Sri Lanka Institute of Information Technology

Artificial Intelligence and Machine Learning | IT2011

2025 | Year 2 Semester 1

Group ID: 2025-Y2-S1-MLB-B6G2-07

Customer Churn Prediction

Student ID	**Name**
IT24102374	Wickramaarachchi M. H
IT24101551	Kuyilini. T
IT24101536	Alwis P.G.R.M
IT24101325	Gammanpila J.P
IT24101520	Athukorala A. P. H. B
IT24101376	Nishshanka N.P.S.M.

B.Sc. (Hons) in Information Technology

1. Introduction and Problem Statement

In the highly competitive and saturated telecommunication market, customer retention has taken over customer acquisition as a key contributor to profitability. Large sums of money are spent by companies to acquire new subscribers, but winning a new customer is estimated to cost five times that of keeping an old one. Customer churn—the act of customers deserting a service provider—therefore translates into an enormous financial loss, running into millions of dollars per year for companies. The challenge for telcos is not so much how to react to churn when it happens, but to detect and retain high-risk customers actively before they churn.

This project addresses this business problem directly by framing customer churn as a binary classification problem. The main concern is to parse through massive amounts of customer data to detect predictive patterns. Through machine learning, we can move away from a reactive retention model to a predictive retention approach. This allows a business to act strategically with the right incentives, for example, price discounts or service improvements, before a customer is about to leave.

The primary aim of this report is to develop, implement, and verify a machine learning model that can well predict the churn of customers.

Project Objectives

- In order to achieve this aim, the project is motivated by the following crisp objectives:
- To perform thorough Exploratory Data Analysis (EDA) on the Telco Customer Churn dataset to identify significant features and trends that are associated with churn.
- To pre-process the data, handling missing values, encoding categorical variables, and scaling features to get a clean, model-ready dataset.
- To design, implement, and train various machine learning classification models.
- To critically compare and validate the models using applicable performance metrics (e.g., Accuracy, F1-Score, AUC-ROC) to establish the best performing solution.
- To discuss the ethical implications and data biases, in this instance the class imbalance, and provide mitigation strategies.

2. Dataset Description

In order to fulfill these objectives, this assignment utilizes the "Telco Customer Churn prediction" dataset provided through Kaggle. The dataset is highly relevant to the problem domain and includes a real-world approximation of a telecommunication customer base.

Dataset Scope and Structure

The dataset is tabular and contains **7,043 customer records (rows)**, with **21 features (columns)** for each customer. Of these features, 20 are predictors, and one is the target variable. The features are a robust mix of data types, capturing a holistic view of the customer:

- **Demographic Information:** Such as gender, senior citizen status, and whether the customer has partners or dependents.
- **Service Subscriptions:** Details on services the customer has, including phone, internet, online security, streaming, and tech support.
- **Account and Billing Information:** Contract type (e.g., month-to-month), payment method, paperless billing, monthly charges, and total charges.

Target Variable

The predictive target for this project is the "Churn" column. This is a binary categorical variable with two possible outcomes:

- **No** (Customer retained)
- **Yes** (Customer churned)

Justification and Limitations

This data set was selected due to its ideal blend of size and complexity. It is over 7,000 records large enough to support successful model training but small enough to be managed within the project scope. Its dense, multi-dimensional feature space is central to identifying advanced behavioral and demographic drivers of churn. The data is good with minimal preprocessing required

One of the primary considerations and possible sources of bias within this data set is its mild class skew. Approximately 26% of the data set's customers have churned. This skew is representative of the real business problem (as churning tends to be a minority event) but needs to be handled with particular care during modeling so that the model does not create a bias in favor of the majority "No Churn" class.

3. Preprocessing & Exploratory

3.1 Data Cleaning

Raw data also had to be sanitized to account for errors and missing entries:

Removal of Duplicates: The database was also scanned for duplicate rows, and 5 duplicates were removed to give 7043 unique records.

Missing Value Identification: The initial test revealed missing values for gender (1), SeniorCitizen (3), tenure (8), MonthlyCharges (10), and TotalCharges (3).

