User Profiling in Recommender Systems

Coursework: ISyE 7406
Advisor: Dr. Yajun Mei

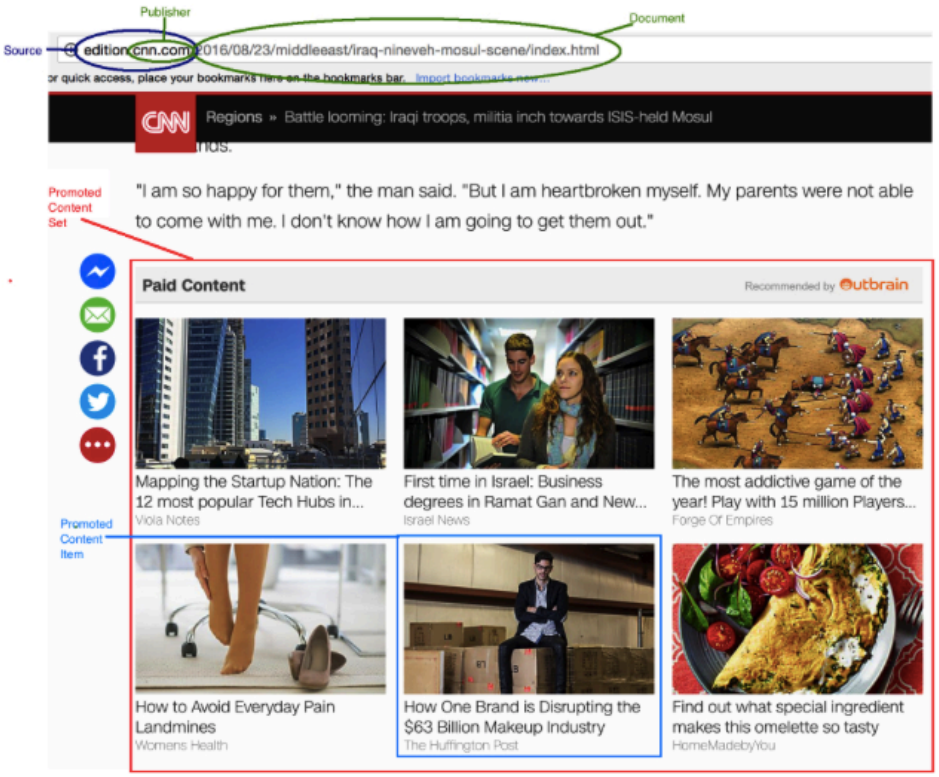
Team:
Abhijeet Mavi (903518738)
Ashwin S. Pothen (903467532)
Sagar Tolani (903472774)

Overview

The internet is a stimulating treasure trove of possibility. Every day we stumble on news stories relevant to our communities or experience the serendipity of finding an article covering our next travel destination. The challenge is to predict which pieces of content users are likely to click on.

Data recorded from 14-Jun-16 to 28-Jun-16 (2 weeks)

We believe that working with a well-structured recommendation data like this would help us to understand the working of a recommender system, provide scope for multiple data combinations, and then further provide a basis for pertinent applications in the education domain.



Data

The Kaggle dataset can be primarily divided into 3 parts: Page Views, Events and Document meta info.

Part I: Page Views – provides behavioral information about the website (document) visited by the user

The first user from New Jersey browses the webpage on his mobile by searching for it at 2,550ms past 12:00AM on 14-Jun-16

uuid	document_id	timestamp	platform	geo_location	traffic_source
53ca43b326638b	1773952	2550	2	US>NJ>501	2
21f3a405b1699d	1262700	6393	2	US>OH>515	1
47d84d241e270d	1167084	9765	1	US>NY>501	1
fbd1bff678b3b7	1340752	19184	2	US>FL>656	1
c745cf53f15066	1467277	21382	1	US>CA>803	1

2,034,275,448 rows (~90GB)

uuid - User ID for 370k unique users

document_id - unique web-document ID

timestamp - ms since first log on 14-Jun-16

geo_location - (country>state>media region)

platform – device used for reading the document *(desktop-1, mobile-2, tablet-3)*

traffic_source – how the user arrives at a document *(internal ads-1, search-2, social-3)*

Data

Part II: Events – Provides information about the ad clicked by the user among the given options

display_id	uuid	document_id	timestamp	platform	geo_location
1	cb8c55702adb93	379743	61	3	US>SC>519
2	79a85fa78311b9	1794259	81	2	US>CA>807
3	822932ce3d8757	1179111	182	2	US>MI>505
4	85281d0a49f7ac	1777797	234	2	US>WV>564
5	8d0daef4bf5b56	252458	338	2	SG>00

23,120,126 rows (~1.5GB)

The events data is a subset of the pageviews file where the user was provided recommendations and the response (click) was recorded.

