Lookalike Modelling

Business Objective

Let’s say you wish to generate brand awareness of your product and increase its sale; online advertising is the easiest way to do it. Online advertising is considered one of the most effective and efficient ways for businesses of all sizes to expand their reach, find new customers, and diversify their revenue streams.

Online advertising is the art of using the internet as a powerful platform to deliver marketing messages to an intended audience. Social media is one the most popular online pastimes for people worldwide, and advertisers have evolved their strategies to target consumers on social media sites like Facebook, Instagram, Twitter, etc. It helps in attracting website traffic and brand exposure, which in turn helps in increasing sales. Hence, online adverting is designed in such a way as to persuade the targeted customer to make a purchase.

But, when these ads get advertised, not all people will be interested in the product/service, and only those interested click on the ads. Click rate is a term that helps us find the percentage of people who have watched your ad online and have ended up clicking it. Hence the goal of online advertising is to reach maximum and relevant users.

So, to find these relevant users/customers, we make use of the lookalike model.
Lookalike models are models used to build larger audiences from smaller segments to create reach for advertisers. The more significant users reflect the benchmark characteristics of the original users, which are known as the seed users.

In this project, we will build a Lookalike model that will help us find similar and relevant customers with the help of the LSH – Locality Sensitive Handling algorithm. This algorithm will hence improve the click rate.

Data Description

The dataset used is from a company called ‘Adform’. This dataset is from a particular online digital online campaign. This ad was shown to several thousand people and recorded whether the people clicked the ad or not. As the dataset is vast, only a subset of the dataset was considered. Following is the description of the data:

The file is gzipped and each line corresponds to a single record, serialized as JSON. The JSON has the following fields:

- "l": The binary label indicating whether the ad was clicked (1) or not (0).
- "c0" - "c9": Categorical features which were hashed into a 32-bit integer.
- Here, c6 and c9 have multiple values per users and the rest have single values per user.

To know more about the dataset, open the following link:

https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/TADBY7

Aim

To build a lookalike model to find similar users using the Locality Sensitive Hashing algorithm and find an increase in click rate w.r.t the default click rate.

Tech stack

- ➢ Language - Python
- ➢ Libraries - scikit-learn, pandas, numpy, pickle, yaml, datasketch

Approach

1. Importing the required libraries and packages
2. Open the config.yaml file. (This is a configuration file that can be edited according to your dataset)
3. Read the json file
4. Clean the json file
    - Reset index in the data
    - Convert list to integers
    - Remove rows above certain threshold values
    - Replace empty values with an empty list if any
    - Store the cleaned file
    - Calculate feature counts for scoring
5. Model Training
    - Create a MinHashForest Model
    - Create an LSH graph object
    - Train the model
    - Save the model to a pickle file
6. Seed set Extension
    - ▪ Read the saved model
    - ▪ Read the seed set data
    - ▪ Retrieve the neighbours of seed set from LSH graph

- Calculate the default click rate
- Score the neighbours
- Create and store the extension file
- Find the increased click rate.

Modular code overview

```
input
   |_config.yaml
   |_data.json
   |_processed.json
   |_count_df.csv
   |_seed.csv


src
   |_engine.py
   |_ml_pipeline
              |_processing.py
              |_model.py
              |_score.py
              |_utilis.py


lib
   |_lookalike.ipynb


output
   |_extn.csv
   |_Ishgraph
```

Once you unzip the modular_code.zip file you can find the following folders within it.

1. input

2. src

3. output

4. lib

   1. Input folder **-** It contains all the data that we have for analysis.
      - A config file, with some basic configuration parameters which can be edited according to your dataset.

    - There is a json file that contains the main data
    - The processed json file (after cleaning the data)
    - The count_df.csv file **-** Calculates feature counts for scoring
    - The seed csv – the seed set users that responded in the past by clicking on the ad.
2. Src folder - This is the most important folder of the project. This folder contains all the modularized code for all the above steps in a modularized manner. This folder consists of:
    - engine.py
    - ml_pipeline

      The ml_pipeline is a folder that contains all the functions put into different python files which are appropriately named. These python functions are then called inside the engine.py file.
3. Output folder **–** The output folder contains the best fitted model that we trained for this data. This model can be quickly loaded and used for future use and the user need not have to train all the models from the beginning. The output folder also contains the extension.csv file. This file the users after scoring the neighbours.

   Note: This model is built over a chunk of data. One can obtain the model for the entire data by running engine.py by taking the whole data to train the models.

4. Lib folder **-** This is a reference folder. It contains the original ipython notebook that we saw in the videos.

Project Takeaways

1. Understanding the business problem.
2. Understanding the concept of lookalike modelling and click rate.
3. Understanding the idea behind LSH – Locality Sensitive Hashing.
4. Calculation using Jaccard similarity on LSH.
5. Understanding what seed set is and how to form seed set users
6. Understanding model evaluation.
7. Understanding ranking users using feature importance.
8. Understanding how to perform scoring to filter out candidates.
9. Importing the dataset and required libraries.
10. Performing basic Data Cleaning using appropriate methods
11. Using python libraries such as numpy, pandas, datasketch etc
12. Creation of config files.
13. Training a model using the MinHashLSHForest algorithm.
14. Creating an LSH Graph object.
15. Performance metric – Calculating the default and improved click rate.
16. Generating extending seed set.