Create an application in R using Shiny and do text based sentiment analysis and IPL team analysis

Name: Pranav Gurditta
Subject: Introduction to Data Science
Roll No. : 2K15/CS/2002
Course: B.Sc. (Hons.) Computer Science
Submitted to:
Mrs. Geetika Vashisht

Index

1.Problem statement
2.Introduction
3.Packages used
4.Why not the base package?
5.Comparison/Analysis of why a particular plot type has been used
6.Concept
7.Implementation
8.Conclusion
9.References

Problem statement

The problem is to make an application in R which does sentiment analysis using tweets from R following a menu based approach to let the user choose to do text based sentiment analysis or check popularity of IPL teams or do a comparison of chances of winning of two IPL teams.

Introduction

The application is used in R to do the following tasks:

1. Text based sentiment analysis: To check whether the word entered has been used in a positive or negative sense in a tweet.
2. IPL team tweet analysis: To know whether the tweets of the IPL team are positive or negative and hence utilise this data to show advertisements to the user based on the situation of the team that he is following on twitter.
3. IPL team comparison: To compare the chances of winning of two teams and also as to know which team is better for investing money for advertisements and brand promotions as brands generally give advertisements to the winning team.

Packages used

The packages used are:

1. **shiny**: Shiny is an R package that makes it easy to build interactive web apps straight from R. You can host standalone apps on a webpage or embed them in R Markdown documents or build dashboards.
2. **shinydashboard**: This project uses shinydashboard to build dashboards in an application in R.
3. **stringr**: stringr is a set of simple wrappers that make R's string functions more consistent, simpler and easier to use. It does this by ensuring that: function and argument names (and positions) are consistent, all functions deal with NA's and zero length character appropriately, and the output data structures from each function matches the input data structures of other functions.
4. **twitteR**: twitteR is an R package which provides access to the Twitter API. Most functionality of the API is supported, with a bias towards API calls that are more useful in data analysis as opposed to daily interaction.
5. **plyr**: plyr is an R package that makes it simple to split data apart, do stuff to it, and mash it back together. This is a common data-manipulation step.
   Importantly, plyr makes it easy to control the input and output data format with a consistent syntax.
6. **plotrix**: It is used in this project to create a pie chart in 3D.

Why not the base package?

The base package is not used for this as shiny and shinydashboard package help us to make an application using R whereas without importing any package it would not be possible. The plyr package makes it easy to split data apart. The twitter package provides access to the Twitter API. The stringr package makes strings easier to use. The plotrix package is used to create a pie chart in 3D whereas the normal pie() function gives a 2D pie chart in this project.

Comparison/Analysis of why a particular plot type has been used

Pie chart:

Pie chart is used in comparison as winning percentages need to be mentioned and not continuous values are not required as are there in a histogram.

