IDS 400: Classification for App Subscription

A. Mobile Application Business Overview

The global mobile application market was valued at $106.27 billion in 2018, $154.05 billion in 2019 and is expected to grow at a compound annual growth rate (CAGR) of 11.5% from 2020 to 2027.

B. Business Objective

There are two basic mobile application monetization approaches, In-App Purchase (IAP) and App Subscription.

For applications with both monetization features, subscription contributes to up to 60% of their revenue, which means increasing subscription rate and retaining existing users are their primary business goals. Below is a quick comparison chart for the Pro and Cons of IAP and Subscription.

	IAP	Subscription
Pros ✅	• Ease of promotion • Ease of payment	• Reliable and Predictable income • More engaged audiences
Cons ❌	• Do not guarantee long-term profit	• The need for continuous release of new features and content. • Ongoing customer support.

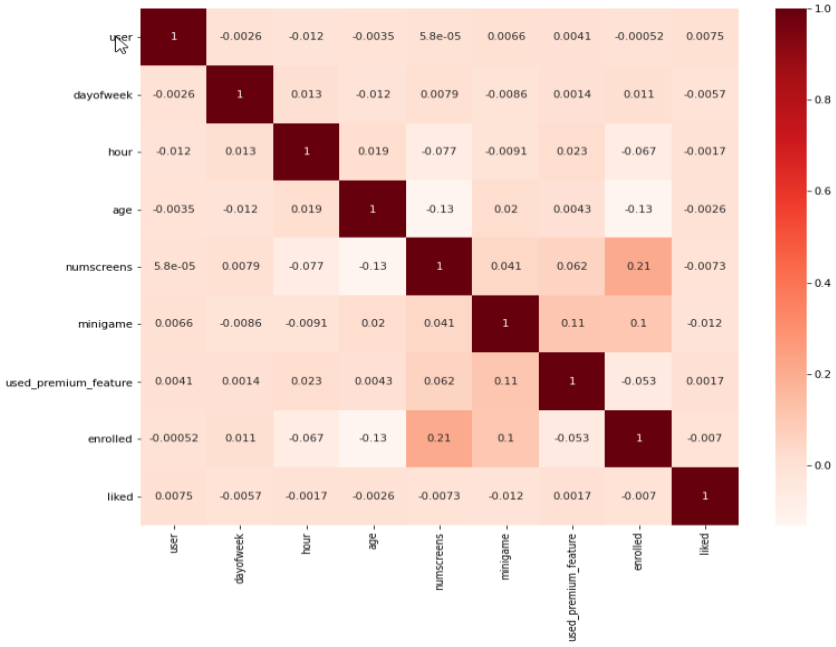
C. Data Exploration

Customer behavior must be captured so FinTech has released a mobile app with two versions: free and premium. So, in the free version, they provide a premium feature for 24 hours. We have 50000 rows (total number of survey participants) and 12 attributes (describe the behavior tendency of each participants) which are:

- **user:** Unique ID for each user
- **First_open:** Date(yy-mm-dd) and time (Hour:Minute:Seconds:Milliseconds) of login on app for the first time.
- **dayofweek:** On which day users' login.
  0:Sunday
  1: Monday
  2: Tuesday
  3:Wednesday
  4:Thursday
  5:Friday
  6:Saturday
- **hour:** Time of a day in 24-hour format customer login. It is correlated with the dayofweek column.
- **age:** The age of the registered user.
- **screen_list:** The name of several screens that consumers view, separated by a comma.
  *This describe the behavior of users and tell how much time a user is spending on screen
- **numscreens:** The total number of screens seen by customers.
- **minigame**: The app contains small games related to finance. If the customer played a mini-game, then 1 otherwise 0.
- **used_premium_feature:** If the customer used the premium feature of the app, then 1 otherwise 0.
- **Enrolled (Target Variable):**If the user bought a premium feature app then 1 otherwise 0.
- **enrolled_date:** On the date (yy-mm-dd) and time (Hour:Minute:Seconds:Milliseconds) the user bought a premium features app.
- **liked:** Each screen of the app has a like button if the customer likes it then 1 otherwise 0.

