# R Individual Assignment

*Paine (HE Peiwei, 54471952)*

## 1 Introduction and research question

### 1.1 Introduction

This project is intended to conduct social media data analysis of AC Milan Official Twitter, consisting with data collecting, descriptive analysis, dictionary analysis, sentiment analysis and topic model building.

### 1.2 Research Question

How does the Twitter coverage of AC Milan mainly about?

## 2 Data Collecting

Load required packages.

```
library(twitteR)
library(quanteda)
library(corpustools)
library(RColorBrewer)
library(ggplot2)
library(igraph)
library(RTextTools)
library(slam)
library(reshape2)
```

Set up Twitter API, using direct authentication.

```
## [1] "Using direct authentication"
```

### 2.1 Gather Tweets by harshtags

Catch English tweets about AC Milan, n=5000, and catch Juventus tweets for further comparision.

```
acm_tweets <- plyr::ldply(searchTwitter("#acmilan", n=5000, lang="en"), as.data.frame)
acm_tweets=subset(acm_tweets,language=="en")

juv_tweets <- plyr::ldply(searchTwitter("#juventus", n=5000, lang="en"), as.data.frame)
juv_tweets=subset(juv_tweets,language=="en")
```

Use the ACM and JUV tweets, corpus the tweets text and convert into dfm. Remove stopwords to clean the dfm.

```
load("acmtweets.rda")
load("juvtweets.rda")
stopwords = c(stopwords("english"), 'a','"', ',','&','the',"?","-","[","]","(",")","coa","apps","https"
acm_corpus = corpus(acm_tweets$text)
acm_dfm = dfm(acm_corpus,removePunct=T,removeNumbers=T, removeTwitter=T)
```

```
acm_dfm = dfm_select(acm_dfm, stopwords,selection=c("remove"),valuetype=c("fixed"))
acm_dfm = dfm_select(acm_dfm,stopwords("english"),"remove")
juv_corpus = corpus(juv_tweets$text)
juv_dfm = dfm(juv_corpus,removePunct=T,removeNumbers=T, removeTwitter=T)
juv_dfm = dfm_select(juv_dfm, stopwords,selection=c("remove"),valuetype=c("fixed"))
juv_dfm = dfm_select(juv_dfm,stopwords("english"),"remove")
```

### 2.2 Get the friends' connections

Get the basic information and friends connection of AC Milan official Twitter. Convert the data into dataframe.

```
u = twitteR::getUser("acmilan")
followings = u$getFriends()
followings.df = plyr::ldply(u$getFriends(), as.data.frame)
```

Convert the followings list into dataframe.

```
friends = u$getFriendIDs()
followings.df = subset(followings.df, id %in% friends)
followings.df = plyr::arrange(followings.df, -followersCount)
ids = head(followings.df$id, 10)
```

Create a function to get a user's friends.

```
followingslist = function(u) {
  message(u$screenName)
  f = u$getFriends()
  f = plyr::ldply(f, as.data.frame)[c("screenName")]
  data.frame(leader=u$screenName, following=f)
}
```

And apply the userlist to the function, and get 2 layers of connections based on AC Milan Twitter. There are totally over 5000 connections.

```
userlist = c(u, followings[ids])
userlist
connections = plyr::ldply(userlist, followingslist, .id=NULL)
```

```
connections=readRDS("acmilanconnections.rds")
head(connections)
```

```
##    leader     screenName
## 1 acmilan     Locampos15
## 2 acmilan  gerardeulofeu
## 3 acmilan MarcoVanBasten
## 4 acmilan         Dugout
## 5 acmilan   mattia_desci
## 6 acmilan     RenzoRosso
```