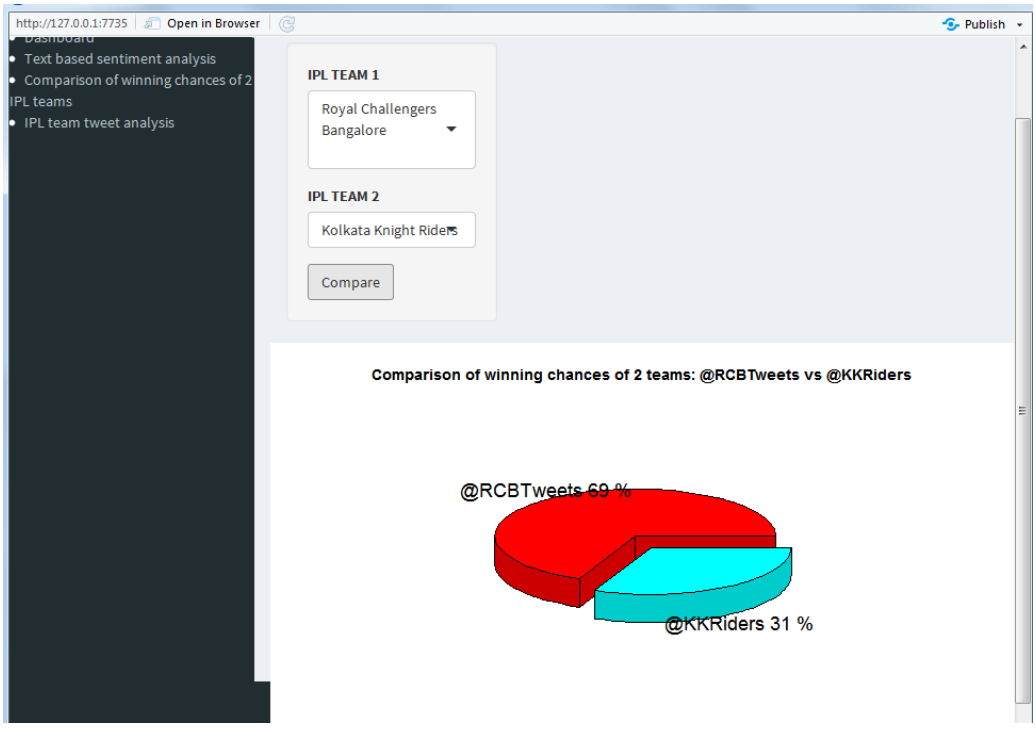

Figure 1: Displaying how pie chart is used in the project

Histogram

It is used to calculate the score to display of whether the text is entered in positive or negative sense in a tweet. We want to know the level of positivity or negativity in the tweet that

contains the particular word entered in text based sentiment analysis hence we use histogram in it.

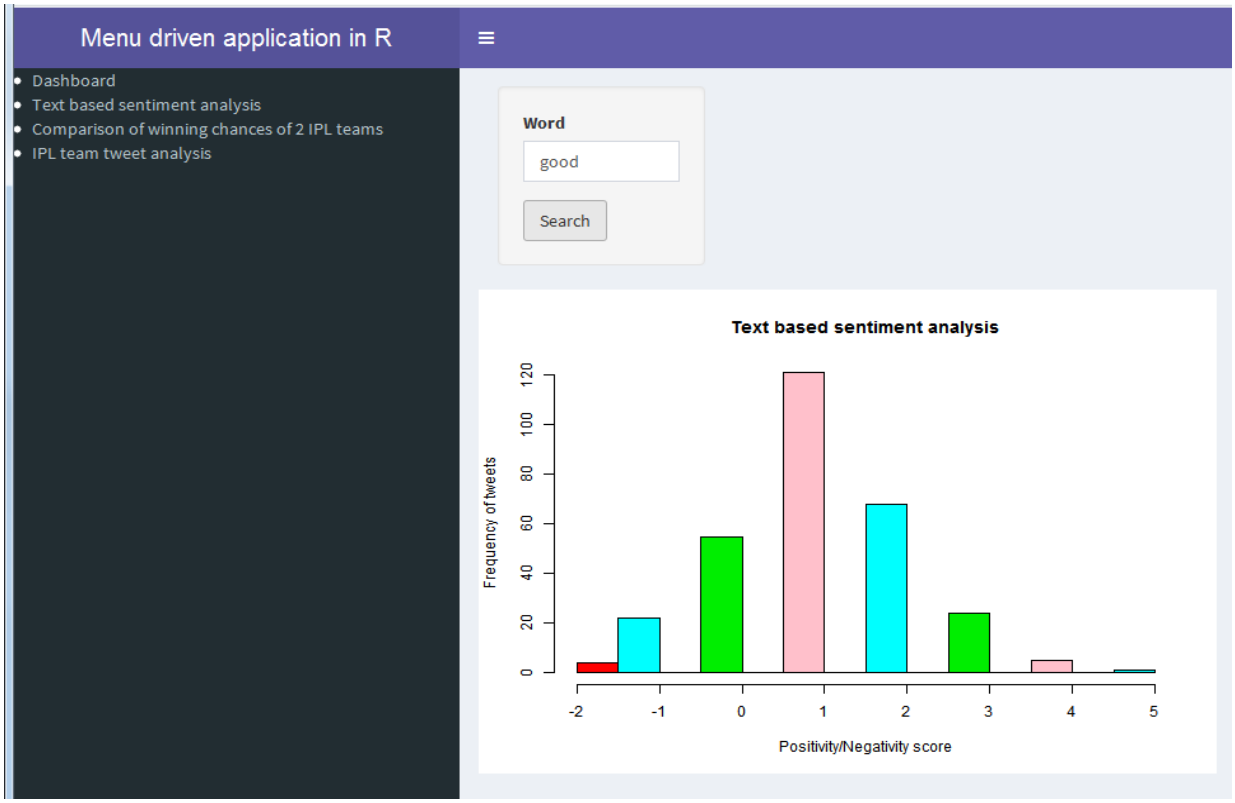

Figure 2: Displaying how histogram is used in the project

Concept of the project

1. **Make a shiny App , first let us separate the App code into two files:**

   ui.R : It contains only the user interface

   server.R: It acts like the server and uses input values sent from ui.R to it to process the output.

