CSIT332
Principles of Machine Learning

PROJECT REPORT

Machine Learning Final Project

December 10, 2024

Abhipsha Luitel
Computer Science and Design
FLAME University
abhipsha.luitel@flame.edu.in

1 Introduction

1.1 Background

Customer churn, where customers stop doing business with a company, poses significant challenges to industries worldwide. High churn rates can lead to substantial revenue loss. In competitive markets, understanding and mitigating customer churn is essential for maintaining a stable customer base and ensuring long-term profitability. Data-driven solutions, including predictive modeling, have emerged as effective tools to address this issue.

1.2 Problem Statement

This project addresses the problem of accurately predicting customer churn using historical customer behavior data. Despite advancements in machine learning techniques, many businesses struggle to implement robust churn prediction models tailored to their datasets. The specific challenge lies in identifying the factors contributing to churn and leveraging them to build an accurate and interpretable predictive model.

1.3 Objective

The primary objective of this project is to develop a machine learning model that can effectively predict customer churn. Key goals include:

- Analyzing the dataset to identify significant features contributing to churn.
- Building and evaluating machine learning models to achieve high predictive accuracy.
- Providing actionable insights to help businesses reduce churn and improve customer retention strategies.

1.4 Data

The dataset used for this project is the *Customer Churning Dataset*, which includes customer information such as demographics, service usage patterns, and past interaction history. The dataset consists features like customer tenure, service type, monthly charges, and payment methods. This dataset serves as the basis for feature extraction and model training.

2 Data Analysis and Preprocessing

2.1 Exploratory Data Analysis (EDA)

The dataset was first examined to understand its structure, identify missing values, and explore the distribution of key numerical and categorical variables. Statistical summaries and data types were reviewed using Python's `info()` and `describe()` functions.

Key insights include:

- Missing values were identified in the `TotalCharges` column, which were handled by converting the column to numeric and dropping rows with missing values.
- Numerical features such as `tenure`, `MonthlyCharges`, and `TotalCharges` were visualized using histograms and kernel density estimates to understand their distributions.
- A correlation heatmap was generated to analyze relationships between numerical features and the target variable `Churn_Yes`.

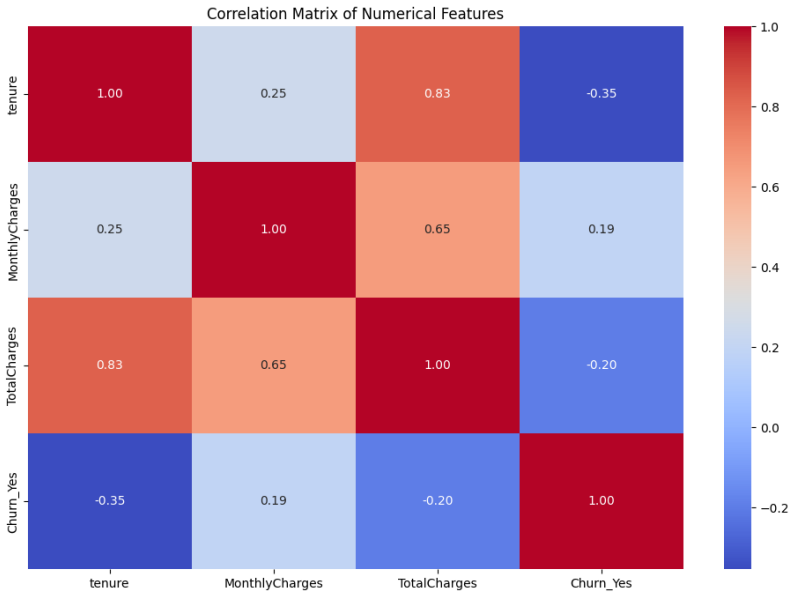


Figure 1: Correlation Matrix of Numerical Features

2.2 Data Cleaning

The cleaning process addressed issues such as missing values, outliers, and categorical encoding:

- Missing values in `TotalCharges` were handled by dropping affected rows after converting the column to numeric.
- Categorical variables were identified and encoded using one-hot encoding, ensuring the dataset was ready for machine learning models.
- Non-informative columns such as `customerID` were dropped.