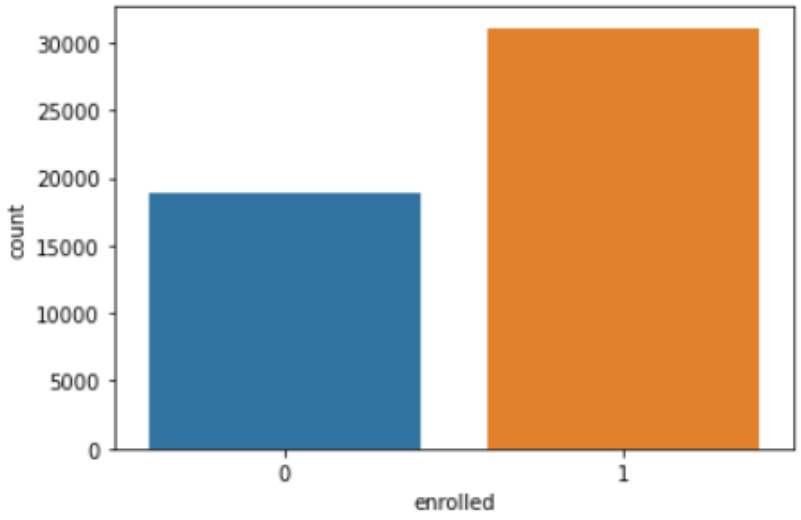
We are interested in finding the correlation between the variables and found that there is no substantial correlation between any characteristics in the fineTech appData2 dataset. The terms 'numscreens' and 'enrolled' have little in common. It suggests that those users who have seen more screens are opting for the premium app. The terms 'minigame', 'enrolled,' and 'used_premium_feature' have a small link. The inverse relationship between

'age', 'enrolled,' and 'numscreens.' It indicates that older clients aren't using the premium app and aren't seeing the various displays.

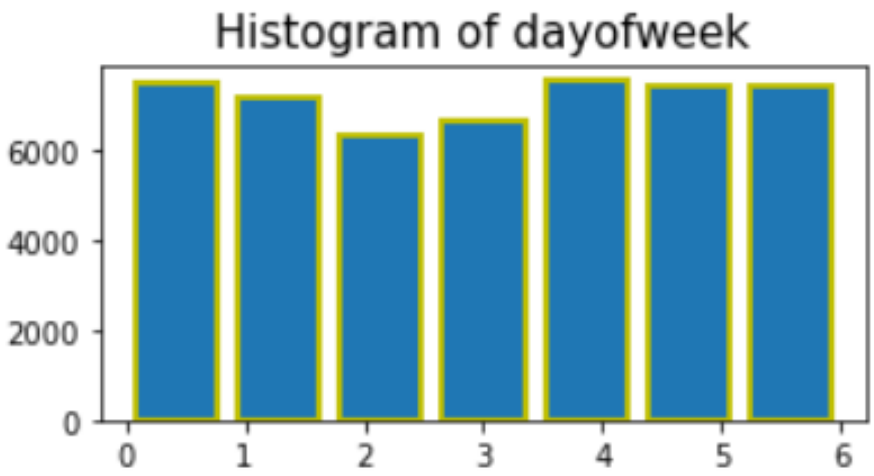


.

Then we calculated total number of Enrolled and Not enrolled users:

