com/open?id=0B3VdPmUhlac1TENtTDJwOWxlQVU

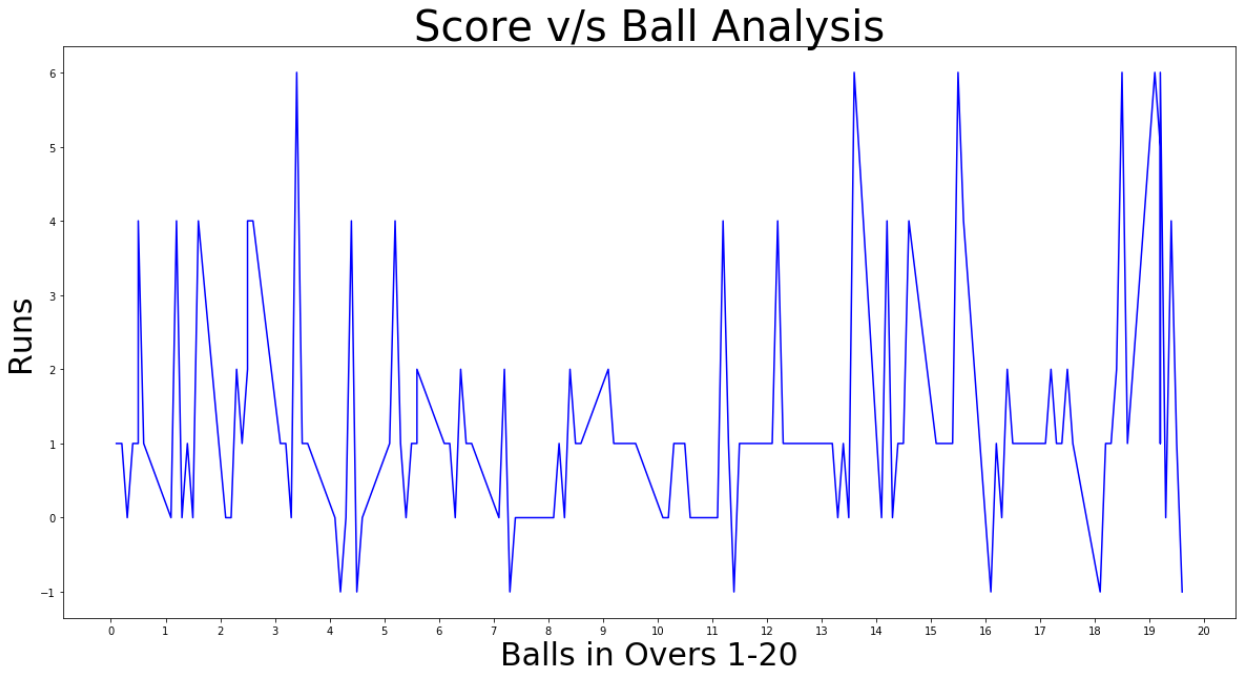
Conclusion:

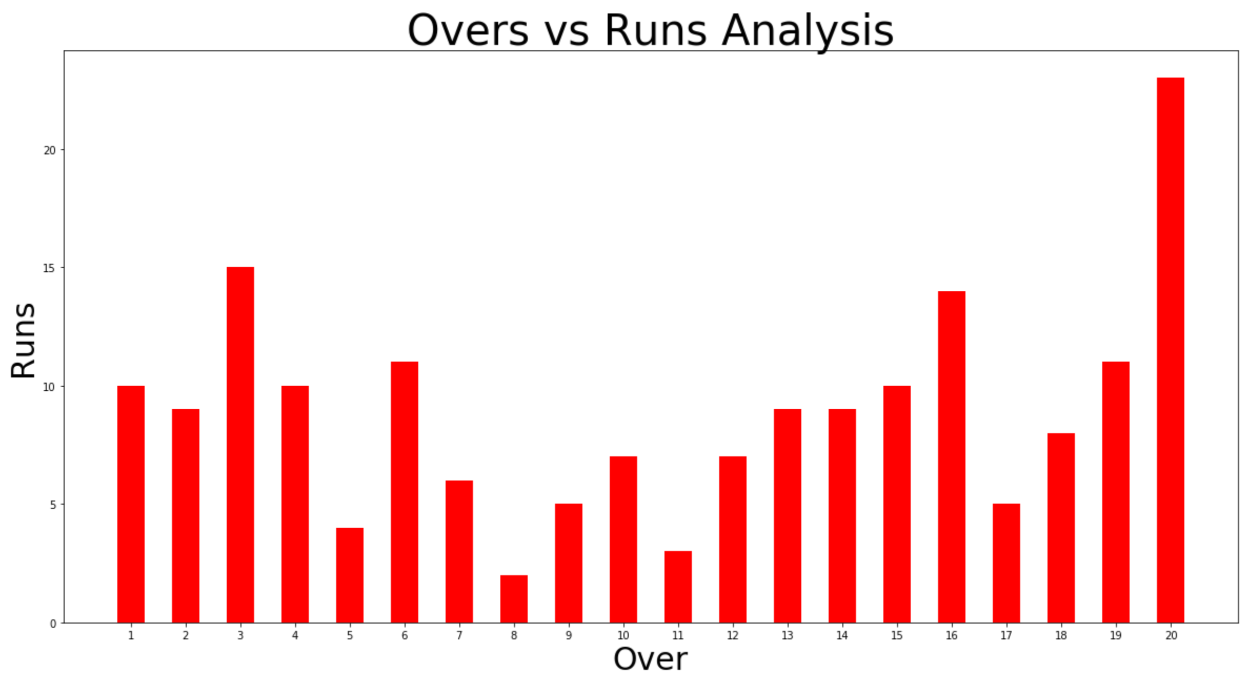
We concluded from the results that after doing this analysis we are now able to generate the top 50 major events in the match, and now using this data we are able to predict what is status of each player in the given match and with the help of these stats and more data which will be given to us in the upcoming stage we will able to predict who were the star players during the

