Report on Predicting Employee Exit Status

Tanay Nagarkar
IMT2022083
Team Name - Tanay's Team

November 15, 2024

Introduction

This report documents the approach to solving the problem of predicting employee exit status. It covers all key steps, including data processing, experimental design, model selection, hyperparameter tuning, and insights from model performance. The goal is to develop a robust predictive model for identifying whether an employee is likely to leave a company, based on various organizational and personal factors.

1 Data Processing Steps

1.1 Data Loading and Overview

The training and test datasets were loaded to analyze the structure, column types, and any missing values. Initial rows and descriptive statistics were displayed to understand feature distributions and potential outliers.

1.2 Exploratory Data Analysis (EDA)

Missing values were identified and visualized. The class distribution of the target variable `exit_status` was analyzed using a count plot, confirming class imbalance. Distributions of key numerical features and relationships between categorical features and `exit_status` were plotted.

1.3 Handling Missing Values

Missing values in numerical columns, such as `monthly_income`, `dependents_count`, and `company_tenure`, were filled with the median. Categorical columns were filled with the mode to maintain consistency and prevent bias.

1.4 Feature Engineering

New features such as `years_at_company` (derived from `company_tenure`) and `family_size` (derived from `dependents_count` and `marital_status`) were created. A correlation matrix for numerical features was calculated to identify potential multicollinearity.

1.5 Class Imbalance Handling

The training data was resampled using `RandomUnderSampler` to balance the classes and improve model performance.

1.6 Feature Scaling

MinMax scaling was applied to the numerical features to prepare the data for models sensitive to feature scaling.

2 Experimental Design and Models

2.1 Target and Features Setup

The target variable `exit_status` was separated from the feature variables, where `X` represents all features except `exit_status` and `response_id`, and `y` represents the target variable. The test dataset, `X_test`, was prepared for making predictions without `response_id`.

2.2 Model Selection

The following three experiments were chosen based on data complexity and required performance:

- **Random Forest with GridSearchCV**: Initially used for hyperparameter tuning but resulted in an F1 score of around 0.73.
- **Random Forest with RandomizedSearchCV**: Improved performance slightly to around 0.74, but further improvement was needed.

- **CatBoost with RandomizedSearchCV**: Chosen due to CatBoost's suitability for categorical data and automatic handling of outliers and categorical encoding. This yielded the best performance, with an F1 score of 0.754.

2.3 Rationale for CatBoost

Categorical Data Handling: CatBoost internally manages categorical features without one-hot encoding, preserving feature relationships and reducing dimensionality.

Outlier Robustness: The model performs well on unscaled and unhandled outliers, which was a significant benefit given the dataset's range of values and missing data.

3 Hyperparameter Tuning

3.1 Parameter Grid Setup

A small parameter grid was configured for CatBoost's main parameters (iterations, learning rate, depth, and l2_leaf_reg) to optimize computational resources and performance. RandomizedSearchCV was run with a limited number of iterations for faster execution, applying a three-fold stratified cross-validation strategy to ensure stable F1 score estimates.

3.2 Best Parameters and Model Finalization

Best Parameters: Documented optimal values for each parameter, achieving a final F1 score of 0.754.
Final Model: Used the tuned CatBoost model, trained on all resampled data to make final predictions.

4 Model Performance and Insights

4.1 Cross-Validation

An Optimal number of folds cross-validation was conducted on the entire resampled dataset, recording F1 scores for each fold to validate the model's stability and generalization ability. The final mean F1 score across folds indicated consistent performance, supporting the choice of CatBoost for final predictions.

4.2 Feature Importances

CatBoost's feature importances were analyzed, revealing which features significantly impacted the model's predictions. Important features included `job_level`, `marital_status`, and `remote_work`. Surprisingly, features like `years_at_company` and `monthly_income` had minimal impact on predictions, suggesting they may not be as critical for this dataset.

4.3 Final Test Set Prediction and Submission

Predictions for the test set were generated and saved in the format required for Kaggle submissions (`response_id`, `exit_status`).

5 Additional Insights and Reflections

5.1 Data Insights

Factors such as `job_level` and `marital_status` suggest that structural workplace attributes and personal circumstances play substantial roles in employee retention. The low importance of traditionally significant features like `years_at_company` may imply that tenure and income are less critical in this specific organizational context.

5.2 Model Comparisons

While Random Forest performed reasonably, it required extensive preprocessing. CatBoost's ability to handle categorical data efficiently made it a better choice, reducing preprocessing complexity and improving model robustness.

5.3 Unexpected Findings

The lower-than-expected importance of `work_life_balance` suggests it may not be a decisive factor for employees in this dataset, contradicting general assumptions. This finding could reflect specific cultural or organizational factors influencing employee retention.

5.4 Accuracy Observations

Attempts to address outliers using the Z-score method resulted in a decrease in model performance, indicating that the model benefits from not removing outliers. Similarly, using the Iterative Imputer to handle missing values

led to a drop in accuracy. In contrast, the use of mean for filling missing numerical values and mode for categorical values yielded the best results. While SMOTE was tested for class balancing, it did not improve accuracy significantly compared to the RandomUnderSampler method. This suggests that balancing the classes with under-sampling might be more effective in this case. I observed that CatBoost internally handles outliers much better than other methods, and using external outlier correction techniques led to a decrease in accuracy. This reinforces the robustness of CatBoost when working with real-world, imperfect data.

6 Future Work

Exploring Neural Networks could provide an opportunity to improve the model's accuracy. Neural networks are capable of capturing non-linear relationships and complex patterns that other models, such as CatBoost, might miss. They have shown strong performance in predictive tasks involving large datasets, especially when fine-tuned for specific tasks like predicting employee exit status. Additionally, using Support Vector Machines (SVMs) could also be explored to improve the accuracy. SVMs are particularly powerful for classification tasks and could provide a good alternative to tree-based methods, especially when tuned for optimal performance.