Business Values from Wikipedia Clickstream Data

Peter LIU

Wikipedia

Facts about Wikipedia

- A non-profit database for people to explore knowledge
- Widely used as a go-to reference website
- March 2016 release for English Wikipedia has 6.8 billion views

Wikipedia Clickstream Data

- Clickstream data for Nov 2017

```
df.head()
```

	prev	curr	type	n
0	other-empty	Holly_Hull	external	43
1	other-search	Holly_Hull	external	18
2	other-empty	Boy_Slaves	external	57
3	Peon	Boy_Slaves	link	12
4	List_of_films_featuring_slavery	Boy_Slaves	link	27

Attributes:

1. Number of Clicks
2. Click type

Hot topics in Nov 2017

The most viewed article in the month

- Celebrities
- Hot movies in Nov
- News will greatly influence the topic ranking

```
# Aggregate the clicks according to topics
article_link_in = df.groupby('curr').sum()

# Sort n of clicks by descending
article_link_in.sort_values(by='n', ascending=False).head(10)
```

	n
curr	
Main_Page	505262821
Hyphen-minus	39017884
XHamster	9537393
Meghan_Markle	8289694
Justice_League_(film)	5480246
Stranger_Things	5231810
Charles_Manson	4553276
Thor:_Ragnarok	3725272
Deaths_in_2017	3299351
Lil_Peep	2910143

The most cited sources in the month

-Search engines and website

- Hot movies
- Summary of events

```
# The top refer sources are from search engine, empty string, e
article_link_out = df.groupby('prev').sum()
article_link_out.sort_values(by='n', ascending=False).head(10)
```

	n
prev	
other-search	2978059866
other-empty	2291181777
other-external	132415436
other-internal	116059001
Main_Page	29716738
Stranger_Things	2815939
Justice_League_(film)	2230895
Thor:_Ragnarok	1887976
Deaths_in_2017	1664140
Muscat_and_Oman	1226358

Business Values

- **Topic recommendation**

  Recommendations based on connections with topics, and updated by user's browsing history

- **Topic classification**

  Identify topics based on web traffic data. This is specifically useful when classification is the business concern, such as fraud alert, virus detection and spam detection

- **Advertisement profits prediction**

  Suppose we launched an advertisement campaign for the movie "Stranger Things". We plan to post this ad in different topics and would like to develop algorithm to 1) predict the profits, and 2) identify the most profitable topics

Topic Recommendation

- **Proposed Approach**

1. Build an undirected network (*e.g.*, once two topics are linked more than 100 times) using networkx package
2. Recommend neighbors (linked topics) of the topic, ranked by the weights of links
3. Based on list of topics (browsers' history), find the common neighbors of topic list

```
# We select links that higher than 100 clicks and drop NAs

df_link = df[df.type=='link']
df_link = df_link[df_link.n>100]
df_link = df_link.dropna()
```

```
# Build the network using networkx package

import networkx as nx
clickstream = nx.Graph()

# Add nodes
clickstream.add_nodes_from(df_link.prev)
clickstream.add_nodes_from(df_link.curr)

# Add edges
clickstream.add_edges_from(zip(df_link.prev, df_link.curr))
```

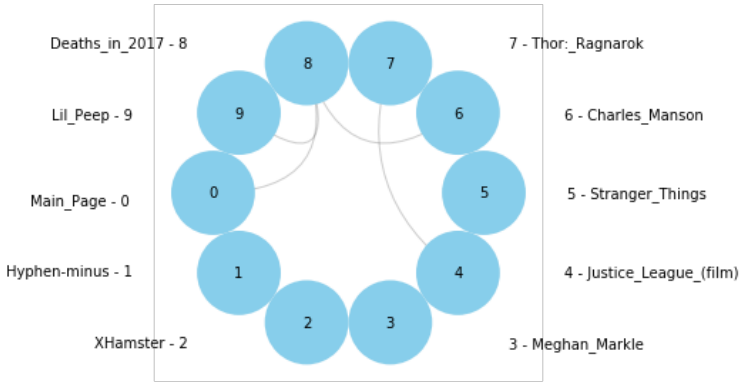


Figure. Top 10 hot topics and their relations

Topic Recommendation

- **One Topic**

```
topic_recommendation('Data_science',10)
```

```
['Data_mining',
 'Machine_learning',
 'Business_analytics',
 'Information_science',
 'Data-driven',
 'Cluster_analysis',
 'Statistics',
 'Information_explosion',
 'Big_data',
 'Business_analyst']
```

- **List of Topics (Browser's history)**

```
# Show recommendations of topics related with 'United States' and 'Donald_Trump'
l = []
for i in topic_recommendation(['United_States','Donald_Trump']):
    l.append(i)
l
```

```
['Barack_Obama', 'President_of_the_United_States', 'Mike_Pence']
```

Features from Network Analysis

- Degree of centrality
- Betweenness centrality
- Number of clicks
- Click through rate(ctr)
- Number of views
- Number of refers

