DESCRIPTION OF DESIGN CHOICES FOR THE MODEL

Building a Propensity Model for an Insurance Company

Introduction

The battle that most insurance marketing undergoes is comprehending customer behavior and, henceforth, predicting their actions. As a Data Scientist, I would be tasked with developing a propensity model that predicts whether people are likely to engage in certain actions, such as purchasing an insurance policy. This model will help the insurance company in optimizing its marketing efforts by using resources effectively on those who can be potential customers.

Why Information About Customers is So Important

Too many businesses capture a lot of information about their customers, but that goes to waste if it is not utilized properly. The point is to take all that data and compile it into insights that will basically help the company grow. Evolving one such way is propensity modeling.

What is Propensity Modeling?

Propensity modeling applies statistical techniques to compute the probability of a customer making a specific purchase, say, insurance. In doing so, it takes into account several variables that may influence their behavior: holder age, income, geographical location, and any other characteristic relevant to the case in hand. It helps a company to make a decision on which customers are likely to be interested in a given campaign and thus concentrate its efforts on them.

Data

An insurance company provided a file, train.csv, as a historical dataset, and test.csv containing the list of people whom they want to target with their marketing. From this list of people, one needs to determine the yes/no decision about whether to market to them by using the relevant columns provided in the datasets.

DATA PRE-PROCESSING:

MISSING VALUES HANDLING:

• **Identification of Missing Values:** There was a scan of missing values throughout the dataset.

• **Decision:** Instances with missing values, less than 5% of the total amount of data, were dropped in order to ensure that only instances free of missing values remained and were representative.

OUTLIER REMOVAL:

• **Method:** The Interquartile Range (IQR) method was employed for the detection of outliers in the case of numerical features.

• **Decision:** Outliers were removed from the dataset so that they didn't affect the model in its
understanding of the underlying patterns and relationships in the data.

STANDARDIZATION:

• **Approach:** StandardScaler was used for the standardization of numerical features to a mean of zero and a variance of one.

• **Purpose**: Standardization ensured that features were on a comparable scale, and this is
important for algorithms sensitive to feature magnitude, such as logistic regression.

FEATURE ENGINEERING:

CREATION OF NEW FEATURES:

• **Aggregated Information:** New features were engineered to calculate the sum and mean of many existing features, thereby capturing aggregated information that could enhance the predictive power of the model.

• **Non-linear Relationships:** Additional features, including the square of a feature ('x1') and the hour component of timestamp data, were derived to catch non-linear relationships and temporal patterns that may exist within the dataset.

MODEL SELECTION:

ALGORITHM CHOICE:

Classification Models Considered for Training

- **Logistic Regression**: Chosen for its simplicity and interpretability, making it easy to understand the relationship between the dependent variable and one or more independent variables. It is particularly useful for binary classification problems and provides probabilistic outcomes which can be beneficial for decision-making processes.
- **Decision Tree Classifier**: Selected for its ability to handle both numerical and categorical data and its interpretability through tree visualization. Decision trees provide clear, visual representations of decisions and their possible consequences, which is helpful for understanding how different variables contribute to the prediction.
- **Random Forest Classifier**: Considered for its robustness and ability to handle large datasets with higher accuracy and stability by averaging multiple decision trees. Random forests reduce overfitting by combining the results of multiple trees, leading to better generalization on unseen data.
- **Gradient Boosting Classifier**: Included for its ability to improve predictive performance by combining the strengths of multiple weak learners. Gradient boosting focuses on reducing errors from previous models and is effective in handling various types of data and complex relationships.
- **Support Vector Classifier**: Chosen for its effectiveness in high-dimensional spaces and its ability to find the optimal hyperplane for classification. SVCs are useful when there is a clear margin of separation between classes and are robust to overfitting, especially in high-dimensional feature spaces.
- **K-Nearest Neighbors Classifier (KNN)**: Considered for its simplicity and ease of implementation, especially useful for small to medium-sized datasets. KNN is a non-parametric method that makes predictions based on the proximity of data points, making it intuitive and straightforward to apply.
- **XGBoost Classifier**: Selected for its efficiency, speed, and performance, often outperforming other algorithms in classification tasks. XGBoost is highly optimized and scalable, making it suitable for large datasets and complex models, while also being effective in handling missing values and providing feature importance metrics.
- **Neural Network Classifier**: Included for its ability to model complex relationships in data and its flexibility in capturing non-linear patterns. Neural networks can learn

from a large amount of data and adapt to various types of input, making them powerful for capturing intricate patterns in the data.

MODEL TRAINING:

DATA SPLITTING:

• **Ratio:** The dataset was divided into a training and testing set in an 80-20 ratio.

• **Purpose**: This ensured the model was trained on a sufficiently large portion of the data, with a portion reserved for evaluating the performance of the model, thus reducing the risk of overfitting.

STANDARDIZATION:

Preprocessing: Numerical features were standardized with the StandardScaler before model training to mitigate the impact of varying scales and ease convergence during the model training process.

PERFORMANCE EVALUATION OF THE MODEL:

METRIC CHOICE:

• **Main Metric:** Accuracy was chosen as the main metric because it is straightforward and effective to use while measuring the general correctness of the predictions.

• **Other Metrics:** Additionally, precision, recall, and F1-score were evaluated to reveal a more detailed performance of the model in its ability to classify instances correctly and reduce false positives.

MODEL EVALUATION:

• **Training Set Performance:** The model has been evaluated on the training set, and high accuracy, precision, recall, and F1 score were achieved, showing that the model actually learned well the underlying trends in the data.

• **Test Set Performance:** The model was tested on unseen test data, and similar performance metrics were achieved, showing that it could generalize well to new instances.

• **ROC AUC Score:** The ROC AUC score was computed as a measure of how well the model could separate the two classes, and the results point to satisfactory discriminative power.

CONCLUSION:

• **Model Performance:** In this project, a logistic regression model achieved strong performance in predicting machine breakdowns with a set of provided features. A high accuracy was achieved on both the training and test datasets.

• **Success Metric:** The success metric for this project, set at 75%, has been met through an accuracy that exceeds it, and the validity of the approach used is thereby confirmed.

• **Insights:** From the thorough performance evaluation of the model using a suite of various metrics, very valuable insights were drawn, confirming that it was suitable for real-world applications.