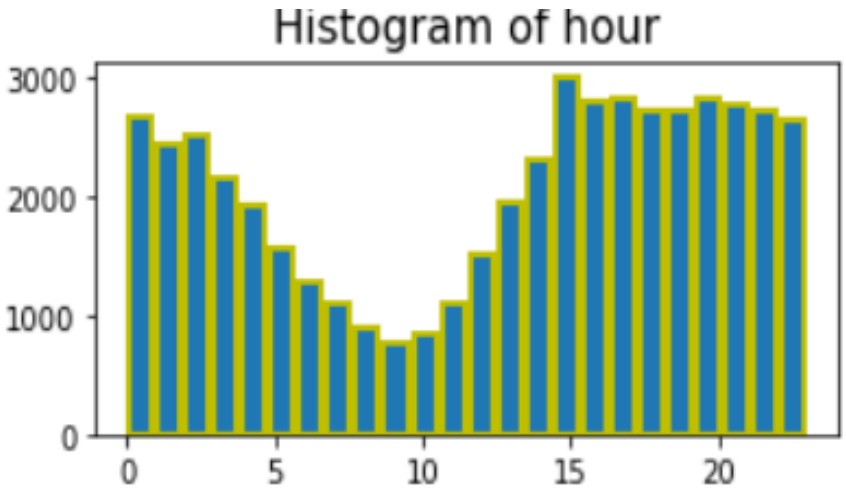


Not enrolled user = 18926 out of 50000
Enrolled user = 31074 out of 50000

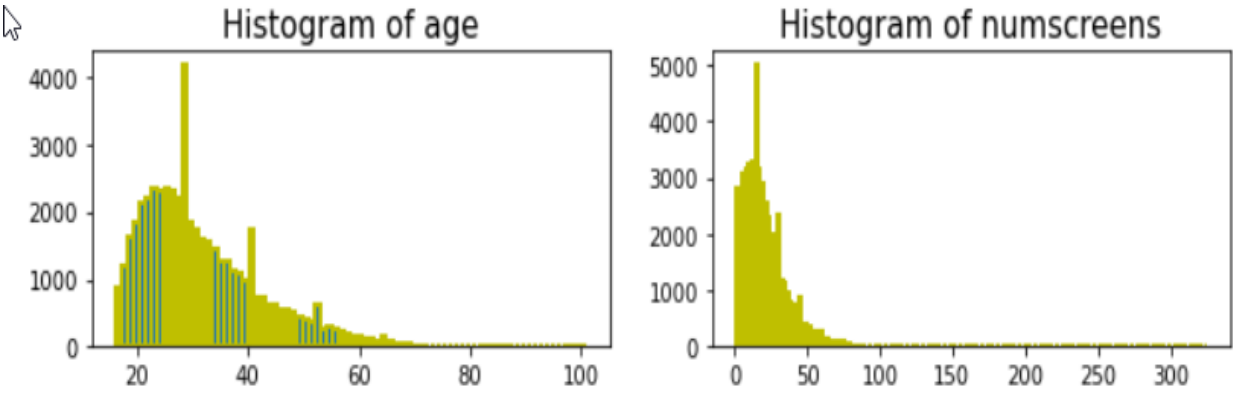


The 'dayofweek' histogram reveals that on Tuesday and Wednesday, significantly fewer customers enrolled for the app.

The 'hour' histogram demonstrates that around 10 a.m., fewer customers register on the app.



The ‘age’ histogram shows, the maximum customers are younger.
The histogram 'numsreens' reveals that just a few clients viewed more than 40 screens.



- Some missing values are found within this variable.

user	0
first_open	0
dayofweek	0
hour	0
age	0
screen_list	0
numscreens	0
minigame	0
used_premium_feature	0
enrolled	0
enrolled_date	18926
liked	0

To complete the analysis we use python programming and it's libraries like:

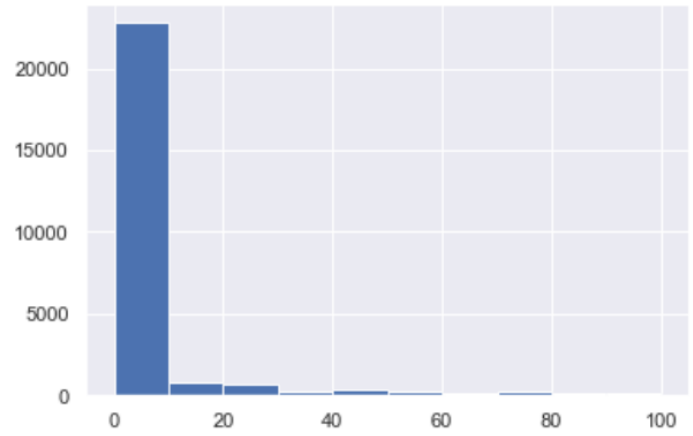
- NumPy
- Pandas
- Matplotlib

- Seaborn

To find out that the majority of customers register within 10 hours after registering we converted the object data type to datetime data type.

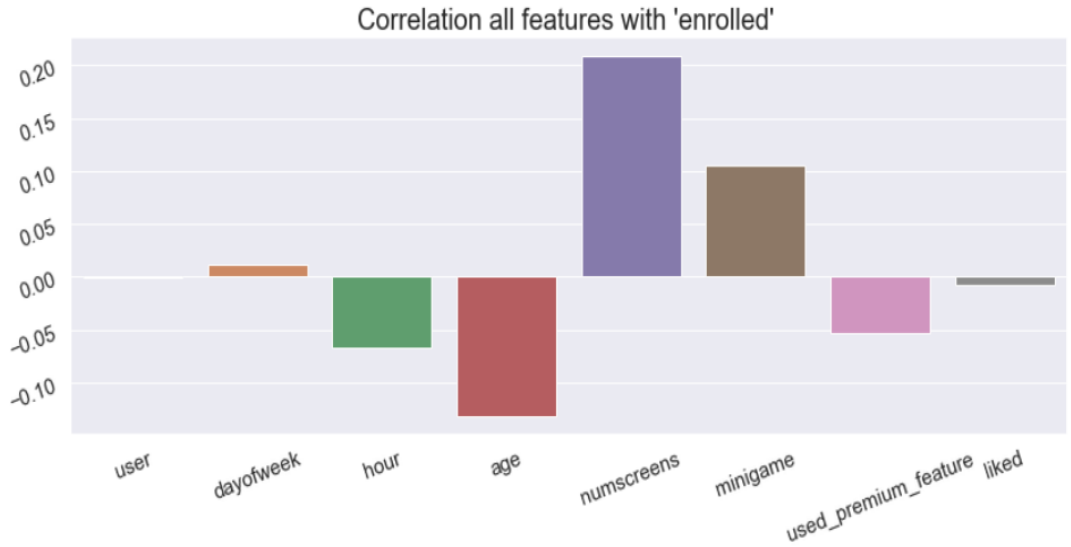
user	int64	user	int64
first_open	**object**	**first_open**	**datetime64[ns]**
dayofweek	int64	dayofweek	int64
hour	object	hour	int64
age	int64	age	int64
screen_list	object	screen_list	object
numscreens	int64	numscreens	int64
minigame	int64	minigame	int64
used_premium_feature	int64	used_premium_feature	int64
enrolled	int64	enrolled	int64
enrolled_date	**object**	**enrolled_date**	**datetime64[ns]**
liked	int64	liked	int64

