Data Mining and Business Analysis with QWE

Shen haowen 20307110108

June 13, 2023

Abstract

In this project I tried to explore the QWE dataset and build a model to predict the potential churning customers. First I conducted the data pre-processing and EDA with the QWE dataset. Visualization was also implemented to help achieve a better understanding. Then I found that the biggest difficulty in prediction is the sample imbalance. As a result I tried different models and implemented several modifications to achieve better performance. Eventually I reached the highest AUC score of 0.77. Data Insights and business analysis were also conducted based on that.

1 Data Description

The QWE dataset contains 6347 rows (each representing a unique customer) with 13 columns: 12 features, 1 target feature (Churn). The data is composed of both int and float feature. Here lists the brief description of each column in the QWE dataset.

Target:
Churn - Whether the customer churned or not (1,0)
Features:
Customer_Age_inmonths - represents customer longevity in months
CHI_M0 - Customer Happiness Index at the current moment
CHI_M01 - Difference of CHI between this month and last month
Support_M0 - number of support cases at the current moment
Support_M01 - Difference of support cases between this month and last month
SP_M0 - average support priority at the current moment
SP_M01 - Difference of average support priority between this month and last month
Logins_M01 - Difference of Clients' login time between this month and last month
Blog_M01 - Difference of clients' log amount between this month and last month
Views_M01 - Difference of clients' visit between this month and last month

DaysLastLogin_M01 - Difference of visit interval between this month and last month

2 Data pre-processing

First I checked whether there contains abnormal values in the dataset. We do not have any missing or duplicated data and our data-types are in order. At the top of the data, we see two columns that are irrelevant, 'ID' and 'Churn', as the former is a unique identifier of the customer and the latter is the target. We quickly remove these features from our DataFrame.

```
Data columns (total 13 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   ID                      6347 non-null   int64
 1   Customer_Age_inmonths   6347 non-null   int64
 2   Churn                   6347 non-null   int64
 3   CHI_M0                  6347 non-null   int64
 4   CHI_M01                 6347 non-null   int64
 5   Support_M0              6347 non-null   int64
 6   Support_M01             6347 non-null   int64
 7   SP_M0                   6347 non-null   float64
 8   SP_M01                  6347 non-null   float64
 9   Logins_M01              6347 non-null   int64
 10  Blog_M01                6347 non-null   int64
 11  Views_M01               6347 non-null   int64
 12  DaysLastLogin_M01       6347 non-null   int64
dtypes: float64(2), int64(11)
memory usage: 644.7 KB
```

Figure 1: Data information

Also I did a data description to explore the distribution of both target and features. As we can see in the figure, the churned targets only occupy 5% of the whole 6347 samples, which indicates a significant sample imbalance. Meanwhile, Our data is full of numeric data now, but they are all in different units. Comparing the number of support cases with a continuous value of days since last login will not give any relevant information because they have different units. The variables simply will not give an equal contribution to the model. To fix this problem, we will standardize our data values before training the model.

	Customer_Age_inmonths	Churn	CHI_M0	CHI_M01	Support_M0	Support_M01	SP_M0	SP_M01	Logins_M01
count	6347.000000	6347.000000	6347.000000	6347.000000	6347.000000	6347.000000	6347.000000	6347.000000	6347.000000
mean	13.896802	0.050890	87.316685	5.058610	0.706318	-0.006932	0.812781	0.030169	15.727903
std	11.160078	0.219791	66.282788	30.828767	1.723961	1.870942	1.320530	1.460336	42.119061
min	0.000000	0.000000	0.000000	-125.000000	0.000000	-29.000000	0.000000	-4.000000	-293.000000
25%	5.000000	0.000000	24.500000	-8.000000	0.000000	0.000000	0.000000	0.000000	-1.000000
50%	11.000000	0.000000	87.000000	0.000000	0.000000	0.000000	0.000000	0.000000	2.000000
75%	20.000000	0.000000	139.000000	15.000000	1.000000	0.000000	2.666667	0.000000	23.000000
max	67.000000	1.000000	298.000000	208.000000	32.000000	31.000000	4.000000	4.000000	865.000000

Figure 2: Data description

3 EDA

In this part I conducted some exploratory data analysis(EDA) and visualization within the QWE dataset. The sample imbalance is demonstrated by the pie chart below. You can check the **EDA.ipynb file** for all the detailed code.

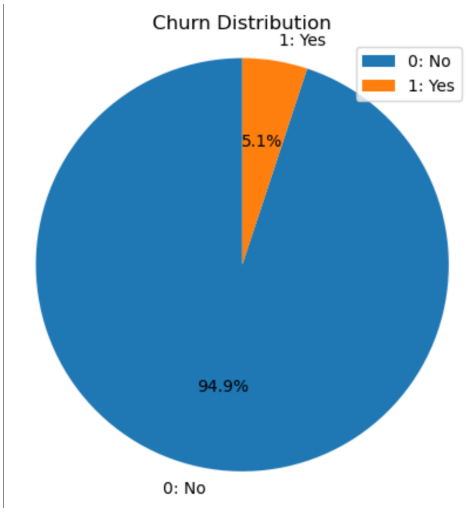


Figure 3: Total Churn Numeber Distribution

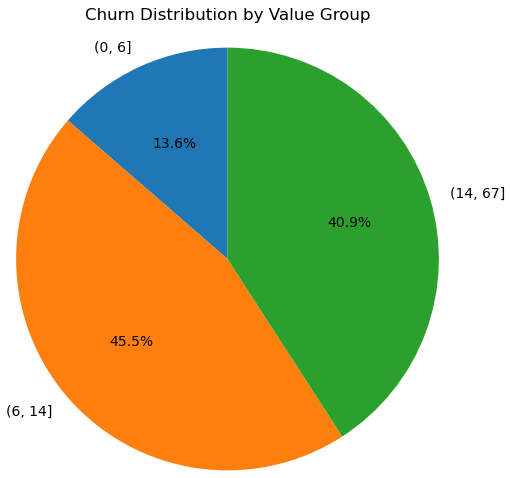


Figure 4: Churn Number Distribution by Age Value Group

As we can see in the right pie chart, customer longevity from 6 months to 14 months accounts for the largest percentage of churned customers, which is 45.5%. Customer longevity above 14 months occupies 40.9% and 13.6% for Customer longevity below 6 months.Let us