Here, document_id refers to the webpage on which recommendations were provided.

display_id	ad_id	clicked
1	42337	0
1	139684	0
1	144739	1
1	156824	0
1	279295	0

ad_id	document_id	campaign_id	advertiser_id
1	6614	1	7
2	471467	2	7
3	7692	3	7
4	471471	2	7
5	471472	2	7

Using display_id, we have information about the recommendations on that page and the clicked recommendation.

Here, document_id refers to the webpage recommended to a user.

Data

Part III: Document Meta – Provides information about features of the webpage (document)

document_id	source_id	publisher_id	publish_time
1595802	1.0	603.0	2016-06-05 00:00:00
1524246	1.0	603.0	2016-05-26 11:00:00
1617787	1.0	603.0	2016-05-27 00:00:00
1615583	1.0	603.0	2016-06-07 00:00:00
1615460	1.0	603.0	2016-06-20 00:00:00

source_id – the specific segment that the content was published in (ex. CNN->History, CNN->Travel)

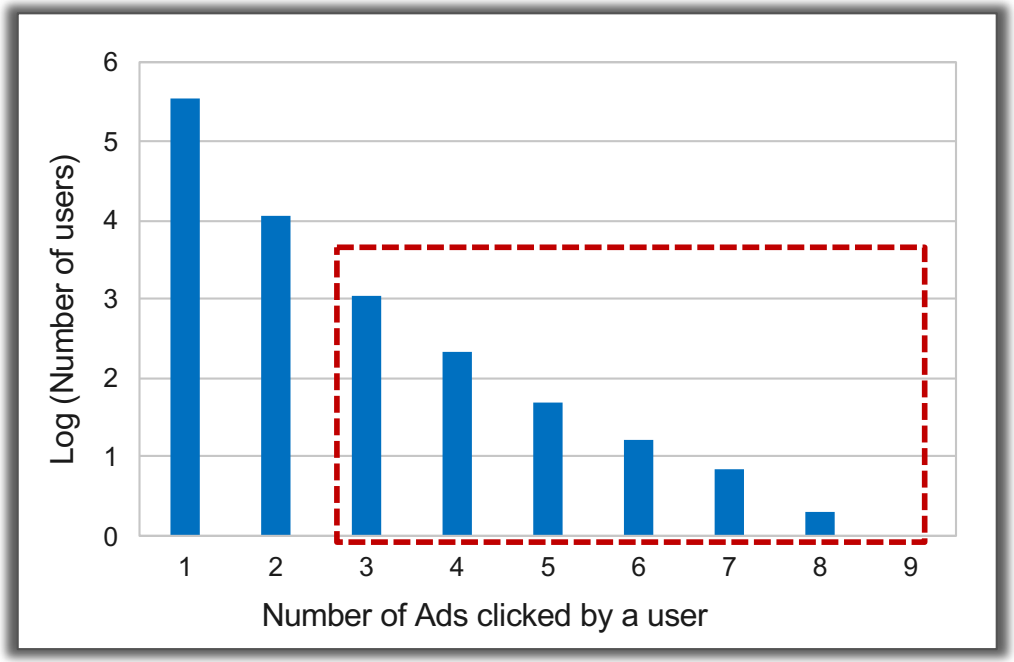
publisher_id – publishing house (ex. CNN, Fox News)

document_id	category_id	confidence_level
1782590	1608	0.715721
1782590	1511	0.054457
455149	1305	0.920000
455149	1608	0.070000
925820	1302	0.920000

document_id	topic_id	confidence_level
1782590	260	0.071256
1782590	113	0.020727
1782590	281	0.015755
455149	89	0.276621
455149	260	0.066864

The 2 datasets provide information about the categories and topics in a document, and Outbrain's confidence of the pair.