And find the difference between enrolled_date and first_open and plot the graph.



According to the histogram above, the majority of users enrolled in the app within 10 hours after registration.

Using **Lasso** to select important variables and we plotted those variables to find correlation.



We saw the heatmap correlation matrix, but it didn't illustrate correlation very well. However, using the above barplot, you can quickly see which features are most connected with the 'enrolled' feature.
The 'numscreens' and ‘minigame' features are substantially connected with the 'enrolled' feature, however the other features are not.
The 'enrolled' feature is substantially negatively linked with the 'hour’, 'age’ and 'used premium feature' features.

Feature selection

- We are considering those customers who have enrolled after 48 hours as 0.
- Drop some of the variables which are not strongly correlated with target variables like time_to enrolled, enrolled_date, first_open.
- We can't utilize the 'Screen list' since it contains string values. To fix this problem, we append each screen name from 'fineTech app screen Data' to 'fineTech appData' as a column of the same name. Then, in the added column, see if this screen name is accessible in ‘screen list'. If it is, set value 1; otherwise, set value 0.

Before appending columns

screen_list
idscreen,joinscreen,Cycle,product_review,ScanP...
joinscreen,product_review,product_review2,Scan...
Splash,Cycle,Loan
product_review,Home,product_review,Loan3,Finan...
idscreen,joinscreen,Cycle,Credit3Container,Sca...
idscreen,Cycle,Home,ScanPreview,VerifyPhone,Ve...
product_review,product_review2,ScanPreview
Splash,Cycle,Home,Credit3Container,Credit3Dash...
product_review,product_review2,ScanPreview,Ver...
Home,Loan2,product_review,product_review,produ...

After the columns are added. And we get 69 features from 12.

(50000, 12) → (50000, 69)

