Analysis of the Xgboost algorithm model for precise online marketing to bank customers.

Antonio(Songming Zhou), Xu XinYi

August 25, 2023

1 Project Introduction

In recent years, the nation has been vigorously promoting a comprehensive digital transformation. At the beginning of 2020, with the outbreak of the COVID-19 pandemic, online digital business processing became an essential means to prevent the spread of the virus and guide crowd diversion. Mobile banking apps have thus become a crucial tool for banks in their digital transformation journey. In the future, mobile banking is bound to become a super platform for banks to build a scenario ecosystem and cultivate user value.

The number of active mobile banking customers is a critical indicator reflecting the effectiveness of digital transformation. Currently, the Hunan branch of the Bank of China faces challenges with a low monthly average of active mobile banking customers and easy customer attrition. They have taken many measures to enhance customer activity. In the past, they often used expert experience rules to bulk select groups for mobile recharge message sending. However, these expert rules, often derived from multi-round marketing experiences, can be inefficient and unstable.

To delve deeper into mobile banking operations and marketing types to reduce user attrition, we employed machine learning algorithms in this project. For instance, when analyzing users' reactions to different marketing types, machine learning algorithms might discover and learn certain behaviors or characteristics of users (like frequently browsing certain products or purchasing behaviors within a specific time frame) that correlate with their reactions to a particular marketing campaign. Once the machine learning model learns this relationship from historical data, it can predict which marketing activities these users might respond positively to when given new user behavior data.

Machine learning can analyze customer consumption behaviors, online banking activities, and other data to help banks segment the market more precisely, thereby offering more personalized services.

2 Project Target

The project objective refers to the overall goal set at the beginning of project planning.

2.1 Target One

- Identify the connection between online marketing campaigns of mobile banking and user characteristics based on complex and redundant data. Use machine learning models to learn the relationships between marketing campaigns and various features, such as the connection between occupation and marketing campaign preferences. Select the most influential features that affect marketing campaign preferences.

2.2 Target Two

- After identifying the key features that most influence marketing campaign preferences, we train machine learning models using algorithms like XGBoost to recognize and learn the relationship between marketing campaigns and these features. We then use a test set to predict the preferred activities for each customer with different features and compare the results to calculate metrics such as accuracy and recall rates to evaluate the model's performance on the test set.

3 Project Progress Steps and Logical Analysis

3.1 Problem Understanding and Data Collection

Define the project objectives and requirements. Gather all available data, including target variables (e.g., mobile banking SMS marketing type) and feature variables (e.g., age group, mobile banking SMS activity level, number of days logged into mobile banking, etc.).

3.2 Data Exploration and Preprocessing

Firstly, we need to ensure our data is clean and formatted correctly. This involves data cleaning, handling missing values, treating outliers, converting data types, encoding categorical variables (for instance, the ”occupation” field can be processed using one-hot encoding or label encoding), and standardizing or normalizing numerical variables.

3.3 Feature Engineering

This is a crucial step that can significantly enhance model performance. In our case, we can create new features to capture more information. For example, deriving a feature like ”active young users” based on age group and mobile banking SMS activity level. Then, we proceed with feature selection, eliminating irrelevant or redundant features. Here, we aim to capture features strongly correlated with our target, which is the 'SMS marketing type'. We'll employ the logistic regression machine learning algorithm, first hot-encode the different features, and then input the features and target into the machine learning model. This allows the model to learn the relationship between different features and the target, which is then reflected through printed coefficients.

3.4 Model Training and Validation

Here, we've used both the logistic regression algorithm and the decision tree algorithm to learn the relationship between different features and the target variable (SMS marketing type). We split the dataset into training and test sets, train the model using the training data, and then compute penalty coefficients for different features (these coefficients reflect the relationship between features and the target). After identifying features with strong correlations, we input these features into the XgBoost machine learning model. This model also learns the relationship between different features and the target variable (SMS marketing type). We then use the test set to predict user preferences for different marketing types based on their features and evaluate the model's performance using appropriate metrics (e.g., accuracy, area under the ROC curve, etc.).

3.5 Model Optimization

Based on the model's performance on the training and test sets, we optimize its parameters to enhance its predictive performance. This might involve techniques like cross-validation and grid search.