2.3 Feature Engineering

New features were created to enhance the predictive power of the dataset:

- A `ChargeRatio` feature was engineered as the ratio of `MonthlyCharges` to `TotalCharges`.
- A `TotalServices` feature was calculated by summing the subscription statuses of various services (e.g., `OnlineBackup_Yes`, `DeviceProtection_Yes`).
- Feature scaling was applied using standardization to ensure numerical features were on a comparable scale for machine learning algorithms.
- Significant features were selected using a **Decision Tree classifier** (Figure 2). Features with importance scores above a threshold (e.g., 0.01) were retained, including `tenure`, `MonthlyCharges`, and `TotalServices`. This process reduced dimensionality and focused the dataset on predictive attributes.

2.4 Target Variable Analysis

The target variable `Churn_Yes` was analyzed to understand its distribution and relationship with other features. Churn rates for various categorical features were calculated, revealing significant patterns such as higher churn rates among certain feature.

2.5 Dataset Preparation

The cleaned and engineered dataset was split into training and testing sets, with an 80:20 ratio. The training set was used for model building, while the test set was reserved for evaluating model performance.

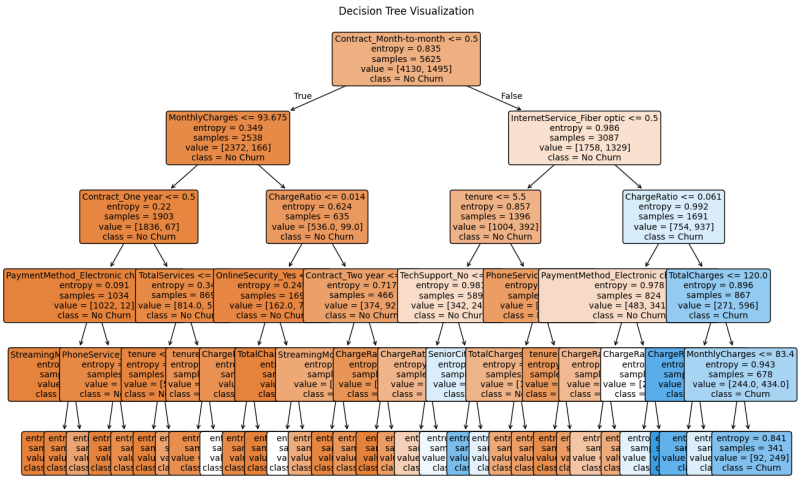


Figure 2: Decision Tree for Feature Selection

3 Model Selection and Implementation

In this section, multiple machine learning models are trained and evaluated using the concepts discussed. For each model, the architecture, hyperparameter tuning, and training and validation processes are described.

3.1 Model Architecture of Logistic Regression

Logistic Regression is a linear model used for binary classification problems. The model predicts the probability of the target variable belonging to a specific class using the logistic function. In this case, Logistic Regression was chosen due to its interpretability and suitability for problems where the relationship between features and the target variable is approximately linear.

3.1.1 Hyperparameter Tuning

The hyperparameters of Logistic Regression were optimized using GridSearchCV with a 5-fold cross-validation strategy. The following hyperparameter grid was defined:

- `C`: [0.001, 0.01, 0.1, 1, 10, 100] (Regularization strength)
- `penalty`: ['l1', 'l2'] (Regularization type)
- `solver`: ['liblinear', 'saga'] (Solvers for optimization)
- `max_iter`: [1000] (Maximum number of iterations)

3.1.2 Training and Validation

The model was trained with optimized hyperparameters and standardized training data for uniformity. Performance was validated on a test dataset using metrics like accuracy, precision, recall, F1 score, and ROC-AUC. A confusion matrix and ROC curve were plotted, confirming the model's effectiveness in predicting the target variable.

3.2 Model Architecture of Random Forest

The Random Forest classifier is an ensemble learning method that constructs multiple decision trees during training and outputs the class that is the mode of the classes output by individual trees. This approach helps reduce overfitting and improves predictive accuracy. Random Forest is chosen for its robustness and ability to handle non-linear relationships effectively.

