**Click Fraud Detection**

**Project Documentation**

**Submitted By-**

**Team 2 BFS Track**

**Rahul Tomar – Team Lead**

**Dipanshu Upreti – Python Track**

**Vaishnavi Subramanya Desai – Python Track**

**Girish Kumar Reddy Tokala – Python Track**

**Ragini Gaurav - UI Track**

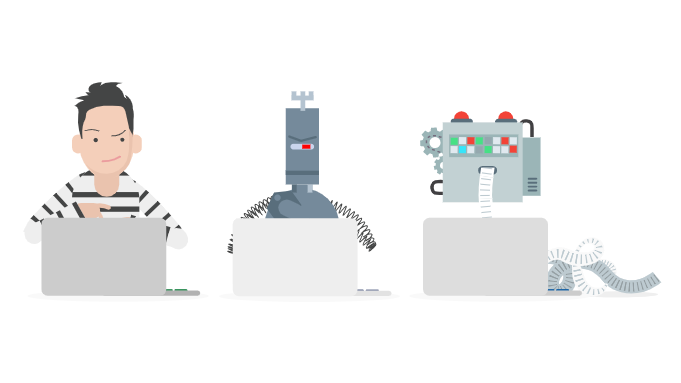
**Table of Contents**

**Introduction to Prohject**

**CLICK FRAUD DETECTION PROJECT**

**Click fraud** is a type of [fraud](https://en.wikipedia.org/wiki/Fraud) that occurs on the [Internet](https://en.wikipedia.org/wiki/Internet) in [pay-per-click](https://en.wikipedia.org/wiki/Pay_per_click) (PPC) [online advertising](https://en.wikipedia.org/wiki/Online_advertising). In this type of [advertising](https://en.wikipedia.org/wiki/Advertising), the owners of [websites](https://en.wikipedia.org/wiki/Website) that post the ads are paid an amount of money determined by how many visitors to the sites click on the ads. Fraud occurs when a person, [automated script](https://en.wikipedia.org/wiki/Auto_clicker), or computer program imitates a legitimate user of a [web browser](https://en.wikipedia.org/wiki/Web_browser), clicking on such an ad without having an actual interest in the target of the ad's link.

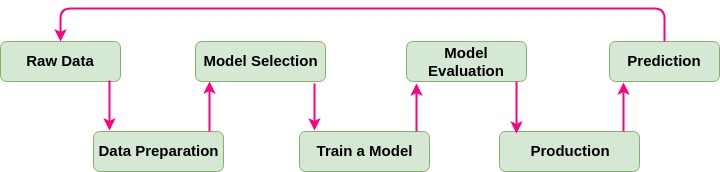
Fraud risk is everywhere, but for companies that advertise online, click fraud can happen at an overwhelming volume, resulting in misleading click data and wasted money. Ad channels can drive up costs by simply clicking on the ad at a large scale.

****

Although Google Adsense has checks in place it is still advisable to monitor your traffic details and analyzing them from time to time to monitor the data.

That’s where our product come in the picture. It serves as way to predict the spamming IP and Devices and find the rate at which Channels are generating clicks to find and isolate wasteful channels and increase spending in the better ones.

**Model Development**



**Dataset:-**

Dataset used by us was picked from Kaggle TalkingData AdTracking Fraud Detection Challenge([https://www.kaggle.com/c/talkingdata-adtracking-fraud-detection/data](https://www.kaggle.com/c/talkingdata-adtracking-fraud-detection/data%20) )

Each row of the training data contains a click record, with the following features.

* IP: IP address of click.
* app: app id for marketing.
* device: device **type** id of user mobile phone (e.g., iPhone 6 plus, iPhone 7, Huawei mate 7, etc.)
* os: os version id of user mobile phone
* channel: channel id of mobile ad publisher
* click\_time: timestamp of click (UTC)
* attributed\_time: if user download the app for after clicking an ad, this is the time of the app download
* is\_attributed: the target that is to be predicted, indicating the app was downloaded

Note that IP, app, device, os, and channel are encoded.

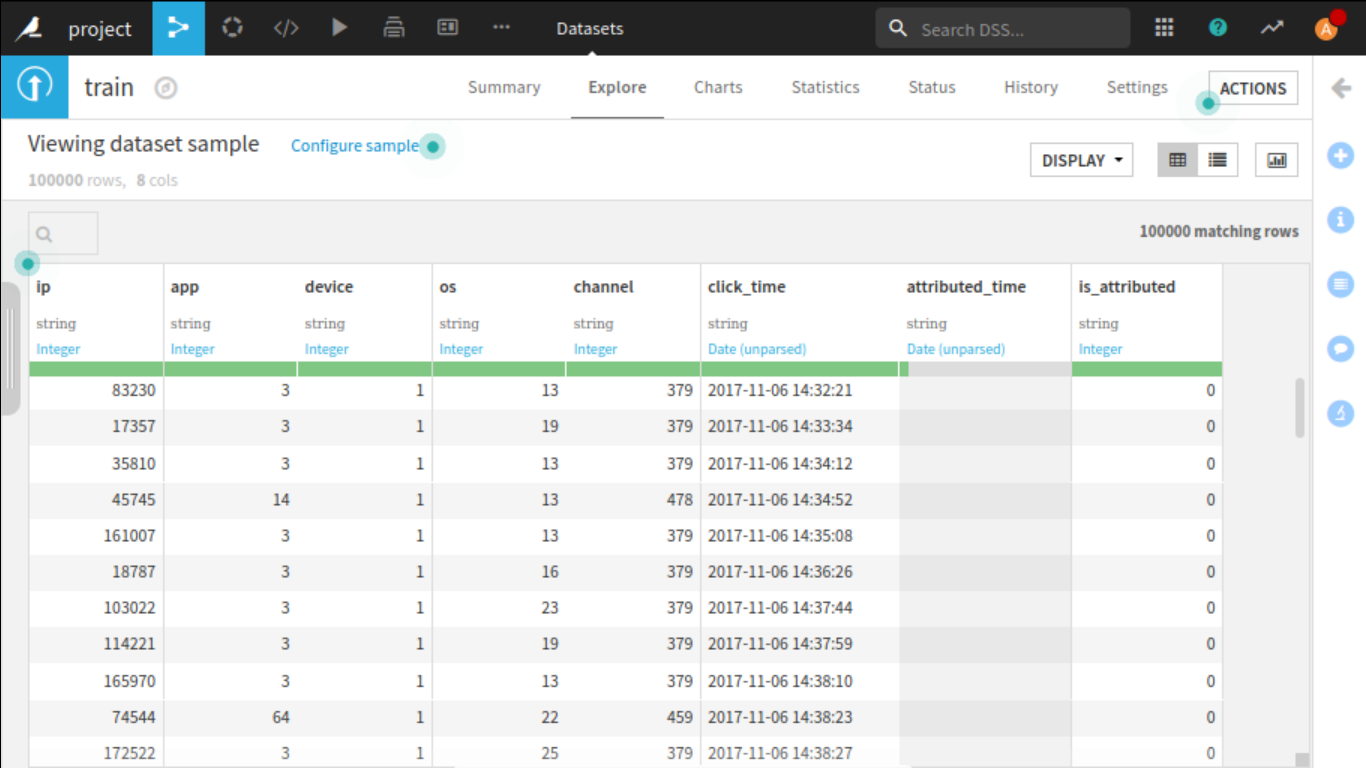
The test data is similar, with the following differences:

* click\_id: a reference for making predictions
* is\_attributed: not included

Our task was to build a machine learning model using **Dataiku DSS**(Data Science Software) which is a collaborative data science software platform for teams of data scientists, data analysts, and engineers to explore, prototype, build and deliver their data products more efficiently.