- **Shortest path between topics**

```
# Return the shorted path between Kevin Sparcey and Meghan Markle.
# Cutoff 2 is chosen because the shortest path between these two topics is 2

list(nx.all_simple_paths(clickstream,'Kevin_Spacey','Meghan_Markle',cutoff = 2))
```

```
[['Kevin_Spacey', 'Horrible_Bosses', 'Meghan_Markle'],
 ['Kevin_Spacey', 'IMDb', 'Meghan_Markle']]
```

- **Most Connected Topic v.s. Most Influential Topic**

- “United States” is most connected topic which directly connects 1,897 nodes with 537,227 clicks and 283 clicks/link
- “Stranger_Things” is most influential topic which directly connects with 213 nodes with 13,109 clicks per link and totally 2,792,254 clicks

Topic Classification

Data Preparation

1. Scraping movies and actors/actresses names from Wiki
   12784 Movie names from 1990 to 2019, 5603 Actors/Actresses names

2. Generate features and labels for each movies and movie stars
   Movies label 1, Names label 0, 81551 records, with 8 features

```
df_new.head()
```

	n_neighbors	clicks_link	clicks_other	betweenness	ctr	size_maximal_clique	n_views	n_referred	label
A	3.0	0.0	0.0	0.0	0.000000	3.0	195278.0	0.0	0
AJ_Michalka	26.0	15278.0	42.0	0.0	0.351404	4.0	43477.0	16088.0	0
Aaliyah_Haughton	0.0	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0
Aaron_Carter	19.0	12290.0	167.0	0.0	0.173502	4.0	70835.0	14365.0	0
Aaron_Eckhart	47.0	12931.0	41.0	0.0	0.233294	4.0	55428.0	14219.0	0

3. Scale the data
   Mean = 0, Variance = 1

Modeling for Topic Classification

Logistic Regression

```
with open('pickle/lr.pickle','rb') as f:
    lr = pickle.load(f)

lr_predict = lr.predict(xtest)

model_performance(ytest, lr_predict)
```

```
Accuracy:      0.7142565192512057
Roc_Auc_Score: 0.7243622894355001
F1_Score:      0.7205947004516208
Confusion Matrix:
[[8460 5039]
 [1952 9015]]
```

Random Forest

```
with open('pickle/rf.pickle','rb') as f:
    rf = pickle.load(f)

model_performance(ytest, rf.predict(xtest))
```

```
Accuracy:      0.8679800539524237
Roc_Auc_Score: 0.8747944565737972
F1_Score:      0.8646383371050206
Confusion Matrix:
[[10920  2579]
 [  651 10316]]
```

KNN

```
with open('pickle/knn.pickle','rb') as f:
    knn = pickle.load(f)

model_performance(ytest, knn.predict(xtest))
```

```
Accuracy:      0.8552685359274095
Roc_Auc_Score: 0.8630527447625829
F1_Score:      0.8531984577753824
Confusion Matrix:
[[10635  2864]
 [  677 10290]]
```

```
feature_eval = pd.DataFrame.from_dict({'Feature':df_new.columns[:-1],
                                       'Importance': rf.feature_importances_,
                                       'Coefficient':lr.coef_[0]})

feature_eval = feature_eval.set_index('Feature')

feature_eval.sort_values('Importance', ascending = False)
```

	Coefficient	Importance
Feature		
n_views	-2.843770	0.293820
ctr	1.636485	0.190153
n_referred	3.058649	0.189774
n_neighbors	0.076477	0.103567
size_maximal_clique	-1.403904	0.086974
clicks_link	-2.217717	0.080158
clicks_other	0.413321	0.049645
betweenness	0.201414	0.005908

- Feature importance selected from random forest and coefficients obtained from logistic regression
- Top 4 important features explained 76% variance
- Movie star topics result more views than movies
- Movies are connected with more topics and thus have higher ctr rates

Profits from Advertisement

Scenario

We launched an advertisement campaign for the movie “Stranger Things”. We planed to post this ad in different topics/articles and would like to predict profits earned from ad clicking.

The expected profits were calculated as ctr*(profit/click)*clicks. So if profit per click is $2, ctr is 0.04 and clicks are 1,000, our anticipated profit would be $80

Data Preparation

- Select topics that link to “Stranger Things”
- Generate web traffic features
- Calculate the profit of ad clicks for each topic