2. **Next, shinydashboard is used for ui.R to make a menu driven program.**

   A dashboard has three parts: a header, a sidebar, and a body. Here's the most minimal possible UI for a dashboard page.

```
library(shinydashboard)

dashboardPage(
```

dashboardHeader(),

dashboardSidebar(),

dashboardBody()

)

Basic dashboard

Next, we can add content to the sidebar. For this example we'll add menu items that behave like tabs. These function when you click on one menu item, it shows a different set of content in the main body.

There are two parts that need to be done. First, you need to add menuItems to the sidebar, with appropriate tabNames.

```
18
19    # Sidebar with a slider input for number of bins
20    dashboardSidebar(
21      width=350,
22    menuItem("Dashboard",tabName="Dashboard"),
23    menuItem("Text based sentiment analysis",tabName="t1"),
24    menuItem("Comparison of winning chances of 2 IPL teams",tabName="t2"),
25    menuItem("IPL team tweet analysis",tabName="t3")
26    ),
```

3. Using TwitteR package to get data from Twitter API in server.R

First get a twitter account if you don't have one by registering on twitter.com website. Next go to (https://dev.twitter.com) with the same credentials. Here we will be setting up the API. This API is required for authentication to search tweets from a third party application. It uses an industry standard process called OAuth. OAuth creates the handshake between twitter and R using something called as "Consumer Key" and "Consumer Secret".

```
29
30  consumer_key <- "CzUJcMqojfMOEczNBmN47QI7x"
31  consumer_secret <- "IbN82fBTOrcmxgS42yepcvez3as7G4fLZsHVCd4Mutv8bs5z8j"
32  access_token <- "4845037632-M9vy2bZTQuI1RInEECFu4572WZTAGK3z5D95h7J"
33  access_secret <- "l9Vhang7KluC1ddPFgKDNrwBgFeSBwWWINpejcaDgySEa"
34
35  setup_twitter_oauth(consumer_key, consumer_secret, access_token, access_secret)
36
37
```

4. Obtaining data from twitter API

userTimeline(user,n) function is used to get/retrieve tweets from various timelines within the Twitter universe. The twListToDF() is used to convert a list of tweets to a dataframe.

```
47      tweet1 <- userTimeline(x,n=3000)
48      tweet2 <- userTimeline(y,n=3000)
49      tweet1_df <- twListToDF(tweet1)
50      tweet2_df <- twListToDF(tweet2)
```

5. Getting input of menu choice from user.

The input entered in ui.R can be traced in server.R by using "input$variable_name" and then using observeEvent() function to do the task chosen by the user. The

observeEvent() function responds to "event-like" reactive inputs, values, and expressions.

```
observeEvent( input$search2,{
  x11 <- input$word

  tweet32 <- searchTwitter(x11,n=300)
  tweet32 <- twListToDF(tweet32)

  #save all the spaces, then get rid of the weird characters that break the API, then convert back the URL-encoded spaces.
  tweet32$text <- str_replace_all(tweet32$text, "%20", " ");
  tweet32$text <- str_replace_all(tweet32$text, "%\\d\\d", "");

    pos.words2 = scan('C:/Users/yati/Desktop/Sentiment Analysis/positive-words.txt',what='character',comment.char=';')
  neg.words2 = scan('C:/Users/yati/Desktop/Sentiment Analysis/negative-words.txt',what='character',comment.char=';')
  teamscore2 <- score.sentiment(tweet32$text,pos.words2,neg.words2,.progress='text')
  output$sum4<-renderPrint(paste("The use of this word in negative or positive is ",x11," are :",sum(teamscore2$score)))

  output$distPlot5<-renderPlot({hist(teamscore2$score,main="Text based sentiment analysis",xlab="Positivity/Negativity score",
                                     ylab="Frequency of tweets",col=c("red","cyan","blue","green2","brown","pink"))  })


})
```

6. **Cleaning data obtained from twitter in the sentiment score calculation function described next.**

The gsub(pattern, replacement, x) function searches for the pattern in x and then replaces with replacement variable in x.