also have a look at the detailed churn rate by customer longevity.

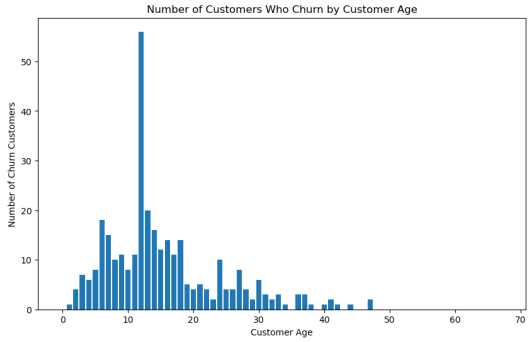


Figure 5: Number of Churning Customers by Age

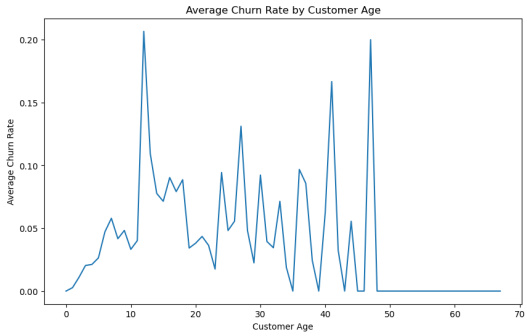


Figure 6: Average Churn Rate by Age

Also the density of customer longevity by different churn outcomes was drawn. It can be clearly seen that the churned customers and unchurned customers have different customer longevity density distribution. Churned distribution peaks around 13 months and has a relatively small variance, while unchurned distribution has a wider distribution and a Poisson like distribution form.

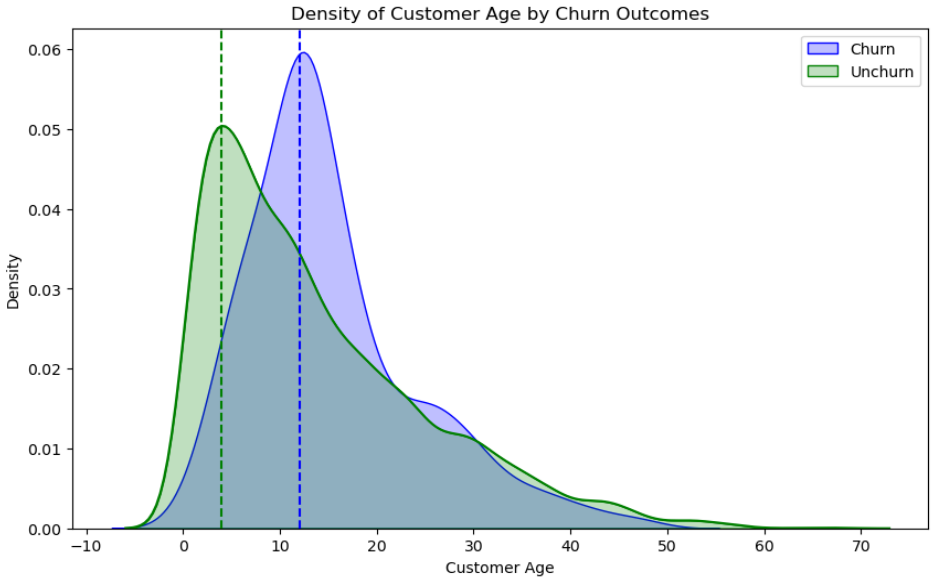


Figure 7: Density of Customer Age by Churn Outcomes

CHi(Customer Happiness Index) has a large role in determining whether a customer will churn. I drew the density of CHI at the current moment and difference of CHI between this month and last month in Churned customers. Meanwhile, Density of CHI at the current moment between Churn and unchurned customers was drew. It is clear that churn customers tend to have relatively lower CHI.

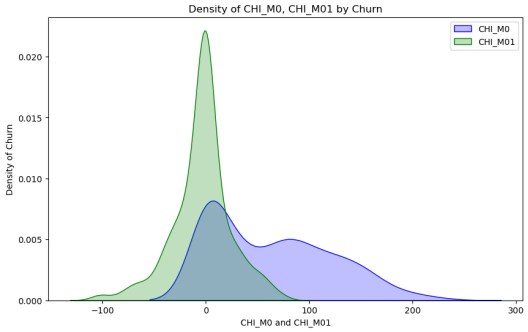


Figure 8: Density of CHI_M0, CHI_M01 by Churn

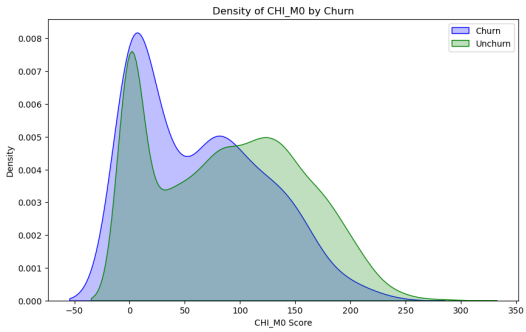
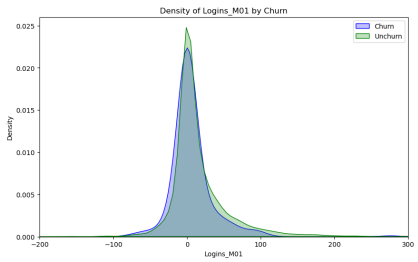