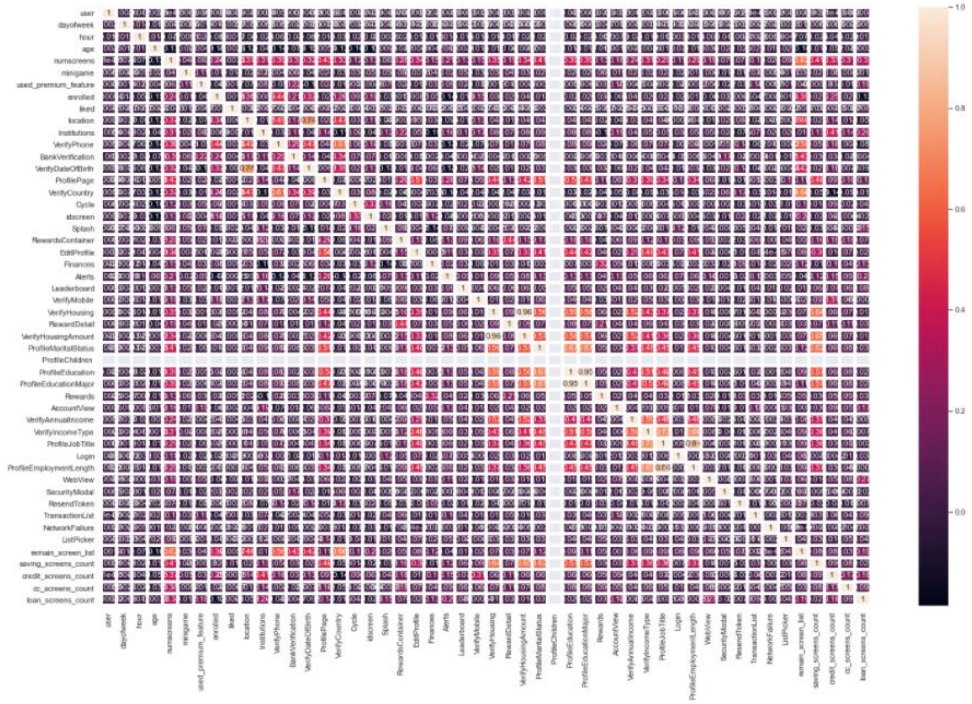
	user	first_open	dayofweek	hour	age	screen_list	numscreens	minigame	used_premium_feature	enrolled
0	235136	2012-12-27 02:14:51.273	3	2	23	joinscreen,product_review,ScanPreview,VerifyTo...	15	0	0	0
1	333588	2012-12-02 01:16:00.905	6	1	24	joinscreen,product_review,product_review2,Scan...	13	0	0	0
2	254414	2013-03-19 19:19:09.157	1	19	23		3	0	1	0
3	234192	2013-07-05 16:08:46.354	4	16	28	product_review,Home,product_review,ReferralCon...	40	0	0	1
4	51549	2013-02-26 18:50:48.661	1	18	31	joinscreen,ScanPreview,VerifySSN,Home,SelectIn...	32	0	0	1
...	...	...	...	...	...	...	...	...	...	...
49995	222774	2013-05-09 13:46:17.871	3	13	32	Home,ScanPreview,VerifySSN,product_review,prod...	13	0	0	1
49996	169179	2013-04-09 00:05:17.823	1	0	35	Home,	4	0	1	0
49997	302367	2013-02-20 22:41:51.165	2	22	39	joinscreen,product_review,product_review2,Scan...	25	0	0	0
49998	324905	2013-04-28 12:33:04.288	6	12	27	Home,product_review,product_review,product_rev...	26	0	0	1
49999	27047	2012-12-14 01:22:44.638	4	1	25	product_review,ScanPreview,ProfileVerifySSN,Pr...	26	0	0	0

50000 rows × 69 columns

ProfileJobTitle	Login	ProfileEmploymentLength	WebView	SecurityModal	Loan4	ResendToken	TransactionList	NetworkFailure	ListPicker
0	1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...
0	0	0	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

All the saving screens are associated with one another, thus we save the sum of all saving screens in each row in a single row for all customers, and similarly for credit, CC1, and loan screens in a single row.

Now with the change data set we will find the correlation matrix using **heatmap.**



D. Data Preprocessing

- Separate dataset into training and testing
  (using set seed to make sure we get the same training and testing set at each run)

```
from sklearn.model_selection import train_test_split
appDataDP=appDataDP.loc[:, (appDataDP != 0).any(axis=0)]
seed = 7
X_train, X_test, y_train, y_test = train_test_split(appDataDP, target, test_size = 0.3, random_state=seed)
```

- Standardize our dataset

```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
appDataDP_sd=sc.fit_transform(appDataDP)
X_train_sc = sc.fit_transform(X_train)
X_test_sc = sc.transform(X_test)
```

E. Analysis

Model Building

We must use supervised classification techniques since the target variable is categorical type 0 and 1. To create the best model, we must first train and test

the dataset using several Machine Learning methods, after which we will be able to identify the best ML model.

The needed packages imported:

from sklearn.metrics import confusion_matrix, classification_report, accuracy_score

- **Logistic Regression Model**

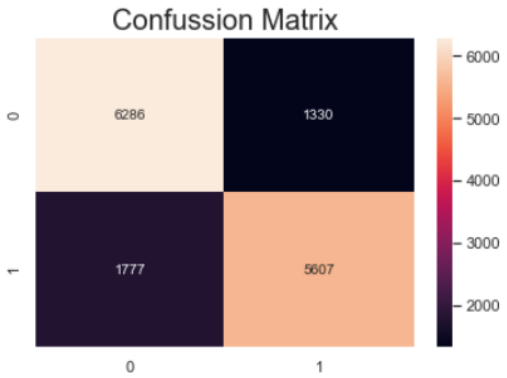
```
# train with Standard Scaling dataset
logit_model = LogisticRegression(random_state = seed, penalty = 'l2')
logit_model.fit(X_train_sc, y_train)
y_pred = logit_model.predict(X_test_sc)
print(accuracy_score(y_test, y_pred))
print('Accuracy of logistic regression classifier on test set: {:.2f}'.format(logit_model.score(X_test, y_test)))
```

```
0.7807333333333333
Accuracy of logistic regression classifier on test set: 0.70
```

