Demographic (Age, Gender) Prediction based on Mobile App Usage

**Predictors:**

* Revanth dattuluri
* Srujith guduri
* Kun zhou

Graphical user interface, application

Description automatically generated

Table Of Contents:

1.Data Source

Background Research

1. Market-Value Estimates
2. App Revenue Data

2. Data Quality & Description

3. Data Cleaning

4. Data Joining

5. Architecture

6. ETL

7. Data Preparation

8. Results

9. References

Data Source

Talking Data Icon

Description automatically generated

Founded in 2011, TalkingData is China's leading third-party data intelligence solution provider. With Smart Data Platform as the core of its data intelligence application ecosystem, TalkingData empowers enterprises and helps them achieve a data-driven digital transformation.

In the last seven years, TalkingData's vision of using "big data for smarter business decisions and a better world" has allowed it to gradually become China's leading data intelligence solution provider.

TalkingData creates value for clients and serves as their "performance partner," helping modern enterprises achieve data-driven transformation and accelerating the digitization of clients from various industries. Using data-generated insights to change how people see the world and themselves, TalkingData hopes to ultimately improve people's lives.

Background Research

1.Market-value estimates

Number of smartphone subscriptions worldwide 6.4bn, globe revenue from smartphones sales

409bn USD, globe revenue from smartphones sales 409bn USD.

Despite the recent slow-down for the constant market growth for years, China has remained

the world’s largest smartphone market since 2012. In 2020, [**smartphone shipments in China**](https://www.statista.com/statistics/387046/smartphone-shipments-in-china/)

reached over 325 million units, accounting for about 25 percent of the total volume of global

smartphone shipments. The [mobile phone subscriptions](https://www.statista.com/statistics/278204/china-mobile-users-by-month/) as of February 2020 have already

reached about 1.59 billion in China.[(Ref)](file:///C:\Users\revan\Downloads\1.https:\www.statista.com\topics\1416\smartphone-market-in-china\)

Chart, bar chart

Description automatically generated

Globe market share of Chinese smartphone, huawei alone 14.6%. Number of mobile phone subscriptions per 100 inhabitants in China from 2009 to 2020. The mobile phone service in China is provided by three [**domestic telecommunication network operators**](https://www.statista.com/statistics/291795/china-mobile-subscribers/), namely China Mobile, China Unicom, and China Telecom.

Why are we limiting only to apps usage?

With over [**3.2 billion smartphone users**](https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/) across the world, it’s no surprise that the mobile app industry is thriving. App usage and smartphone penetration are still growing at a steady rate, without any signs of slowing down in the not-long future.

Now factor in the [**1.14 billion tablet users**](https://www.statista.com/statistics/377977/tablet-users-worldwide-forecast/) worldwide, which a number that’s grown about 36% over the past six years.

What’s everyone doing on their phones? Well, [**88% of mobile time**](https://www.emarketer.com/content/the-majority-of-americans-mobile-time-spent-takes-place-in-apps) is spent on apps.

Chart, bar chart

Description automatically generated

If you want to compete and claim your share of this multi-billion dollar industry, you need to have a better understanding of exactly how people are downloading and using mobile apps.(ref:<https://www.appinchina.co/blog/app-purchase-revenue-share-in-china/>)

China, with total app revenue at close to $40 billion, accounts for nearly 40% of global app revenue by this reckoning – around double US 2018 app revenue Chinese app revenue has grown by 140% since 2016 Sensor Tower H1 2019 app revenue data shows total revenue of $39.7 billion; this reflects a 15.4% increase over 2018.

2.App revenue Data

(ref: [**https://www.businessofapps.com/data/app-revenues/#:~:text=China%2C%20with%20total%20app%20revenue%20at%20close%20to,this%20reflects%20a%2015.4%25%20increase%20over%20H1%202018**](https://www.businessofapps.com/data/app-revenues/#:~:text=China%2C%20with%20total%20app%20revenue%20at%20close%20to,this%20reflects%20a%2015.4%25%20increase%20over%20H1%202018))

The value of our research lies on the existing huge revenue out there on the market. If we can predict the demographics of a user (gender and age) based on their app download and usage behaviors, we can almost predict more needs of the people behind the usage of the apps, giving them the recommendation. not only the apps but also the biomatrix-related needs like medical care needs. Talkingdata is one of the companies that would like to provide that convenience. [TalkingData](https://www.talkingdata.com/), as China’s largest third-party mobile data platform, is seeking to leverage behavioral data from more than 70% of the 500 million mobile devices active daily in China to help its clients better understand and interact with their audiences.