3.6 Model Evaluation

Evaluate the model's performance on the test set using various metrics such as precision, recall, F1 score, AUC-ROC, etc.

3.7 Model Deployment and Monitoring

Deploy the model in a production environment for real-time predictions. Also, monitor the model's performance to ensure the accuracy of its predictions. If the model's performance deteriorates, it might need retraining or adjustments.

3.8 Results Interpretation and Reporting

Interpret the model's results, extract valuable insights, and then draft a report or presentation to share your findings with stakeholders.

3.9 Task dismantling

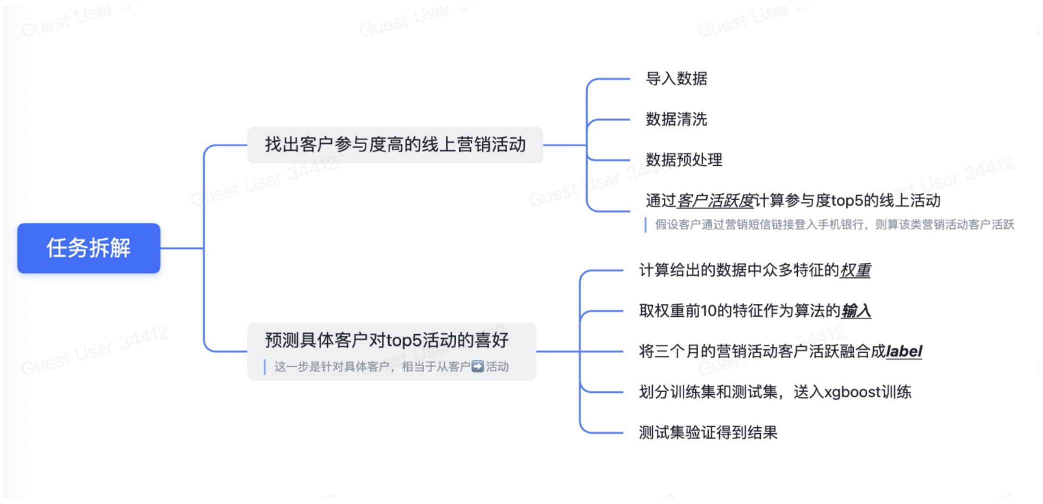

任务拆解
找出客户参与度高的线上营销活动
导入数据
数据清洗
数据预处理
通过客户活跃度计算参与度top5的线上活动
假设客户通过营销短信链接登入手机银行，则算该类营销活动客户活跃
预测具体客户对top5活动的喜好
这一步是针对具体客户，相当于从客户➡活动
计算给出的数据中众多特征的权重
取权重前10的特征作为算法的输入
将三个月的营销活动客户活跃融合成label
划分训练集和测试集，送入xgboost训练
测试集验证得到结果

Figure 1: Task_dismantling_uml_graph

4 Project Detail Analysis

4.1 Should All Features Be One-Hot Encoded First?

Before processing the data, it's essential to understand its characteristics. Specifically, we need to identify the data type of each feature. Some features might be categorical (e.g., gender, region), while others might be numerical (e.g., age, income).

For categorical features, if they are nominal (i.e., without a natural order, like city names or colors), we typically need to apply One-Hot Encoding. This encoding creates a new binary feature for each possible category, allowing models to handle unordered categorical data.

If our categorical features are ordinal (e.g., ratings like "high," "medium," "low"), we can use Label Encoding to map each category to an integer. This approach assumes a natural order among categories, which should be meaningful for the prediction task.

For numerical features, one-hot encoding is usually unnecessary. However, other treatments, like normalization or standardization, might be needed. This is because features with vastly different scales might dominate the model, affecting its performance.

It's worth noting that tree-based models (e.g., decision trees, random forests, XGBoost) inherently don't require one-hot encoding. They can directly handle categorical features without converting them to numerical ones. However, if there are many categories, one-hot encoding might inflate data dimensions, potentially impacting model performance.

4.2 How to Address Data Imbalance in Machine Learning?

4.2.1 Resampling the Training Set

Different datasets can be used. There are two methods to balance an imbalanced dataset—undersampling and oversampling.