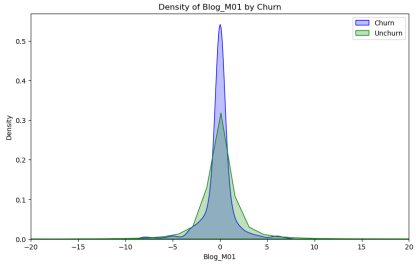
Figure 9: Density of CHI_M0 by Churn

For the usage information, I compared the difference of churn and unchurned customers between this month and last month. From the figures I found that it was hard to identify whether a customer would churn merely on the Usage Information. A customer may use the system more frequently if he encounters a problem that could not be fixed, which will certainly increase the probability of churn. In contrast, he may use the system more only because he finds the system helpful and convenient. It is hard to tell which situation is right.

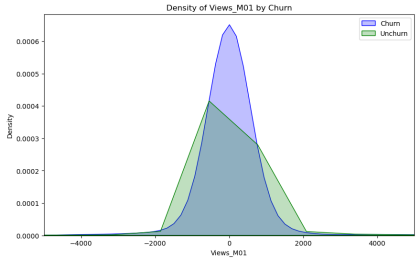
As we can see in the figures, churned and unchurned customer features data have large overlap, which indicates that it is difficult for the model to precisely classify the two groups.



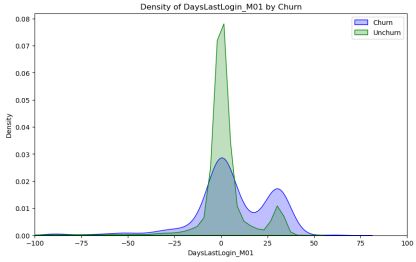
(a) Logins_M01



(b) Blog_M01



(c) Views_M01



(d) DaysLastLogin_M01

Figure 10: Density of usage information by Churn

For each feature in the dataset, I conducted a Correlation Analysis using Pearson correlation coefficient test with the target ”Churn”. The results are listed in the following table.

Taking the significance as 0.05, I found that ”Churn” has a relatively high correlation with CHI, SP_M0, Logins, DaysLastLogin, which are all statistically significant. A heatmap is also drawn for more straightforward understanding.

Table 1: Pearson Correlation Analysis

Column	**Correlation**	**p-value**
Customer_Age_inmonths	0.0302	0.0161
CHI_M0	-0.0840	2.04×10^{-11}
CHI_M01	-0.0661	1.38×10^{-7}
Support_M0	-0.0450	0.0003
Support_M01	0.0055	0.6638
SP_M0	-0.0549	1.19×10^{-5}
SP_M01	-0.0074	0.5539
Logins_M01	-0.0421	0.0008
Blog_M01	-0.0129	0.3046
Views_M01	-0.0141	0.2610
DaysLastLogin_M01	0.0609	1.22×10^{-6}

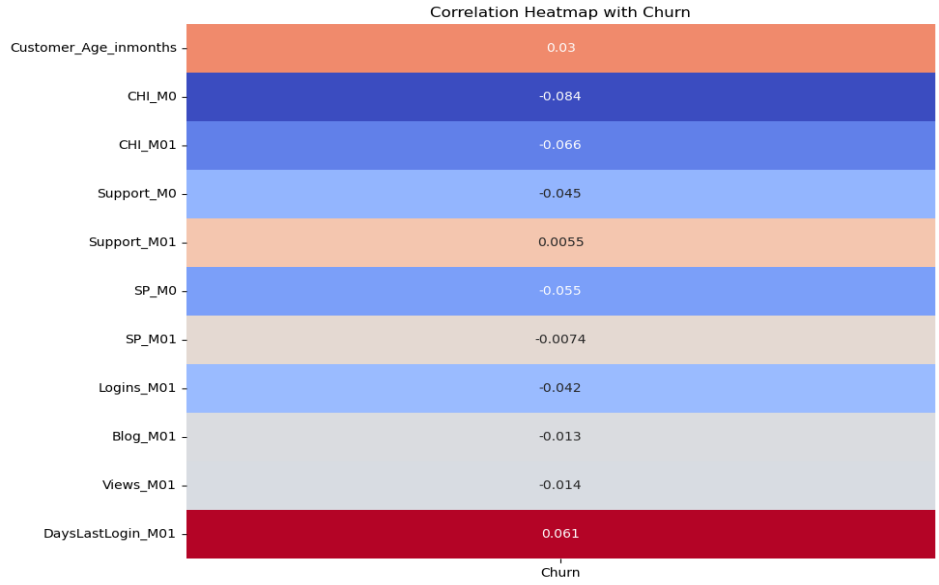


Figure 11: Correlation Heatmap with Churn

4 Models and Prediction

As I have mentioned before, we will need to standardize our data values before training the model. So, I rescale the original variable to have equal range and variance as the remaining variables. In this case, I use Min-Max Scaling [0,1] because the standardize value will lie within the binary range.