3.2.1 Training and Validation

The Random Forest model was implemented with default hyperparameters. Specifically, the model was initialized with:

- `n_estimators`: 100 (Number of trees in the forest)
- `max_depth`: None (Unlimited depth for each tree)
- `random_state`: 42 (Ensures reproducibility)

The model was trained using the scaled training dataset. Its performance was validated on the test dataset, and the following metrics were computed to evaluate its effectiveness: Accuracy, Precision, Recall, F1 Score. The confusion matrix was generated to visualize the distribution of true positive, true negative, false positive, and false negative predictions. Additionally, the Receiver Operating Characteristic (ROC) curve was plotted to assess the model's ability to distinguish between the two classes.

3.3 Model Architecture of Neural Network

The Neural Network model implemented was a Multi-Layer Perceptron (MLP) classifier. This architecture is a type of feedforward artificial neural network that maps input data to output predictions using a series of interconnected layers. Neural networks are chosen for their capability to model complex, non-linear relationships in the data.

3.3.1 Training and Validation

The MLP classifier was implemented with the following default configuration:

- `hidden_layer_sizes`: (100,) (Single hidden layer with 100 neurons)

- `activation`: 'tanh' (Hyperbolic tangent activation function)
- `solver`: 'adam' (Adaptive Moment Estimation optimizer)
- `max_iter`: 2500 (Maximum number of iterations for convergence)
- `random_state`: 42 (Ensures reproducibility)

The model was trained on the scaled training dataset and evaluated on the test dataset. The following performance metrics were computed to assess its effectiveness: Accuracy, Precision, Recall, and F1 Score. The classification performance of the Neural Network was analyzed using a confusion matrix, which illustrates its ability to classify the test data correctly while identifying misclassifications. Additionally, a Receiver Operating Characteristic (ROC) curve was plotted to further examine the model's ability to distinguish between the classes.

3.4 Model Architecture of Support Vector Machine (SVM)

The Support Vector Machine (SVM) is a supervised learning algorithm primarily used for classification tasks. It works by finding the hyperplane that best separates the classes in the feature space, with a goal of maximizing the margin between the closest points (support vectors) of each class. In this implementation, the radial basis function (RBF) kernel was used, which allows the model to handle non-linear decision boundaries.

3.4.1 Training and Validation

The SVM model was configured with the following default parameters:

- `kernel`: 'rbf' (Radial Basis Function kernel)
- `C`: 1.0 (Regularization parameter)
- `gamma`: 'scale' (Kernel coefficient)
- `probability`: True (Required for probability-based metrics and ROC curve)
- `random_state`: 42 (Ensures reproducibility)

The model was trained on the scaled training dataset, and its performance was evaluated on the test dataset. The following metrics were computed: Accuracy, Precision, Recall, F1 Score. These evaluations indicate the model's effectiveness in classification tasks. To evaluate the classification performance, a confusion matrix was created, offering a detailed view of the correct and incorrect predictions. Additionally, the Receiver Operating Characteristic (ROC) curve was plotted to assess the model's ability to distinguish between the two classes.

3.5 Model Architecture of K-Means Clustering

K-Means clustering is an unsupervised machine learning algorithm used to partition data into clusters based on similarity. The model works by iteratively assigning each data point to one of k clusters, where k is the number of clusters chosen by the user. The algorithm optimizes the positions of the cluster centroids to minimize the variance within each cluster.

3.5.1 Training and Validation

The optimal number of clusters for the K-Means algorithm was determined using both the Elbow Method and Silhouette Scores. The process was as follows:

- **Elbow Method**: The Elbow method involves plotting the inertia (sum of squared distances from each point to its assigned cluster center) against the number of clusters. The optimal number of clusters is chosen at the "elbow" point, where inertia decreases at a slower rate.
- **Silhouette Score**: The silhouette score measures how similar each point is to its own cluster compared to other clusters. A higher silhouette score indicates better-defined clusters.

For this analysis, the number of clusters was set to 2, based on the plots generated by the Elbow Method.

The clustering process was then performed on the scaled training data, and the resulting cluster labels were added as a new feature to the training set. The silhouette score for the chosen clustering solution was calculated to evaluate the quality of the clusters.

3.5.2 Cluster Visualization

To visualize the clusters, Principal Component Analysis (PCA) was used to reduce the dimensionality of the data to two dimensions. The clusters and their corresponding centroids were plotted to provide a visual understanding of the data grouping. The centroids are marked with red "X" markers, and the data points are color-coded according to their cluster label.