**Loading dataset in Dataiku**

This is how our dataset looked like:-

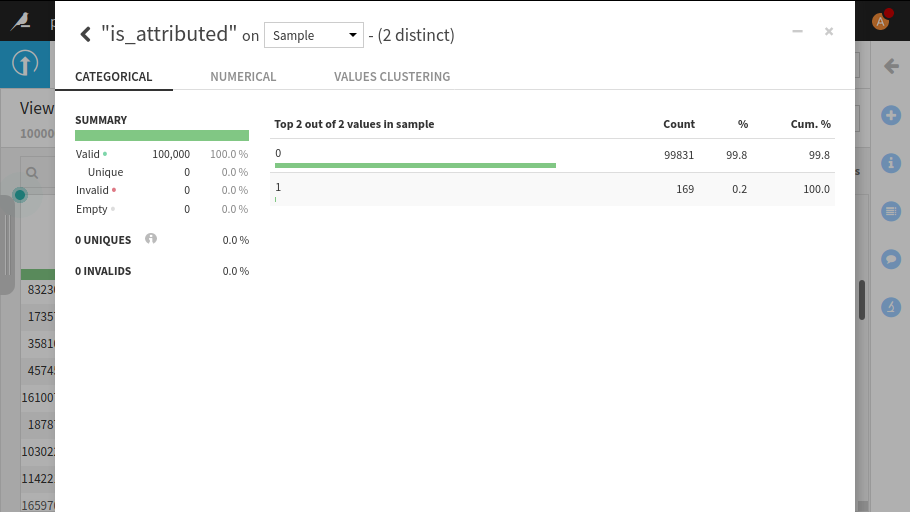


The actual train dataset was very large around **7GB** in size containing 180 million entries. With our systems it was not feasible to work in such big dataset.

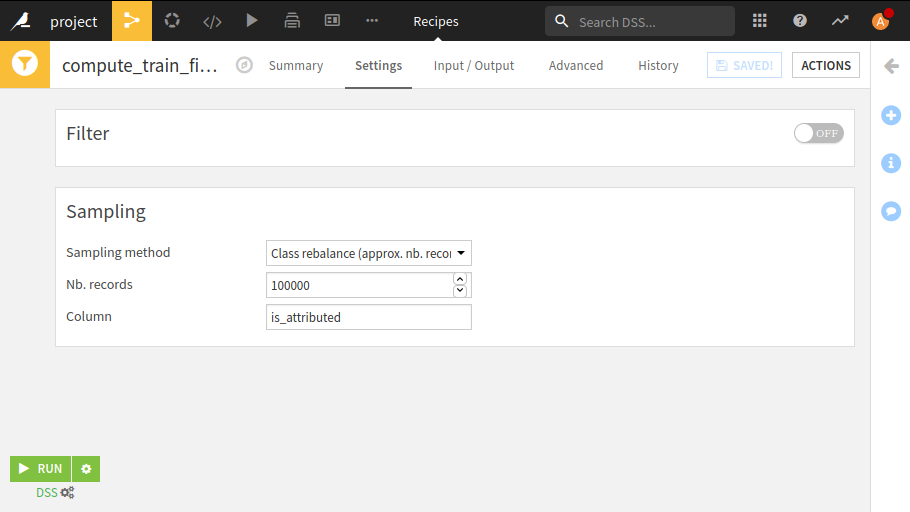
So, we worked only with 100k entries considering our system capabilities. Although the model is completely scalable and can be trained with a larger dataset in future.

**Basic Exploratory Data Analysis:-**

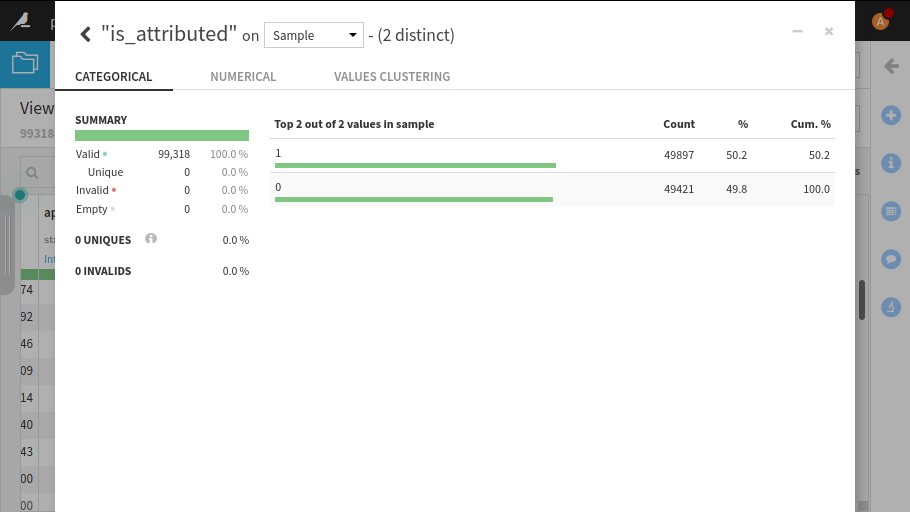
It can be seen how skewed our dataset was from a first look. It was very highly unbalanced and biased towards the 0 value that corresponds to no attribution.



Our task was to balance the dataset for creating a more accurate model which we achieved using Dataiku sampling recipe.