The tolower(sentence) converts each and every character in sentence to lowercase.

The str_split(sentence,separator) is a function in stringr package to split a string into pieces.

The unlist(x) function when given a list structure x, unlist simplifies it to produce a vector which contains all the atomic components which occur in x.

```
scores <- laply(sentences, function(sentence, pos.words, neg.words){
  sentence <- gsub('[[:punct:]]', "", sentence)
  sentence <- gsub('[[:cntrl:]]', "", sentence)
  sentence <- gsub('\\d+', "", sentence)

  sentence <- tolower(sentence)
  word.list <- str_split(sentence, '\\s+')
  words <- unlist(word.list)
```

7. **Matching positive and negative words after fetching them from files against tweets retrieved and then giving the sentiment score.**

match() function returns a vector of the positions of (first) matches of its first argument in its second.

isna() function is used to check whether the value in the vector is not available.

data.frame() function is used to make a dataframe

```r
score.sentiment<-function(sentences, pos.words, neg.words, .progress='none')
{
  scores <- laply(sentences, function(sentence, pos.words, neg.words){
    sentence <- gsub('[[:punct:]]', "", sentence)
    sentence <- gsub('[[:cntrl:]]', "", sentence)
    sentence <- gsub('\\d+', "", sentence)

    sentence <- tolower(sentence)
    word.list <- str_split(sentence, '\\s+')
    words <- unlist(word.list)
    pos.matches <- match(words, pos.words)
    neg.matches <- match(words, neg.words)
    pos.matches <- !is.na(pos.matches)
    neg.matches <- !is.na(neg.matches)
    score <- sum(pos.matches) - sum(neg.matches)
    return(score)
  }, pos.words, neg.words, .progress=.progress)
  scores.df <- data.frame(score=scores, text=sentences)
  return(scores.df)
}
```

8. Using sentiment score to do analysis and plot a visualization to better understand the data.

The hist() function is used to make a histogram whereas pie() function is used to make a pie chart and pie3D() function of plotrix package is used to plot a pie chart in 3D.

The variable name “output$variable_name” is used to refer to variables which are mentioned as output variables in ui.R.

```r
    pos.words2 = scan('C:/Users/yati/Desktop/Sentiment Analysis/positive-words.txt',what='character',comment.char=';')
  neg.words2 = scan('C:/Users/yati/Desktop/Sentiment Analysis/negative-words.txt',what='character',comment.char=';')
  teamscore2 <- score.sentiment(tweet32$text,pos.words2,neg.words2,.progress='text')
  output$sum4<-renderPrint(paste("The use of this word in negative or positive is ",x11," are :",sum(teamscore2$score)))

  output$distPlot5<-renderPlot({hist(teamscore2$score,main="Text based sentiment analysis",xlab="Positivity/Negativity score",
                                     ylab="Frequency of tweets",col=c("red","cyan","blue","green2","brown","pink"))  })


})
```