1.1. Undersampling

Undersampling balances the dataset by reducing the size of the abundant class. This method is preferable when there's ample data. By retaining all samples of the rare class and randomly selecting an equal number of samples from the abundant class, a balanced dataset can be obtained for further modeling.

1.2. Oversampling

In contrast, oversampling is suitable when data is scarce. It tries to balance the dataset by increasing the number of rare samples, rather than reducing the abundant class. New rare samples are generated using methods like duplication, bootstrapping, or Synthetic Minority Over-sampling Technique (SMOTE).

Both undersampling and oversampling have their merits, and their application depends on the use case and the dataset itself. Combining both can also be effective.

4.2.2 Using K-fold Cross-Validation

When using oversampling to address imbalance, cross-validation should be applied appropriately. Oversampling observes rare samples and generates new random data based on a distribution function. If cross-validation is applied after oversampling, we risk overfitting our model to a specific artificially induced outcome. This is why cross-validation should always be performed before oversampling, similar to feature selection. Only resampling data can introduce randomness to the dataset, ensuring no overfitting issues arise.

K-fold cross-validation involves randomly dividing the original data into K parts. One of these parts is chosen as the test data, and the remaining K-1 parts are used as training data. The process is repeated K times, each time selecting a different part as the test data. The results from the K experiments are then averaged.

4.2.3 Treat as a One-Class Problem

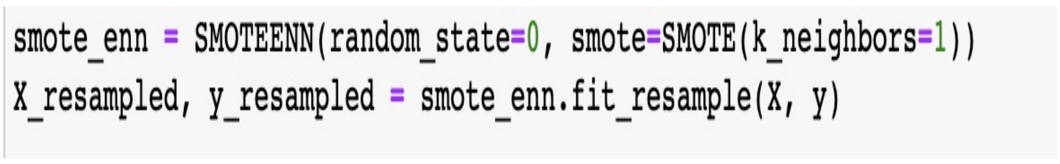
For binary classification problems with a highly imbalanced ratio between positive and negative samples, we can view it as a One-Class Learning or Novelty Detection problem. These methods focus on modeling one class rather than distinguishing between classes. Classic works include One-class SVM.

4.2.4 Combine Different Resampling Datasets

The simplest way to generalize a model successfully is to use more data. However, classifiers like logistic regression or random forests tend to generalize by discarding the rare class. A best practice is to build n models, each using all samples of the rare class and n different samples of the abundant class. If 10 models are to be combined, for instance, retain 1000 samples of the rare class and randomly draw 10000 samples of the abundant class. Then, simply divide the 10000 samples into 10 blocks and train 10 different models.

This is a combined sampling method, integrating Oversampling with Edited Nearest Neighbors (ENN) for Undersampling, known as SMOTE-ENN.

We also adopted this method in our project to balance data. Below is a demonstration code:



```
smote_enn = SMOTEENN(random_state=0, smote=SMOTE(k_neighbors=1))
X_resampled, y_resampled = smote_enn.fit_resample(X, y)
```

Figure 2: Demonstation_code_graph

4.2.5 Resample with Different Ratios

Method 4 can finely adjust the ratio between the rare and abundant classes. The best ratio largely depends on the data and model used. However, not all models are trained at the same ratio overall, so it's worth trying to combine different ratios. If 10 models are trained, it makes sense to have one with a 1:1 (rare:abundant) ratio and another with a 1:3 or even 2:1 ratio. The weight a class gets depends on the model used.

4.2.6 Cluster the Abundant Class

Sergey Quora proposed an elegant method, suggesting not to rely on random samples to cover training sample varieties but to cluster the abundant class into r groups, where r is the number of examples in r. Only the cluster center (medoid) is retained for each group. The model is then trained based on the rare class and the retained categories.

First, we can cluster the abundant class with a large number of samples. Suppose we use the K-Means clustering algorithm. In this case, we can choose the K value as the number of data samples in the rare class and treat the cluster centers and corresponding cluster centers as representative samples of the rich class, with the same class label as the rich class.

After the clustering operation described above, we have filtered the rich class training samples. Next, we can train with supervised learning using an equal number of K positive and negative samples.