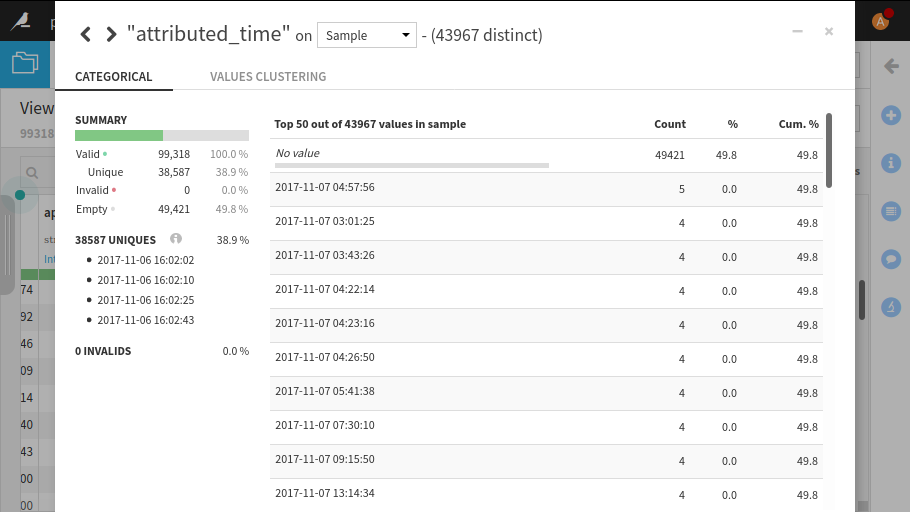
****

We used class rebalance sampling method for balancing the is\_attributed column and for filtering out 100k such entries from the dataset.



After performing sampling we got a new balanced dataset with almost equal values of 0 and 1 in is\_attributed column.

Analysis of attributed\_time showed that it contains mostly null or no values and hence was not helpful for prediction.



So, this feature was dropped and was not included for model creation.

**Feature Engineering:-**

Dataset had some features through which certain properties could be extracted. It had click\_time column through which date components were selected.

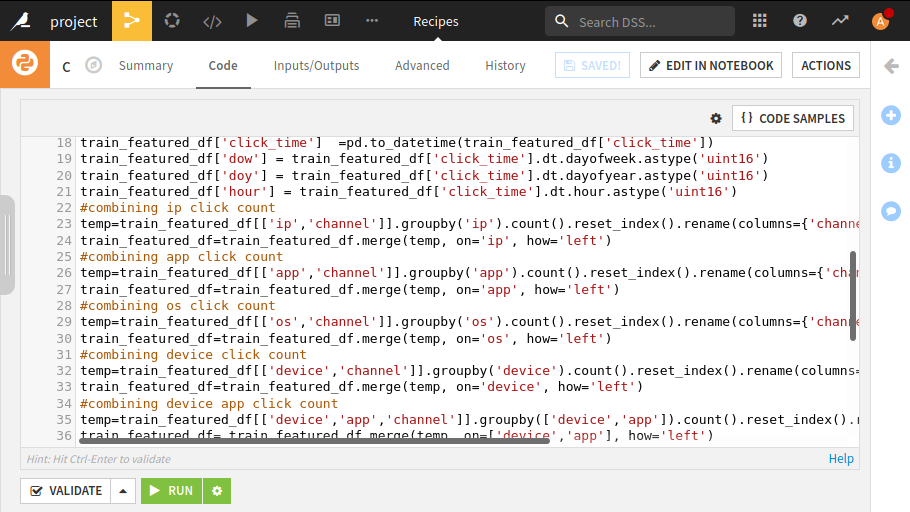
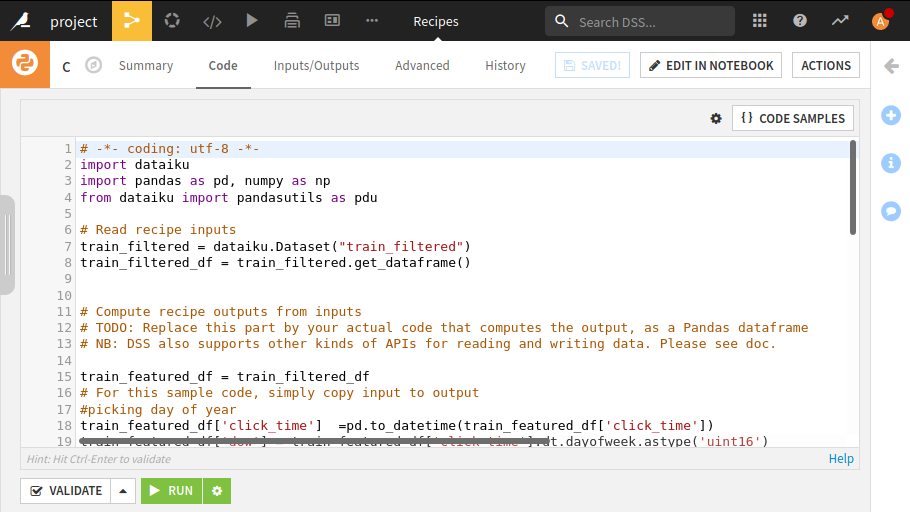
Date Components selected were:-

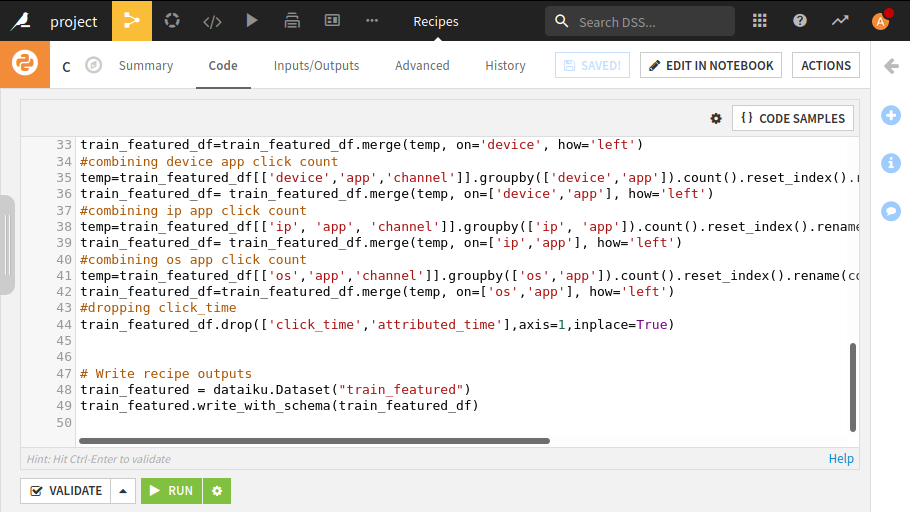
hour,doy(day of year),dow(day of week)