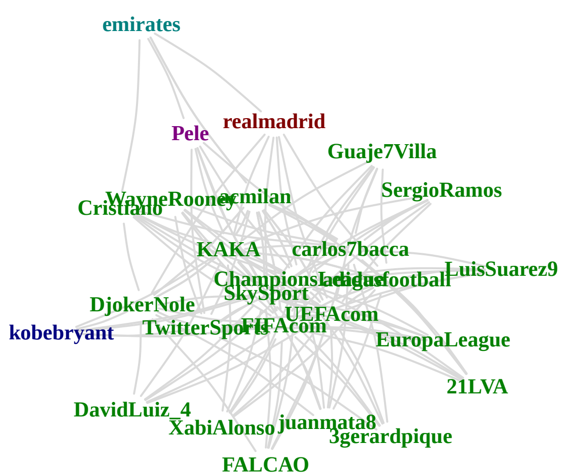
# 3 Data analysis

## 3.1 Descriptive Analysis

### 3.1.1 Social Friends Network

To draw a graph of the connections, keep only the people with in-degree over 5, and adjust the label size based on betweenness centrality, color the labels based on clustering, create a circular layout with the Reingold Tilford layout.

```
g = igraph::graph_from_data_frame(connections, directed=F)
g2 = igraph::decompose(g, min.vertices = 50)[[1]]
g2 =induced_subgraph(g2, degree(g2, V(g2), "in")>5)
centrality = betweenness(g2)
V(g2)$label.cex = 0.7 + 0.5 * centrality / max(centrality)
clusters = edge.betweenness.community(g2)$membership
pal = substr(rainbow(length(unique(clusters)), start=0.33, end=1, v=0.5), 1, 7)
layout <- layout.reingold.tilford(g2, circular=F)
plot(g2, vertex.shape="none",
     vertex.label.font=2, vertex.label.color = pal[match(clusters, unique(clusters))],
     vertex.label.cex=.7, edge.curved=.1,edge.color="gray85")
```




### 3.1.2 Top Features Compare the top 10 features of AC Milan and Juventus. As the results shown, ACM tweets are more related to italian football, while JUV tweets are more related to other european football clubs, especially those with transfer news.

```r
topfeatures(acm_dfm, 10)
```

```
##        acmilan footballitalia          milan     forzamilan         dugout
##           4303            821            800            621            594
##         follow            win           just          shirt         signed
##            591            573            552            549            545
```

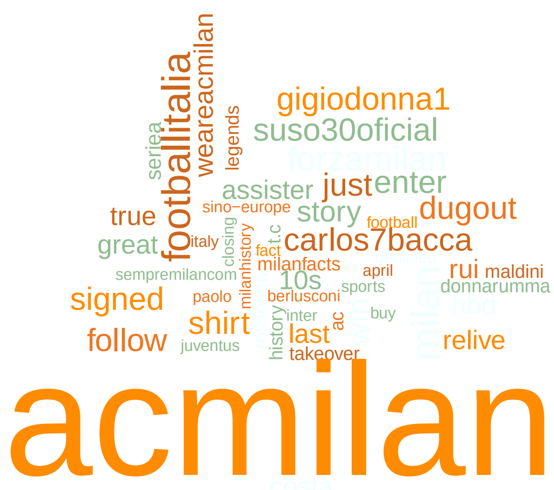
```r
topfeatures(juv_dfm, 10)
```

```
##       juventus footballitalia            cfc            psg            afc
##           3251            995            514            509            451
##  chelstransfer           team           want         alexis        sanchez
##            394            390            377            355            283
```

### 3.1.3 Wordcloud Plot

Plot a naive wordcloud of AC Milan to have a brief view.

```r
textplot_wordcloud(acm_dfm, max.words = 50, random.color = TRUE, rot.per = .25, colors =sample(colors()
```




Comparing word use in different texts. Specifically look at words are overrepresented in ACM tweets compared to JUV tweets.