Profits Prediction

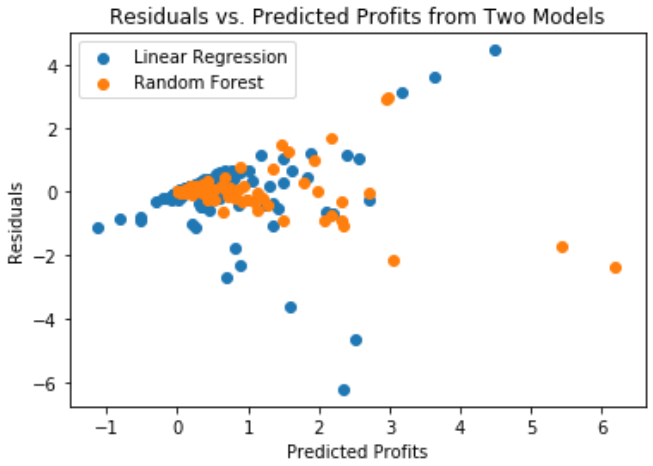
Comparison of Model Performance

```
Linear Regression
==========================================
True Sum: 77.74517654617279
Predicted Sum: 77.70325236130218
Residual Sum of Square: 157.8897164351228


Random Forest Regressor
==========================================
True Sum: 77.74517654617279
Predicted Sum: 79.4741015008377
Residual Sum of Square: 46.135478475486124


KNN Regressor
==========================================
True Sum: 77.74517654617279
Predicted Sum: 70.78252058145742
Residual Sum of Square: 98.19983078218394
```

Residual vs. Predicted Profits by Linear Regression and Random Forest

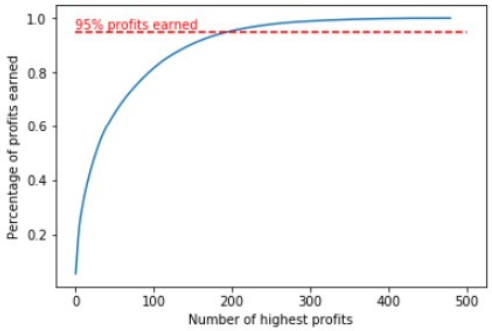


Identify Most Profitable Topics

- **Top 196 topics account for 95% profits**
- **Label the top 196 articles as 1, and the rest as 0.**

```
# 196 highest profits account for 95% profits
_ = plt.plot(range(len(df_ad)), np.cumsum(df_ad.sort_values('profit', ascending=False).profit/df_ad.profit.sum()))
plt.plot([0,500], [0.95, 0.95], 'r--')
plt.annotate('95% profits earned', xy = (0,0.96), xytext = (0,0.96),color = 'r')
plt.xlabel('Number of highest profits')
plt.ylabel('Percentage of profits earned')
```

```
<matplotlib.text.Text at 0x1f7040925f8>
```



- **Extract the topic text using Beautiful Soup, vectorize the text using BOW or TFIDF**

```
df_articles.head()
```

	decision	text	topic
0	1	Matthew Avery Modine (born March 22, 1959) is ...	Matthew_Modine
1	1	Telltale Incorporated, doing business as Tellt...	Telltale_Games
2	1	Daniel Rosenfeld, (born 9 May 1989) also known...	C418
3	1	Sadie Sink (born April 16, 2002) is an America...	Sadie_Sink
4	1	Charlie Heaton is an English actor and musicia...	Charlie_Heaton

Modeling for Profitable Topics Prediction

- **Model Performance**

Naive Bayes

```
model_performance(ytest, nb_pred)
```

```
Accuracy:      0.8214285714285714
Roc_Auc_Score: 0.8214285714285714
F1_Score:      0.8275862068965518
```

RF

```
model_performance(ytest, rf_predict)
```

```
Accuracy:      0.75
Roc_Auc_Score: 0.75
F1_Score:      0.7586206896551724
```

Logistic (Ridge)

```
model_performance(ytest, lr_predict)
```

```
Accuracy:      0.8571428571428571
Roc_Auc_Score: 0.8571428571428572
F1_Score:      0.8666666666666666
```

Logistic (Lasso)

```
model_performance(ytest, lr_lasso_predict)
```

```
Accuracy:      0.8214285714285714
Roc_Auc_Score: 0.8214285714285714
F1_Score:      0.8148148148148148
```

- **Common selected important words by RF and Lasso**

```
# Common important features selected by both random forest and lasso
set(features_rf).intersection(set(features_lasso))
```

```
{'announced', 'based', 'members', 'new', 'played', 'super', 'things'}
```

- **Probability of a topic being profitable**

```
# Probability that the topic will be high profitable topics
lr.predict_proba(count_vector.transform([extract_text('Millie_Bobby_Brown')]))[0][1]
```

```
0.92875650834407297
```

```
# Probability that the topic will be low profitable topics
lr.predict_proba(count_vector.transform([extract_text('Netflix')]))[0][1]
```

```
0.99999999999027334
```

```
# Probability that the topic will be low profitable topics
lr.predict_proba(count_vector.transform([extract_text('Microsoft')]))[0][1]
```

```
3.5820724825128777e-05
```

Conclusion

- We explored **business values** from Wikipedia clickstream data, which could be easily generalized to commercial clickstream data analysis
- We extracted topic networks, web traffic data, and word tokens, which were further successfully applied in **topic recommendation**, **topic classification**, and **profits prediction**