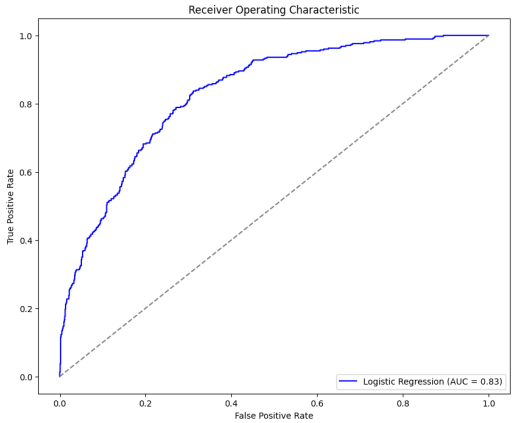
4 Models' Evaluation

4.1 Performance Metrics

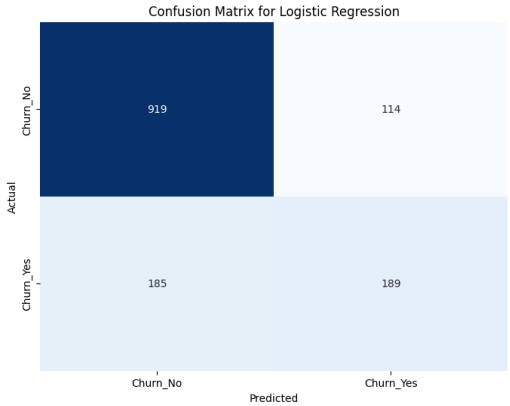
This section provides the evaluation of each model's performance on the test dataset.

4.1.1 Logistic Regression

- **Accuracy:** 0.7875
- **Precision:** 0.6238
- **Recall:** 0.5053
- **F1 Score:** 0.5583
- **ROC-AUC:** 0.83



(a) ROC Curve for Logistic Regression



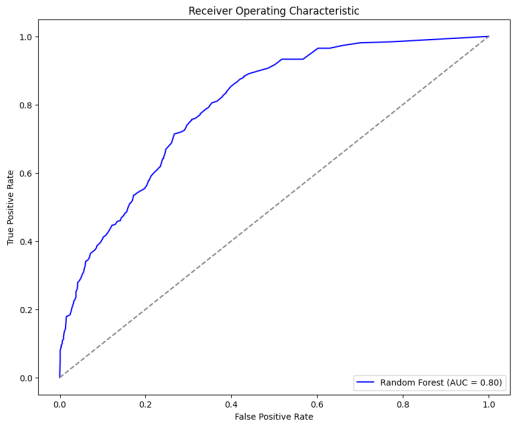
(b) Confusion Matrix for Logistic Regression