Implementation of the code

- ## Text based sentiment analysis

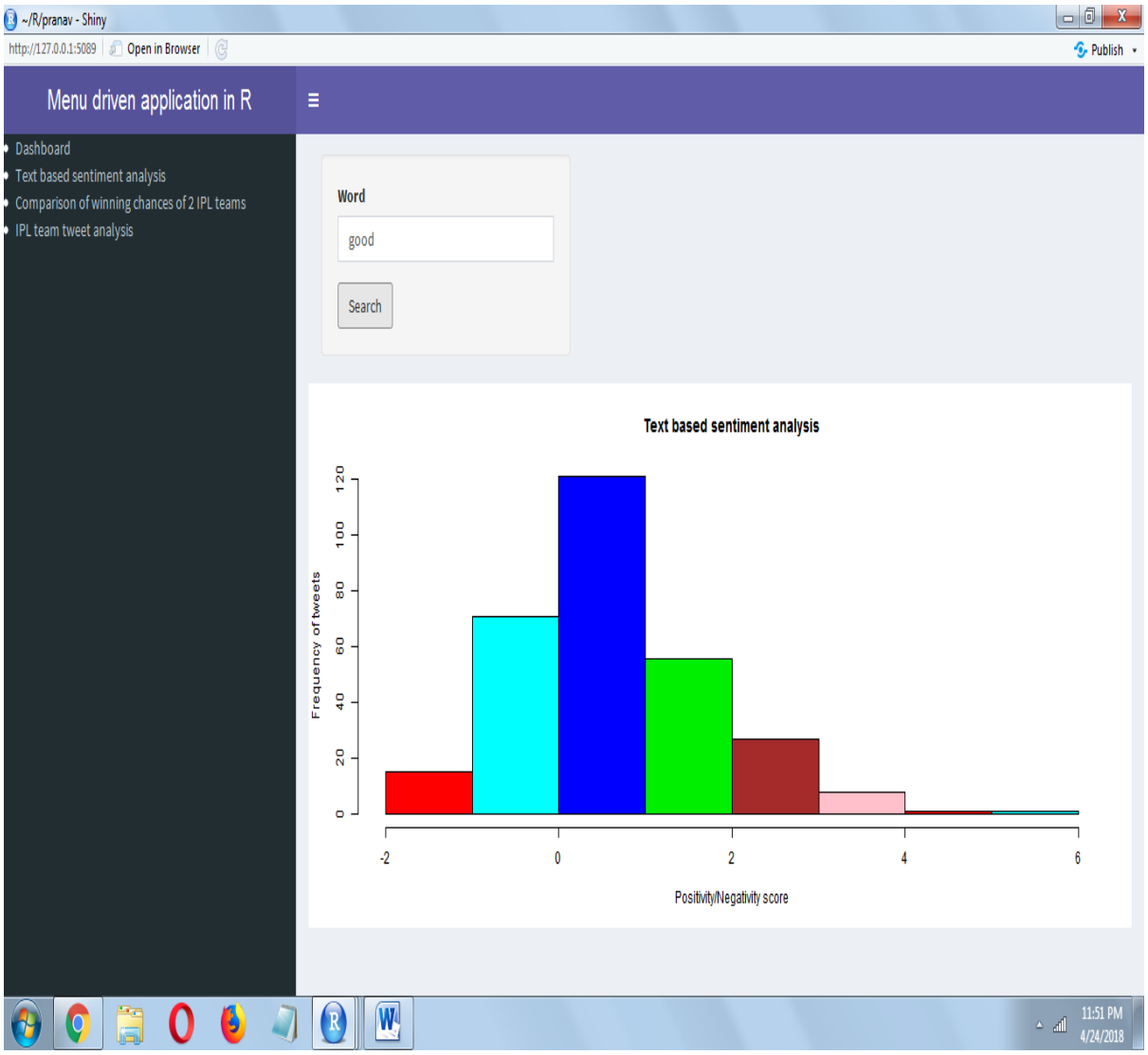

Figure 3: Displaying text based sentiment analysis using histogram

The word entered for example “good” is used as an argument in searchTwitter() function to get recent tweets from Twitter API having the word “good” in it. The tweets are then analysed to know whether the tweet is positive or negative and the positivity or negativity score denoted by positive or negative values on x axis of histogram respectively. Basically it tells us whether the word is used in a negative or positive sense.

Applications of text based sentiment analysis:.