For penalty, we go for L2 Ridge Regression to prevent the model from overfit.
Accuracy for Training Set: 0.7807
Accuracy for Test Set: 0.70

- **SVM**

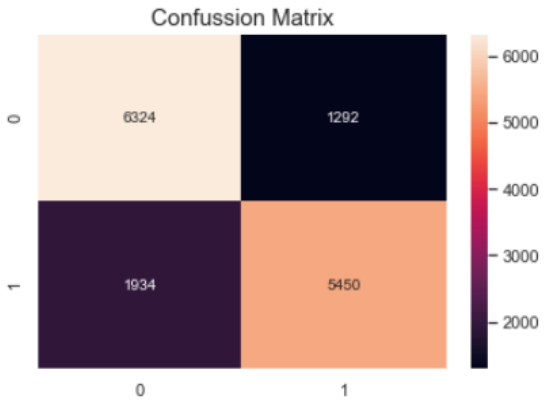
We build an SVM model and calculate the accuracy in 79%, as well as plot the confusion matrix to see how much better our model is.



Recall or Hit rate (% of positive cases caught): 75.9%
Precision (% of positive predictions that are correct): 80.82%

- **Random Forest**

We create a Random Forest model, assess its accuracy in 78%, then plot the confusion matrix to evaluate how much better it is.



Sensitivity or Recall or Hit rate (% of positive cases caught): 73.8%
Precision (% of positive predictions that are correct): 80.83%

F. Business Implementation

- From the Logistic Regression result, we define the attributes possessing positive and negative correlations with the target variable, "enrolled". For users who went through app pages such as "Mini Game", "Credit page", and "Verify Phone/Mobile", will more likely increase their possibility to subscribe to the premier version. On the other hand, if the users click through pages like "Alerts", "Resend Token", "Loan Page" they will predict to be less willing to subscribe.

- According to the result, we came up with three **business recommendations.**

1. Find the potential group of users to promote. Target the customers -> Promote to the right group of people (more cost-effective for the business)

2. Focus on user pain points to optimize the user experience. Let users discover the "more appealing" features in the app by guiding the users to click on pages such as mini games, a well made credit record page (which are the green attributes that have a positive correlation in the logistic analysis result). The company might have a higher possibility to

3. Design a popup tutorial to guide the users to know more about app features. For instance, Alert page and Resend token page -> refine user interface to make it less intimidating or decrease the occasion that alert will pop up.

	Coefficient
mini_game	0.9426
credit_page	2.0914
verify_phone	0.8382
verify_mobile	1.9496
remain_screen_list	0.2984
alert	-0.4314
resend_token	-0.4326
loan_page	-0.5103
location	-0.5322

