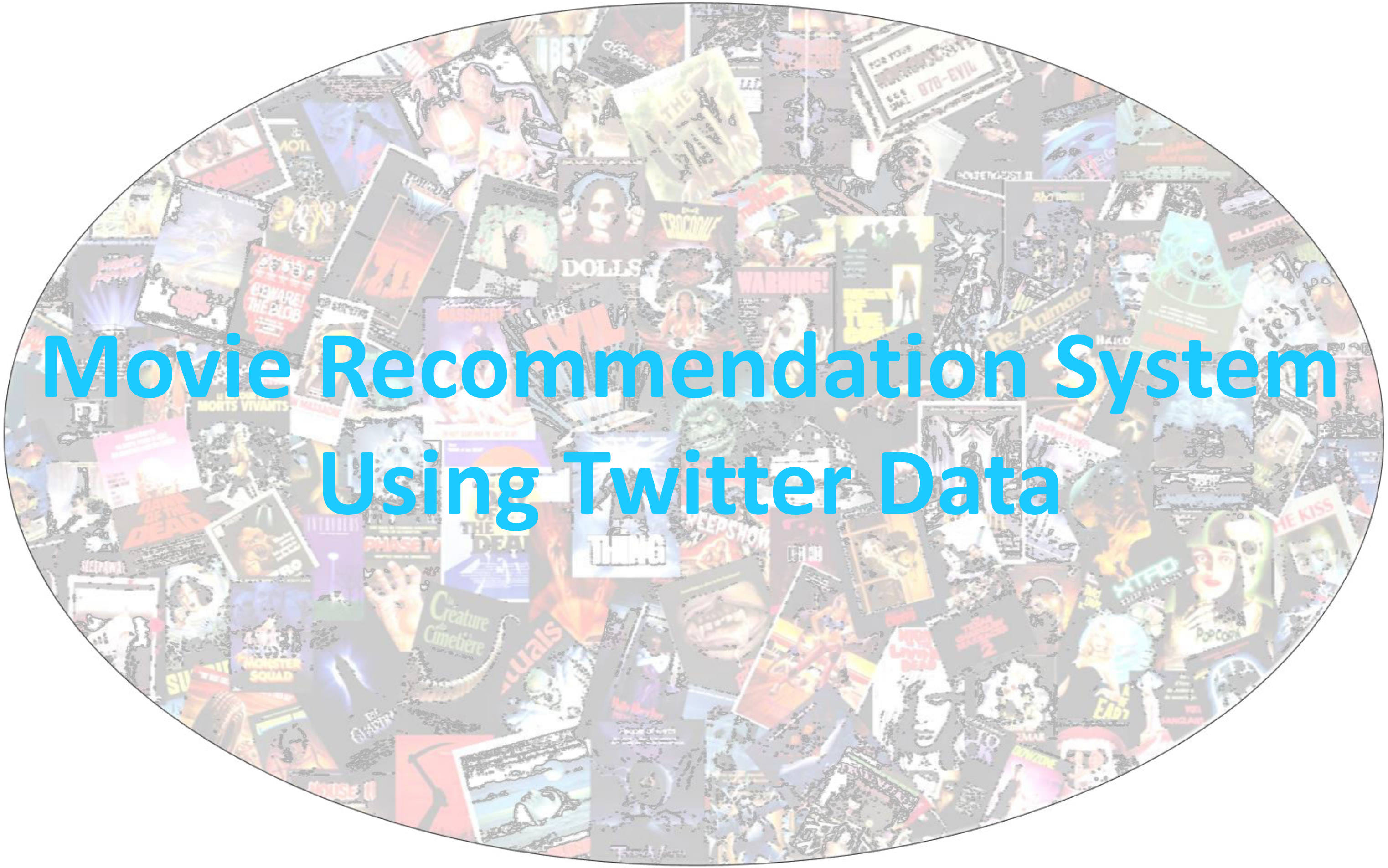
Movie Recommendation System Using Twitter Data