Figure 3: Performance Metrics of Logistic Regression Model

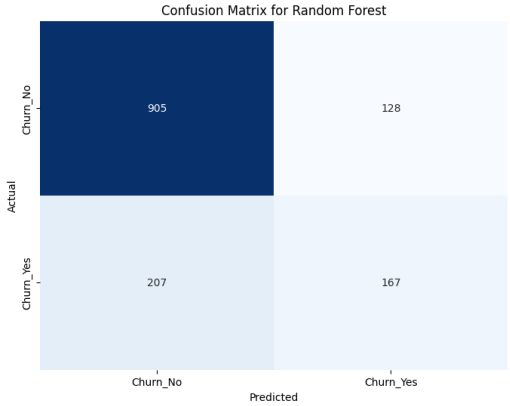
4.1.2 Random Forest

- **Accuracy:** 0.7619
- **Precision:** 0.5661

- **Recall:** 0.4465
- **F1 Score:** 0.4993
- **ROC-AUC:** 0.80



(a) ROC Curve for Random Forest

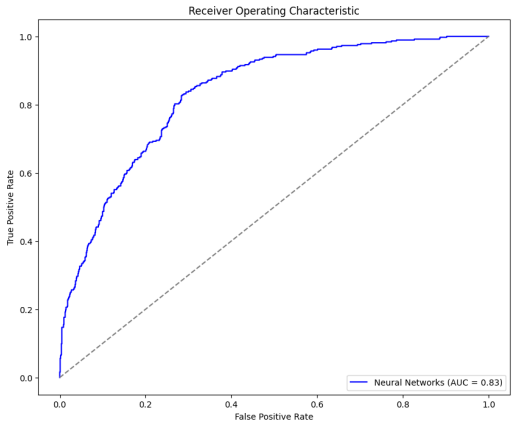


(b) Confusion Matrix for Random Forest

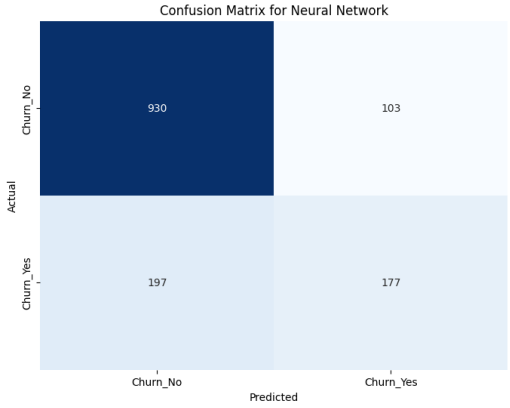
Figure 4: Performance Metrics of Random Forest Model

4.1.3 Neural Network (MLP)

- **Accuracy:** 0.7868
- **Precision:** 0.6321
- **Recall:** 0.4733
- **F1 Score:** 0.5413
- **ROC-AUC:** 0.83



(a) ROC Curve for Neural Network

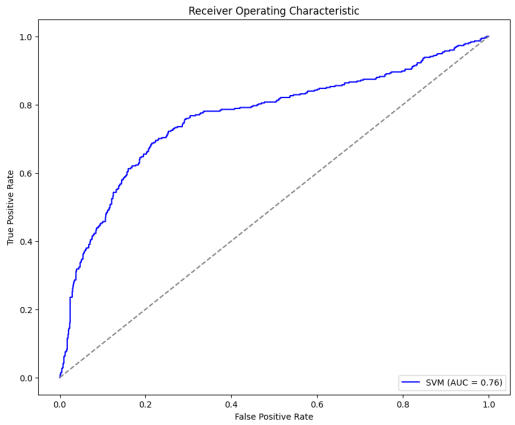


(b) Confusion Matrix for Neural Network

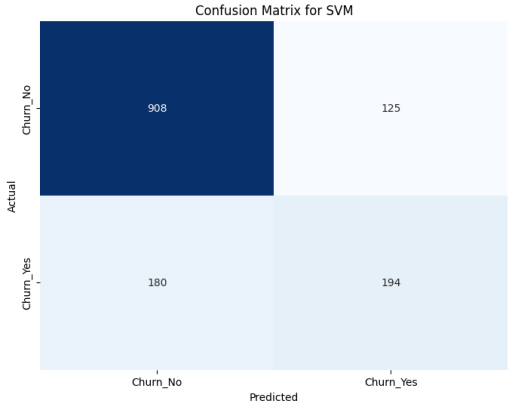
Figure 5: Performance Metrics of Neural Network Model

4.1.4 Support Vector Machine (SVM)

- **Accuracy:** 0.7832
- **Precision:** 0.6082
- **Recall:** 0.5187
- **F1 Score:** 0.5599
- **ROC-AUC:** 0.76



