Business Values from Wikipedia Clickstream Data

Peter Liu

8/24/2018

Wikipedia, a non-profit database for people to explore knowledge, has been widely used as a go-to reference website.To give you a sense of the scale of the data, the March 2016 release for English Wikipedia contained 25 million distinct “(referer, resource)” pairs from a total of 6.8 billion requests. This project is trying to answer questions that might be of business interests based on wikipedia clickstream data, and will also be a good reference for business analysis. The Github link included here.

https://github.com/liudj2008/Clickstream-data-analysis/blob/master/ClickStream.ipynb

1. Problem Statement

Since the born of first web page on August 6, 1991 by Tim Berners-Lee, the internet extensively changed people’s way of thinking. The spread of news, information and knowledge became more and more convenient and affordable. According to Zeitgeist 2012 report, 1.2 trillion searches in 2012 or 3 billion searches per day were conducted through Google. This enormous clickstream data helped Google keep updated with people’s interests, which in turn helps Google create or improve products that better meet people’s needs in the future. Inspired by Google’s success, we herein explored clickstream data of Wikipedia, the largest go-to reference website, for potential business values. The analysis we provided here could be easily generalized to provide real-world business impacts.

- Topic recommendation

  Based on clickstream data analysis and market basket analysis, we built a topic recommendation system, in which recommendations were ranked according to the weighted connection with topics, and recommendation results could be more customized based on user’s browsing history

- Topic classification

  This project offered a solution for topic classification based on web traffic data analysis, where movie names and movie star topics were separated with more than 85% confidence. This is specifically useful when classification is the business concern, such as fraud alert, virus detection and spam detection

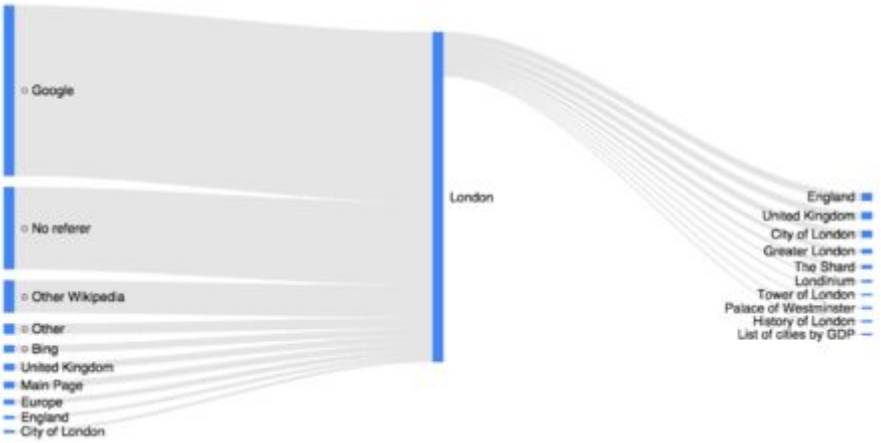
- Profits prediction

Nowadays, profits obtained from ad clicks are the major source of profits for internet companies. The algorithm we developed here will 1) identify the most profitable articles for our ads based on NLP and 2) predict the profits for specific ad based on clickstream data

2. Description of the dataset

All the clickstream data are public available and could be downloaded from https://meta.wikimedia.org/wiki/Research:Wikipedia_clickstream

The Wikipedia Clickstream dataset contains counts of (referer, resource) pairs extracted from the request logs of Wikipedia. A referer is an HTTP header field that identifies the address of the webpage that linked to the resource being requested. The data shows how people get to a Wikipedia article and what links they click on. In other words, it gives a weighted network of articles, where each edge weight corresponds to how often people navigate from one page to another. To give an example, consider the figure below, which shows incoming and outgoing traffic to the "London" article on English Wikipedia during January 2015.



Herein, we adopted most recent 8 months data from Nov 2017 to June 2018. The dataframe includes four columns: “prev”, “curr”, “type” and “n”. For Nov 2017 data, there are total 26 million records, with only 47 NA values.