Exploratory Data Analysis

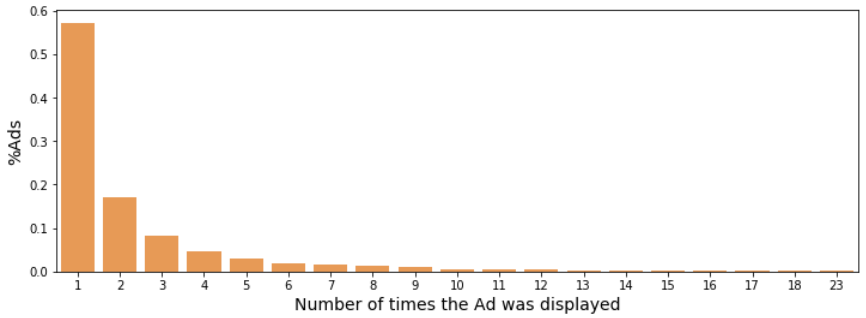


Ads clicked by user: We are provided with data where >99% of the users have <2 clicks, which isn't meaningful in the objective of user profiling.

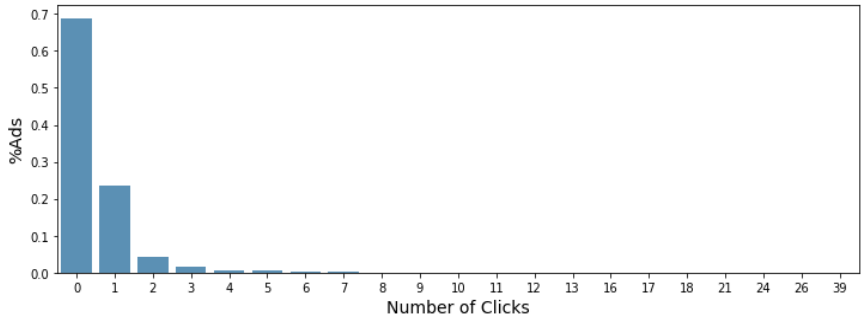
Hence, we proceed with analyzing the users with atleast 3 clicks giving us a total of 1,352 unique users.

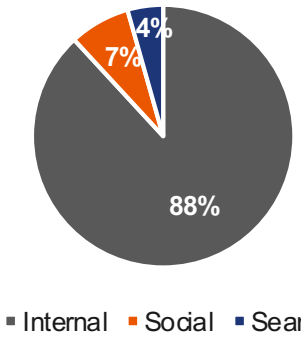
Exploratory Data Analysis

Recurrence of ads: Depicts that >80% of ads appear <3 times validating the *user-specific recommendations*



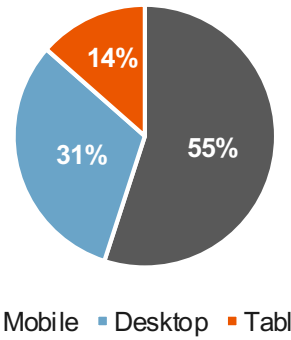
Click-through-rate: Depicts that >90% of ads have less than 1 click validating the *user-specific taste*