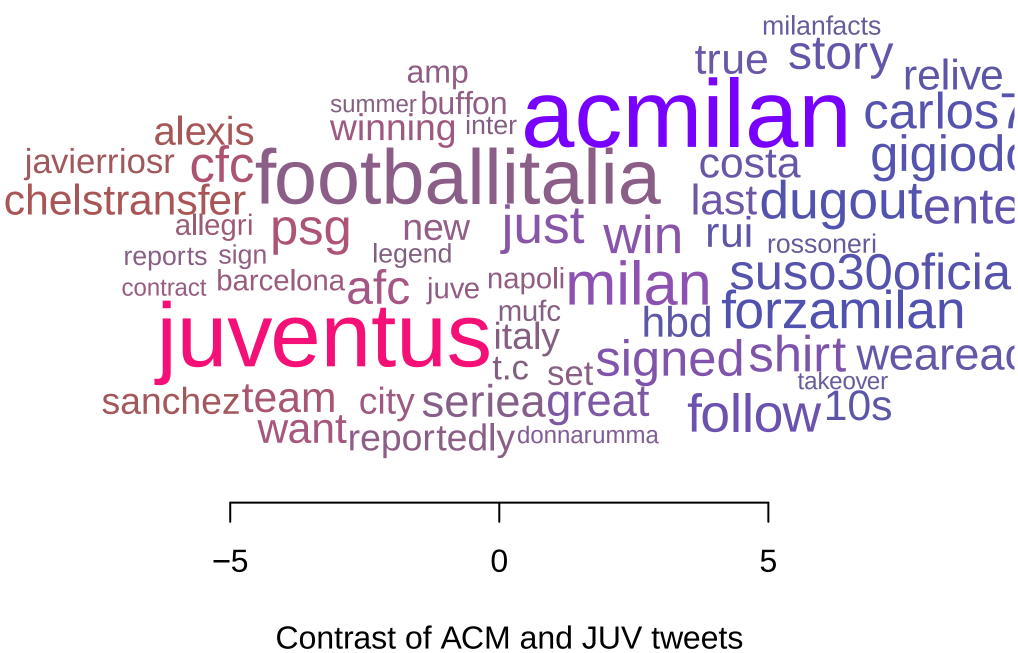
```r
cmp = corpustools::corpora.compare(acm_dfm, juv_dfm)
cmp = cmp[order(cmp$over, decreasing = F), ]
h = rescale(log(cmp$over), c(1, .6666))
s = rescale(sqrt(cmp$chi), c(.25,1))
```

```
cmp$col = hsv(h, s, .33 + .67*s)
head(cmp,1)
```

```
##                term termfreq.x termfreq.y termfreq    relfreq.x   relfreq.y
## 3220 chelstransfer          0        394      394 2.432853e-05 0.009992916
##              over      chi     col
## 3220 0.002434578 422.6829 #A95555
```

Then, plot the contrast wordcloud of ACM and JUV in one plot.

```
cmp = arrange(cmp, -termfreq)
with(head(cmp, 60), plotWords(x=log(over), words=term, wordfreq=termfreq, random.y = T, col=col, scale=3
title(xlab="Contrast of ACM and JUV tweets")
```




### 3.2 Dictionary Analysis

Create a dictionary of 4 concepts, including "FC.Inter", "Napoli", "AS Roma" and "European". Set some keywords for these 4 concepts.

```
d = dictionary(list(Inter=c("internazionale", "inter*","icardi","pioli","FCIM"),Napoli=c("napoli", "napo
acmd = as.data.frame(dfm_lookup(acm_dfm, d))
juvd = as.data.frame(dfm_lookup(juv_dfm, d))
```

Apply the dictionary to the ACM and JUV dataframe, and calculate the frequency.