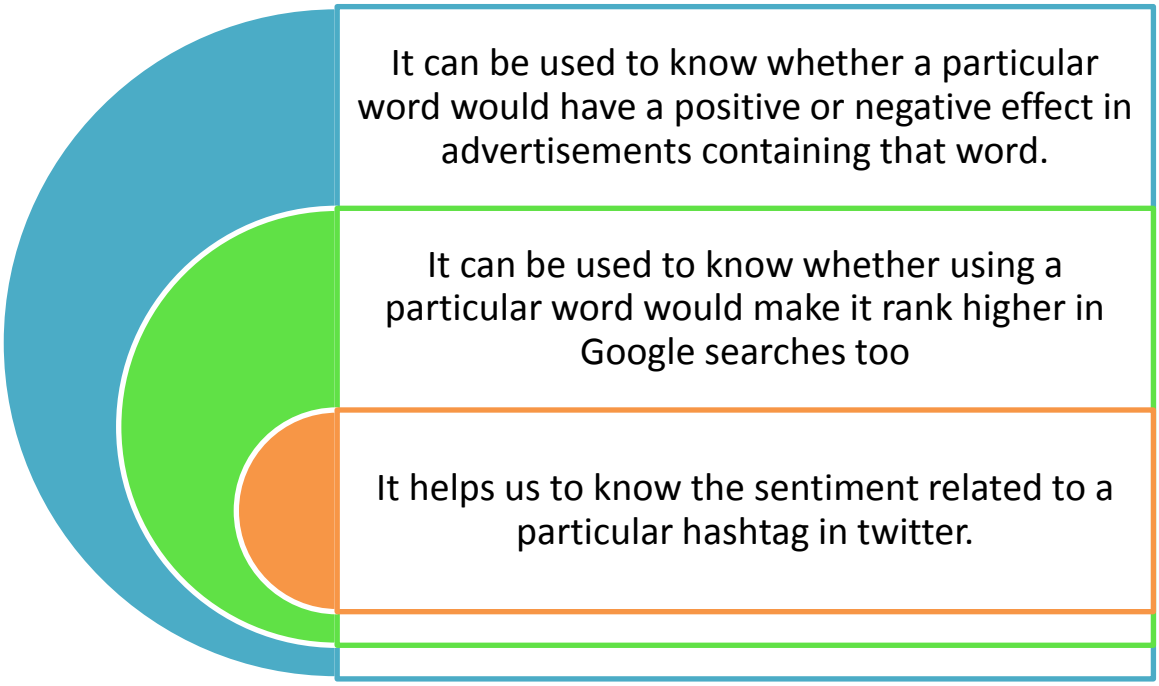

Figure 4: Applications of text based sentiment analysis done in the project

- ## Comparison of winning chances of 2 IPL teams

The IPL teams upload each and every detail of how the match they are playing is going on hence analysing those tweets and counting the positive and negative tweets which we assume to represent good situation of the team and bad situation of the team respectively in a particular match. We use the data to analyse and predict which team will win based on past performances.

