Insurance Prediction

DEBAJYOTI MAITY

July 17, 2023

1 Introduction

Insurance plays a vital role in today's society by providing financial protection against unforeseen events and risks. In this project, we aim to develop a predictive model that can accurately predict the likelihood of premium non-payment by policyholders.

The dataset used for this project consists of various features related to policyholders' demographics, payment history, underwriting score, and other relevant information. The features include *id*, *perc premium paid by cash credit*, *age in days*, *Income*, *Count 3-6 months late*, *Count 6-12 months late*, *Count more than 12 months late*, *application underwriting score*, *no of premiums paid*, *sourcing channel*, *residence area type*, *premium*, and the target variable *target* indicating premium non-payment.

The primary objective of this project is to train a machine learning model using the provided dataset to accurately predict whether a policyholder is likely to default on premium payments. By identifying policyholders who are at a higher risk of non-payment, insurance companies can proactively take necessary measures to mitigate potential losses and manage their portfolios more effectively.

In addition to developing the predictive model, we will also analyze the importance of different features in determining premium non-payment. This analysis will provide valuable insights into the factors that significantly influence the likelihood of non-payment and enable insurance companies to make data-driven decisions when assessing risks and setting premiums.

By leveraging advanced machine learning techniques and utilizing the provided dataset, we strive to create a model that can enhance the efficiency and effectiveness of premium payment management in the insurance industry.

2 Methodology

To develop an accurate predictive model for premium non-payment, we followed a systematic approach that involved the following steps:

2.1 Data Preprocessing

1. Train Data Preprocessing :

   Before training the machine learning model, we performed data preprocessing to ensure the data's quality and prepare it for analysis. There are 97 missing values in `Count_3-6_months_late` column. There are 97 missing values in `Count_6-12_months_late` column. There are 97 missing values in `Count_more_than_12_months_late` column. There are 2,974 missing values in `application_underwriting_score` column. No other column in the dataset has any missing value.

   Now I have to treat these missing values; otherwise, they'll affect my final predictions. I'll treat the missing values of all the three columns by filling up zero in place of the missing values. This is due to the fact that I don't know whether a customer is 0 premium late or 1 or 2 or more premiums late. The customer can be practically any number of premiums late, which is impossible to find out. So it'd be better if I assume he/she is 0 premium late. Because if I put 1 in front of someone who has really paid all of his/her premiums on time and I treat my model with these wrong values, then it'll result in bad predictions in the test dataset.

   Now for the missing values in `application_underwriting_score` column , I'll fill them with the mean of this column. There's a condition that applications under the score of 90 do not get insured. So obviously if I fill the missing values of this column with 0 and train my model then similar kind of data in test dataset will get affected. There are various methods to fill up a missing value, apart from using 0. Few of them are mean, median, mode, minimum value, maximum value, etc. The mean is the average of all the values of a particular column. The median is the middle value of all the values of the column. The mode is the most common value in the entire column. So filling the underwriting score with mean seems to be the best option.

2. Test Data Preprocessing : There are 31 missing values in `Count_3-6_months_late` column. There are 31 missing values in `Count_6-12_months_late` column. There are 31 missing values in `Count_more_than_12_months_late` column. There are 1323 missing values in `application_underwriting_score` column. No other column in the dataset has any missing value.

   Now I have to treat these missing values; otherwise, they'll affect my final predictions. I'll treat the missing values of all the three columns by filling up zero in place of the missing values. This is due to the fact that I don't know whether a customer is 0 premium late or 1 or 2 or more premiums late. The customer can be practically any number of premiums late, which is impossible to find out. So it'd be better if I assume he/she is 0 premium late. Because if I put 1 in front of someone who has really paid all of his/her premiums on time and I treat my model with these wrong values, then it'll result in bad predictions in the test dataset.

