

# Talking Data Kaggle competition

Sowmya Vasam

Daisy Du

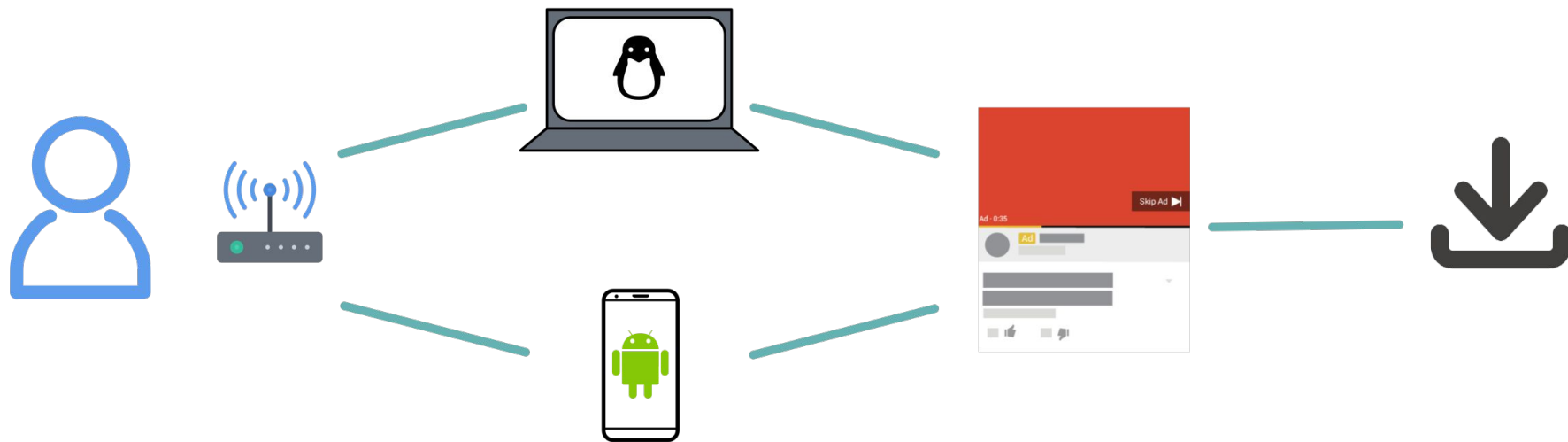
Erika Pelaez

Mentor: Nick Janetos





# Click journey

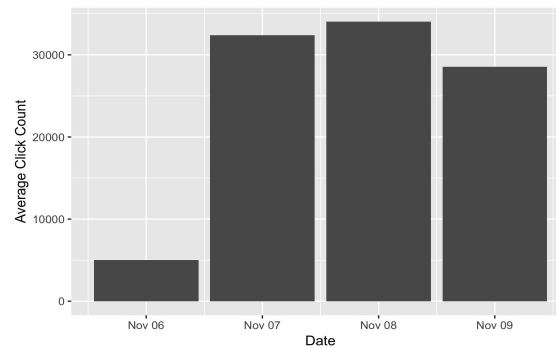
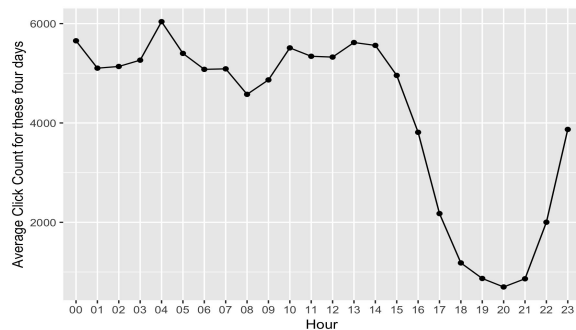




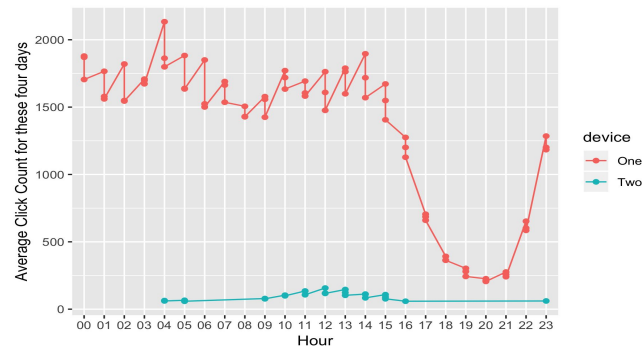
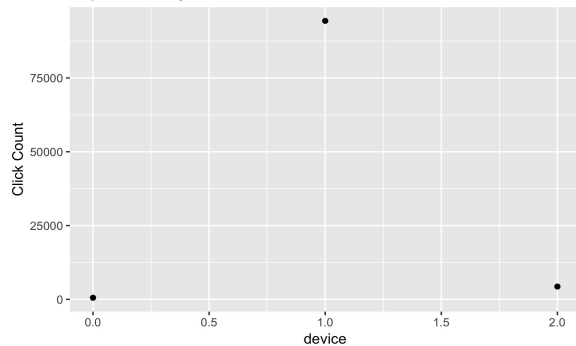
# Explanatory Analysis

Feature	Unique Count
IP address	277396
Device	3475
OS	800
App type	706
Channel	202