Dataset descriptions and data quality

Full recognition and consent from individual user of those apps have been obtained, and appropriate anonymization have been performed to protect privacy. Due to confidentiality, they didn’t provide details on how the gender and age data was obtained. Please treat them as accurate ground truth for prediction. I think for academic study practice, it’s not a concern for time being.

After study business case, the relevant logic relationship should be the as following:

* gender\_age\_train.csv, gender\_age\_test.csv - the training and test set
* group: this is the target variable you are going to predict
* events.csv, app\_events.csv - when a user uses TalkingData SDK, the event gets logged in this data. Each event has an event id, location (lat/lon), and the event corresponds to a list of apps in app\_events.
* timestamp: when the user is using an app with TalkingData SDK
* app\_labels.csv - apps and their labels, the label\_id's can be used to join with label\_categories
* label\_categories.csv - apps' labels and their categories in text
* phone\_brand\_device\_model.csv - device ids, brand, and models
* phone\_brand: note that the brands are in Chinese

Diagram

Description automatically generated

**gender\_age:**

* device\_id(PK, int)
* gender (char)
* age (int),
* group(int)

**phone\_brand\_device\_model:**

* device\_id(PK, int),
* phone\_brand(char),
* device\_model(int)

**events:**

* event\_id(PK, int)
* device\_id(FK, int)
* timestamp(timestamp)
* latitude (float)
* longtitude (float))

**Label\_categories**:

* label\_id(PK, int)
* category(char)

**app\_labels:**

* app\_id ( comp\_PK, FK, int)
* label\_id (comp\_PK, FK, int)

**app\_events**:

* event\_id(comp\_PK, FK, int)
* app\_id(comp\_PK, FK, int)
* is\_installed(boolin)
* is\_active(boolin)

Data quality

Generally speaking, this dataset is not ready for directly use but useful. After back and forth visualization and cleaning, it has normal not available ( NA) and duplication, some need logically flat or compound into new feature. The total records for training are74,645 and there is no duplicate in training dataset but here is NA in it. Now, looking back, there is still enough data for later research.

Diagram

Description automatically generated

Shortly speaking, that we start with a tangible handset and end with the handset’s owner’s bio-information,  gender and age. Bio-info is more hard to collect than handset information, while the bio-info brings a huge market needs from medical business to political strategy. For that purpose, we need cleaning all the six tables one by one.

In this research, we just narrow down to 12 group for simple, eg M32-38, which means the owner is a male aged from 32 to 38. In order to synchronize these two tables gender\_age and phone\_brand\_device\_model, all the intermediate tables should be treated as they need data wrangling. We found that the there is duplicates in the events table. There is same device\_model name for different brand handset. This needs to make new columns for specific band and that specific model. For example: brand Huawei and Samsung may have the same model named M2. That way, we found out there are 54 entries with the same model name from different phone brands.

For better understanding, I listed the code and output.

**gender\_age table:**

gender\_age table is the training dataset, already having the label (dependent variable: group) in it. There is no duplicate or missing or other odd data in it. It looks like this:

Graphical user interface

Description automatically generated with medium confidence

After Exam the total entry and unique of devide\_id, there is no duplicate. The reports shows like this:

Graphical user interface

Description automatically generated with medium confidence

**phone\_brand\_device\_model:**

For short, we call this table phone\_data. It looks like this:

Word

Description automatically generated with low confidence

Obviously, there is no duplicate. But the common sense told us there might be different phone\_brand have the same device\_model name as well as the one phone\_brand has multiple device\_models. We need to handle this concern while merge the tables.

**events table:**

when we inspect the event table by events.event\_id.nunique(), events.device\_id.nunique(), events.shape[0]

, the result is : (3252950, 60865, 3252950)

Which indicate device(device\_id) to events is one to many. We already know that there are 186716 device ids in training dataset alone, 3252950 when training and test combined together.   However, in the events table, there are only 60865 devices registered. That means the loss is 67.4%. So, we have to use the PhoneBrand. For that purpose, we have to come back to exam the completion of device\_id in test dataset. If it's yes, we go back to delete the non-related device ids and their related events.

Events table is the center joint for all other tables. The original table has missing timestamp and doesn’t have hour and night\_active. For real world analysis, after add columns hour and night\_active, Event looks like this:

Graphical user interface

Description automatically generated with medium confidence

Some of them have missed the timestamp, some of them missed longitude or latitude.

For event alone, it tells how’s people’s individual life which after hour at night. The longitude and latitude give us information how the geolocation of these events. If it links with gender\_age tables, it also show different gender and ages behavior patterns.

**Label\_categories tables**:

Here categories is the apps category, with label\_id to specified it and later use the Tfidf to transferred into math matrix. We can tell there are 930 entities while only 835 unique, which no need to drop NaN and duplicates for practice at this point.