2.1 Load Nov 2017 data

```
# Load dataframe and save as pickle file
df = pd.read_csv('https://dumps.wikimedia.org/other/clickstream/2017-11/clickstream-enwiki-2017-11.tsv.gz',
                 sep='\t',
                 names=['prev','curr','type','n'],
                 compression = 'gzip')
```

2.2 Nov 2017 data frame

```
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

Data Instructions

Prev

Sources leading to wiki topics/pages. In addition to wiki topics, other sources include 1. other-empty: direct enter through http; 2. other-search: link through search engine; 3. other external: link through other website; 4. other-internal: through non english wikipedia database; 5. other-other: Unknown source. NA means the sources have been removed or updated to a new one

Curr

Visited Wiki pages/topics. NA means the topics being updated or removed

Type

Three classes including external, other and link. External means sources are from external, other means both the source and curr are wiki topics, but there is no direct link between them. Link means both source and curr are wiki topics and there is direct link between them.

n

Number of clicks

3. EDA

3.1 Global’s hot topics in Nov 2017

- **The most viewed article in the month**

People were interested in knowing about celebrities, and hot movies on Nov. The reason why “Meghan_Markle” climbed to the top was due to the engagement with Prince Harry on Nov 27. The first episode of “Stranger Things” was watched by 15.8 million people within the first three days from Oct 27 and thus arouse people's interests throughout November. In addition, the distribution of the number of clicks is

heavily right skewed (mean>>median), and almost 50% of articles are viewed less than 100 times, we will use this as the cut off

```
In [8]: # 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)
```

Out[8]:

	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

Figure. Top 10 viewed articles in 2017

- **The most cited sources in the month**

The top refereed sources are other sources instead of internal links, mainly from search engines and enter through http. For the internal links, the highest cited articles are hot movies such as *stranger things*, *justice league* and summary of events, such as *deaths in 2017*.

```
# 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

Figure. Top 10 refereed articles in 2017

3.2 Relations between hot topics

The way of exploring the relations between different topics is to build a topic network. To simplify the problem, we built an undirected network (e.g., once two topics are linked and linked more than 100 times, they has relations) using networkx package.

```
# 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))
```

- **Visualization of connections between different topics**

There is an intuitive visualization of the network by nxviz. It showed that hot topics are closely related. For example, top 10 hot topics such as Deaths_in_2017 connected Lil_Peep, David_Cassidy and Charles_Manson (Figure below) are related. And with more nodes added in, more connections will be revealed. For top 20 hot topics, Meghan markle is connected with Prince Harry and Matt Lauer. Stranger things are connected with Millie Bobby Brown.

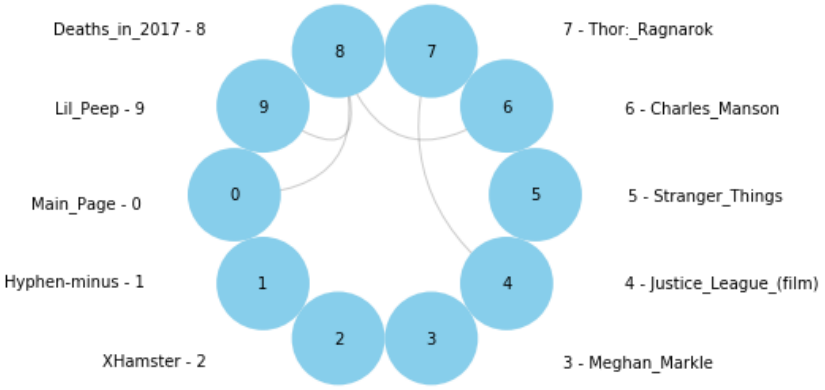


Figure. Top 10 hot topics and their relations

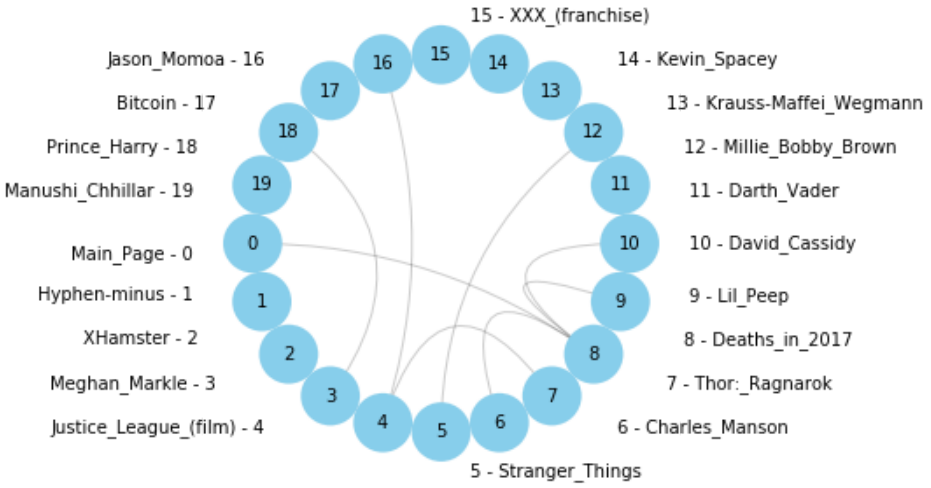


Figure. Top 20 topics and their relations

- **How to find connection between topics that are not directly connected?**

Use network analysis, we can find the shortest path between indirectly linked topics, such as "Kevin Spacey" and "Meghan Markle". It is indicated that they both feature in Horrible Bosses, and their movies/shows are rated by IMDb

```
# 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']]
```