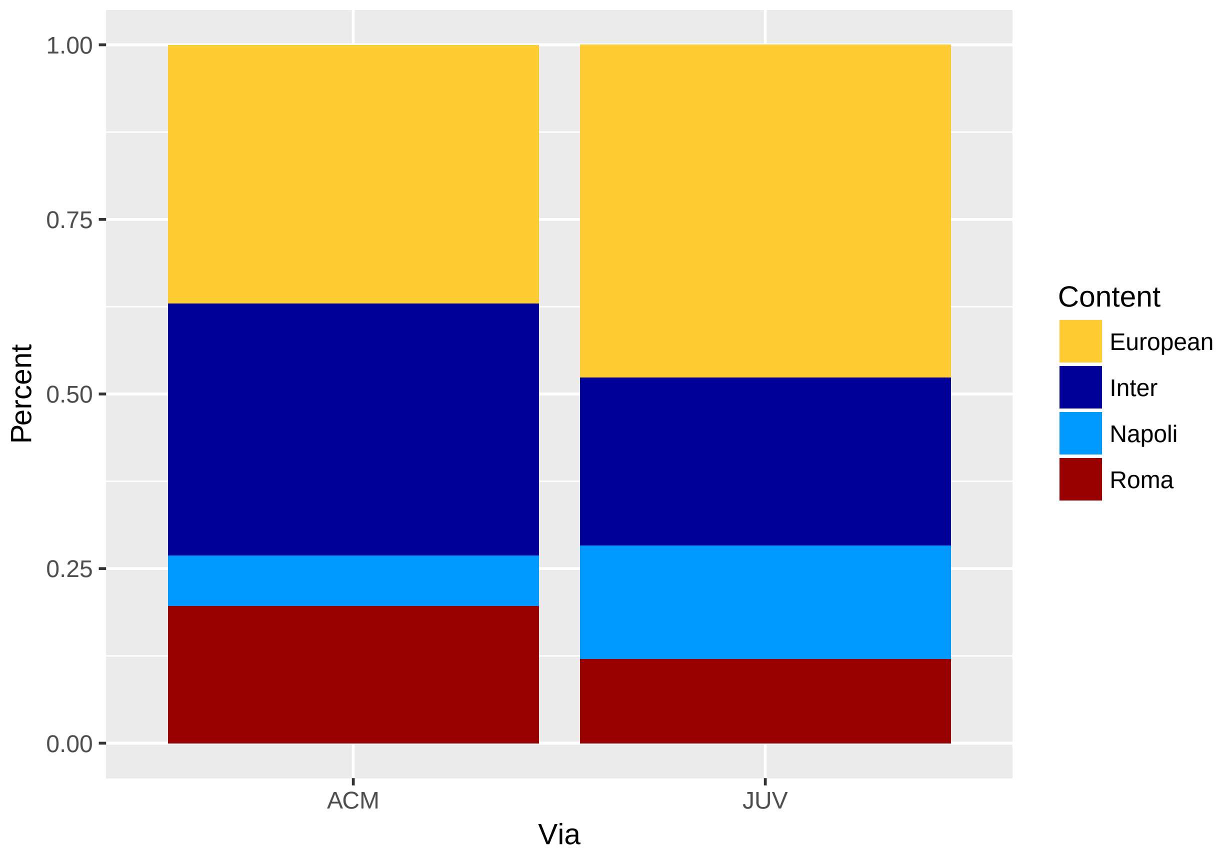
```
x1=data.frame(lapply(data.frame(acmd), sum))
y1=data.frame(lapply(data.frame(juvd), sum))
```

```r
all= rbind(x1, y1)
rownames(all) <- c("via ACM","via JUV")
all
```

```
##         Inter Napoli Roma European
## via ACM   210     42  114      215
## via JUV   291    197  146      577
```

Plot the bar chart with ggplot, to compare the tweets contents of ACM and JUV. As the plot shown, ACM is mostly interested in FC Inter, which has been its greatest competitor in Serie A from past till nowadays. However, JUV is more related to other european football issues than ACM.

```r
ggplot(counts, aes(fill=Content, y=Percent, x=Via)) + geom_bar(stat="identity",position="fill")+scale_fi
```




## 3.3 Topic Model

### 3.3.1 Bulid Topic Model

Build a topic model of ACM tweets by using LDA, automatically assigns topics to text documents.

```r
d = convert(acm_dfm, to="topicmodels")
m = LDA(d, k = 5, method = "Gibbs", control=list(alpha=.1, iter=100))
ignore = c(stopwords('english'),'acmilan','milan','footballitalia', 'forzamilan','peopl*','americ*')
d = dfm(acm_dfm, remove=ignore, stem=T)
d = convert(d, to="topicmodels")
set.seed(123)
```

```
m = LDA(d, k = 5, method = "Gibbs", control=list(alpha=.1, iter=100))

topic=data.frame(terms(m, 10))
colnames(topic) = c("Berlusconi", "Rui Costa", "Rising Star","News","Transfer")
topic
```

```
##    Berlusconi Rui Costa Rising Star         News     Transfer
## 1      legend       rui     maldini weareacmilan         sign
## 2         t.c     costa      legend    milanfact          win
## 3  berlusconi     great       paolo         fund       dugout
## 4      footbal     stori    deulofeu       seriea        shirt
## 5         buy      last          ac      histori       follow
## 6      chines    assist      gerard         fact         just
## 7      silvio      true  donnarumma        sport  gigiodonna1
## 8     italian       10s       thank       report  suso30ofici
## 9       inter       hbd       itali        close carlos7bacca
## 10     takeov     reliv        mufc        april        enter
```

Compare with JUV topic model:

```
##    Bayern   Serie A    Transfer   Contract    Compete
## 1    cont    buffon         cfc     report       goal
## 2  report      juve         psg   contract       fcim
## 3    sign     itali        want       sign      match
## 4    told    seriea         afc    allegri      europ
## 5  bayern   tolisso       alexi     dybala     napoli
## 6   alaba       amp chelstransf      czech  strongest
## 7    high barcelona         win realmadrid fiorentina
## 8    keen    latest     sanchez        set     defens
## 9    term     thank        citi        max     averag
## 10     lb    napoli javierriosr        new     conced
```

### 3.3.2 Computing Perplexity

Evaluate the topic model by computing perplexity.

```
train = d[sample(1:nrow(d), 2400), ]
eval = d[setdiff(1:nrow(d), train), ]

d = list()
for (k in c(10, 25, 50, 100)) {
  message(k)
  for (i in 1:5) {
  m = LDA(train, k = k, method = "Gibbs", control=list(alpha=5/k, iter=100))
  p = perplexity(m, eval)
  d = c(d, list(data.frame(k=k, i=i, p=p)))
  }
}
```

Test the perplexity of train and eval dataset seperately. Perplexity was 58.9 on the training data, and a little higher 60.6 on the evaluation set.