In this section, I implemented several models including logistic regression, random forest, SVM and neural network. From the data description we know that sample imbalance is the main difficulty of prediction, as a result of which I tried several ways to help reach better precision and recall.

Here I listed the **final performance of my main models**:

Table 2: Model Performance Comparison

Model	Confusion Matrix	Recall	Precision	Fb Score	AUC
Logistic Regression	$\begin{bmatrix} 45 & 20 \\ 394 & 811 \end{bmatrix}$	0.6923	0.1025	0.3219	0.69
Random Forest	$\begin{bmatrix} 35 & 30 \\ 337 & 868 \end{bmatrix}$	0.5385	0.0941	0.2769	0.70
SVM	$\begin{bmatrix} 47 & 18 \\ 481 & 724 \end{bmatrix}$	0.7231	0.0890	0.2982	0.72
Neural Network	$\begin{bmatrix} 46 & 19 \\ 431 & 774 \end{bmatrix}$	0.7077	0.0964	0.3314	0.74
Neural Network with GAN	$\begin{bmatrix} 40 & 25 \\ 220 & 985 \end{bmatrix}$	0.6154	0.1538	0.3846	0.77

4.1 Evaluation Indicators

Confusion Matrix:

The confusion matrix is a matrix used to evaluate the performance of a classification model, typically in binary classification problems. It compares the predicted labels with the actual labels and categorizes the samples into four categories: true positive (TP), true negative (TN), false positive (FP), and false negative (FN). The confusion matrix provides insights into the model's accuracy and misclassification on different classes.

$$\begin{bmatrix} \text{True Positive (TP)} & \text{False Negative (FN)} \\ \text{False Positive (FP)} & \text{True Negative (TN)} \end{bmatrix}$$

AUC (Area Under the Curve):

AUC is a metric used to evaluate the performance of a binary classification model. It represents the area under the Receiver Operating Characteristic (ROC) curve. The ROC curve is a plot of the true positive rate (TPR) against the false positive rate (FPR). AUC measures the model's ability to distinguish between positive and negative samples, where a higher AUC indicates better performance.

PR (Precision-Recall):

PR curve is another metric used to evaluate binary classification models. It focuses on

the precision and recall of the positive class. Precision represents the proportion of correctly predicted positive samples among all predicted positive samples, while recall represents the proportion of correctly predicted positive samples among all actual positive samples. The PR curve is a plot of precision against recall. In the case of imbalanced datasets, where the positive class is rare, the PR curve is often more informative than the ROC curve as it captures the model's performance in correctly identifying positive samples.

AUC Formula:

The AUC can be calculated as the integral of the ROC curve:

$$\text{TPR} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

$$\text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}}$$

$$\text{AUC} = \int \text{TPR}(f)\, d\text{FPR}(f)$$

F_β score:

The F_β score is a metric commonly used in binary classification tasks to measure the balance between precision and recall. It is defined as the weighted harmonic mean of precision and recall, where the weight is determined by the parameter β.

When β is greater than 1, the F_β score emphasizes recall more than precision. This means that false negatives (missed positive samples) are considered more important than false positives (misclassified negative samples), resulting in a higher weight on recall. Conversely, when β is less than 1, the F_β score places more emphasis on precision, indicating that false positives are considered more important, leading to a higher weight on precision. When β is equal to 1, the F_β score reduces to the F1 score, which equally balances precision and recall.

PR Formula:

Precision, Recall, and F_β-score can be calculated based on the confusion matrix:

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

$$F_\beta = \frac{(1 + \beta^2) \cdot \text{Precision} \cdot \text{Recall}}{(\beta^2 \cdot \text{Precision}) + \text{Recall}}$$

4.2 Main objective of the project

As we have mentioned in the data description, our main objective is to build a model to predict the potential churning customers. In this case, the overall accuracy is not that

important to us because we don't need to predict exactly who will not churn. In terms of confusion matrix, we only care about the recall and precision because they show the model's performance in predicting customer churn. Meanwhile, it is the company's decision whether to place more importance on recall or precision. Here in F_β score, I set $\beta = 2$, which means more importance is given to recall for the reason that I don't want to miss any potential churn customer. ROC and PR score are also presented in every model so as to demonstrate more comprehensive model classification capabilities.

4.3 sample imbalance

There contains only 5% of the churn samples within the whole dataset. However, our purpose is to identify the potential churn customers. As a result we should focus more on the recall and precision which represents the model's performance on identifying the churn customers. With such imbalanced proportion of targets, the model will inevitably identify all samples as unchurned to reduce loss. For example, in a logistic regression model, the model tends to identify all the samples as unchurned because nearly 95% samples are unchurned. In this way we get a converged model with low loss and high accuracy up to 94.88%. But it is clearly not what we want. So, I tried different ways including Oversampling, undersampling, SMOTE algorithm, Borderline-SMOTE algorithm, ADASYN algorithm, and changed the class weight hyperparameter. Taking logistic regression model as an example, the following table represents the model performance after different modification.

With sample imbalance modification, the model tends to shift its focus to small samples. However, as we can see in the table, simply implementing modifications doesn't bring huge improvement to the model's performance. From EDA we find that the churned and unchurned group features have large overlap, indicating that it is difficult for the model to generate clear decision boundaries.

