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

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

"I see dead people"

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

"Why so serious?"

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

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

Mapping User’s Preferences With Different Genres

- Extract Tweets
- Create Dictionaries
- Mapping Algorithm
Project Methodology

Extracting Twitter Data

- **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

### R Code Examples
- `userTimeline`
- `twListToDF`
- `regmatches`
- `TermDocumentMatrix`
- `tm_map`
- `Corpus`

```r
# Create term frequency matrix
term.freq <- rowSums(as.matrix(tdm))

# Keep terms with frequency >= 1
term.freq <- subset(term.freq, term.freq >= 1)

df <- data.frame(term = names(term.freq), freq = term.freq)

# Calculate weights using TF-IDF
wm <- data.frame(df, weight = (1+log10(term.freq)) * log10(sum(df$freq)/term.freq))

# Function to get word stems
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

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
<th>Action</th>
<th>Animation</th>
<th>Mystery</th>
<th>Comedy</th>
<th>Drama</th>
<th>Family</th>
<th>Horror</th>
<th>Romance</th>
<th>Thriller</th>
<th>Adventure</th>
<th>SciFi</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>State of the World</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>Garage</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>Frank</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>Mission: Impossible III</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>Star Trek</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>Rana's Wedding</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>Paradise Now</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>Omar</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Project Methodology
Creating Genre Dictionaries

<table>
<thead>
<tr>
<th>Action</th>
<th>Animation</th>
<th>Comedy</th>
</tr>
</thead>
<tbody>
<tr>
<td>mutant</td>
<td>infant</td>
<td>singles</td>
</tr>
<tr>
<td>boxer</td>
<td>anime</td>
<td>student</td>
</tr>
<tr>
<td>army</td>
<td>disney</td>
<td>romance</td>
</tr>
<tr>
<td>hero</td>
<td>magic</td>
<td>comedy</td>
</tr>
<tr>
<td>marvel</td>
<td>pixar</td>
<td>college</td>
</tr>
<tr>
<td>mission</td>
<td>castle</td>
<td>amnesia</td>
</tr>
<tr>
<td>pirate</td>
<td>walt</td>
<td>date</td>
</tr>
<tr>
<td>hero</td>
<td>cartoon</td>
<td>firemen</td>
</tr>
<tr>
<td>bond</td>
<td>joy</td>
<td>lie</td>
</tr>
<tr>
<td>confidential</td>
<td>child</td>
<td>speed</td>
</tr>
</tbody>
</table>

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 <-ssqlf("SELECT term, SUM(total_weight_horror) AS total_weight_horror FROM horror_bind GROUP BY term")```
Assigning Weights to Words and Finding User’s Genre Preference

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

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

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

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

\[
\text{weight}, t_{w,d} = TF \times IDF
\]
Mapping Algorithm
Identifying Movies as per User’s Genre Preferences

\[
\begin{bmatrix}
\text{Death Race} & 1 & 0 & 1 & 0 & \ldots & 1 \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
\text{OldBoy} & 0 & 1 & 1 & 0 & \ldots & 1 \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
\text{KungFu Panda} & 1 & 0 & 1 & 1 & \ldots & 0 \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
\text{Blood Diamond} & 0 & 0 & 1 & 1 & \ldots & 1 \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
\text{Beowulf} & 1 & 0 & 1 & 1 & \ldots & 0
\end{bmatrix}
\]

\[
\begin{bmatrix}
\text{Action} & 31.288 \\
\vdots & \vdots \\
\text{Comedy} & 31.462 \\
\vdots & \vdots \\
\text{Scifi} & 30.885
\end{bmatrix}_{11 \times 1}
\]

- 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

\[
\text{Name} \quad \text{Score}
\]

- 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
How the Output can be used?

- 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
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