- **Topics with most connections and influences**

The topic that has the most connections are "United States", which directly connects 1,897 nodes with 537,227 clicks and 283 clicks/link. 99% of articles have neighbors less than 44. So 1897 connections that "United States" has are very significant large number. However, if we want to know which articles are more influential, *e.g.* more clicks than other articles, we can tell "stranger_things", which directly connects with 213 nodes, lead the board with 13,109 clicks per link and totally 2,792,254 clicks.

- **Topics that has largest number of similar topics**

Size of similar topics was calculated based on the size of maximal cliques for each topic. It is interesting to notice that the *top tiers are countries and the second tiers are football teams*.

```
sorted(size_maximal_clique.items(), key = lambda x:x[1], reverse=True)[:20]
```

```
[('Russia', 15),
 ('United_States', 15),
 ('Germany', 15),
 ('United_Kingdom', 15),
 ('Japan', 15),
 ('Australia', 15),
 ('China', 15),
 ('France', 15),
 ('India', 15),
 ('List_of_countries_by_GDP_(nominal)_per_capita', 15),
 ('Canada', 15),
 ('List_of_countries_by_GDP_(nominal)', 15),
 ('List_of_countries_by_GDP_(PPP)_per_capita', 15),
 ('Gross_domestic_product', 15),
 ('List_of_countries_by_GDP_(PPP)', 15),
 ('Purchasing_power_parity', 15),
 ('Italy_national_football_team', 14),
 ('2006_FIFA_World_Cup', 14),
 ('Argentina_national_football_team', 14),
 ('Brazil_national_football_team', 14)]
```

Figure. Topics that have largest number of similar topics

3.3 Topics Recommendation based on user's specific interests

Heavily weighted links of specific topic will reveal other topics that are closely related with this topic, and thus could be used for recommendation systems. We will answer questions such as: are there any other topics that might also interest you? The recommendation results will rank recommended topics according to number of clicks

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

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

Figure. Top 10 recommended topics related with "Data Science"

But what if customers are interested in two or more interests? Or will the machine provide more customized recommendation based on customer's browsing history?

This is a typical market basket analysis, and we achieved this by searching for common related topics between customers' browsing history. For example, when user viewed both "United States" and "Donald Trump", we could recommend three topics (e.g., "Barack Obama", "President of the United States", and "Mike Pence") as

the top common related topics

```
# 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']
```

4. Predict the topic class

Based on website traffic data, such as degree of centrality, betweenness centrality, number of clicks leading to other topics, click through rate(ctr), and times of viewed and referred, we can predict the class of the article (whether it is movie stars or movie names). Movie names were collected from 1990 to 2019. All the data were scraped from wiki. We generated features for each topic, and labeled them as 0 (artist) and 1 (movie name). We further concat all 8 month' data and scaled all the columns, resulting a dataframe with 81,551 rows and 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

Figure. Data collected for movie stars and movie names. Data was labeled artist as 0 and movie name as 1. Data was scaled before using ML model for classification.

4.1 Model Performance

Using default parameters, we can see the random forest and knn performs better than logistic regression. It is revealed that we can separate movie names and movie star topics with more than 85% confidence purely based on web traffic data. This is specifically useful for fraud alert, virus detection and spam detection based on web traffic data.

a)

```
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]]
```

b)

```
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]]
```

c)

```
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]]
```

Figure. Model performance of linear regression, random forest, and knn for prediction of binary classes