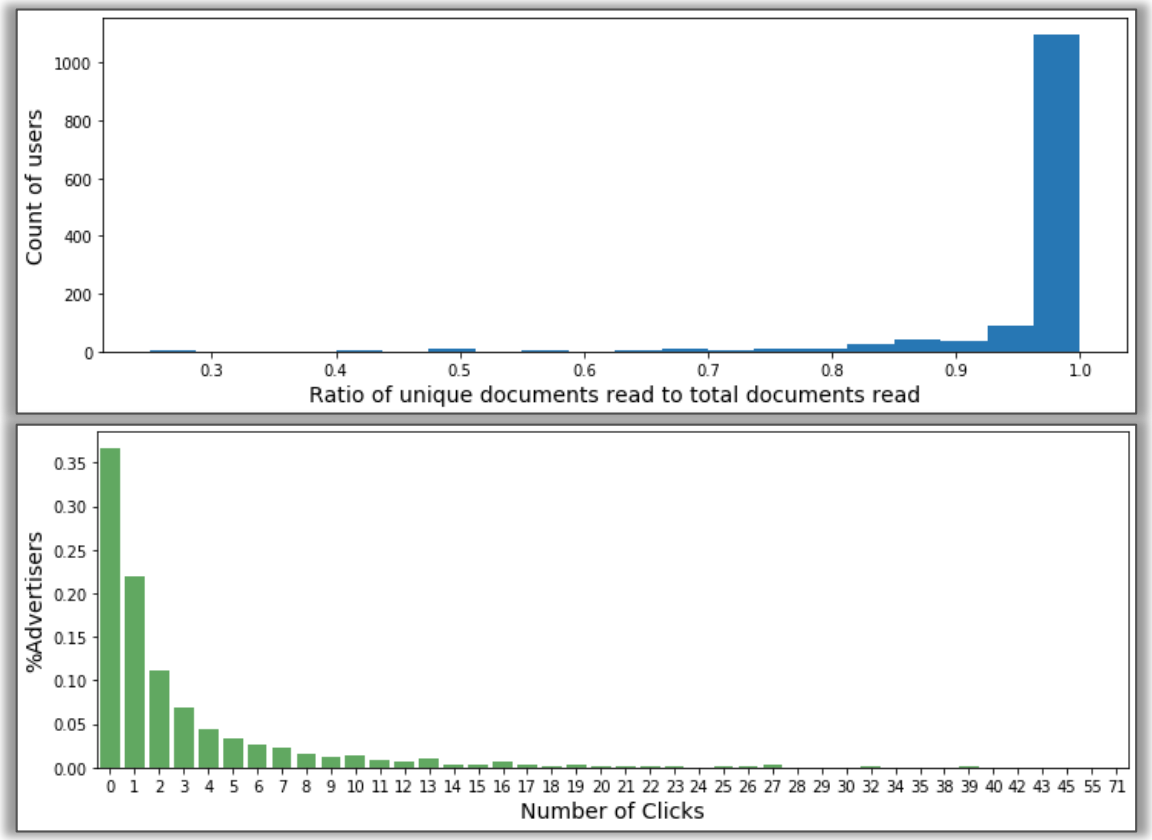


Most visits through internal recommendations highlights the **importance of quality recommendations**.

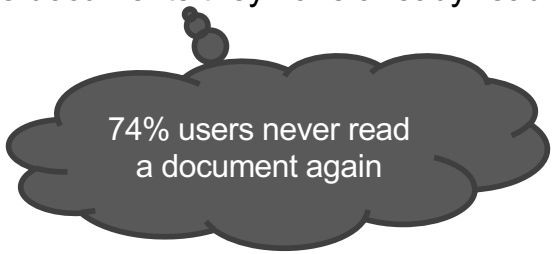
A **considerable use of all platforms** indicates that it should be included in the model.



Exploratory Data Analysis



Unique documents visited by a user: Depicts that majority users do not revisit the documents they have already read



Click-through-rate of ad features: Depicts that ~70% of advertisers got <3 clicks, which signifies that the user is not influenced by the advertiser.

Feature Engineering

Why? One-hot-encoding makes the data sparse. Hence, to make the data more intuitive and representative of the underlying problem, we engineer the features based on the following categories:

- **Time based-** Time influences user's preference

Timebins	Weekday
• 4 quadrants of the day	• Activity on weekday/weekend

- **User profile based-** Individuality of the user: Eagerness of a user "to read", to read "different" or "same" documents

User_pageviews	User_pageviews_unique	User_doc_read_count
• Total number of documents read by the user	• Unique number documents read by the user	• Number of times the user has read that document

Feature Engineering

- **Document based -** "Popularity" of the document and the publisher; "where" the document came from?