(a) ROC Curve for SVM

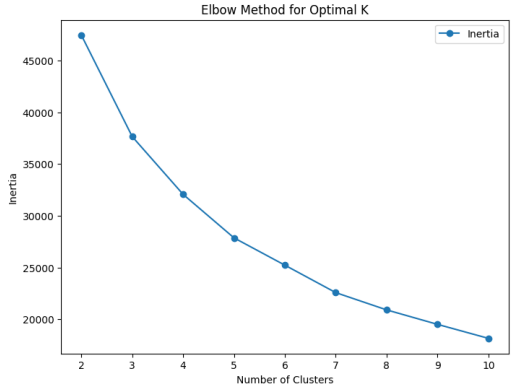


(b) Confusion Matrix for SVM

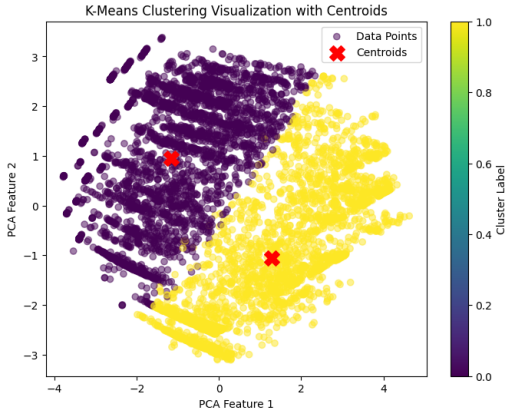
Figure 6: Performance Metrics of SVM Model

4.1.5 K-Means Clustering

- **Silhouette Score:** 0.2196



(a) Elbow-Method for Optimal K



(b) Clusters of K-means

Figure 7: Elbow Method and Clusters of K-means

5 Results and Discussion

5.1 Model Performance

The performance of four machine learning models Logistic Regression, Random Forest, Neural Network (MLP), and Support Vector Machine (SVM)âwas evaluated using accuracy, precision, recall, F1 score, and ROC-AUC. These metrics were calculated based on predictions made on the test dataset. Table 1 summarizes the comparative results.

- **Logistic Regression:**
  - Achieved the highest **accuracy** (0.7875) and **F1 score** (0.5583).
  - Demonstrated balanced performance across all metrics with **ROC-AUC** of 0.83, indicating strong separability between classes.
- **Random Forest:**
  - Achieved an **accuracy** of 0.7619 and had the lowest **F1 score** (0.4993), indicating room for improvement in predicting the minority class.
- **Neural Network (MLP):**
  - Showed the highest **precision** (0.6321), indicating fewer false positives.
  - However, its **recall** (0.4733) was comparatively lower, suggesting difficulties in identifying true positives.
  - Achieved an **ROC-AUC** of 0.83, demonstrating good classification ability overall.
- **Support Vector Machine (SVM):**
  - Had the highest **recall** (0.5187), making it effective for capturing true positives.
  - However, it achieved the lowest **ROC-AUC** score (0.76), indicating less robust separability between classes.

5.2 Comparative Study

Table 1 highlights the performance metrics of each model.

Table 1: Comparative results of the Machine Learning Models under the best configurations.

Model	Accuracy	Precision	Recall	F1 Score	ROC-AUC
Logistic Regression	**0.7875**	0.6238	0.5053	**0.5583**	**0.83**
Random Forest	0.7619	0.5661	0.4465	0.4993	0.80
Neural Network (MLP)	0.7868	**0.6321**	0.4733	0.5413	**0.83**
Support Vector Machine (SVM)	0.7832	0.6082	**0.5187**	0.5599	0.76

5.3 Discussion

The results indicate that:

- **Logistic Regression** is a well-balanced model, especially for datasets with linearly separable features. Its high **F1 score** highlights its ability to balance precision and recall effectively.

  Due to which Logistic Regression was selected as the final model and was hyper parameter tuned further to improve its performance.

  It was also employed to derive insights into factors influencing customer churn.

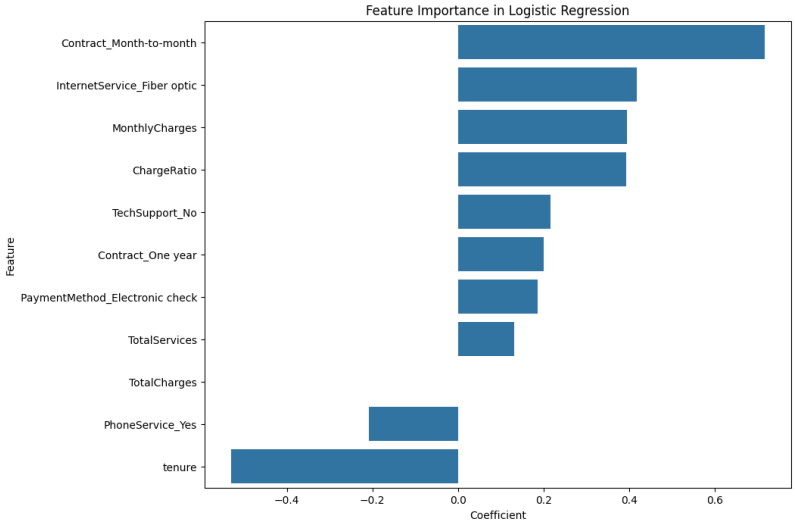


Figure 8: Feature Importance in Logistic Regression

5.3.1 Feature Importance Analysis

Key Observations:

- **Dominant Features:**
  - `Contract_Month-to-month` has the highest positive impact on the prediction, with the largest coefficient.