Graphical user interface

Description automatically generated with low confidence

**app\_labels table:**

The original table look like this

Graphical user interface, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Label\_id actually is the categorical data in numerical format. Here need to convert it into catagolical. But there is a technique issue that one device (one device\_id)  has multiple events, one events has multiple apps (active or not), one app has multiple categories to belong to. Flat all of them, we need treat the app category( label\_id) as natural terms. Using TfidfVectorizer to turn lable\_ids under one devide\_id into math matrix, otherwise can’t calculate by computer. After conversion, it looks like the last form of the following picture. If the device( device\_id) has that categorical app, the value of that cell will be 1 otherwise it will be 0.

Graphical user interface, application

Description automatically generated

**app\_events table**:

A picture containing application

Description automatically generated

One event has multiple applications(app\_id).

We need to collect all the app category for one event , the new column is app\_label and map them into new categorical columns for later dummy them up to do math transfer.

Graphical user interface, text, application

Description automatically generated

We mapped all the app and their categories and the phone\_brands and it’s models into a superwide table( dataframe). All feature are ready for math calculations. It’s not only NA or missing revomed but all categorical variables were transferred into dummy.  It look like this:

A picture containing screenshot, different, line, several

Description automatically generated

**Data Cleaning:**

Removed duplicate device\_ids from phone brand table after duplicate entries were found wrt device\_id, brand\_name, model\_name combination. All other tables data was found to be clear of issues except Chinese characters of phone brands which had to be translated using key pairs and transformed in table.

**Data Joins:**

Joins were performed wrt gender\_age\_test and train tables with device\_id as primary key on Events and phone\_brands table.

Resulting table is joined with App\_data and App\_categories table to get categorical data along with is\_active flag.

**Architecture:**

We are getting the data from Kaggle to the Google Drive,finally through Google colab

**Following the steps below to download and use kaggle datasets in Google Colab:**

Downloading Kaggle data to Google drive and readily using it in Google Colab

Icon

Description automatically generated

Kaggle is the world's largest data science community with powerful tools and resources to help you achieve your data science goals.

Google Colaboratory is a free platform (environment might be a better word though!) for programmers to do coding! Beyond that, it is in essence developed to facilitate Machine Learning and Deep Learning research by providing free GPU and TPU resources!

In Google Colab:

* You have access to a free GPU with limited runtime!
* You write your code in a nice ready-to-use notebook.
* Installing new packages is very easy!

Google Colab basically provide descent computation resources for whoever around the world that

* desire to do Machine Learning and
* have a Gmail account!

Taking about Machine learning, Deep Learning, or NLP require quite serious processing. Many people tend to use Google Colab for model training since the GPUs and TPUs available for free. One drawback of Colab is that the model no longer remains when quitting the notebook (unless you are the pro user). The solution is to move the workspace to Google Drive because it can be saved all information after leaving the notebook. The download and upload are consuming time.

A picture containing text, clipart, vector graphics

Description automatically generated

## How to directly download the Kaggle dataset to Google Drive?--- Everything will be on the cloud for sure!

Follow the below 6 easy steps:

1. Create New API Token from Kaggle Account
2. Mount your Google drive to Google Colab
3. Configure environment on Google Colab
4. Get the Kaggle API
5. Use the Copied API command in Colab
6. Unzip the file to use it!

## 1. Create New API Token from Kaggle Account

Go to your Kaggle website. If you do not have a Kaggle account then first you need to create it. Once done open your account details, scroll down to the API section click on “Create New API Token”. Graphical user interface, text, application, email

Description automatically generatedGraphical user interface, text, email

Description automatically generated

Your web browser will ask you to download JSON file, named **kaggle.json**. The file contains the **Username** and the **Key** that will be used for config the Google Colab environment. After downloading it successfully, you will get the below message.



## 2. Mount your Google Drive to Google Colab

Use the below code to mount the Google drive. You may change the directory to some folder in your drive but remember dataset will be downloaded to that directory.

*from google.colab import drive*

*drive.mount("/content/drive")*

After executing the above two lines of code, it will ask you to go to a URL, click on it and sign in to your Google Drive account and copy the code.

Graphical user interface, application

Description automatically generated

Paste the copied code under ‘Enter your authorization code’.

Graphical user interface, text, application, email

Description automatically generated

After clicking enter, in some time you will get the following message:

Don’t forget to mount the Google Drive. Otherwise, your data will be lost anyway!

## 3. Configure environment on Google Colab

Now create the new notebook for the coding config environment. Insert a new cell and type the below code for configuring the environment.

Please note that the Username and the Key which you will get from kaggle.json are case-sensitive. You have to type exactly the same Username and the Key, otherwise, it will throw an error.