Doc_read_count	Publisher_read_count	Source_read_count
• Unique users that read the document	• unique users that read the publisher's document	• unique users that read the source's document

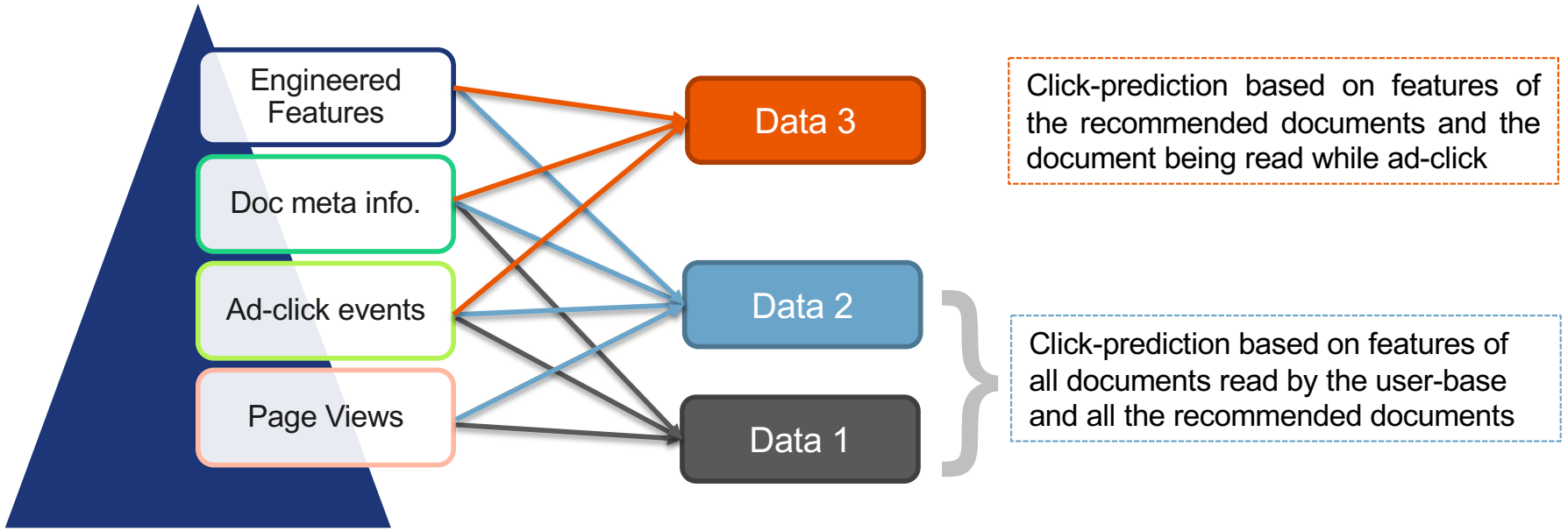
- **Click based –** click through rate for the ad-document and it's features

Click_through_rate	Feature_clicked	Display_ad_count
• Total number of clicks on an ad document	• Total number of clicks on an ad feature (campaign, advertiser, source, publisher)	• Number of choices that the user had on the webpage

Behaviour based - Subconscious decisions that greatly influence the user's preference. We consider the above engineered features and document features in the BLSTM model to find the relevant behavior based features.