    - `InternetService_Fiber optic`, `MonthlyCharges`, and `ChargeRatio` are also highly important contributors.
  - **Negative Contributions:**
    - `Tenure` has the largest negative coefficient, indicating that longer tenure decreases the likelihood of the predicted event (e.g., customer churn).
    - `PhoneService_Yes` also has a smaller negative impact.
  - **Moderate Effects:**
    - Features like `TechSupport_No`, `Contract_One year`, and `PaymentMethod_Electronic check` have moderate positive contributions.
  - **Interpreting Coefficients:**
    - **Positive coefficients (right side):** These features increase the likelihood of the outcome.
    - **Negative coefficients (left side):** These features decrease the likelihood of the outcome.

  Lastly, it was also deployed on a sample dataset to make predictions on unseen customer records.

- **Random Forest** performs well in terms of **ROC-AUC**, but its lower recall and F1 score suggest that it may require hyperparameter tuning (e.g., increasing the number of estimators or adjusting the maximum depth) to enhance performance.
- **Neural Network (MLP)**, despite achieving high precision, suffers from relatively low recall, which may stem from its sensitivity to the data's complexity or overfitting to the training set. Adjusting the hidden layers or activation functions could improve its performance.
- **SVM** stands out in recall, making it suitable for applications where identifying true positives is crucial. However, its lower **ROC-AUC** suggests a need for better class separation, potentially through hyperparameter tuning.

5.4 Limitations and Future Work

- **Limitations:**
  - The models were trained without hyperparameter tuning, which may have limited their performance.
  - The dataset's size and class imbalance could have impacted the recall and F1 scores, especially for Random Forest and Neural Network.
  - SVM's lower ROC-AUC highlights a potential issue with feature scaling or class imbalance that needs further investigation.

- **Future Work:**
  - Perform hyperparameter optimization for all models to achieve their best configurations.
  - Explore feature engineering techniques or oversampling methods to address class imbalance.
  - Investigate ensemble methods to improve overall performance and robustness.
  - Evaluate models on additional datasets to ensure generalizability.

6 Conclusion

This project addressed the critical issue of customer churn, a significant challenge for businesses aiming to maintain customer loyalty and profitability. By leveraging the Customer Churning Dataset, we applied advanced data preprocessing, exploratory analysis, and feature engineering to develop machine learning models that predict churn with notable accuracy. Among the evaluated models, **Logistic Regression** emerged as the most interpretable and balanced option, achieving the highest F1 score and ROC-AUC, making it suitable for deployment in business environments.

6.1 Significance of the Work

The importance of this work lies in its practical application: predictive analytics empower businesses to proactively address customer churn by identifying and acting upon key factors of customer behavior. Our contribution includes not only the development of effective predictive models but also the provision of actionable insights into churn predictors, such as **contract type, tenure, and payment methods**.

6.2 Potential Improvements

Despite these successes, there are opportunities for improvement. With additional time, the project could benefit from:

- Comprehensive hyperparameter tuning across all models.
- Integration of ensemble techniques to enhance predictive performance.
- Advanced methods to address class imbalance in the dataset.
- Further refinement of feature engineering to incorporate domain-specific insights.

6.3 Future Work

As future work, we propose:

- Deploying the models on larger, real-world datasets to evaluate scalability and robustness.
- Developing real-time prediction capabilities for operational integration.
- Enriching the dataset with external data sources such as customer feedback for deeper insights.
- Expanding the deployment framework to include advanced customer retention strategies.

6.4 Final Remarks

By demonstrating the potential of machine learning to address customer churn, this project underscores the value of data-driven approaches in solving critical business problems. It not only validates the efficacy of predictive modeling but also opens pathways for future exploration and application.