Table 3: Methods of processing sample imbalance in logistic regression

Modification	**Accuracy**	**Confusion Matrix**	**Recall**	**Precision**
No modification	0.9488	0 65 0 1205	0.0	0.0
Class weight = "Balanced"	0.5567	32 33 530 675	0.4923	0.0569
Undersampling	0.5583	32 33 528 677	0.4923	0.0571
SMOTE algorithm	0.5669	29 36 514 691	0.4462	0.0534
Borderline-SMOTE algorithm	0.5789	33 32 462 743	0.5077	0.0667
ADASYN algorithm	0.5720	35 30 557 648	0.5385	0.0591

As a result, I believe that additional information must be introduced into the model to help achieve better classification. Such additional information should help the model distinguish the features of two groups and expand the samples' distance from the decision boundaries. EDA can be a handy assistant for us to get the additional information.

4.4 Logistic Regression

Here I implemented data standardization, SMOTE algorithm and Class weight = "Balanced" to train the model. Simply using Logistic Regression, the model reached 0.61 AUC with 9.16% precision and 50.79% recall. Also, I listed the coefficients of each feature in the Logistic Regression Model in the following table. It can be seen that the top three influencing factors are CHI, DaysLastLogin and Customer_Age_inmonths.

Table 4: Coefficients of features

Feature	**Coefficient**
Customer_Age_inmonths	-1.511754707785039
CHI_M0	2.269299745235857
CHI_M01	2.998675623376894
Support_M0	0.5365021124591173
Support_M01	-0.3169580862443928
SP_M0	0.06423125939213703
SP_M01	-0.6810421926744276
Logins_M01	0.7654800973561413
Blog_M01	0.210308056287261
Views_M01	0.22608461618013515
DaysLastLogin_M01	-1.7296679695495323

Meanwhile, as we found in the section EDA, Customer_Age_inmonths in 6 14 month tends to have a bigger probability to churn while Customer_Age_inmonths <6 or >14 month tends to stay. As a result, we shouldn't put Customer_Age_inmonths as a linear regression sub into the regression. In this case I created 3 new regression sub named as 'age0-6', 'age6-14', 'age$>$14' instead of Customer_Age_inmonths and ran the regression again. The result was very promising. AUC reached a new high of 0.69. As we can see in the table, the coefficient of age6-14 is 0.08 while age0-6 is 1.59, indicating Customer_Age_inmonths in 6-14 month has a significantly bigger probability to churn than that in <6 month. This also confirms the situation in Figure 6.

Table 5: Coefficients after age grouping

Feature	Coefficient
CHI_M0	2.8112522341779886
CHI_M01	2.4315092152283455
Support_M0	-0.8400921049910017
Support_M01	-1.78198273155569
SP_M0	-0.07098490525357841
SP_M01	-0.5762626086878374
Logins_M01	1.8082313760695083
Blog_M01	0.679889325853613
Views_M01	1.0535002955698523
DaysLastLogin_M01	-3.980425753858899
age0-6	1.5922410363495894
age6-14	0.08141972438804576
age>14	0.26322785523925113

Here I displayed AUC and PR curve of Logistic Regression. After dividing the Customer_Age_inmonths variable into three groups, AUC increased from 0.61 to 0.69.

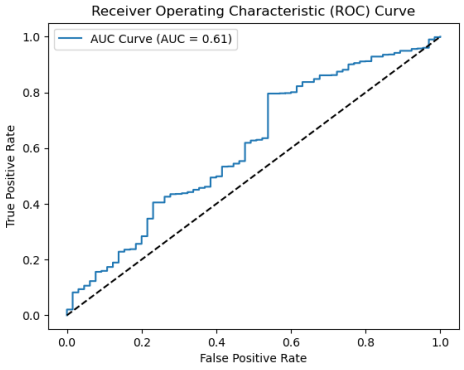


Figure 12: AUC Curve of LR

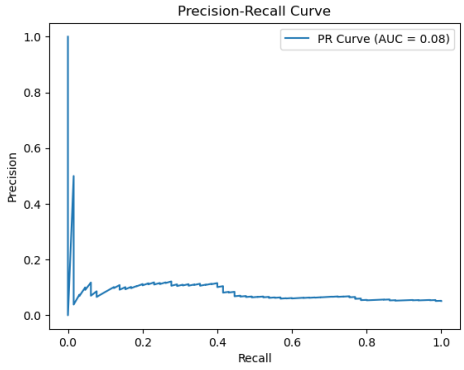


Figure 13: Precision-Recall Curve of LR

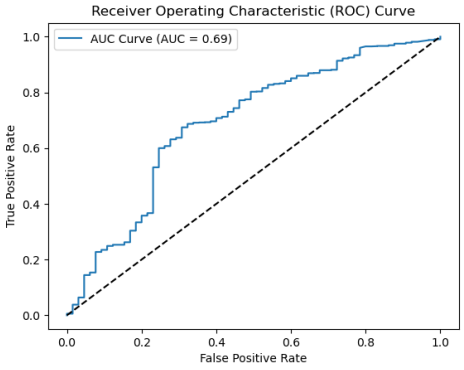


Figure 14: AUC Curve after grouping of LR

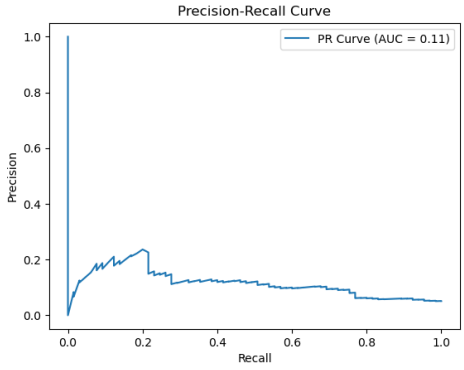


Figure 15: Precision-Recall Curve after grouping of LR

4.5 Random Forest

Random forest is a convenient way to perform classification. I conducted the Random Forest prediction and tried Ensemble learning with Logistic Regression, Random Forest and Gradient Boosting Decision Tree, hoping to increase the model's performance. However, ensemble learning seemed not to live up to my expectation and made little difference. Detailed code can be seen in the **random_forest_train.py** and **Ensemble learning.py** files.