Discussion Agenda

1 PROJECT OBJECTIVE

2 PROJECT METHODOLOGY

3 PROJECT OUTPUT

4 NEXT STEPS

Project Objective

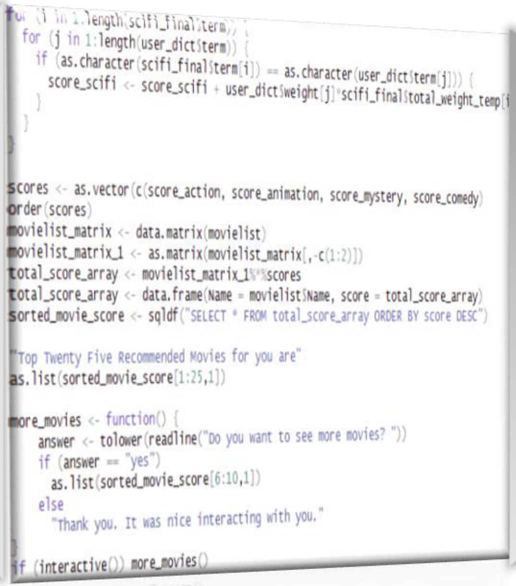
Recommending Movies to a User Based on his Twitter Handle

"Why so serious?"

"Frankly, my dear, I don't give a damn"

*"I'm not a big fat panda. I'm *the* big fat panda"*

"I see dead people"



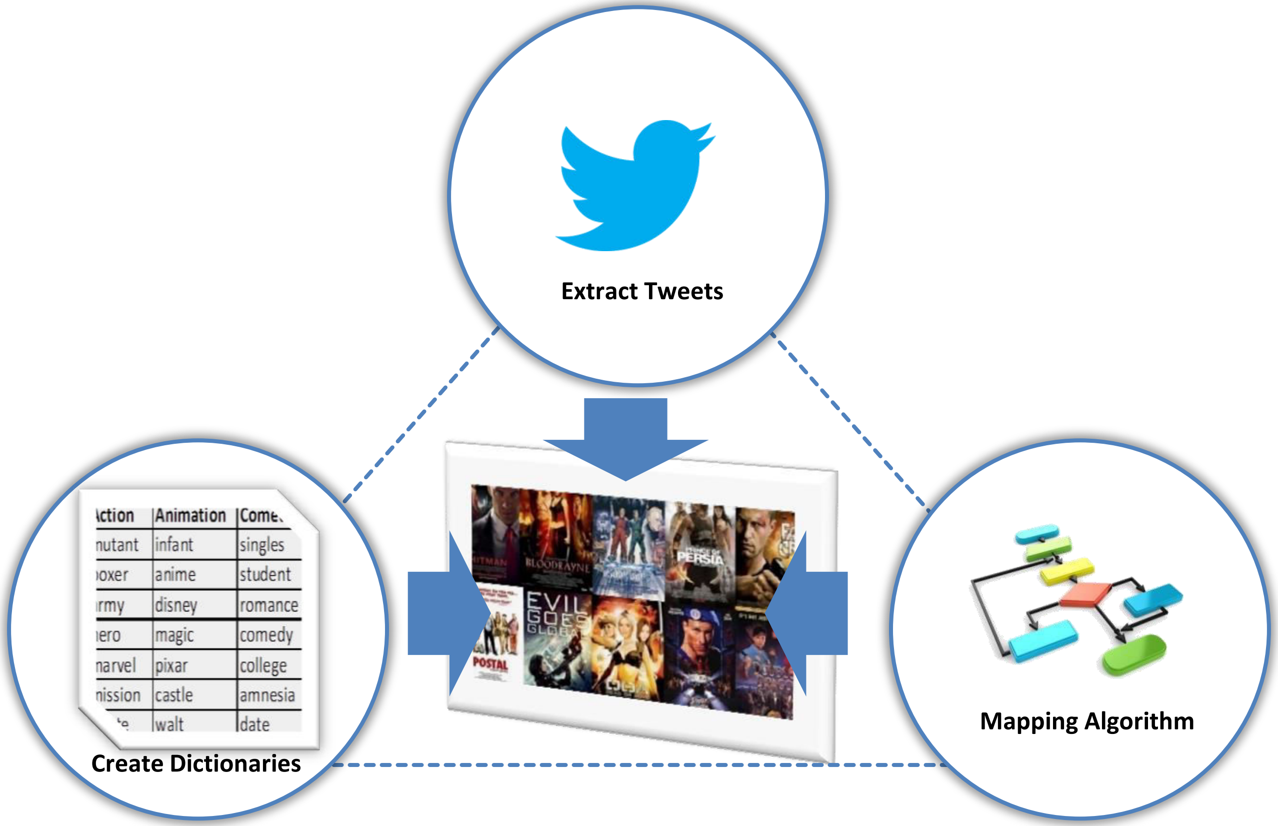
"I'm Gonna Make Him An Offer He Can't Refuse"