Processing Non-Numeric Data: Columns MonthlyCharges, tenure, and TotalCharges contained non-numeric data or were incorrectly typed as object. They were converted to numeric using pd.to_numeric(errors='coerce'), substituting illegal values with NaN.

Imputation

Imputed numeric columns (MonthlyCharges, tenure, TotalCharges) that were created resulting from the conversion or that exist inherently were substituted with the median for the columns

Non-categorical missing values (age, distanceToCenter) were replaced with the mean. Missing.

Categorical Standardization: Text data from the Object column were standardized through the process of converting to lower case and removing leading/trailing space. Inconsistent categories such as '12 months' were replaced with 'one year'. Observations with the ambiguous 'maybe' value from the Churn column were eliminated.

Outlier Handling: The outliers for tenure, MonthlyCharges, and Total Charges were found with the 1.5*IQR rule. The method found no outliers, so no rows were eliminated for this reason.

3.2 Feature Engineering

Following steps were performed for the conversion and feature selection for the models:

Encoding Categorical Features:

Label Encoding: Binary categories (Partner, Dependents, PhoneService, PaperlessBilling, Churn, gender) were changed to numerical (0s and 1s, etc.) using LabelEncoder.

One-Hot Encoding:

Multi-category variables (such as MultipleLines, InternetService, Contract, PaymentMethod) are one-hot encoded with pd.get_dummies with drop_first=True to prevent multicollinearity, increasing the size of the feature space. The size of the resulting dataset became (7042, 32)

Scaling Numerical Features:

The numerical features (tenure, MonthlyCharges, TotalCharges) were scaled with StandardScaler to unit variance and zero mean for being ready for algorithms that are sensitive to the scale of the features.

Dimensionality Reduction (PCA):

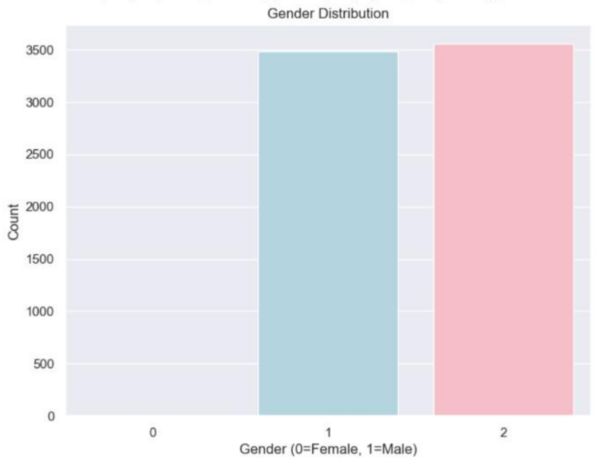
Principal Component Analysis (PCA) was applied to the remaining features (excluding the target 'Churn').

The number of components was chosen to retain **90%** of the variance, resulting in **14 principal components** being selected for the final model training dataset. This reduced the feature count from the original 29 features (after encoding and dropping ID/TotalCharges).

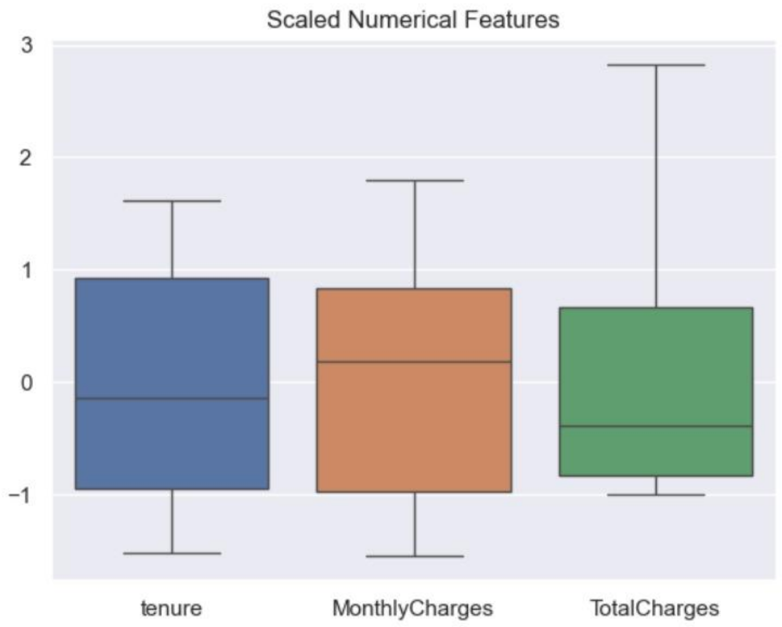
The final dataset containing these 14 components and the target variable was saved.