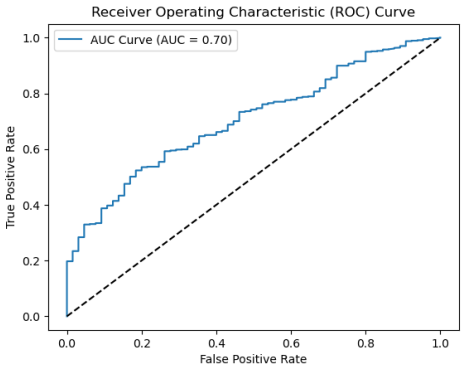


Figure 16: AUC Curve of Random Forest

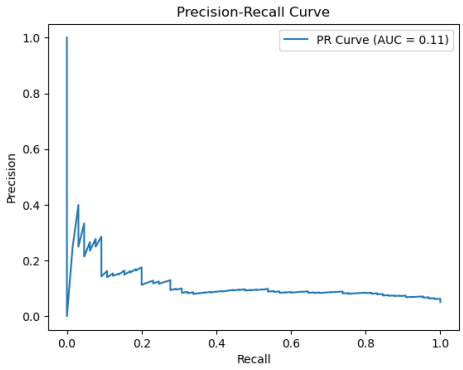


Figure 17: Precision-Recall Curve of Random Forest

Also I randomly chose a tree estimator in the random forest and visualized its classification results. Since the feature dimension is high, it is hard for us to get a straightforward understanding of the classification results. However, we can also find that those with few supports and low average support priority are more likely to churn.

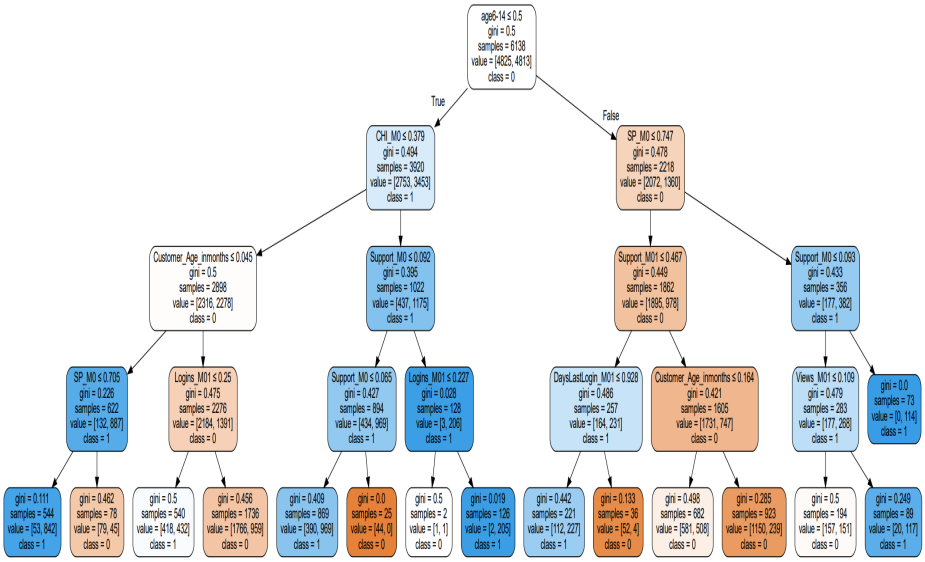


Figure 18: Visualization of tree estimator

4.6 SVM

SVM is effective in high-dimensional spaces and robust against overfitting. Meanwhile, SVM model can be computationally expensive as it takes much longer time to trian a SVM model. Ultimately SVM model reached the AUC score of 0.72, proving the good classification ability of SVM.

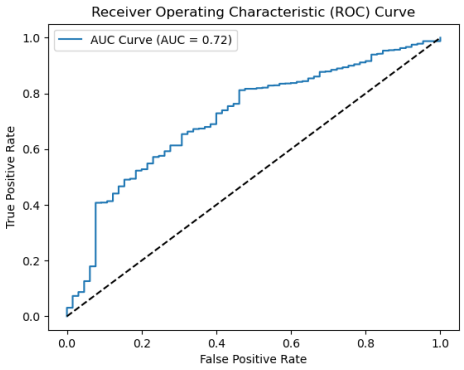


Figure 19: AUC Curve of SVM

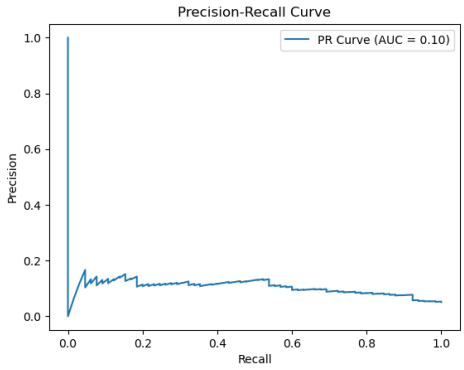


Figure 20: Precision-Recall Curve of SVM

4.7 Neural Network

Here I used Pytorch to build a basic BP neural network. It has two linear layers. The output of the first linear layer is activated by relu and the output of the second linear layer is activated by Sigmoid to give the predicted classification probabilities. Relying on the nonlinear decision boundaries and rich learnable parameters, neural network has excellent learning ability. Here I used the BCEWithLogitsLoss loss function. Also I implemented the Adam optimizer to have adaptive learning rate and flexible gradient descent. I set the hidden union as 64, learning rate as 0.001, epochs as 30. By simpily carrying over the previous features, it reached an auc score of 0.74.