	Coef.	Std.Err.	z	P>|z|	[0.025	0.975]
hour	-0.0065	0.0016	-4.1002	0.0000	-0.0096	-0.0034
age	-0.0064	0.0011	-5.8250	0.0000	-0.0085	-0.0042
numscreens	-0.0083	0.0017	-4.7847	0.0000	-0.0117	-0.0049
minigame	0.9429	0.0421	22.4103	0.0000	0.8604	1.0253
used_premium_feature	-0.1949	0.0354	-5.5109	0.0000	-0.2642	-0.1256
Loan2	-0.4006	0.0310	-12.9348	0.0000	-0.4613	-0.3399
location	-0.5322	0.0361	-14.7602	0.0000	-0.6029	-0.4615
Institutions	-0.0264	0.0362	-0.7288	0.4661	-0.0974	0.0446
Credit3Container	0.3850	0.0364	10.5811	0.0000	0.3137	0.4564
VerifyPhone	0.8210	0.0346	23.7468	0.0000	0.7532	0.8888
BankVerification	0.0907	0.0317	2.8575	0.0043	0.0285	0.1529
VerifyDateOfBirth	0.1599	0.0347	4.6137	0.0000	0.0920	0.2279
ProfilePage	-0.2249	0.0453	-4.9697	0.0000	-0.3136	-0.1362
VerifyCountry	-0.5571	0.0401	-13.8894	0.0000	-0.6357	-0.4784
Cycle	0.1266	0.0274	4.6203	0.0000	0.0729	0.1803
idscreen	0.3434	0.0285	12.0607	0.0000	0.2876	0.3992
Credit3Dashboard	-0.8382	0.0524	-16.0108	0.0000	-0.9408	-0.7356
Loan3	-0.4132	0.0588	-7.0222	0.0000	-0.5286	-0.2979
CCICategory	0.0111	0.0943	0.1175	0.9064	-0.1737	0.1959
Splash	-0.2117	0.0392	-5.4070	0.0000	-0.2885	-0.1350
Loan	-1.0650	0.0489	-21.7993	0.0000	-1.1608	-0.9692
CC1	0.0026	0.0693	0.0369	0.9706	-0.1333	0.1384
RewardsContainer	-0.1017	0.0540	-1.8838	0.0596	-0.2075	0.0041
Credit3	-0.0661	0.0625	-1.0574	0.2903	-0.1887	0.0564
Credit1	2.0914	0.0564	37.0658	0.0000	1.9808	2.2020
EditProfile	0.2239	0.0701	3.1954	0.0014	0.0866	0.3613
Credit2	0.1622	0.0617	2.6296	0.0085	0.0413	0.2830
Finances	0.0529	0.0681	0.7774	0.4369	-0.0806	0.1865
CC3	0.1431	0.1015	1.4100	0.1585	-0.0558	0.3419
Saving9	0.3242	0.1094	2.9644	0.0030	0.1099	0.5386
Saving1	-0.2695	0.1752	-1.5379	0.1241	-0.6129	0.0740
Alerts	-0.4314	0.0517	-8.3465	0.0000	-0.5327	-0.3301
Saving8	0.0041	0.1036	0.0394	0.9686	-0.1990	0.2072
Saving10	0.2116	0.1766	1.1986	0.2307	-0.1344	0.5577
Leaderboard	-0.2727	0.0546	-4.9983	0.0000	-0.3797	-0.1658
Saving4	0.3641	0.1355	2.6876	0.0072	0.0986	0.6296
VerifyMobile	1.9496	0.0837	23.2998	0.0000	1.7856	2.1136
VerifyHousing	-0.0943	0.1992	-0.4734	0.6359	-0.4848	0.2962
RewardDetail	0.0134	0.0840	0.1591	0.8736	-0.1513	0.1781
VerifyHousingAmount	0.0206	0.2063	0.1001	0.9203	-0.3836	0.4249
ProfileMaritalStatus	-0.2399	0.0947	-2.5327	0.0113	-0.4256	-0.0543
ProfileEducation	0.3034	0.1750	1.7338	0.0830	-0.0396	0.6465
Saving7	-0.5299	0.1410	-3.7583	0.0002	-0.8062	-0.2535
ProfileEducationMajor	-0.2870	0.1816	-1.5800	0.1141	-0.6430	0.0690
Rewards	0.0297	0.0894	0.3324	0.7396	-0.1455	0.2050
AccountView	-0.2245	0.0832	-2.6985	0.0070	-0.3876	-0.0615
VerifyAnnualIncome	0.4547	0.1252	3.6327	0.0003	0.2094	0.7000
VerifyIncomeType	-0.2329	0.1179	-1.9748	0.0483	-0.4641	-0.0017
Saving2	-0.6941	0.2829	-2.4537	0.0141	-1.2486	-0.1397
Saving6	0.4644	0.1956	2.3739	0.0176	0.0810	0.8478
Saving2Amount	0.5104	0.2964	1.7218	0.0851	-0.0706	1.0914
Saving5	-0.0219	0.1971	-0.1113	0.9114	-0.4083	0.3644
ProfileJobTitle	-0.2144	0.1646	-1.3023	0.1928	-0.5370	0.1083
Login	-0.7266	0.0701	-10.3647	0.0000	-0.8640	-0.5892
ProfileEmploymentLength	-0.1197	0.1620	-0.7388	0.4600	-0.4373	0.1979
WebView	-0.1460	0.0461	-3.1655	0.0015	-0.2364	-0.0556
SecurityModal	-0.0221	0.1065	-0.2073	0.8358	-0.2308	0.1867
Loan4	-0.5103	0.1158	-4.4056	0.0000	-0.7373	-0.2833
ResendToken	-0.4326	0.0990	-4.3712	0.0000	-0.6265	-0.2386
TransactionList	-0.1006	0.1093	-0.9204	0.3574	-0.3149	0.1136
NetworkFailure	0.2744	0.1334	2.0579	0.0396	0.0131	0.5358
ListPicker	-0.5871	0.1343	-4.3726	0.0000	-0.8502	-0.3239
remain_screen_list	0.2984	0.0066	44.9961	0.0000	0.2854	0.3114
dayofweek.0	-0.4596	1621332.0033	-0.0000	1.0000	-3177752.7932	3177751.8739
dayofweek.1	-0.5591	1621332.0033	-0.0000	1.0000	-3177752.8926	3177751.7745
dayofweek.2	-0.5123	1621332.0033	-0.0000	1.0000	-3177752.8458	3177751.8213
dayofweek.3	-0.5042	1621332.0033	-0.0000	1.0000	-3177752.8377	3177751.8294
dayofweek.4	-0.6087	1621332.0033	-0.0000	1.0000	-3177752.9422	3177751.7248
dayofweek.5	-0.4609	1621332.0033	-0.0000	1.0000	-3177752.7944	3177751.8726
dayofweek.6	-0.4760	1621332.0033	-0.0000	1.0000	-3177752.8095	3177751.8576
first_month.1	-0.3972	1621332.0033	-0.0000	1.0000	-3177752.7308	3177751.9363
first_month.2	0.5025	1621332.0033	0.0000	1.0000	-3177751.8310	3177752.8360
first_month.3	0.3677	1621332.0033	0.0000	1.0000	-3177751.9659	3177752.7012
first_month.4	-0.1313	1621332.0033	-0.0000	1.0000	-3177752.4649	3177752.2022
first_month.5	-0.1272	1621332.0033	-0.0000	1.0000	-3177752.4608	3177752.2063
first_month.6	-0.5643	1621332.0033	-0.0000	1.0000	-3177752.8978	3177751.7692
first_month.7	-0.9314	1621332.0033	-0.0000	1.0000	-3177753.2650	3177751.4021
first_month.11	-1.0949	1621332.0033	-0.0000	1.0000	-3177753.4285	3177751.2386
first_month.12	-1.2044	1621332.0033	-0.0000	1.0000	-3177753.5379	3177751.1292