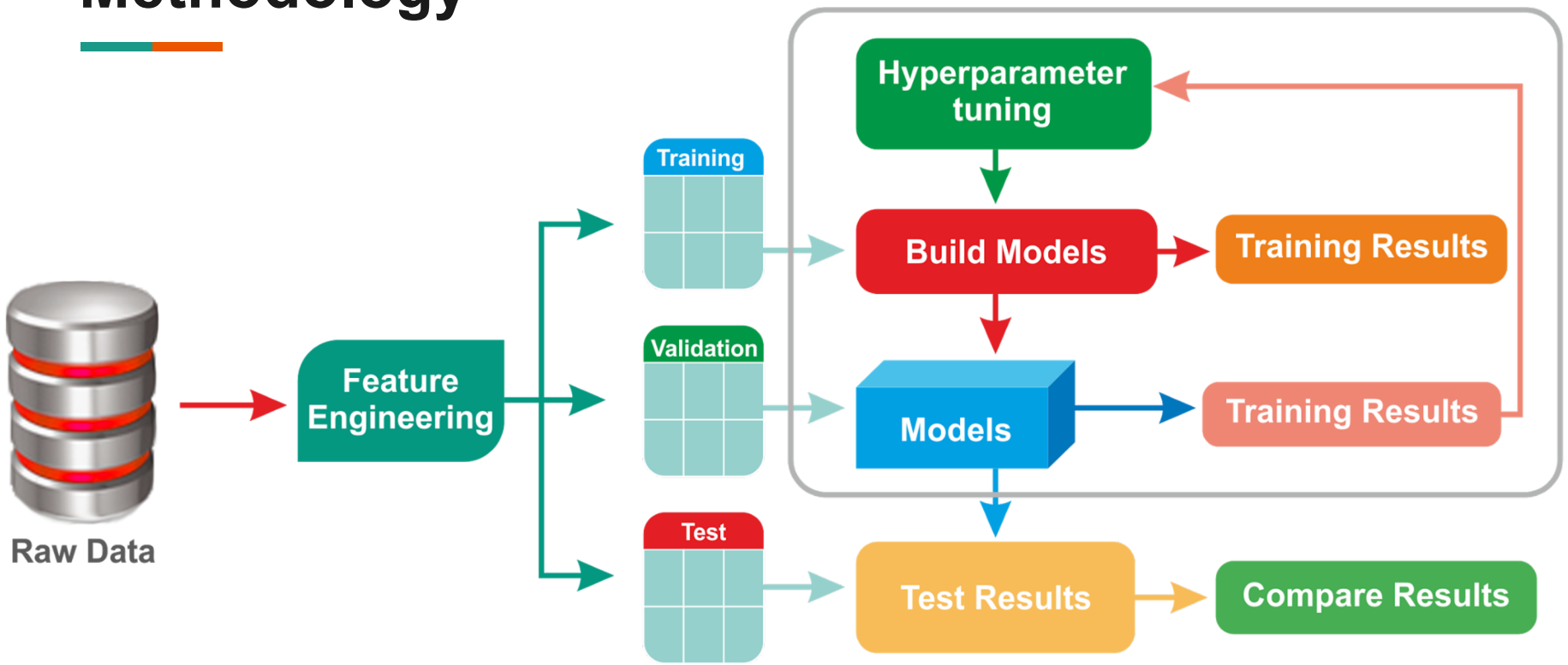
Preparation of Data

With the expanse of click-prediction data available, we had the opportunity of creating multiple datasets with the hope that we would be better prepared to tackle relevant applications, as we take this project forward…



(Prepared on Google Cloud Platform)

Methodology



Baseline Models

- Benchmarking LSTM models via two baseline models which provides a metric for accuracy, runtime and computing power
- The data was split into train (80%), validation (10%) and test set (10%); validation set was used to tune hyperparameters
- Time dummies were included based on minutes (60), hours (24) and day (14) to signify time sequence
- Cross-entropy loss function was used for misclassification rate

Logistic Regression

- **Why Logistic?** Natural binary classifier
- Data is sparse and hence **regularisation** Ridge, LASSO and Elastic Net stacked on logistic regression
- Cross validation, to find optimal penalty (L1 and L2 norm)
- For elastic net, L1-Ratio was tuned to find the optimal mix

Random Forests

- **Why RF?** Runs efficiently with large datasets and many categorical variables
- Improvised version of bagging
- Tree-depth tuned to avoid overfitting

Baseline model - Performance metrics

Data 1: PageViews + Events + document features

Model	Error rate (Validation set)	Error rate (Test set)	Run time for tuning	Server Utilized	Computational burden
Logistic (Ridge)	28.88%	53.36%	6 hours	GCP -16 CPU, 104 GB	Moderate
Logistic (LASSO)	**17.12%**	**49.07%**	**6.5 hours**	**GCP -16 CPU, 104 GB**	**Moderate**
Logistic (Elastic Net)	17.16 %	49.10%	7.5 hours	GCP -16 CPU, 104 GB	Moderate
Random Forest	32.40%	57.60%	2 hours	Google Colab - 25GB + GPU	Low

- **Logistic LASSO** performed the best variable selection
- Logistic regression was time consuming; RF was fastest with the worst error rate
- All models are observed to be overfit given the higher test error than validation error