4.2.7 Design Models Suitable for Imbalanced Datasets

All previous methods focus on the data, keeping the model as a fixed component. However, if a model is designed for imbalanced data, resampling isn't necessary. The renowned XGBoost is already a great starting point, so designing a model suitable for imbalanced datasets is meaningful.

Many models naturally generalize to the rare class by designing a cost function to penalize misclassification of the rare class rather than classifying the abundant class. For example, adjusting SVM to penalize misclassification of the rare class.

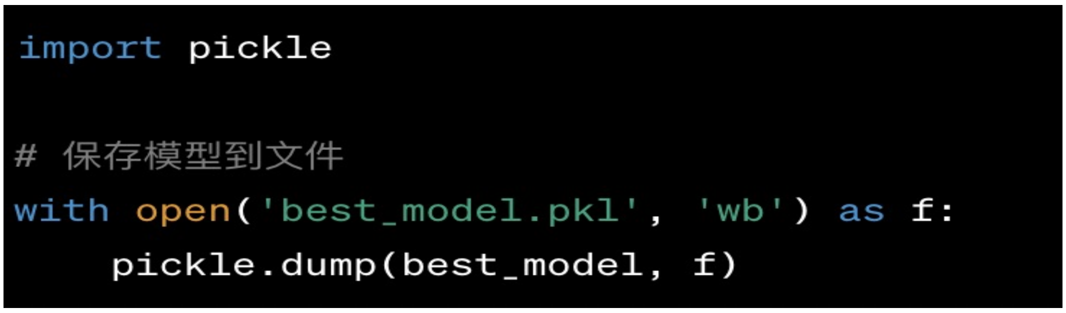
4.3 How to Avoid Wasted Time in Model Training

A challenge observed when training advanced gradient boosting models (like Xgboost) on a set of complex feature columns is the computational intensity of the training process. This not only prolongs the training duration but might also lead to operational issues, like losing the training set due to mishaps or other time factors during validation set evaluation.

To address this and maximize computational efficiency, one strategy is to serialize and export the best model to persistent storage after training. In subsequent experiments or validation phases, we can directly load the pre-trained model, saving time and resources needed for retraining. This method not only enhances workflow robustness but also offers flexibility for further model validation and deployment.

4.3.1 Exporting the Trained Model

Method to export the model to a file using the pickle library:



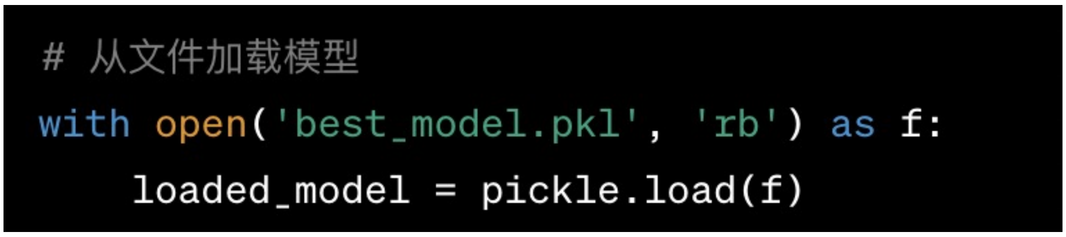
```
import pickle

# 保存模型到文件
with open('best_model.pkl', 'wb') as f:
    pickle.dump(best_model, f)
```

Figure 3: Exported_Trained_Model_graph

4.3.2 Reloading the Model

To reload the model, follow this method:



```python
# 从文件加载模型
with open('best_model.pkl', 'rb') as f:
    loaded_model = pickle.load(f)
```

Figure 4: Reloaded_Model_code_graph

5 Model Predictions and Algorithm Results

5.1 Logistic Regression Model Calculates Penalty Coefficients for Different Features

In the realm of data science and machine learning, logistic regression is a widely used statistical model for predicting binary outcomes. In our context, this model is used to infer the relationship between different features and the target marketing type.

For instance, suppose our marketing type of interest is a binary classification problem. The logistic regression model can help us understand how different customer attributes influence their response to a specific marketing type. By training the logistic regression model, we can obtain the weight or so-called penalty coefficient for each feature. These coefficients describe the mathematical relationship between the features and the response variable.