*#Configuration environment*

*import os*

*os.environ['KAGGLE\_USERNAME'] = "……………." # username from the json file*

*os.environ['KAGGLE\_KEY'] = "9f9d3c58210dda1dc770b62db060e304" # key from the json file*

Graphical user interface, text, application, email

Description automatically generated

## 4. Get the Kaggle API

We are now ready to use Kaggle API. To use it, go to the link we needed and find “three dots” somewhere on the right side of the page. Then, select **Copy API command**.

Graphical user interface, text, application, email, website

Description automatically generated

## 5. Use the Copied API command in Colab

Come back to Google Colab notebook. Insert the new cell and type **'!'**at the beginning followed by the API command.

*#Download hr-analytics-job-change-of-data-scientists data from Kaggle API*

*!kaggle datasets download -d arashnic/hr-analytics-job-change-of-data-scientists*

See the below image, you will get the message after successfully downloading the data files. You will get the size of the downloaded files also.

Text

Description automatically generated

## 6. Unzip the file to use it!

Use the below code on the next cell.

*Eg:- !unzip -q "/content/hr-analytics-job-change-of-data-scientists.zip"*

Copy the path of the file to want to read or perform any other operation.

A picture containing graphical user interface

Description automatically generated

**ETL**

we are getting the data from Kaggle, so the data source is of Kaggle

Data Source

[TalkingData Mobile User Demographics | Kaggle](https://www.kaggle.com/c/talkingdata-mobile-user-demographics/data)

Data:

Let's focus on “mobile usage data” here.

Diagram

Description automatically generated

As shown in the above diagram,7 files are provided where gender\_age is the train data and the other 5 files have the data of train and test devices. The 7th file is similar to gender\_age, containing only device ids and skipping the rest.

We have device\_id as our primary key indicating the device, which is linked with events data. app\_events, labels, and label categories are linked through event\_id,app\_id, and label\_id.

Transformations: -

We removed few device id’s that has repeated for the same app events. We have joined the app\_labels and app\_categories first on the label \_id as shown in the ER diagram, then we joined events and app\_events on event\_id . Also same with gender\_age and phone\_brand\_device\_model data sets as well.

Loading: -

Finally the data is saved into a Google Drive from where import those data files into Google colab by mounting. Where we go ahead to do further Analysis.

Data Preparation:

* BOW approach of phone brand (one-hot encoding, since these consist of different languages).

Graphical user interface, application, Teams

Description automatically generated

* BOW approach of phone model (one-hot encoding, since these consist of different languages).
* TFIDF Hour of the Event.

Text, letter

Description automatically generated

* mode/median of longitude(of all events and timestamps).
* mode/median of latitude(of all events and timestamps).
* TFIDF approach of apps used(active) by that particular device(challenges).
* BOW approach of labels of apps used(installed) by that particular device.
* TFIDF weekday the event has occurred.
* BOW Hour bin of the Event.
* The ratio of active apps and installed apps.
* Clustering of location into 10 clusters using latitude and longitude.

Results:

* Final table from the features created with help of one hot encoding, Bag of words & TF-IDF approaches resulted in a total of ~21k columns of numeric datatype.
* The table was then split into two different parts: one with events and one without events.
* We tried neural network using keras libraries with different results for the same hyperparameters when they were run two times independent of each other.
* This above behavior was a result of variance neural networks exhibit commonly.

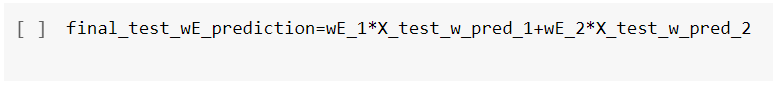
Graphical user interface, text, application

Description automatically generated

* It was then decided to train two neural networks with different parameters producing best possible results wrt logloss as seen below and combine the results to get an average based on CV logloss weightages.

Text

Description automatically generated



* Final prediction looks like below spreadsheet which can be uploaded to Kaggle for a final score.

Diagram

Description automatically generated

**References:**

* phone\_brand: Brands are in Chinese. Used translation provided by [fromandto](https://www.kaggle.com/zhaohao10).
* <https://towardsdatascience.com/cross-entropy-loss-function-f38c4ec8643e>
* <https://analyticsindiamag.com/guide-to-tensorflow-keras-optimizers/>
* <https://towardsdatascience.com/3-deep-learning-algorithms-in-under-5-minutes-part-2-deep-sequential-models-b84e3a29d9a8>
* <https://www.oreilly.com/library/view/tensorflow-for-deep/9781491980446/ch04.html>
* <https://towardsdatascience.com/natural-language-processing-feature-engineering-using-tf-idf-e8b9d00e7e76>
* <https://medium.com/@sewwandikaus.13/bow-vs-tf-idf-in-information-retrieval-a325b5e61984>