Agenda

- Introduction to our topic (use simple questions to intrigue audiences)
- Our data set (cam use some visualization)
- Business Objectives → The problem we trying to solve
- Our assumption for the answer
- Some research to outline the direction of our idea
- Data analysis
- Visualization
- Does the result meet our expectations?

Slides overview

1. Heading
2. Agenda
3. Introduction (our motivation)
4. Business Objectives → The problem we trying to solve
5. Use Visualization to showcase our data
6. Distinguish how many types of users ? (K-mean)
7. Prep Our data set
8. Analysis
9. Result (Does the result meet our expectations?)
10. Reference

-

Dataset in mind

- Customers to Subscription Through App Behavior

Customers_to_Subscription_Through_App_Behavior

In fineTech_appData DataFrame, there are 50,000 users data with 12 different variables. Let's know each and every variable in brief.

1. user: Unique ID for each user.

2. first_open: Date *(yy-mm-dd)* and time *(Hour:Minute:Seconds:Milliseconds)* of login on app for the first time.

3. dayofweek: On which day users log in.

- 0: Sunday
- 1: Monday
- 2: Tuesday
- 3: Wednesday
- 4: Thursday
- 5: Friday
- 6: Saturday

4. Hour: Time of a day in 24-hour format customer logon. It is correlated with the dayofweek column.

5. age: The age of the registered user.

6. screen_list: The name of multiple screens seen by customers, which are separated by a comma.

7. numscreens: The total number of screens seen by customers.

8. minigame: The app contains small games related to finance. If the customer played a mini-game then 1 otherwise 0.

9. used_premium_feature: If the customer used the premium feature of the app then 1 otherwise 0.

10. (target variable) enrolled: 0 or 1, If the user bought a premium feature app then 1 otherwise 0.

11. enrolled_date: On the date (yy-mm-dd) and time (Hour:Minute:Seconds:Milliseconds) the user bought a premium features app.

12. liked: Each screen of the app has a like button if the customer likes it then 1 otherwise 0

Common Questions

Q: What are we trying to solve here?

A: To distinguish which kind of users will end up subscribing to their premium versions. After finding the pattern, what can business do to, then, get the rest of your users to do more of those actions?

Q: What are the business insights of this analysis?

A: Subscription contributes an average of more than 50% percent of the application's revenue. Increasing subscribers is an important means of monetization. Such as targeting your promotion campaign to those who are less likely to subscribe might be more cost-effective than targeting all the users.

Q: How to deal with this dataset?

A: This is a classification problem, whether users enroll in the premium version or not, we can use Logistic Regression/KNN/SVM to predict.

Q: How to showcase visually?

A: At the beginning, we can use bar charts or any kind of plot to present users' profiles.

Reference

Directing-Customers-to-Subscription-Through-App-Behavior-Analysis/Directing Customers to Subscription Through App Behavior Analysis.ipynb at master · bleso-a/Directing-Customers-to-Subscription-Through-App-Behavior-Analysis

ML Project: Directing Customers to Subscription Through Financial App Behavior Analysis