Now for the missing values in `application_underwriting_score` column , I'll fill them with the mean of this column. There's a condition that applications under the score of 90 do not get insured. So obviously if I fill the missing values of this column with 0 and train my model then similar kind of data in test dataset will get affected. There are various methods to fill up a missing value, apart from using 0. Few of them are mean, median, mode, minimum value, maximum value, etc. The mean is the average of all the values of a particular column. The median is the middle value of all the values of the column. The mode is the most common value in the entire column. So filling the underwriting score with mean seems to be the best option.

2.2 Feature Selection

To determine the most relevant features for predicting premium non-payment, we conducted feature selection The selected features are:

- `id`
- `perc_premium_paid_by_cash_credit`
- `Income`
- `Count_3-6_months_late`
- `Count_6-12_months_late`
- `Count_more_than_12_months_late`
- `application_underwriting_score`
- `no_of_premiums_paid`
- `sourcing_channel`
- `residence_area_type`
- `premium`
- `target`
- `age`

```
# Creating dummies of both train_x and test_x sets:
train_x = pd.get_dummies(train_x)
test_x = pd.get_dummies(test_x)
```

We convert the categorical value column to numerical value column .

2.3 Model Selection

We evaluated various machine learning algorithms suitable for classification tasks, such as logistic regression, decision trees,Naive Baise , K-nearest neighbors (KNN). We selected the model that provided the best balance between performance metrics, interpretability, and computational efficiency.

2.4 Model Training and Evaluation

We split the dataset into training and testing subsets, with a common practice of 80% for training and 20% for testing. We trained the selected model using the training data and evaluated its performance on the testing data. Evaluation metrics such as accuracy were used to assess the model's performance.

1. Logistic Regression : In the logistic regression model we get the accuracy 0.9363200901662441 in the train dataset(80%),0.9417694571410682 in the test dataset(validation dataset)

2. Decision Tree : In the decision tree model we get the validation accuracy 0.942834 .

3. Naive Baise : In the Naive Baise model we get the accuracy 0.9363 in the train dataset(80%),0.9417694571410682 in the test dataset(validation dataset) .

4. KNN : In the KNN model we get the accuracy 0.9417 in the train dataset(80%),0.9406 in the test dataset(validation dataset) for the best k is 11 .

2.5 Validation and Interpretation

To validate the model's performance and generalization capability, We see that the four model gives the approximately same accuracy . So we predict the test dataset by using this four model .

3 Results

The four model have the same score like the precision score is 0.9417694571410682 and the recall score is 1.0. This means that the model achieved a high precision, correctly identifying 94.18% of the positive predictions, and a perfect recall, correctly capturing all the actual positive instances in the test data.

The precision and recall values suggest that the model performs well in terms of correctly classifying positive instances and identifying all actual positive instances.

4 Future Work

Although the models achieved high precision and recall scores, there are several areas for future work and improvement. Some potential avenues to explore include:

1. Feature Engineering: Further investigate and engineer additional features that could provide valuable insights into the prediction task. This may involve domain knowledge exploration or incorporating external data sources to enhance the models' performance.

2. Model Ensemble: Explore ensemble techniques, such as model averaging or stacking, to combine the predictions of multiple models. Ensemble methods have the potential to improve overall prediction accuracy and robustness by leveraging the strengths of different models.

3. Imbalanced Data Handling: Address the potential issue of class imbalance in the dataset. Techniques such as oversampling the minority class or using weighted loss functions can help the models better handle imbalanced data and prevent biases towards the majority class.

4. Hyperparameter Tuning: Perform a more comprehensive search for optimal hyperparameters by employing techniques like grid search or Bayesian optimization. This can help fine-tune the models and potentially improve their performance further.

5. External Validation: Validate the models on unseen, real-world data to assess their generalization capabilities. Evaluating the models on different datasets or in different scenarios can provide valuable insights into their robustness and potential for real-world deployment.

6. Interpretability and Explainability: Investigate techniques to enhance the interpretability and explainability of the models' predictions. This could involve methods such as feature importance analysis, surrogate models, or generating explanations for individual predictions.

By addressing these future work areas, we can potentially enhance the models' performance, interpretability, and generalization capabilities, making them more valuable for real-world premium prediction tasks.