5.1.1 Weight Interpretation

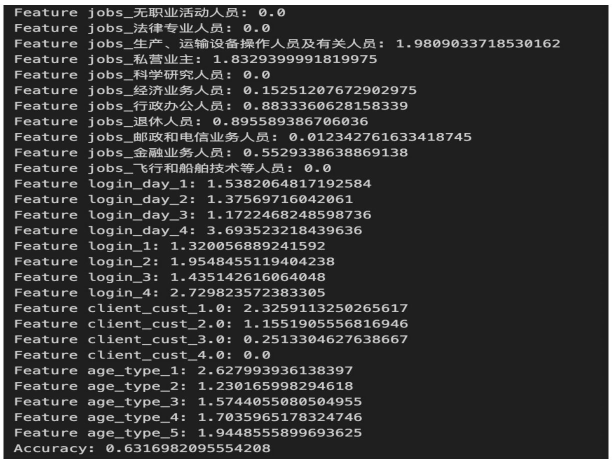
The weight of each feature informs us about the direction and strength of its relationship with the target variable. A positive weight indicates a positive correlation between the feature and the target category, while a negative weight indicates a negative correlation. The absolute value of the weight indicates the degree of influence the feature has on the target category.

5.1.2 Model Validation

To ensure the reliability and accuracy of the model, it's common to evaluate its performance using metrics like cross-validation, confusion matrix, precision, recall, etc. These validation steps help us understand the model's generalization capability on unseen data and provide guidance for further optimization.

5.1.3 Obtaining Penalty Coefficients

Display of penalty coefficients and model accuracy:



```
Feature jobs_无职业活动人员: 0.0
Feature jobs_法律专业人员: 0.0
Feature jobs_生产、运输设备操作人员及有关人员: 1.9809033718530162
Feature jobs_私营业主: 1.8329399991819975
Feature jobs_科学研究人员: 0.0
Feature jobs_经济业务人员: 0.15251207672902975
Feature jobs_行政办公人员: 0.8833360628158339
Feature jobs_退休人员: 0.895589386706036
Feature jobs_邮政和电信业务人员: 0.012342761633418745
Feature jobs_金融业务人员: 0.5529338638869138
Feature jobs_飞行和船舶技术等人员: 0.0
Feature login_day_1: 1.5382064817192584
Feature login_day_2: 1.37569716042061
Feature login_day_3: 1.1722468248598736
Feature login_day_4: 3.693523218439636
Feature login_1: 1.320056889241592
Feature login_2: 1.9548455119404238
Feature login_3: 1.435142616064048
Feature login_4: 2.729823572383305
Feature client_cust_1.0: 2.3259113250265617
Feature client_cust_2.0: 1.1551905556816946
Feature client_cust_3.0: 0.2513304627638667
Feature client_cust_4.0: 0.0
Feature age_type_1: 2.627993936138397
Feature age_type_2: 1.230165998294618
Feature age_type_3: 1.5744055080504955
Feature age_type_4: 1.7035965178324746
Feature age_type_5: 1.9448555899693625
Accuracy: 0.6316982095554208
```

Figure 5: Penalty_Coefficients_graph

5.1.4 Applied in bank business

Finally, by analyzing the penalty coefficients of the features, we can better understand how different customer attributes influence their response to specific marketing campaigns. These insights can be directly applied to tailor marketing strategies to enhance customer response rates and ROI.

In summary, logistic regression not only offers us a powerful predictive tool but also reveals the key factors and mechanisms behind the issues we care about.

5.2 Xgboost Model Predicts Preferences of Users with Different Features

In this section, we utilize the Xgboost model to predict user preferences and link them to their specific attributes. Here are the key steps undertaken for this task:

5.2.1 Data Preparation

Initially, the code cleansed the dataset, removing rows with missing values, and queried the count of various activity types. The features include personal information, gender, shopping habits, etc. Additionally, normalization and one-hot encoding were performed.

5.2.2 Handling Imbalanced Data

Given the potential class imbalance in the dataset, we employed a combination of SMOTE and Edited Nearest Neighbours (ENN) for oversampling and undersampling to balance the categories before training the model.

5.2.3 Data Splitting

The dataset was split into training and test sets, with 40 percent of the data used for testing.