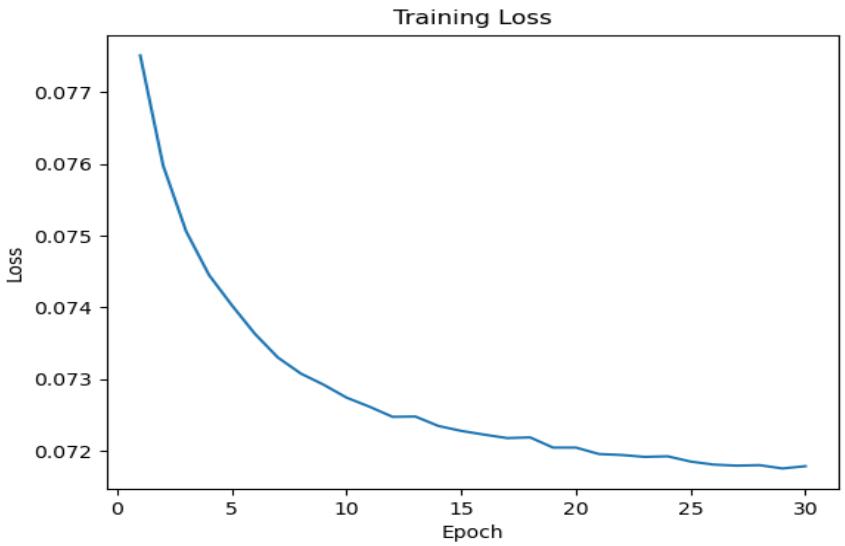


Figure 21: Training loss of Neural Network

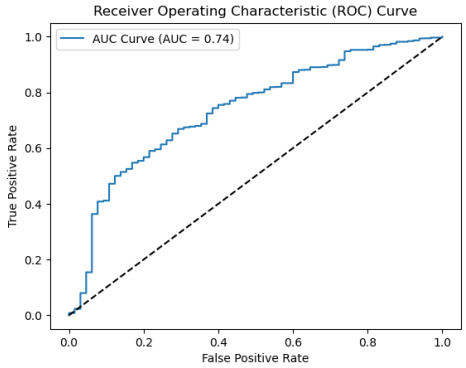


Figure 22: AUC Curve of Neural Network

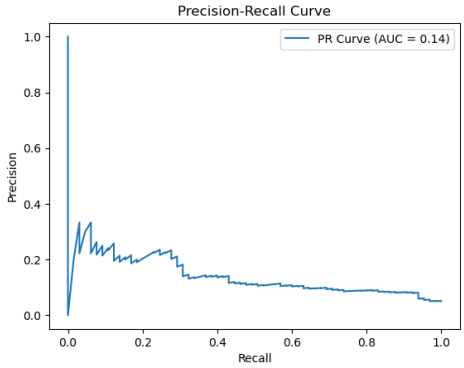


Figure 23: Precision-Recall Curve of Neural Network

Considering the limited performance of SMOTE algorithm, here I implemented GAN to generate small samples. GAN (Generative Adversarial Network) is a framework that consists of two neural networks: a generator network and a discriminator network. The generator network generates synthetic data samples, while the discriminator network learns to distinguish between real and fake data.

GAN can generate synthetic samples that closely resemble the real data. The generator learns the complex data distribution, allowing it to produce samples that are more realistic and representative of the minority class. In contrast, SMOTE generates synthetic samples by interpolating existing samples, which may not capture the true data distribution as effectively.

In our model, first I trained GAN on the churned data and then generated 5000 new churned samples based on the trained GAN. With the new generated churned data and the original data, the total training data now has equal churn and unchurned samples. Second, I continued to use the neural network model and trained it on the new total data. As we can see, different from SMOTE which generates new data based on its neighbours, GAN is a trained model with better generalization ability. You can check the detailed code in **gan.py file**.
Eventually Neural Network with GAN reached the recall of 0.6154 and precision of 0.1538. Compared with other sample imbalance algorithms(precision = 0.09), model with GAN increased precision significantly while maintaining high value of recall. With the help of GAN, we obtained the highest value of AUC —— 0.77.