Groupby Aggregations done are by click count of:-

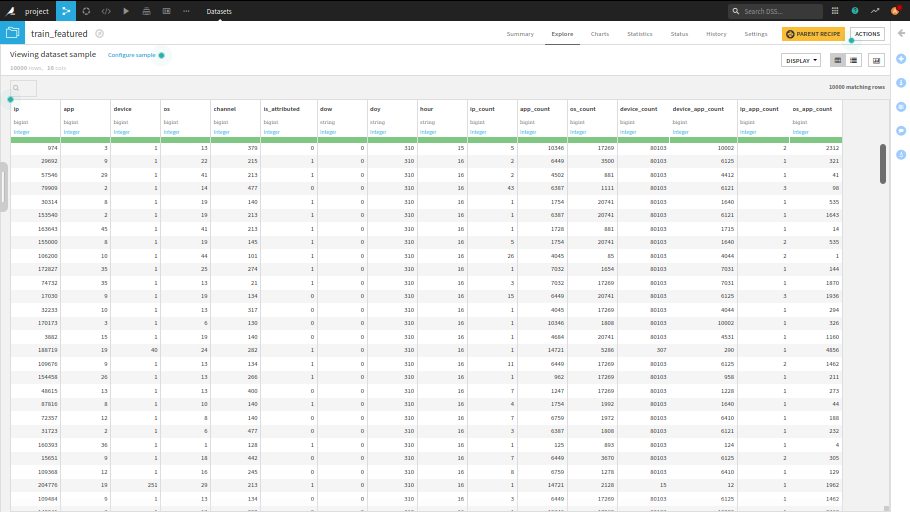
ip, os, device, app , ip and app, os and app, device and app

The python code recipe used was:-





After running this code the dataset looked like:-



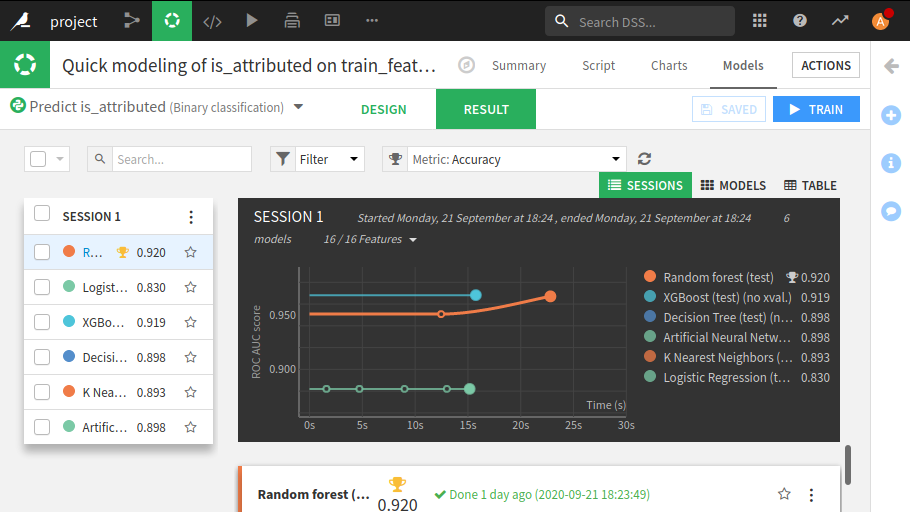
This dataset was now ready to be served to ML models.

**Choosing Model**

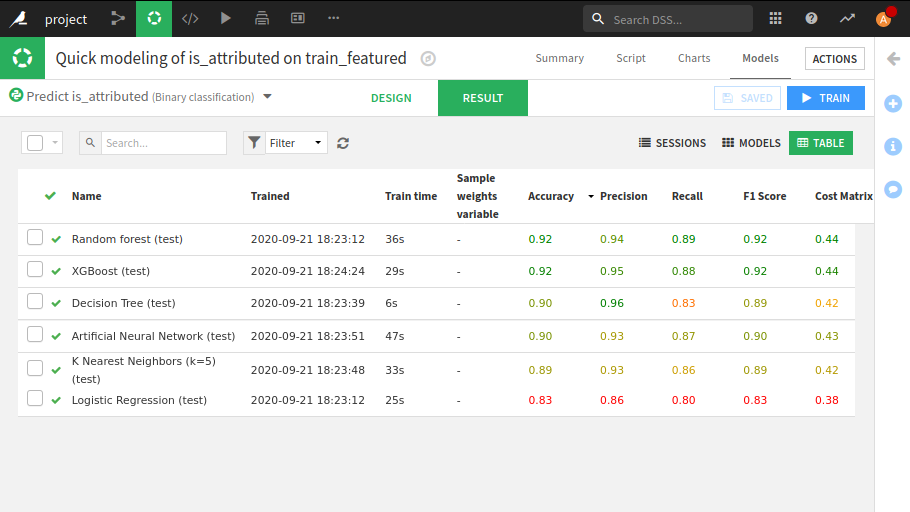
For building model, we took the advantage of Dataiku DSS auto ML capabilities. We chose 6 different classifiers for building our models which are:-

* Random Forest Classifier
* Logistic Regression
* XGBoost Classifier
* Decision Tree
* K Nearest Neighbor
* Artificial Neural Network

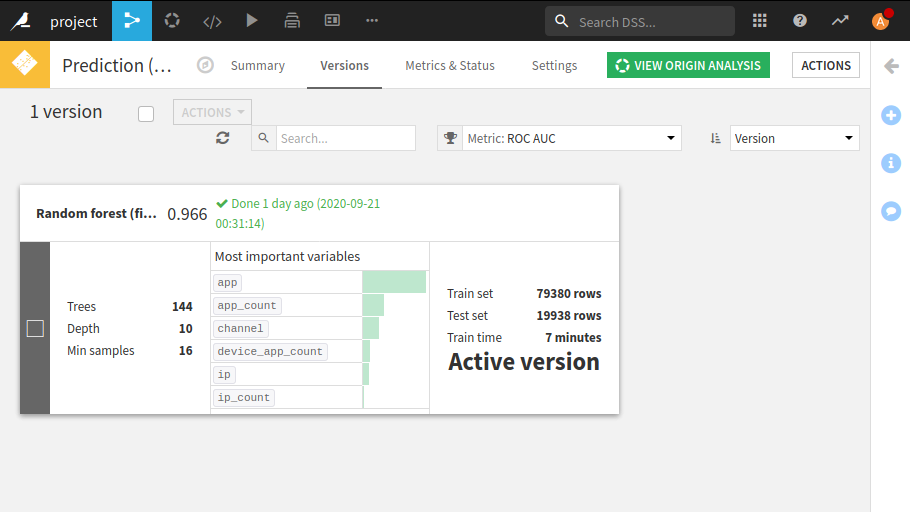
The training looked like:-



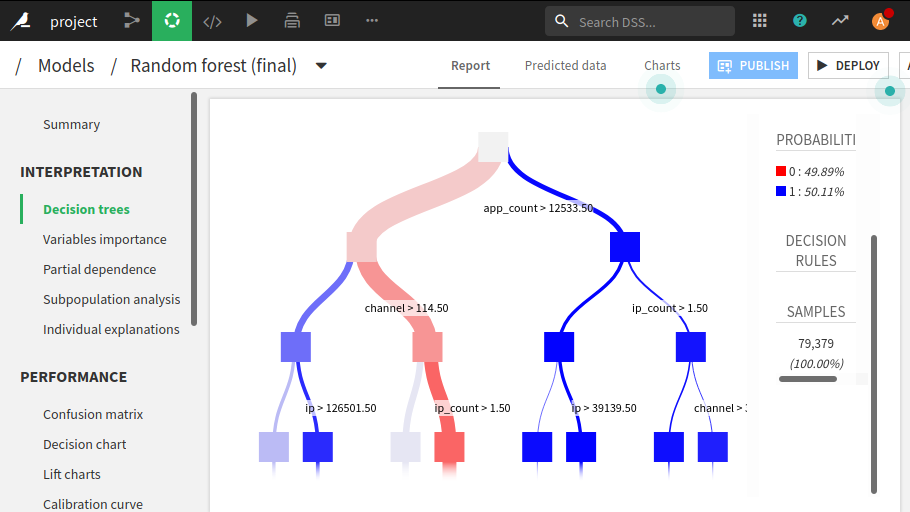
We got the best results for Random Forest Classifier, detailed metrics of all the classifiers as:-