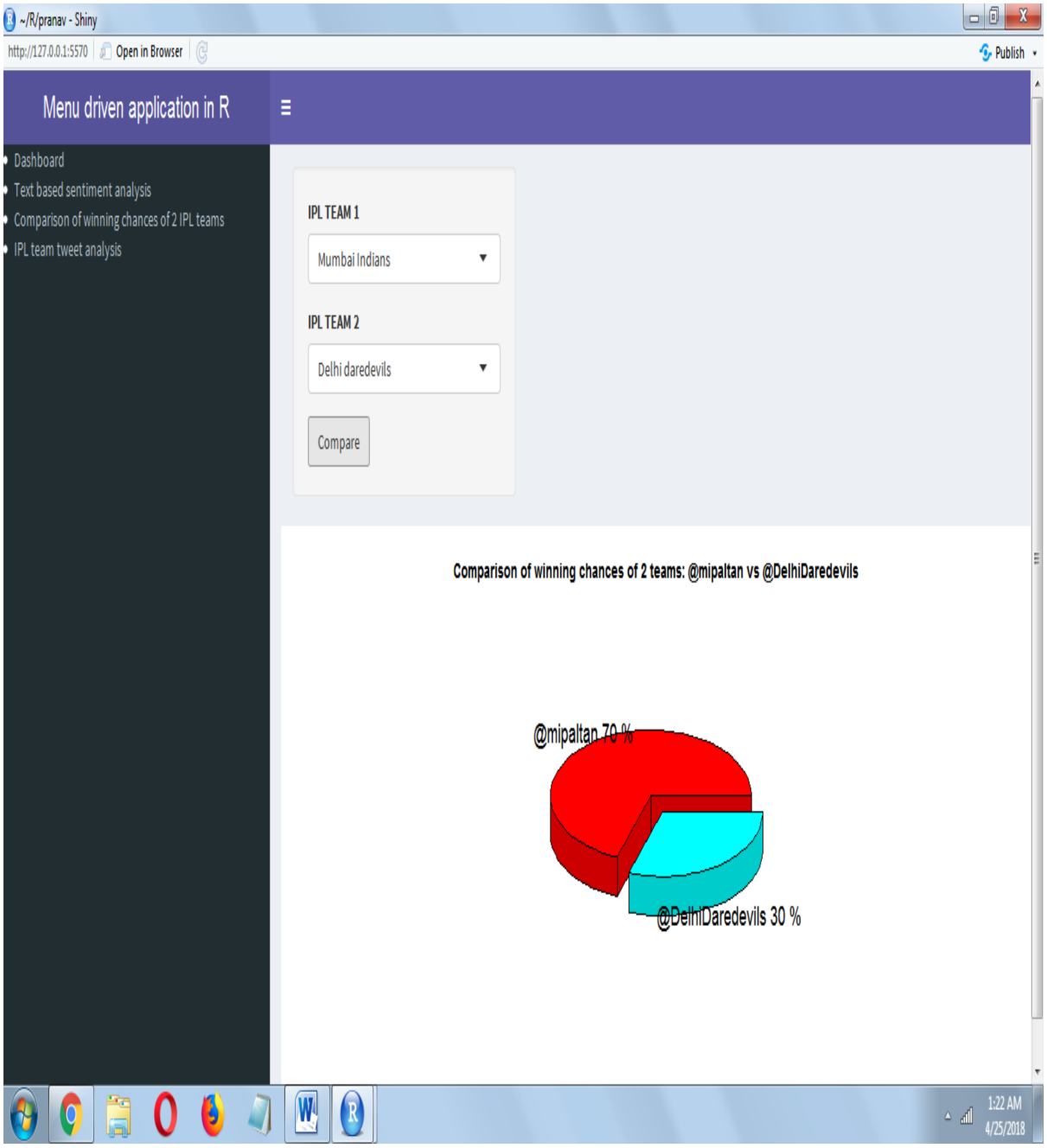

Figure 5: Displaying comparison of chances of winning of two teams

Applications of comparison of chances of winning of two IPL teams:

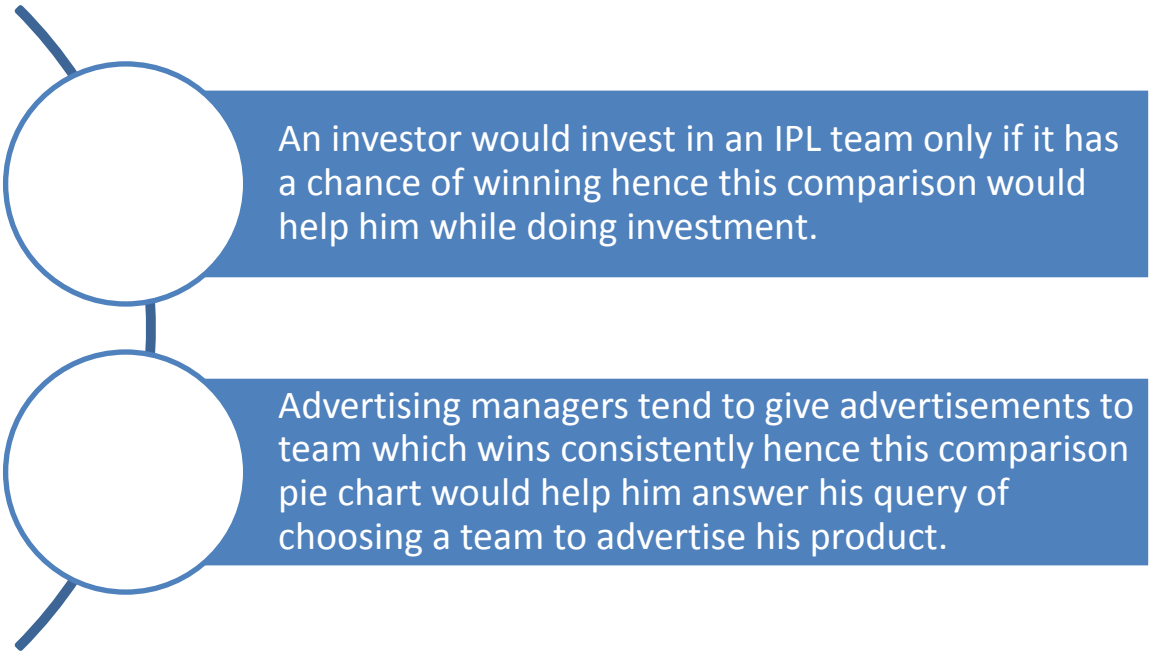

Figure 6: Applications of comparison of winning chances of two teams done in this project

- # IPL team tweet analysis

This analysis checks which team has a positive mind set and posts more positive or negative tweets as it is believed that positivity in a team could help them do wonders in the match. The analysis basically helps advertisers know the current mind set of a person who is following that team and send him advertisements based on team's situation.