3.3 EDA

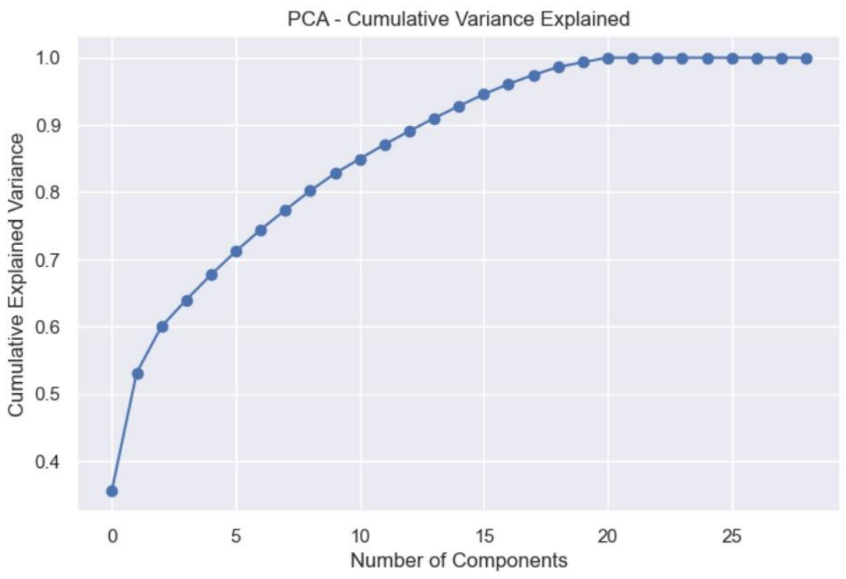
```
C:\Users\ASUS TUF\AppData\Local\Temp\ipykernel_21924\1781214837.py:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  ax = sns.countplot(x='gender', data=df, palette=['lightpink', 'lightblue'])
C:\Users\ASUS TUF\AppData\Local\Temp\ipykernel_21924\1781214837.py:2: UserWarning:
The palette list has fewer values (2) than needed (3) and will cycle, which may produce an uninterpretable plot.
  ax = sns.countplot(x='gender', data=df, palette=['lightpink', 'lightblue'])
```



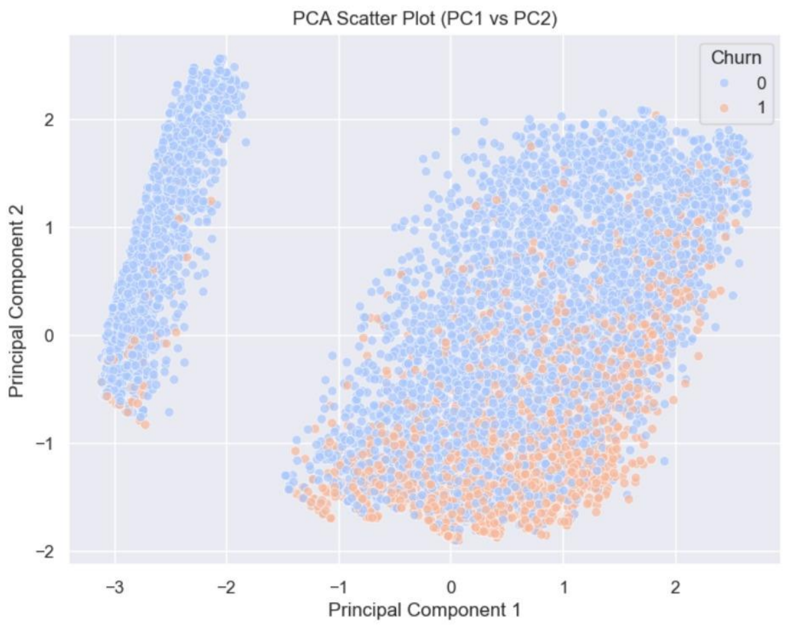
This bar chart visually represents the number of customers in each encoded gender category. It offers a quick insight into the gender balance of the dataset. This helps establish a basic understanding of the customer demographics being analyzed.