season and this will also help us getting the best all-rounders, best openers, best bowlers, best keeper and best fielders during the season. So all this done we will choose the top 11 players which will form the best team and will surely win any of the match in the next season.

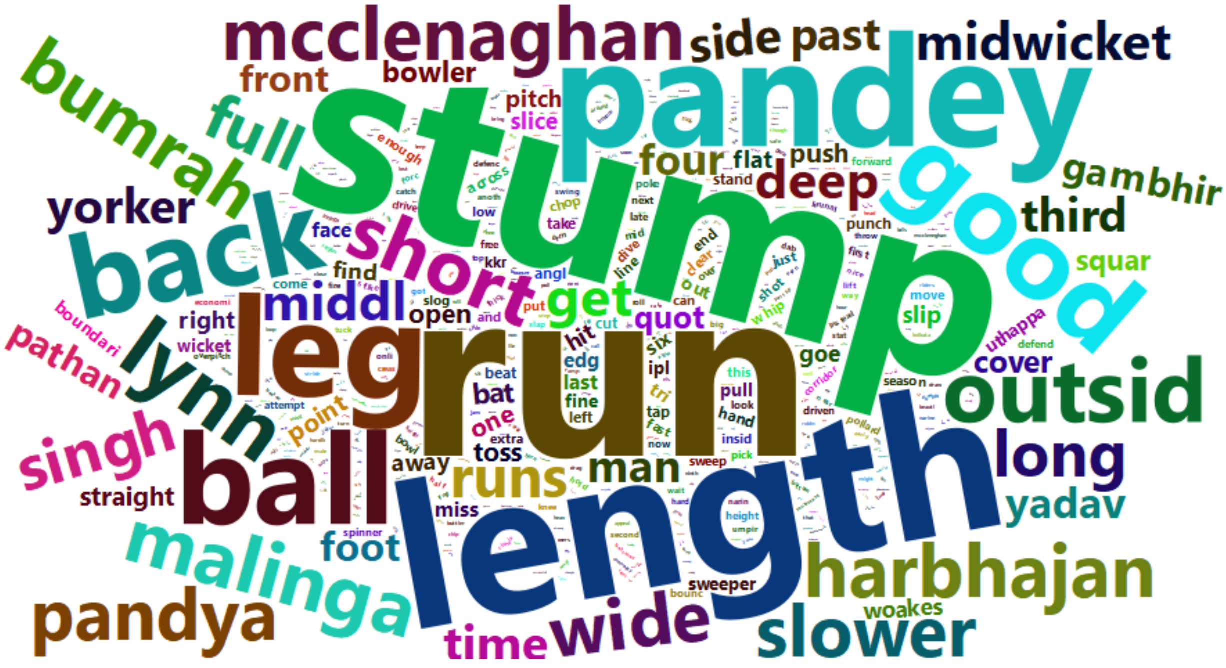
Alternate Approaches –

This is the alternative approach that we can use for the major event extraction. Unlike data driven approach which we have used to extract the 50 major events from the data, it is often on patterns that express rules representing expert knowledge. It is inherently based on linguistic and lexicographic knowledge, as well as existing human knowledge regarding the contents of the text that is to be processed. In this approach we could have divided the commentary by means of three keywords (score, ball, batsman) that together describe the major events. For example let's take the instance if a batsman score 6 runs in the last ball of the last over it is considered as a major event rather than the ball of any middle over. Also the batsman that is already playing good throughout the series, if that batsman gets out then it will be the major event. So to find these relations we need the expert knowledge and create the rules manually. This approach is known as knowledge driven approach which we could have use for the event extraction.




Overs vs Runs Analysis
Runs
Over
0
5
10
15
20
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20

Word Cloud


stump
run
length
pandey
mcclenaghan
leg
ball
back
good
short
bumrah
lynn
malinga
outsid
harbhajan
pandya
midwicket
side
past
front
bowler
full
yorker
middl
four
deep
third
gambhir
long
wide
slower
singh
pathan
foot
runs
man
yadav
time
squar
cover
slip
get
quot
straight
right
wicket
pitch
slice
flat
push
woakes
sweeper