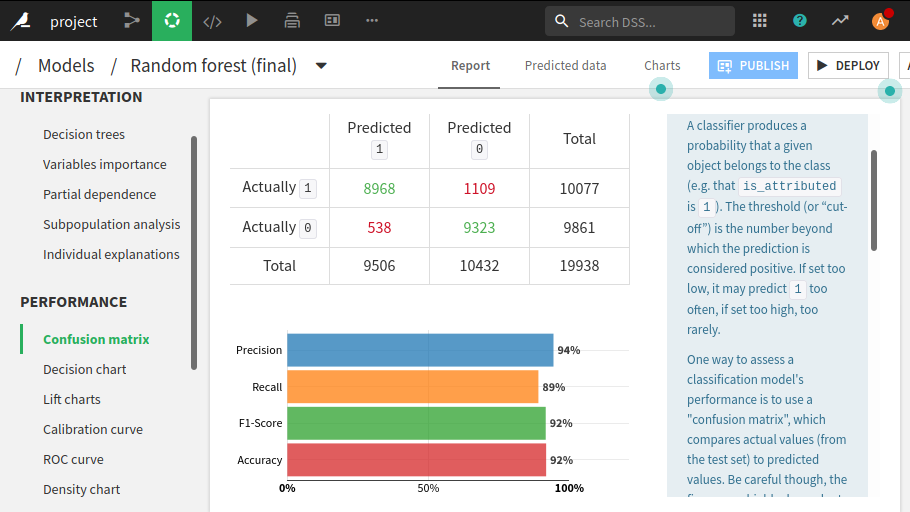
So, we chose Random Forest Classifier for our final model.