This plot displays the distributions of standardized numerical features like tenure and Monthly Charges. It visually confirms that scaling centered the data near zero and brought features to a comparable range. This insight is important for verifying data readiness for scale-sensitive algorithms.



This line graph shows the cumulative percentage of dataset variance explained by adding principal components sequentially. It provides the crucial insight that 14 components capture 90% of the variance. This justifies the dimensionality reduction choice for subsequent modeling.



This scatter plot projects the customers onto the first two principal components, colored by their churn status. It gives a 2D glimpse into the data's structure after dimension reduction. This visualization helps assess if the most significant components offer any inherent visual separation between churned and non-churned groups.

4. Model Design and Implementation

In this subsection, the design and implementation of the machine learning models developed for customer churn prediction, a binary classification issue, are explained.

4.1 Dataset Preparation for Modeling

The preprocessed and dimensionally reduced dataset obtained from the last step was used to train and cross-validate the models. The final dataset contains 14 important components explaining 90% of the variance in the original, and binary target feature 'Churn'.

For training and testing models, data was split into a train set and a test set by using train,_test_,split in scikit-learn. An 80:20 split was performed where 80% was used for training and 20% was left for testing. A random state=42 was used while splitting to allow reproducibility of results for different runs and models.

4.2 Model Selection and Justification

A collection of classification algorithms was selected to assess a variety of modeling strategies to the churn prediction problem:

Logistic Regression: Selected because it is simple, easy to interpret, and a good linear baseline classifier for binary class classification.

K-Nearest Neighbors (KNN): Selected because it is a non-parametric, instance-based learning algorithm that classifies by majority class among its nearest neighbors. It is capable of learning non-linear decision boundaries.

Support Vector Machine (SVM): Included for its robustness in high-dimensional space (even after PCA dimension reduction) and ability to find optimal separating hyperplanes using different kernels (like the RBF kernel used here) to handle non-linearities.

Random Forest: Ensemble method included for its resistance to overfitting, ability to handle intricate interaction, and intrinsic ability to prevent overfitting by generating a multitude of decision trees and aggregating predictions in an ensemble.

Gradient Boosting Machine (GBM): Another powerful ensemble technique selected as it adheres to a sequential tree creation approach where every tree strives to correct the mistakes of the previous ones, which is expected to lead to high prediction accuracy.

Multi-layer Perceptron (MLP): A feedforward neural network included with the objective to experiment with deep learning techniques and to discover very complex, non-linear data patterns.

4.3 Implementation Details

All models were implemented primarily using the scikit-learn library in Python. Key implementation specifics for each model are outlined below:

Model	**Implementation (Library/Class)**	**Key Parameters (as implemented)**
Logistic Regression	sklearn.linear_model.LogisticRegression	Default parameters, random_state=42
K-Nearest Neighbors	sklearn.neighbors.KNeighborsClassifier	n_neighbors=5 (default)
Support Vector Machine	sklearn.svm.SVC	kernel='rbf' (default), random_state=42, probability=True
Random Forest	sklearn.ensemble.RandomForestClassifier	n_estimators=100 (default), random_state=42
Gradient Boosting	sklearn.ensemble.GradientBoostingClassifier	n_estimators=100, learning_rate=0.1, max_depth=3 (defaults), random_state=42
Multi-layer Perceptron	sklearn.neural_network.MLPClassifier	hidden_layer_sizes=(100,), activation='relu', solver='adam' (defaults), max_iter=300, random_state=42

5.Evaluation and Comparison

For a customer churn prediction system, the primary goal is to effectively identify customers who are at high risk of leaving. This business problem typically involves an **imbalanced dataset**, where the number of "churners" (the positive class) is much smaller than the number of "non-churners."

