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

Revanth Dattuluri

Srujith Guduri

Kun Zhou

Graphical user interface, application

Description automatically generated

Contents

[1 Market value 1](#_Toc90072349)

[Why are we limiting only to apps usage? 1](#_Toc90072350)

[2 Task 1](#_Toc90072351)

[3 Who buys? Audience analysis 1](#_Toc90072352)

[4 Data source 1](#_Toc90072353)

[5 Dataset descriptions and data quality 1](#_Toc90072354)

[6 Approach choice 1](#_Toc90072355)

[7 modeling 1](#_Toc90072356)

[7.1 keras 1](#_Toc90072357)

[7.2 Model XGBoost . 1](#_Toc90072358)

## 1 Market value

Why we do this research? Because there is a huge revenue on the market and the manufacture power( the hareware) already exiting there. What we can do is to surf in the tide and make profit.

The 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

This highly centralized market gives us a chance to concentrate on one-size-fits-all business tragegy. 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 only 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?

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.

What about out of China? 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

What ‘s the App revenue?

(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>)

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.

Above all, we try to study on one Chinses case, to set up a business model to help solve the same problem in other regions of the world.

## 2 Task

Our task here is to predict the demographics of a user (gender and age) based on their app download and usage behaviors. .

## 3 Who buys? Audience analysis

For time being, the audience is professor Vijay Kumar, you, who’s a highly professional while pretending 3rd grader. So, we need include the information on: market value, the research task and analysis (plots for 3rd grader), the models we built and their relevant performance for a Ph.D CTO and results based on the data proof. We also use the Tw Cen MT fond to facilitate your reading.

## 4 Data source

The Data is collected from TalkingData SDK integrated within mobile apps. TalkingData serves under the service term between TalkingData and mobile app developers. It’s legal to collect these data in China because the data contributor eventually be financially rewarded by the policy via Alipay or Wechat pay, which are widely used back in China. TalkingData SDK itself was developed by a tencent company who also developed Wechat and QQ to social media back in China. The company Talking Data is just a branch company of tencent.

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.

## 5 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)

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 (dependant 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 PhoneBran. 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 timestap 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 longtitude or latitude.

For event alone, it tells how’s people’s individual life which after hour at night. The longtitude 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 dupicates 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 convert into mathematical matrix.

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 removed but all categorical variables( app category, time session, brand, devide\_model, etc) were transferred into dummy. It look like this:

A picture containing screenshot, different, line, several

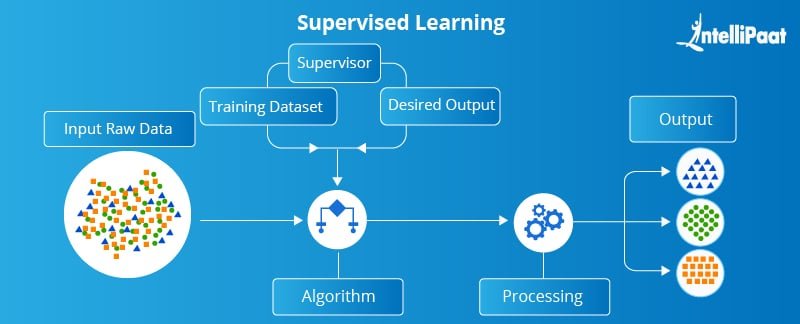
Description automatically generated

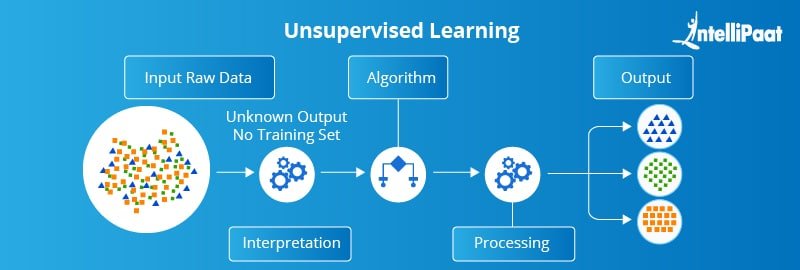
If we slide the bar we can see more dummy columns.