5.2.4 Model Training and Hyperparameter Tuning

We used the Xgboost classifier and employed GridSearchCV to find the optimal hyperparameters under cross-validation. The search space included different numbers of weak learners, learning rates, and maximum tree depths.

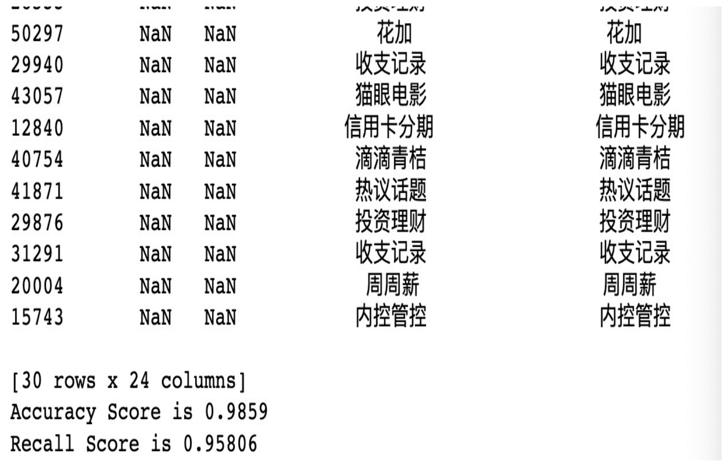
5.2.5 Prediction and Evaluation

The best model obtained from training was used to predict on the test set, and accuracy and recall were calculated. A confusion matrix was also plotted to visually display the model's performance.

5.2.6 Model Saving

For further use, the best model was saved as a pickle file.

5.2.7 Display Model Predicted Marketing Types and Accuracy



```
50297       NaN   NaN        花加              花加
29940       NaN   NaN      收支记录          收支记录
43057       NaN   NaN      猫眼电影          猫眼电影
12840       NaN   NaN     信用卡分期        信用卡分期
40754       NaN   NaN      滴滴青桔          滴滴青桔
41871       NaN   NaN      热议话题          热议话题
29876       NaN   NaN      投资理财          投资理财
31291       NaN   NaN      收支记录          收支记录
20004       NaN   NaN       周周薪            周周薪
15743       NaN   NaN      内控管控          内控管控

[30 rows x 24 columns]
Accuracy Score is 0.9859
Recall Score is 0.95806
```

Figure 6: Predicted graph

6 Project Conclusion

This project underwent various stages, from data preprocessing, feature engineering to model selection and tuning. Here's a summary of these steps and the primary conclusions of the project:

6.1 Data Cleaning and Balancing

The first step of the project involved data cleaning, where missing values were removed, and necessary preprocessing like normalization and one-hot encoding was done. Considering potential class imbalance, SMOTE and ENN techniques were used to balance the samples of each category.

6.2 Logistic Regression Feature Association Analysis

We inferred the association of different features with the target marketing type using the logistic regression model, identifying the features with the highest penalty coefficients with the target marketing type. This helped us understand the relationship between each feature and the response variable, enabling better model construction for subsequent steps.

6.3 Xgboost Model Predicts User Preferences

Next, we predicted the preferences of test set users using the Xgboost model. This phase covered the training of the model, hyperparameter tuning, and predictions on the test set.

6.4 Model Validation

The final stage involved validating the trained Xgboost model on a validation set. By observing the model's accuracy and other evaluation metrics, it was evident that the Xgboost model had superior predictive capabilities compared to the logistic regression model.

6.5 Summary

From the project's perspective, the introduction of the Xgboost model significantly enhanced prediction accuracy and provided in-depth insights into user preferences. Compared to the logistic regression model, Xgboost demonstrated greater capability and flexibility in handling the project dataset. Future work can continue to explore other advanced models and feature engineering techniques to further enhance prediction accuracy and interpretability. Additionally, the project emphasized the importance of balancing data and thorough data cleaning, which are key components of any successful data science project.

6.6 Future Directions

Despite the successes achieved, there are potential future directions worth exploring. These might include experimenting with different feature selection methods, using more complex models, and conducting more comprehensive hyperparameter tuning. These improvements might further enhance the model's performance and interpretability.