Baseline model - Performance metrics

Data 2: PageViews + Events + document features + engineered features

Model	Error rate (Validation set)	Error rate (Test set)	Run time for tuning	Server Utilized	Computational burden
Logistic (Ridge)	**16.08%**	**16.62%**	**16 hours**	GCP -16 CPU, 104 GB	**High**
Logistic (LASSO)	16.12%	16.85%	18 hours	GCP -16 CPU, 104 GB	High
Logistic (Elastic Net)	16.39%	16.65%	19.5 hours	GCP -16 CPU, 104 GB	High
Random Forest	**16.20%**	**18.50%**	**3 hours**	Google Colab - 25GB + GPU	**Moderate**

- Feature engineered data improved the performance of all models
- Almost all models have the same misclassification rate
- **Random forest** performed the best in terms of accuracy, time and computing power

Baseline model - Performance metrics

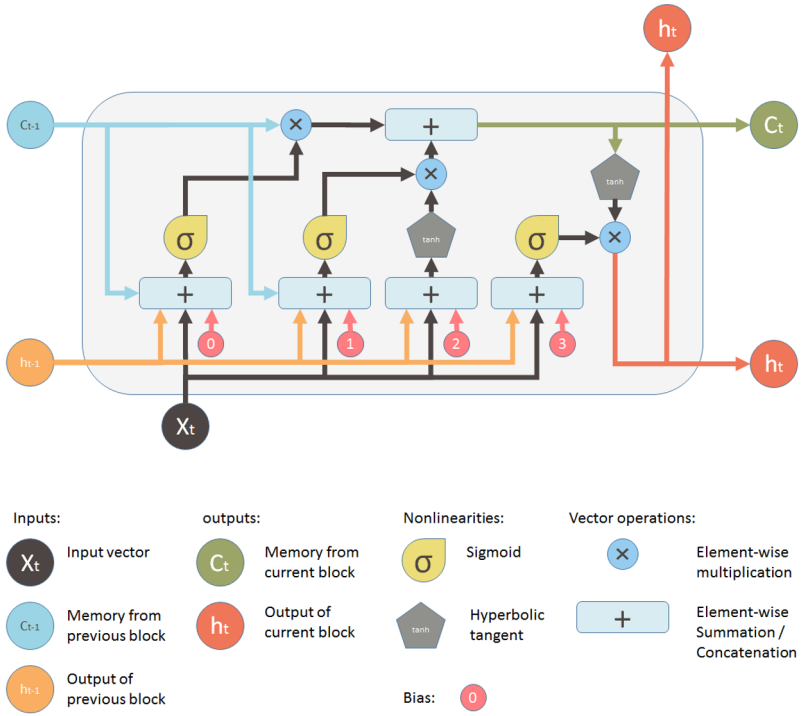
Data 3: Events + document features + engineered features

Model	Error rate (Validation set)	Error rate (Test set)	Run time for tuning	Server Utilized	Computational burden
Logistic (Ridge)	16.95%	18.26%	10 hours	GCP -16 CPU, 104 GB	High
Logistic (LASSO)	16.86%	18.17%	12 hours	GCP -16 CPU, 104 GB	High
Logistic (Elastic Net)	16.78%	18.23%	13 hours	GCP -16 CPU, 104 GB	High
Random Forest	**16.30%**	**18.26%**	**2 hours**	**Google Colab - 25GB + GPU**	**Low**

Random forest was the fastest, most accurate and required least computational burden

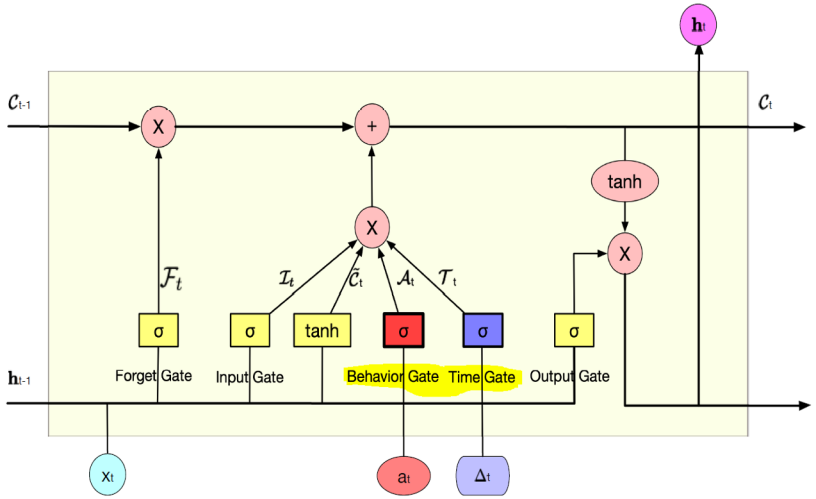
LSTM: The way forward

- Based on recurrent learning, LSTM cell forgets the irrelevant features at each step
- **Why LSTM over traditional time series?** Highly sparse features demand right selection at each step of the sequence
- LSTM conserves relevant memory in cell state