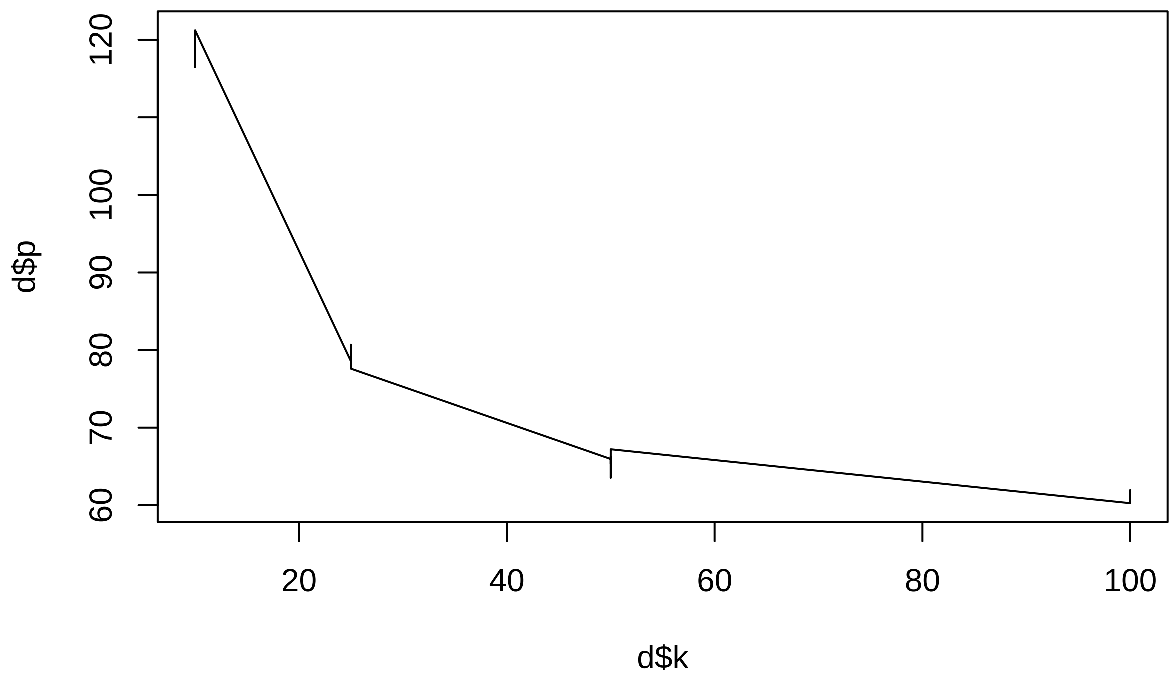
```
perplexity(m, newdata = train)
```

```
## [1] 57.42046
```

```
perplexity(m, newdata = eval)
```

```
## [1] 60.29319
```

```
d = rbind.fill(d)
d2 = tapply(d$p, d$k, mean)
plot(d$k, d$p, type="l")
```




### 3.4 Sentiment Analysis

#### 3.4.1 Applying Sentiment Dictionary to DTM

Read lexicon as dictionary to define positive and negative words.

```
lexicon = readRDS("lexicon.rds")
pos_words = lexicon$word1[lexicon$priorpolarity == "positive"]
neg_words = lexicon$word1[lexicon$priorpolarity == "negative"]
```

Read ACM tweets texts, and apply the sentiment dictionary to create matrix of positive and negative words.

```
textacm=acm_tweets[c("text")]
acmdtm = create_matrix(textacm, language="english",removePunctuation=T, stemWords=F)
acm_tweets$npos = row_sums(acmdtm[, colnames(acmdtm) %in% pos_words])
acm_tweets$nneg = row_sums(acmdtm[, colnames(acmdtm) %in% neg_words])
```