4.2 Interpretation of features and their importance in prediction

Herein, we summarized the feature importance obtained from random forest along with coefficients obtained from logistic regression, and we ranked the importance from high to low. It is interesting that the top 4 features explained 76% variance. Notice that movie stars are labeled as 0, movie names are labeled as 1. It is observed that movie star topics result more views than movies, while movies tend to be connected with more topics and thus have higher ctr rates.

```
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

5. Predict profits from advertising clicks

Herein, we made a bold assumption that Wiki is for profit and our goal to predict how much Wiki will earn from clicks of the advertisement. Suppose we launched an advertisement campaign for the movie 'Stranger Things'. We plan to post this ad in different topics/articles and would like to predict profits earned from ad clicking. The expected profit coming from ad clicks can be 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.

5.1 Model performance

Our hypothesis is that using 6 basic web traffic features, e.g., degree of centrality, betweenness centrality, number of clicks leading to other topics, click through rate(ctr), and times of viewed and referred, we can predict the ad profits. We applied linear regression, Random Forest Regressor, and KNN Regressor for profit prediction. It turned out Random Forest model rendered the lowest RSS.

```
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
```

Figure. Comparison of Residual Sum of Squares by different regression models

5.2 Model selection for ad profits prediction

We will choose the best model leading to the lowest residual sum of square. Herein, Random Forest Regressor rendered the lowest RSS, and thus is better than linear regression and KNN. It is interesting to observe that although linear regression shows closer sum of predicted profits to sum of true profits, the individual predicted profits show larger deviation from the true profits.

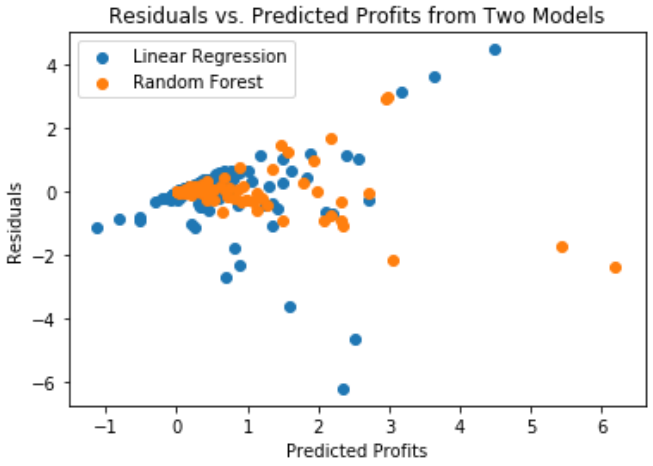


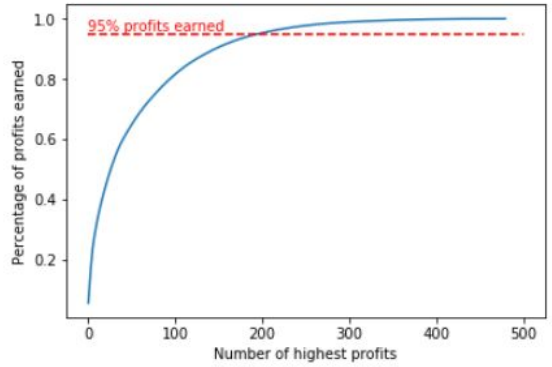
Figure. Residual vs. Predicted Profits by Linear Regression and Random Forest

6. Predict most profitable topics

It is noted that from total 480 topics that have our advertisement “Stranger Things”, 196 topics earned 95% of total profits. Our goal herein is to identify and predict the most profitable topics based on natural language analysis, so we can find the best topics to post our advertisement in the future.

```
# 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>
```



After labeling the top 95% profitable article by 1 and the rest by 0, we can extract texts from each topics using Beautiful Soup package, vectorize texts using Bag-of-Words and TFIDF package and remove the stop words. The resulting dataset has 93 rows and 10540 columns, and was trained using Naive Bayes, Random Forest and Logistic Regression. It is indicated that BOW data resulted slightly better model formance, and logistic regression with L2 regularization performed the best among all of these three algorithm (0.86, 0.86, 0.87 for accuracy, AUC score and F1-score respectively). In addition, we compared the important words selected by random forest and lasso, and ended up with 7 common important features, e.g., “announced”, “based”, “members”, “new”, “played”, “super”, “things”. Using the logistic regression by L2 regularization, we were able to predict topic such as “Millie_Bobby_Brown” and “Netflix” is extremely likely to be most profitable topics, while topic such as “Microsoft” is extremely unlikely to be most profitable topics.

7. Conclusion

Herein, we explored business values from Wikipedia clickstream data, which could be easily generalized to commercial clickstream data analysis. Based on clickstream data and Wikipedia webpage analysis, we extracted topic networks, web traffic data, and word tokens, which were further successfully applied in topic recommendation, topic classification, and profits prediction.