"This is your life and it's sending one minute at a time"

"Why do we fall? So we can learn to pick ourselves up"

Project Methodology

Mapping User's Preferences With Different Genres



Project Methodology

Extracting Twitter Data

1

Extract Tweets
- *Create Twitter app and get access key and token*
- *Complete authentication and extract Tweets*

Convert into data frame
- *Create a corpus of extracted tweets*
- *Keep only the text part of tweets*

Clean data
- *Remove #hashtags*
- *Remove other stop words, prepositions, helping verbs*

Create term document matrix
- *Create term document matrix*
- *Keep words which have word length > 2 characters*

Assign weights
- *Create data frame of word stems of selected words*
- *Assign weights to each word using TF-IDF method*

- userTimeline
- twListToDF
- regmatches
- TermDocumentMatrix
- tm_map
- Corpus

```
term.freq <- rowSums(as.matrix(tdm))
term.freq <- subset(term.freq, term.freq >= 1)
df <- data.frame(term = names(term.freq), freq = term.freq)
wm <- data.frame(df, weight = (1+log10(term.freq))*log10(sum(df$freq)/term.freq))
user_dict <- data.frame(term = wordStem(wm$term), weight = wm$weight)
```

Project Methodology

Building Movie Database

Movie Description<>Genres<>Recommended Movies

250 Movie Description

11 Genres

1800 Movie to Recommend

ID	Name	Action	Animation	Mystery	Comedy	Drama	Family	Horror	Romance	Thriller	Adventure	SciFi
1	State of the World	0	0	0	0	1	0	0	0	0	0	0
2	Garage	0	0	0	0	1	0	0	0	0	0	0
3	Frank	0	0	1	1	1	0	0	0	0	0	0
4	Mission: Impossible III	1	0	0	0	0	0	0	0	1	1	0
5	Star Trek	1	0	0	0	0	0	0	0	0	1	1
6	Rana's Wedding	0	0	0	1	1	0	0	0	0	0	0
7	Paradise Now	0	0	0	0	1	0	0	0	0	0	0
8	Omar	0	0	0	0	1	0	0	0	1	0	0

Project Methodology

Creating Genre Dictionaries

250 Movie Description

Action	Animation	Comedy
mutant	infant	singles
boxer	anime	student
army	disney	romance
hero	magic	comedy
marvel	pixar	college
mission	castle	amnesia
pirate	walt	date
hero	cartoon	firemen
bond	joy	lie
confidentia	child	speed

Clean → Keep Frequent words → Word Stem

themoviedb.org

Tag words for every movie

Manually selecting related words

Mapping these words to specific genres, which they represent

```
horror_df <- data.frame(term = horror_df$term, total_weight_horror = horror_df$total_weight_horror*dict_factor)
horror_key_words <- data.frame(term = unique(moviegenrekeywords[,7]), total_weight_horror = mean(horror_df$total_weight_horror)*keyword_factor)
horror_bind <- rbind.data.frame(horror_df, horror_key_words)
horror_bind <- data.frame(term = wordStem(horror_bind$term), total_weight_horror = horror_bind$total_weight_horror)
horror_final <- sqldf("SELECT term, SUM(total_weight_horror) AS total_weight_horror FROM horror_bind GROUP BY term")
```

Project Methodology

Assigning Weights to Words and Finding User’s Genre Preference

UserTweets	Weight
hero	0.002589
marvel	0.238937
mission	0.025734
comedy	0.006662
college	0.114809
joy	0.302830
child	0.217025
date	0.472366
day	0.098997
fight	0.198521

List of words from user’s Twitter Handle

Action	Weight	Action
mutant	0.002355	**Animation**
boxer	0.023459	**Mystery**
army	0.123588	**Comedy**
hero	0.007896	**Drama**
marvel	0.003698	**Family**
mission	0.087352	**Horror**
pirate	0.314879	**Romance**
hero	0.361255	**Thriller**
bond	0.098763	**Adventure**
confidentia	0.198756	**SciFi**

11 Dictionaries, one for each genre

Genre	User's Score
action	31.2888101
animation	31.4013252
mystery	31.3814179
comedy	31.4627622
drama	30.9160581
family	31.0038619
horror	30.6097828
romance	31.2182833
thriller	31.0338976
adventure	30.7779819
scifi	30.8852927