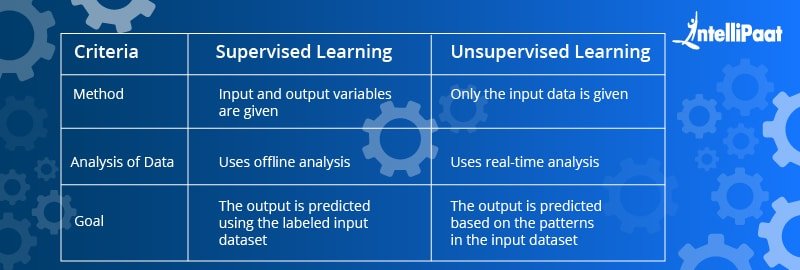
## 6 Approach choice

After cleaning the data and get the dataframe ready for models . We encounter a question: what approach we have on hand and which of them solve this practical problem?

The tools available for use are three: supervised modeling/learning, unsupervised modeling and reinforcement modeling (including deep reinforcement learning). For out case, we have already had the labeled variable named “group” as the output and the independent input is the device\_id, phone\_brand and etc.. Although we knew it’s a sophistic supervised learning case, we still want to discuss the three learning types for academic study purpose. (reference : pictures are from: https://intellipaat.com/blog/supervised-learning-vs-unsupervised-learning-vs-reinforcement-learning/)

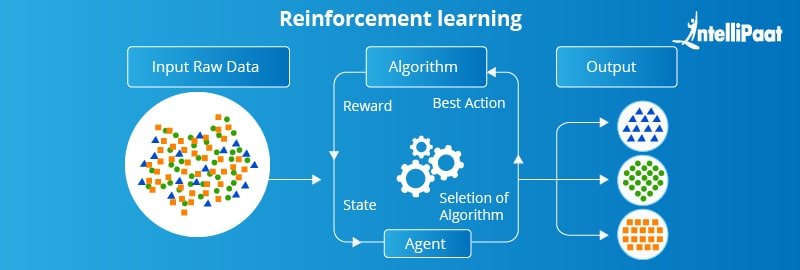






In reinforcement learning (RL), **algorithms learn to react to an environment on their own. To my own understand, RL is unlike the previous two ways putting all dataset in. RL is handling dataset on ad-hoc batch, let it go through a bag of algorithms. Output the best algorithm for that specific batch of subdataset. The whole output is a sequential of best algorithms matching with the on-going sub-datasets.**

**This reminds us differential in math, if the average speed can’t the solve in-situ problem, we use differential speed. Supervised learning is like running at the average speed, unsupervised learning is like running at a certain number of different average speeds while reinforcement learning is like running at a dynamic speed at a time. The last one requires a high speed hardware to process the small batch of sub dataset through all built-in algorithms within a very short of period of time. It’s definitely a first choice to solve the problem on-going games like flappy bird, self-driving car.**



For our case, the most practical ML approach is supervised learning. What we can do a better job is to choose a more accurate model(s). That way, I choose to use classic Logistic Regression at beginning, lowest cost and effective to see if this case can be converted into mathmatica matrix or not, and this matrix has solution or not. If its matrix convergent and has solution, it should use supervised learning approach otherwise choose unsupervised learning approach to let one matrix split into several matrix to converge. Only when both of the previous approach can’t work, we can consider the reinforcement approach, the dataset is of more diversity at a time. We don’t need that. Just supervised approach is good enough and we choose Keras API to accurately get the job done.

## 7 modeling

### 7.1 keras

Because we waste a lot of time trying to use colab, which later in the forum got a lot of user complain about the compatibility with packages and APIs. I switched back to notebook. I only sample out 10% data from original dataframe for model practice.

import numpy as np

import pandas as pd

from keras.models import Sequential

from keras.layers import Dense, Dropout, Activation

from keras.wrappers.scikit\_learn import KerasClassifier

from keras.utils import np\_utils

from sklearn.cross\_validation import KFold

from sklearn.preprocessing import LabelEncoder

from sklearn.cross\_validation import train\_test\_split

from sklearn.metrics import log\_loss

def baseline\_model():

# create model

model = Sequential()

model.add(Dense(50, input\_dim=Xtrain.shape[1], init='normal', activation='tanh'))

model.add(Dropout(0.5))

model.add(Dense(12, init='normal', activation='sigmoid'))

# Compile model

model.compile(loss='categorical\_crossentropy', optimizer='adadelta', metrics=['accuracy']) #logloss

return model

model=baseline\_model()

ga\_full=ga\_full.sample (frac=0.1)

X\_train, X\_test, y\_train, y\_test = train\_test\_split(Xtrain, dummy\_y, test\_size=0.002, random\_state=42)

fit= model.fit\_generator(generator=batch\_generator(X\_train, y\_train, 32, True),

nb\_epoch=15,

samples\_per\_epoch=50,

validation\_data=(X\_val.todense(), y\_val), verbose=2

)