# Anatomy of the dataset



Top 3 clicks by device





# Feature Engineering

IP is an important feature but is highly dynamic. We decided to create various frequency based features on IP, to retain the importance of IP without actually having the feature, namely:

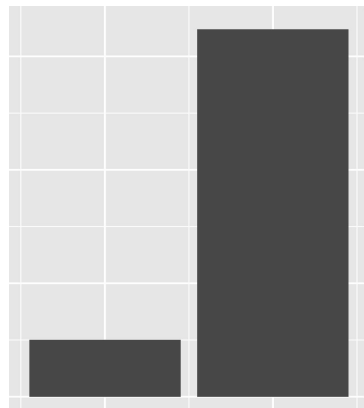
- Number of clicks every hour by an IP
- Number of clicks every hour by an IP-App combination
- Number of clicks every hour by an IP-OS combination
- Number of clicks every hour by an IP-Device combination
- Number of clicks every hour by an IP-Channel combination

Another feature we thought as interesting was to see if time from the previous click of the same click journey had any relation with app downloads hence we incorporated the following feature:

- Time from the previous click for every unique IP-App-OS-Device combination



# Sampling



Training  
20%



Validation  
20%

All Positives +





# Model selection and performance

Xgboost

Learning Rate:0.1

Max Depth : 6

1000 estimators

Early stopping rounds: 10

Subsample : 80% (0.8)

The screenshot shows a Google Colab notebook titled 'TalkingData\_training.ipynb'. The notebook interface includes a menu bar (File, Edit, View, Insert, Runtime, Tools, Help) and a toolbar with options for CODE, TEXT, and CELL. The output of a cell is displayed, showing the training and validation AUC scores for Xgboost across 130 iterations. The training AUC scores generally increase from 0.959038 at iteration 10 to 0.982587 at iteration 130. The validation AUC scores fluctuate, starting at 0.958585 and ending at 0.977031. The notebook also shows a 'Stopping. Best iteration:' message at iteration 129.

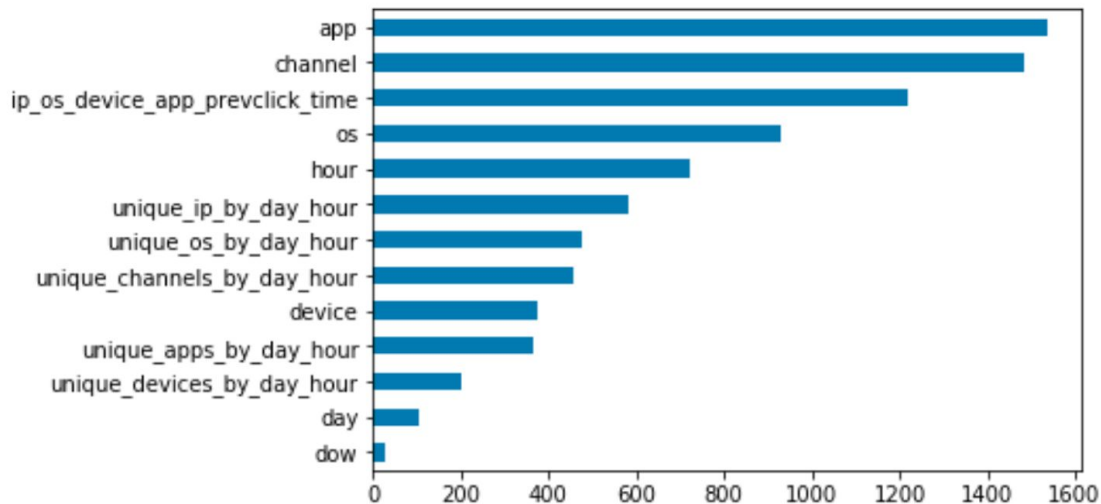
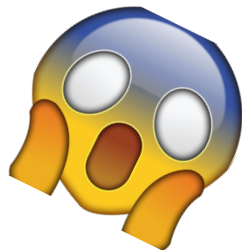
```
[10] train-auc:0.959038 validation-auc:0.958585
[20] train-auc:0.969642 validation-auc:0.96779
[30] train-auc:0.973738 validation-auc:0.971889
[40] train-auc:0.976116 validation-auc:0.973792
[50] train-auc:0.977614 validation-auc:0.974745
[60] train-auc:0.978969 validation-auc:0.975239
[70] train-auc:0.979857 validation-auc:0.975789
[80] train-auc:0.980454 validation-auc:0.97612
[90] train-auc:0.980971 validation-auc:0.976438
[100] train-auc:0.981385 validation-auc:0.976598
[110] train-auc:0.98187 validation-auc:0.976794
[120] train-auc:0.982178 validation-auc:0.976777
[130] train-auc:0.982587 validation-auc:0.976876
Stopping. Best iteration:
[129] train-auc:0.98253 validation-auc:0.977031
```



# Final results

Kaggle AUC Score:

0.82768







## Following steps

- Hyper parameter tuning
- More effective sample selection
- Different evaluation metrics (F1 score, suggestions?)
- Rolling frequency features which is applicable in a production setting

## Things Learnt

- Effective feature engineering when dealing with big data and limited memory
- Efficient way to handle big data with Dask
- Creative ways of handling sample selection with limited memory
- Modeling with large scale imbalanced dataset



**Questions?**



**Thank you!!**

**CONVOY**