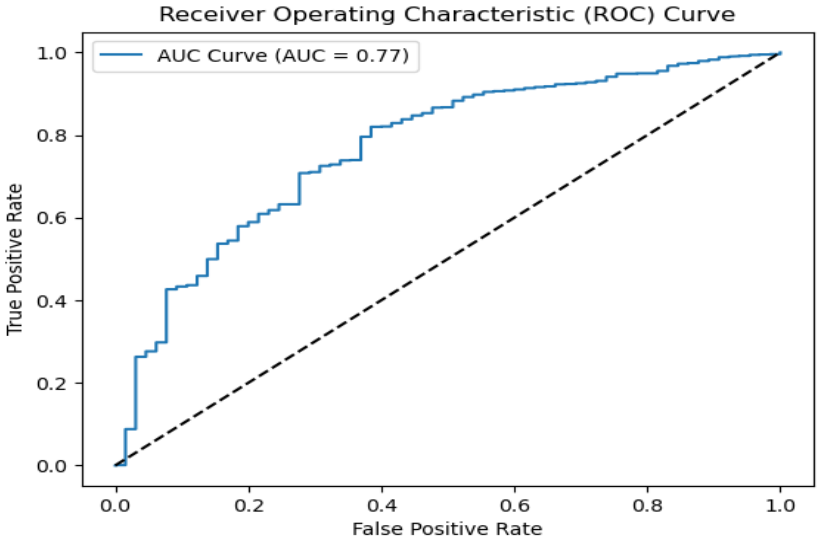


Figure 24: AUC Curve of Neural Network with GAN

5 Data Insights and Conclusions

From the analysis above, I explored the correlation between different features and customer churn rate. Customer longevity is a significant factor related to churn rate. The difference in customer age among churned group versus unchurned is clear. It is worth mentioning that the impact of customer longevity on churn rate is not linear. As I have explored in EDA, customer longevity from 6 months to 14 months accounts for the largest percentage of churned customers. Shorter or longer customer longevity will reduce the rate of churn. The huge improvement of performance after grouping also proves that. As a result more attention should be paid on the 6-14 Customer_Age_inmonths groups. From this case we can see that implementing EDA before running regression is critical.

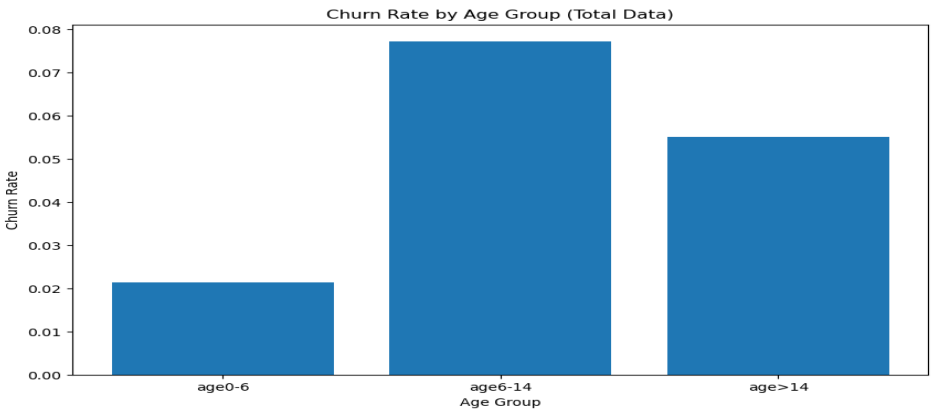


Figure 25: Churn Rate by Age Group (Total Data)

From the Pearson correlation coefficient test(Table 1) and the coefficients of logistic regression(Table3 and 4), we can find that Customer_Age_inmonths, CHI, Logins_M01 and Support_M01 have large impact on churn rate. As we can see in the density figures in the EDA section, churned customers tend to have relatively low CHI, decreased login time and unchanged support cases. It is easy to understand that customers with low happiness index are more likely to churn. Our data also confirms that, as a result of which we should pay attention to the Customer Happiness Index, especially those with low index.

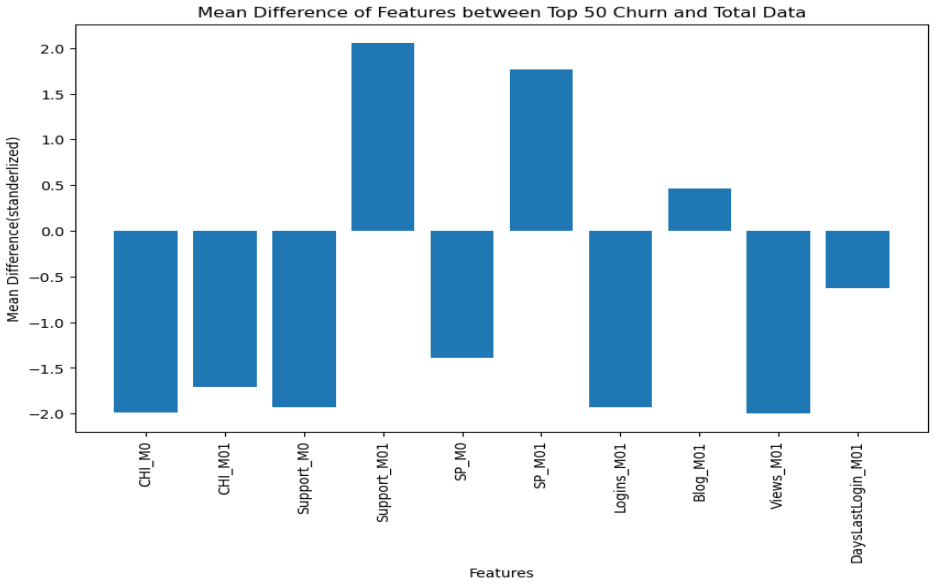


Figure 26: Mean Difference of Features between Top 50 Churn and Total Data

In the above figure I selected the top 50 potential churn customers and compared their standardized features with total data. As we can see from the bar chart, churn customers tend to have relatively lower CHI, fewer logins and views than those of total data, which illustrates the significance of customer usage Indicators like views and logins. What's more, the

difference of support and sp(M01) in churn customers is higher than average, while support and sp(M0) of churn customers is lower than average. That indicates churn customers normally have few number of support cases and low priority, but they demand more support and higher priority than usual a month before they churn. One possible explanation is that they are experiencing difficulties during this time, which increase their probability of churn next month.

As a result, here are my suggestions for the QWE company:

1. Pay more attention to the medium Customer_Age_inmonths groups.
2. Customers with low Customer Happiness Index should be valued.
3. Focus on customers who don't usually seek support but suddenly have a large increase in demand for support.
4. Customer usage data like logins and views should not be ignored. Customers who don't login and browse frequently are more likely to churn.