**Random Forest Model Decision Tree:-**

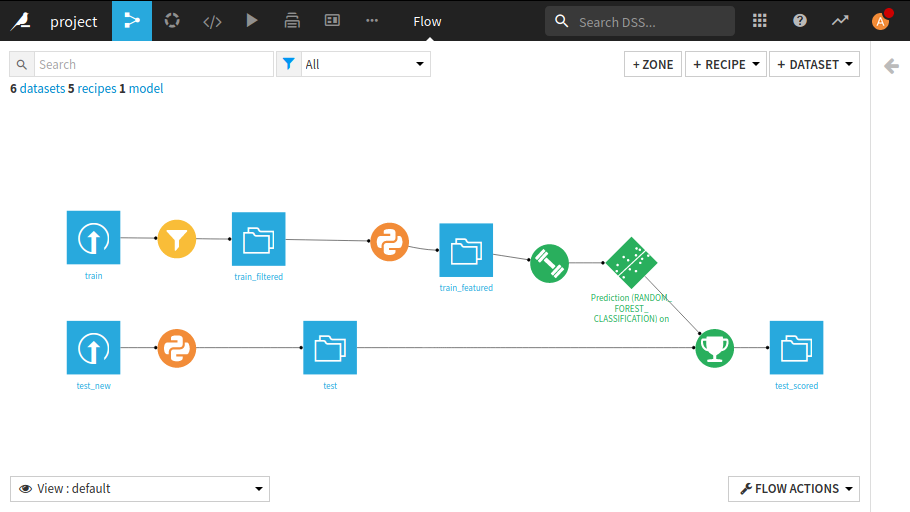


**Confusion Matrix and Detailed Metrics:-**

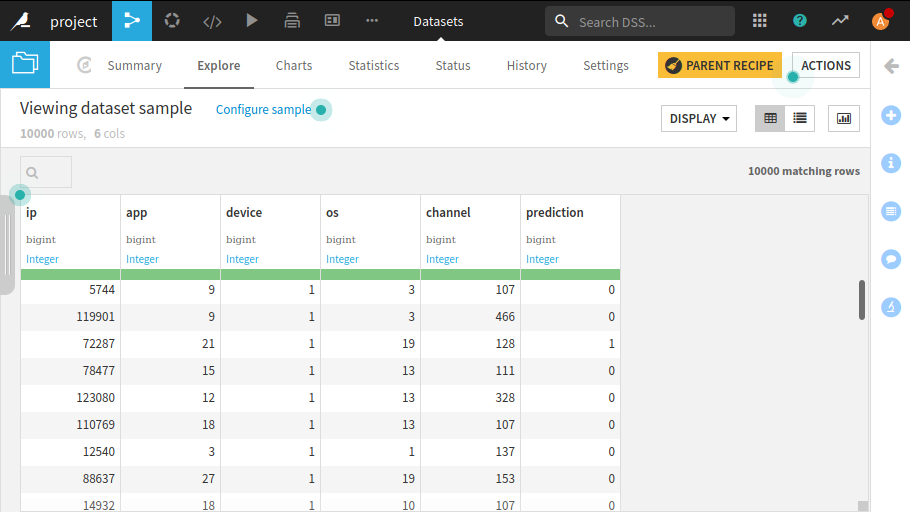


**Final Flow:-**

Our final flow after test predictions looked like this:-



**Test Dataset predictions:-**

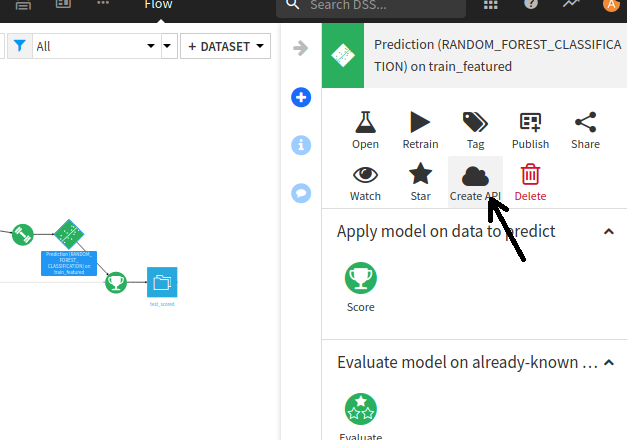


**How to use this model as API?**

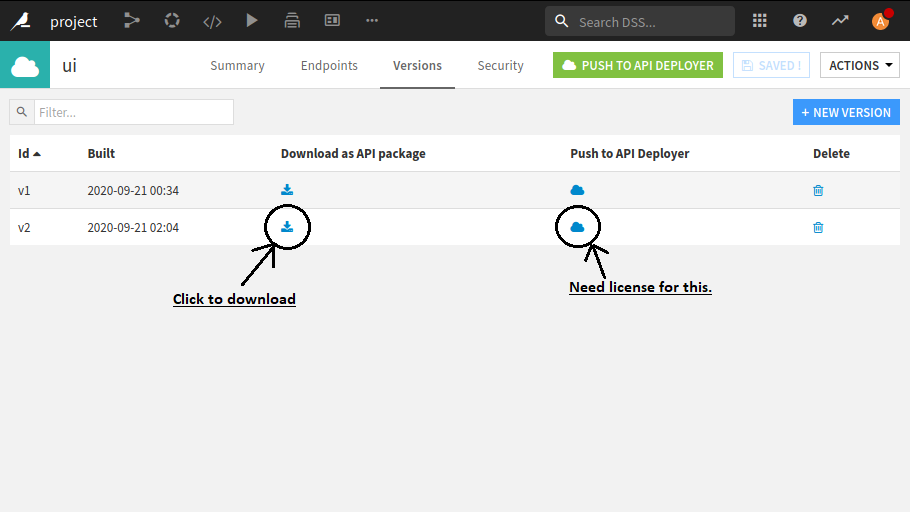
For using Dataiku API capabilities we must have DSS License. So, we had to figure out a way through which we could use this model for predictions outside Dataiku DSS.

**Steps involved were**:-

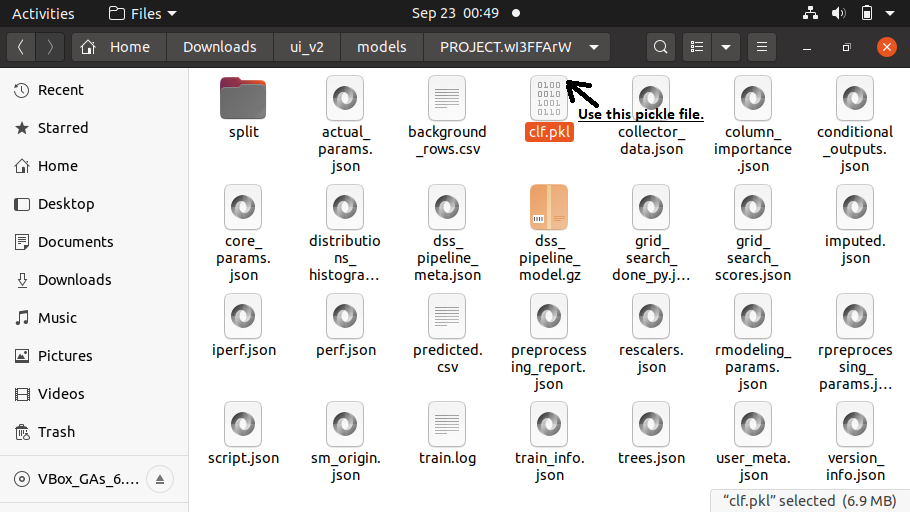
1. Click the create API option.



2. Download it as an API package; we could not use API deployer because of license limitations.



3. After downloading it, we found a pickle file. This file can be used to make predictions for the test dataset outside DSS.



Flask API Documentation

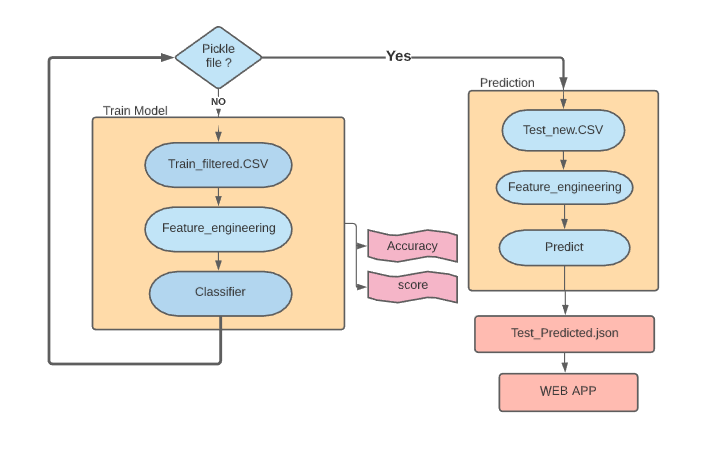
# Introduction

**Flask** is a micro web framework which has a minimal footprint compared to others like Django and allows us to select which components and modules we want to integrate with it. Flask is highly reliable and performant.

The Flask Restful was chosen on the suggestion of Mentor when our team was unable to get an API from Dataiku