# evaluate the model

scores\_val = model.predict\_generator(generator=batch\_generatorp(X\_test, 32, False), val\_samples=X\_test.shape[0])

scores = model.predict\_generator(generator=batch\_generatorp(X\_test, 32, False), val\_samples=X\_test.shape[0])

print('logloss val {}'.format(log\_loss(y\_val, scores\_val)))

#Scaling to 1-0 probs

#for i in xrange(X\_test.shape[0]):

# scores2[i,]=scores[i,]/sum(scores[i,])

pred = pd.DataFrame(scores, index = X\_test.index, columns=targetencoder.classes\_)

pred.head()

pred.to\_csv('Keras on labels and brands -2.csv',index=True)

############

The output should look like this:

Graphical user interface, application, table, Excel

Description automatically generated

### 7.2 Model XGBoost .

Boosting: gradient boosting. A boosting method. Boosting is loosely-defined as a strategy that combines multiple simple models into a single composite model. With the introduction of more simple models, the overall model becomes a stronger predictor. In boosting terminology, the simple models are called weak models or weak learners. This model can’t go through at my end on colab. The feed back is using up RAM and require upgrade to pro user. I am plan to try 10% sample data on jupyter notebook. The code should be like this:

import xgboost as xgb

from sklearn.model\_selection import train\_test\_split

X.set\_index('device\_id', inplace=True)

X\_train, X\_val, y\_train, y\_val = train\_test\_split(X, Y, train\_size=.80)

dtrain = xgb.DMatrix(X\_train, y\_train)

dvalid = xgb.DMatrix(X\_val, y\_val)

params = {

    "objective": "multi:softprob",

    "num\_class": 12, # Y has 12 catagories(groups)

    "booster": "gbtree", # gbtree is the default model(gbliner is another which based on linear model)

    "eval\_metric": "mlogloss",

   "eta": 0.3, # similar as learning rate in GBM 。

    "silent": 0, # how much info will be output.0 is default.

}

watchlist = [(dtrain, 'train'), (dvalid, 'eval')]

gbm = xgb.train(params, dtrain, 140, evals=watchlist, verbose\_eval=True)

test.set\_index('device\_id', inplace=True)

y\_pre = gbm.predict(xgb.DMatrix(test), ntree\_limit=gbm.best\_iteration)

scores = cross\_val\_score(gbm, X, Y, scoring='neg\_log\_loss',cv=10, verbose=1)

the above code is need modify. But due time, I have to do after this submission.

7.3 The wiget design for on-site test:

# https://ipywidgets.readthedocs.io/en/latest/examples/Widget%20List.html

# https://ipywidgets.readthedocs.io/en/latest/examples/Widget%20Basics.html#

import ipywidgets as widgets

import pandas as pd

phones\_short = {

    "三星": "samsung",

    "天语": "Ktouch",

    "海信": "hisense",

    "联想": "lenovo",

    "欧比": "obi",

    "爱派尔": "ipair",

    "努比亚": "nubia",

    "优米": "youmi",

    "朵唯": "dowe",

    "ZOYE": "ZOYE",

    "MIL": "MIL",

    "索尼" : "Sony"

}

phones\_short\_list = list(phones\_short.values())

phones\_short\_list

apps\_short\_list = [

    "waze","baidu","wechat","bili"

]

apps\_short\_list

input\_df = pd.DataFrame(columns=apps\_short\_list+phones\_short\_list)

input\_df

phone\_selection = widgets.RadioButtons(

    options=phones\_short\_list,

    value='samsung',

    #rows=10,

    description="",

    disabled=False

)

print("Select an handset")

phone\_selection

# Inspect value

phone\_selection.value

MULTI-SELECT FOR APPS INSTALLED ON PHONE

app\_selection = widgets.SelectMultiple(

    options=apps\_short\_list,

    value=['waze'],

    #rows=10,

    description="",

    disabled=False

)

print("CTRL+Click to select multiple apps")

app\_selection

app\_selection.value

inputs = list(app\_selection.value) + list([phone\_selection.value])

inputs

inputs = {x:1 for x in inputs}

inputs

input\_df.append(inputs, ignore\_index=True)

input\_df = input\_df.append(inputs, ignore\_index=True)

input\_df

input\_df = input\_df.fillna(0)

input\_df

So far, it’s ready for fitting in the model.

Reference for relevant app in USA:

top 15 apps in China

<https://technode.com/2017/10/20/apps-for-living-in-china/1/>