Finding user’s genre preference by generating score for every genre

$$weight, t_{w,d} = TF \times IDF$$

Term Frequency (TF)

- Frequency of a word in a document

Inverse Document Frequency (IDF)

- Relative importance/impact of a word in a corpus of a document

$$TF = \begin{cases} 1 + log_{10} t_{w,d}\ , & t_{w,d} > 0 \\ 0\ , & otherwise \end{cases}$$

$$IDF = \{log_{10}(total\ words\ in\ the\ document/t_{w,d}),\ t_{w,d} > 0\}$$

- tm_map
- wordStem
- TermDocumentMatrix
- sqldf

```
for (i in 1:length(action_final$term)) {
  for (j in 1:length(user_dict$term)) {
    if (as.character(action_final$term[i]) == as.character(user_dict$term[j])) {
      score_action <- score_action + user_dict$weight[j]*action_final$total_weight_temp[i]
    }
  }
}
```

Mapping Algorithm

Identifying Movies as per User's Genre Preferences

$$\begin{bmatrix} Death\ Race & 1 & 0 & 1 & 0 & \cdots & 1 \\ \vdots & & & \vdots & & & \\ OldBoy & 0 & 1 & 1 & 0 & \cdots & 1 \\ \vdots & & & \vdots & & & \\ KungFu\ Panda & 1 & 0 & 1 & 1 & \cdots & 0 \\ \vdots & & & \vdots & & & \\ Blood\ Diamond & 0 & 0 & 1 & 1 & \cdots & 1 \\ \vdots & & & \vdots & & & \\ Beowulf & 1 & 0 & 1 & 1 & \cdots & 0 \end{bmatrix}_{1814 \times 11} \begin{bmatrix} Action & 31.288 \\ \vdots & \vdots \\ Comedy & 31.462 \\ \vdots & \vdots \\ Scifi & 30.885 \end{bmatrix}_{11 \times 1} =$$

Name	Score
Manchurian Candidate, The	95.00495
Bourne Supremacy, The	95.00495
Paprika	95.00495
Scanner Darkly, A	91.59114
Inception	91.59114
Oldboy	91.59114
Watchmen	89.54713
Death Race	89.54713
Lucy	89.54713
Titan A.E.	88.17734

- *Take dot product of movie matrix (1814 X 11) with the score vector (11 X 1)*
- *Get movie score vector (1814 X 1) having score for every movie*
- *Recommend movie at the top of the list*

```
movielist_matrix <- data.matrix(movielist)
movielist_matrix_1 <- as.matrix(movielist_matrix[,-c(1:2)])
total_score_array <- movielist_matrix_1%*%scores
total_score_array <- data.frame(Name = movielist$Name, score = total_score_array)
sorted_movie_score <- sqldf("SELECT * FROM total_score_array ORDER BY score DESC")

"Top Ten Five Recommended Movies for you are"
as.list(sorted_movie_score[1:10,1])
```

How the Output can be used?

```
[1] "Top Ten Movies Recommended to you are"
> as.data.frame(sorted_movie_score[1:10,1])
   sorted_movie_score[1:10, 1]
1   Manchurian Candidate, The
2        Bourne Supremacy, The
3                      Paprika
4           Scanner Darkly, A
5                    Inception
6                      Oldboy
7                    Watchmen
8                  Death Race
9                        Lucy
10                 Titan A.E.
```

- *Identify user's personality traits*
- *Identify user's interests and likings for specific products*
- *Build Recommendation System, not just for movies but for whole suit of products*
- *Understand user's socio-economic status*

- **Real-time,**
- **Social-Media,**
- **Targeted**

MARKETING

Next Steps??

Related Words

- Matching synonyms and/or related words, in the genres and tweets of the person
- Words which are associated with the genre could impact results significantly

Increasing Number of Genres

- More the number of genres, better would be mapping and hence the results
- Netflix has around 77,000 micro genres, which provide fairly accurate results

Find association of movies

- Movies which are related to each other can be grouped together
- Various aspects for clustering could be time period, cast, director, among others

Including other aspects of Twitter handle

- User personality traits can be used and mapped with different movies
- User’s socio-economic, demographic traits can be used to create clusters

Thank You ☺

PGDBA Group 08
Bharathi R
Faichali Basumatary
Shoorvir Gupta
Vishal Bagla