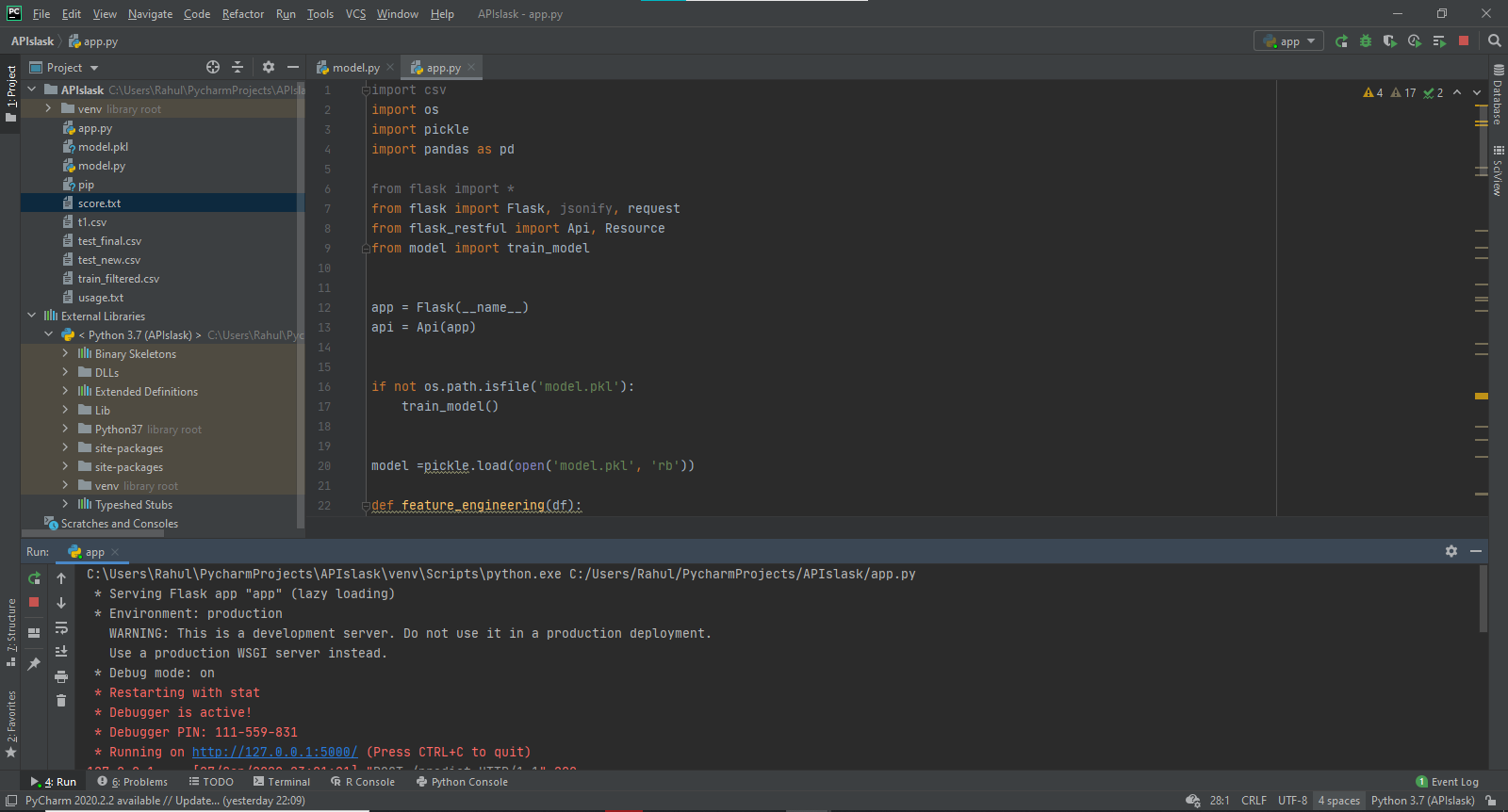
# Project Layout

Project layout looks like the following folder structure.



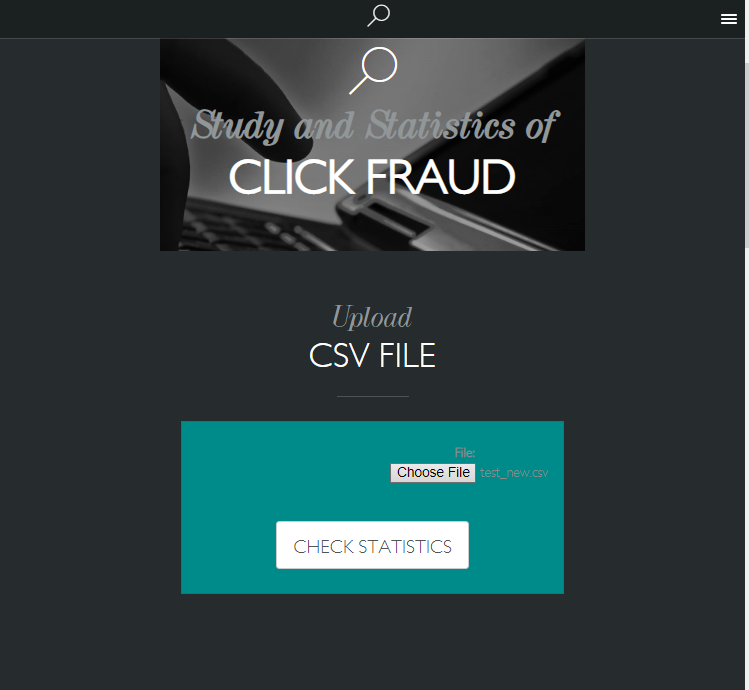
API module will host our application code, from models-

* App.py serves as the main file and hosts the server whereas models.py serves as the backend for the training.
* App.py has 3 classes each serving separate calls to provide a different response to calling application based on request
* Train\_filtered.csv is the dataset used for training the module
* Test\_new.csv is created from the received post request
* Usage and Scores file contains the data about the system resources

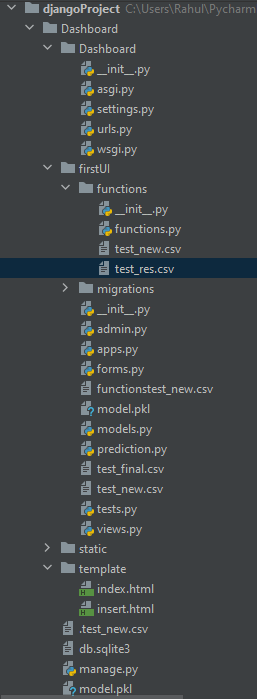


**Dashboard**

Django based web app to upload a csv file and give visual analytics on the given file



File Structure of the project



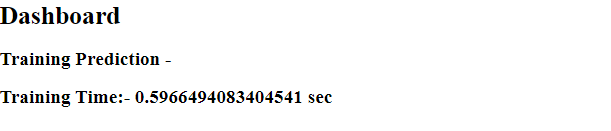
The dashboard can be started using manage.py

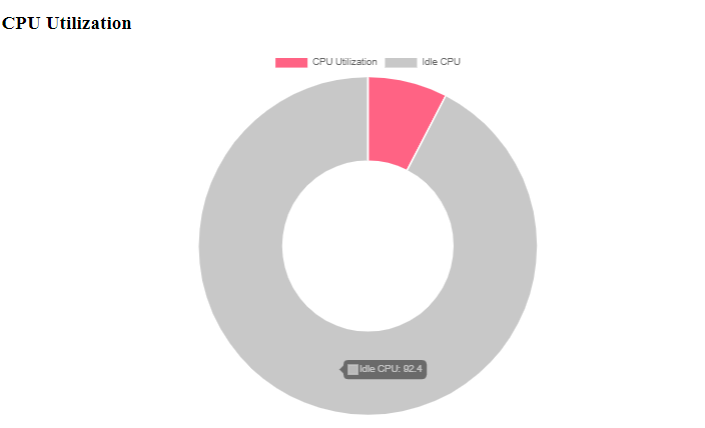


The function in View.py will send requests to API server and get resultant files which can then be used to visualize the result