Behaviour LSTM: Above and Beyond

- Some features are inherently important and they give identity to a user via **behavior conservation**
- LSTM is ruthless in forgetting features; not taking into account the subjective definition of features
- In the fig., BLSTM gives importance to highlighted features (2) through same governing dynamics



Improving BLSTM and LSTM

Specifications of the Model	LSTM	Behavioural LSTM
Hidden feature length	8	8
Epochs	100	100
Batch size x Sequence length	8x32	8x32
Optimizer (loss criterion)	Adam (Cross-entropy loss for misclassified click rate [0,1])	Adam (Cross-entropy loss for misclassified click rate [0,1])

- **Hyperparameter tuning**: Hidden nodes, epochs, batch and sequence length
- **Avoid Overfit** by randomising the batches of a particular sequence length, we reduce the chances of overfitting on training data; choosing the right optimiser and loss criterion also a hyperparameter

Results: LSTM and BLSTM

Data	Model	Error rate (Validation set)	Error rate (Test set)	Run Time	Server Utilised	Behaviors
Data 1	LSTM	18.39%	20.23%	~15mins	Google Colab - 25GB + GPU	-
	BLSTM	**17.03%**	**17.98%**	**~15mins**	**Google Colab - 25GB + GPU**	**category, topic**
Data 2	LSTM	16.37%	18.29%	~15mins	Google Colab - 25GB + GPU	-
	BLSTM	**16.37%**	**18.03%**	**~15mins**	**Google Colab - 25GB + GPU**	**category, topic**
Data 3	LSTM	15.87%	20.09%	~30mins	Google Colab - 25GB + GPU	-
	BLSTM	**15.24%**	**17.56%**	**~30mins**	**Google Colab - 25GB + GPU**	**(Choices, weekday (0/1)), (category, topic)**

CONCLUSIONS

- Data 2, a more refined version of data 1, doesn't show deviation in BLSTM vs LSTM results as engineered features aid in forecasting but none of those qualify to be user behaviour
- Data 3 gives the best testing error based on **choices for a user on a display_id**; defines what he/she clicks and this behavior is further influenced if a page was visited during weekend or a weekday

FINAL CONCLUSIONS AND LEARNINGS

1. In a traditional recommender systems (Data 1), the behaviour is recorded by the content category and topics covered. If the number of choices and the time of weekly visit on a landing page are also considered, then the user profile is more refined and predictions are much better even on a smaller data (Data 3).

2. ML models vs BLSTM: BLSTM is much more time-efficient as only the pertinent features get learnt over time; user-behaviour is preserved via the type of choices he/she is provided and the webpage content

3. Big data hurdles: The problem is not solvable on traditional laptop CPUs, it requires cloud based data processing on GPUs or TPUs on AWS and Google Colab, which was a good learning exposure for us

4. Applications: The concept of **behavior conservation** exercised here will have huge implications in all kinds of user-interactive recommender systems: media, social media feed, e-commerce

5. An application that is of particular interest to us is using user profiling model to recommend educational articles to school students to facilitate holistic learning

References

- Yulong Gu, Zhuoye Ding, Shuaiqiang Wang, Dawei Yin. WSDM '20, February 3–7, 2020, Houston, TX, USA. Hierarchical User Profiling for E-commerce Recommender Systems.
- The Elements of Statistical Learning:Data Mining, Inference, and Prediction. By Trevor Hastie, Robert Tibshirani, Jerome Friedman.
- An Introduction to Statistical Learning with Applications in R. By Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani.
- How Feature Engineering can help you do well in a Kaggle competition link
- Various kaggle kernels: link
- Understanding RNN and LSTM: link
- Various pages: https://towardsdatascience.com
- LinkedIn learning: link