Calculate the sentiment scores.

```r
acm_tweets$sent = (acm_tweets$npos - acm_tweets$nneg) / (acm_tweets$npos + acm_tweets$nneg)
acm_tweets$sent[is.na(acm_tweets$sent)] = 0
```

Use quanteda to apply the sentiment dictionary to ACM tweets.

```r
dict = dictionary(list(pos=pos_words, neg=neg_words))
dfm2 = dfm(paste(acm_tweets$text), dictionary=dict)
head(dfm2)
```

```
## Document-feature matrix of: 3,313 documents, 2 features (0% sparse).
## (showing first 6 documents and first 2 features)
##        features
## docs    pos neg
##   text1   0   0
##   text2   0   0
##   text3   0   0
##   text4   2   0
##   text5   0   1
##   text6   0   0
```

Convert it to a data.frame and combine it with the tweets, according to the sentiment classification and retweetcount.

```r
d2 = as.data.frame(dfm2)
acm_tweets = cbind(acm_tweets, d2)
acmt=subset(acm_tweets,select=c("id","screenName","text","retweetCount","pos","neg","sent"))
acmt$gold = as.factor(ifelse(acmt$retweetCount > 100, "pos", "neg"))
acmt$pred = ifelse(acmt$sent > 0, "pos", "neg")
m = table(acmt$gold, acmt$pred)
m
```

```
##
##        neg  pos
##   neg 1562  816
##   pos   15  920
```

As the result shown, when the retweetCount is more than 100, there seems to be more negative words in the text. So I guess more negative contents about AC Milan are more easily to spread on Twitter.

### 3.4.2 Validating sentiment

Compute accuracy, precision, and recall on a dichotomized version of the predictions and manual sentiment. Firstly calculate overall prediction accuracy:

```r
sum(diag(m)) / sum(m)
```

```
## [1] 0.7491699
```

Then compute precision, recall and FScore per class. As the FScore is not very high, the coorelation between retweetCount and words sentiment is not very significant.

```r
pr = m["pos", "pos"] / sum(m[, "pos"])
re = m["pos", "pos"] / sum(m["pos", ])
f1 = 2*pr*re/(pr+re)
print("FScore:")
```

```
## [1] "FScore:"
```

```r
c(Precision=pr, Recall=re, FScore=f1)
```

```
## Precision    Recall    FScore
## 0.5299539 0.9839572 0.6888806
```

# 4 Conclusion

## 4.1 Social Network Analysis

As the friends connections shown, AC Milan has a close connection with other football clubs and some famous footballers, even sports celebrity in other area like Kobe Bryant, and interestingly, the FIFA is in the center place of this network, and this graph reveals the social network of football world to a certain extent.

## 4.2 Textual Analysis

To conduct the textual analysis, firstly I use JUV tweets to do the comparision. From the wordcloud plots, we could see "footballitalia" is the most frequent neutral word between 2 data sets, and ACM focus on its own team news, while JUV relates to other external news.

Secondly, the dictionary analysis also proved my guess from the simple wordclouds. As the bar graph shown, JUV has more texts related to European football issues, while ACM focus on FC Inter besides its own team news.

Thirdly, from the topic model of ACM tweets, I conclude 5 topics, which are "Berlusconi", "Rui Costa", "Rising Star","News" and "Transfer". Except "Transfer", all the other 4 topics are closely related to the discussion of AC Milan internal news. Comparing with JUV top model, which is hard to classify each topic, ACM model could more qualitative illustrate different topics. And by computing perplexity, the score is around 60, just as the graph shown, 25 is the turning point, which means ACM topic model makes sense.

## 4.3 Sentiment Analysis

By using the lexicon dictionary, I conduct a sentiment analysis of ACM tweets, by testing the correlation between retweetCount and word sentiment. As the result shown, when the retweetCount is more than 100, there seems to be more negative words in the text, which means more negative contents about AC Milan are more easily to spread on Twitter. To validate sentiment analysis result, as the FScore 0.688 is not very high, the coorelation between retweetCount and words sentiment is not very significant.

## 4.4 Summary

AC Milan official twitter can reveal the football world's important association. To compare the tweets of ACM and JUV, ACM is more focus on milan football, including the team news and derby news, while JUV is more related to the discussions on European football, including the transfer news and European Champions League news. And on Twitter, negative tweets about ACM are possibly to be more widely spread among football fans.