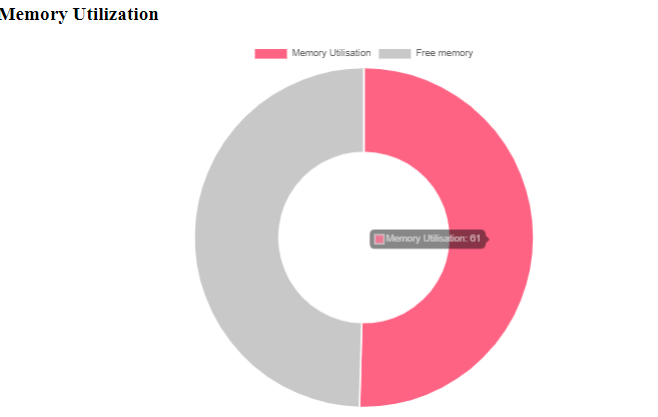
**Data Visualization**

The next page shows the training time for the model

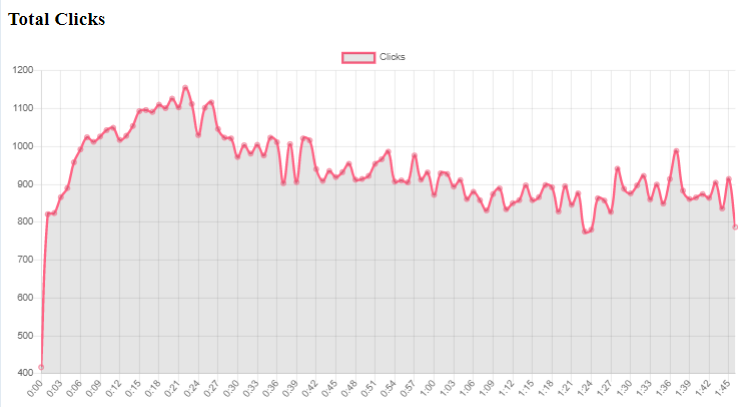
****

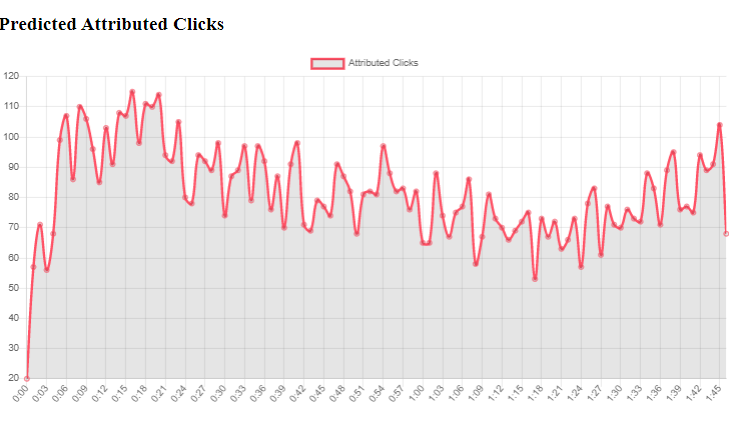
****

This doughnut shows the CPU Utilization while running the server in percentage

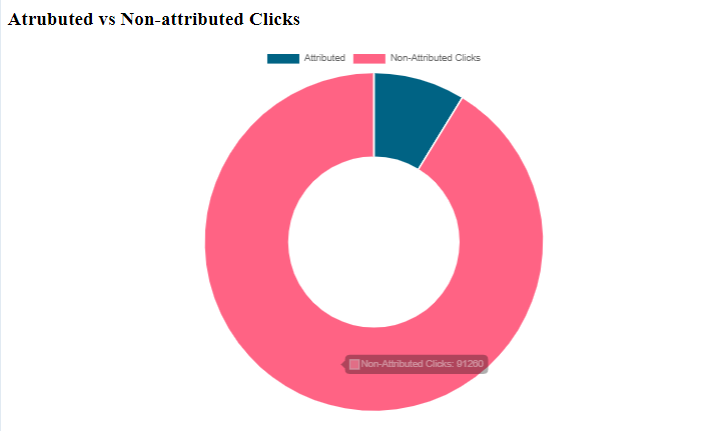


This doughnut shows the Memory Utilization while running the server in percentage

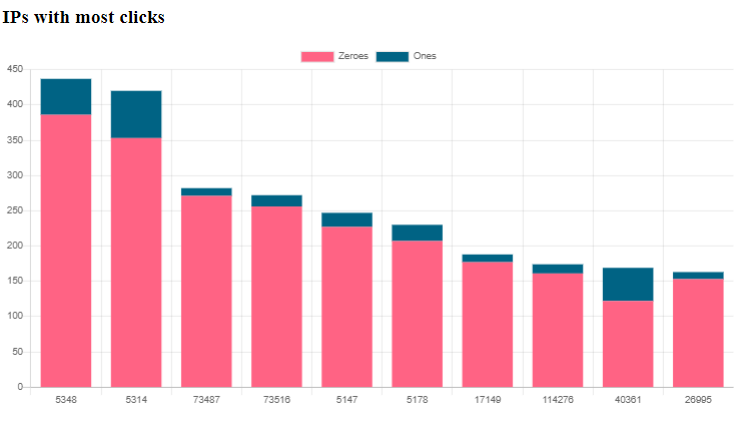
Number of clicks with time on x-axis



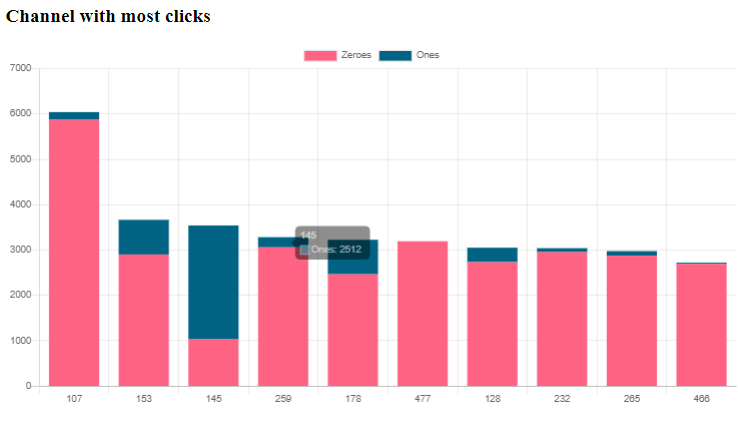
Number of predicted Attributions



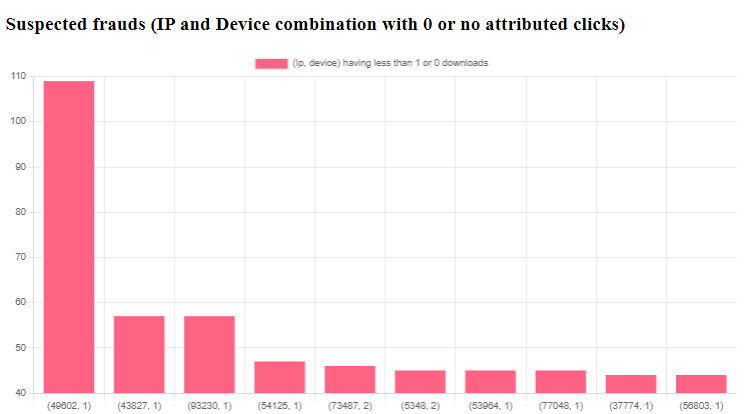
Attributed to non attributed clicks



IPs with the most number of clicks



Channels with the most number of clicks



Suspected Frauds with 0 or 1 attributions