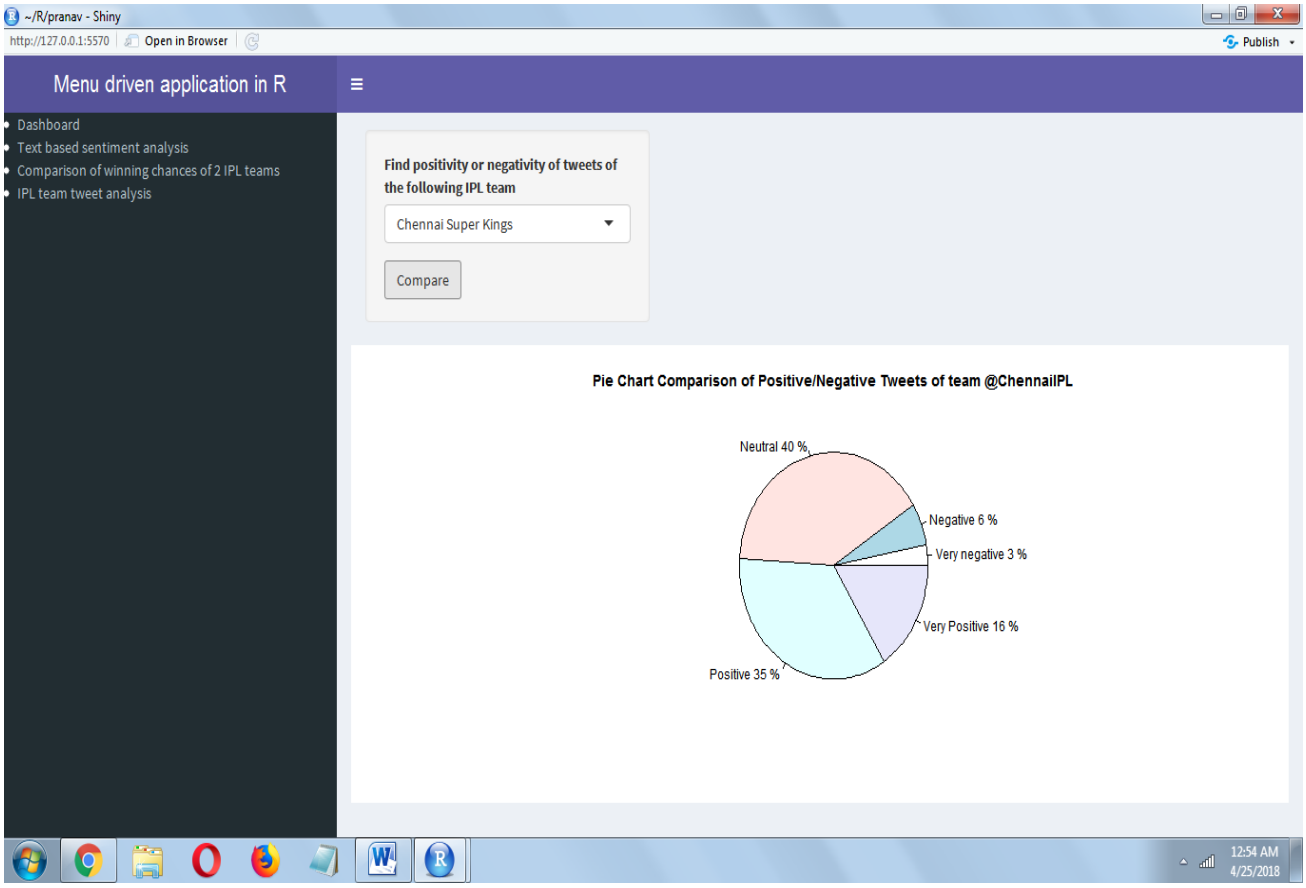

Figure 7: Displaying IPL team tweet analysis

Applications of IPL team tweet analysis:

- The emotions of the people are based on the team they follow on twitter and how they are performing and this IPL team tweet analysis helps the advertiser to send advertisements based on the kind of tweets their team is doing. For example if the number of negative tweets are more than positive tweets and a person follows that team then stress relieving gels could be marketed to such a person.

Conclusion

- The text based sentiment analysis help us to analyse in general which word is used in a positive or negative sense in a tweet.
- It helps us to conclude which team is better among the two teams the user enters and helps us to know the winning chances of both the teams.
- The IPL team tweet analysis helps us to conclude about the behaviour/mood of the person who is following that particular team.

References

- https://www.rdocumentation.org/
- https://www.edureka.co/
- https://youtube.com/
- https://developer.twitter.com/
- Minqing Hu and Bing Liu. "Mining and Summarizing Customer Reviews."