In this scenario, **Accuracy** can be a misleading metric. A model that simply predicts "no churn" for everyone could achieve high accuracy but would be useless for our goal.

Therefore, the **F1-Score** is selected as the most critical evaluation metric. The F1-Score is the harmonic mean of Precision and Recall, providing a single score that balances two essential objectives:

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

- **Recall:** The ability of the model to find *all* actual churners. A high recall is vital because it is more costly to miss a churning customer (a false negative) than to incorrectly flag a loyal customer (a false positive).
- **Precision:** The ability of the model to be correct when it predicts a customer will churn. This is also important to avoid wasting retention resources on customers who were never leaving.

5.2 Model Performance Results

Model	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**AUC-ROC**	**CV_Score**
Gradient Boosting (Tuned)	0.7459	0.5140	0.7861	**0.6216**	0.8451	0.7515
K-Nearest Neighbors (k=5)	0.7665	0.5648	0.5241	0.5437	0.7740	0.7643
Logistic Regression (Tuned)	0.7367	0.5026	0.7781	0.6107	0.8375	0.7501
Neural Network (Deeper MLP)	0.7899	0.6189	0.5428	0.5783	0.8338	-
Random Forest (Tuned)	0.7651	0.5458	0.6845	0.6074	0.8344	0.7727
Support Vector Machine (Linear)	0.7942	0.6346	0.5294	0.5773	0.8349	0.8019

5.3 Comparison and Discussion

Comparing the models based on the primary metric, the **F1 Score** for predicting churn:

1. The **Tuned Gradient Boosting** model achieved the highest F1 Score (**0.6216**). This indicates the best balance between correctly identifying churners (high Recall of 0.7861) and ensuring those predictions are accurate (Precision of 0.5140).
2. The **Tuned Logistic Regression** model followed closely with an F1 Score of **0.6107**, also showing very high Recall (0.7781) but slightly lower Precision (0.5026).
3. The **Tuned Random Forest** model ranked third with an F1 Score of **0.6074**.
4. The **Deeper MLP** (0.5783) and **Linear SVM** (0.5773) performed similarly based on F1 Score.
5. **KNN** had the lowest F1 Score (0.5437).

Considering the importance of correctly identifying potential churners (high Recall) while maintaining reasonable prediction accuracy (Precision) in an imbalanced dataset, the **F1 Score** remains the most suitable primary metric.

Therefore, based on the highest F1 Score achieved, the **Tuned Gradient Boosting model** is selected as the best-performing model for this churn prediction task

6. Ethical Considerations and Bias Mitigation

Using customer data for churn prediction requires careful thought about ethics and bias to ensure fairness.

6.1 Potential Bias Issues

The main risks come from the data itself. We saw a **class imbalance** (more non-churners than churners), which could make the model ignore churners. Also, **historical data** might reflect past unfair practices, and the model could accidentally learn these biases.

6.2 Fairness Concerns

These biases could lead to **unfairness**, like targeting only certain groups for retention or missing others. Complex models and PCA also make it hard to explain *why* someone is flagged as high-risk, which affects **transparency**. Predictions should also be used responsibly, not just to push customers into bad deals.

6.3 Steps Taken to Reduce Bias

We took several steps to address these issues:

- We were **aware** of potential biases during data cleaning and feature selection.
- Because of the class imbalance, we chose the **F1 Score** instead of Accuracy as our main way to judge the models, since it better reflects performance on the smaller 'churn' group. We also looked at **AUC-ROC**.
- We **compared** several different model types to see how they handled the data.
- For real-world use, we'd recommend using **explainability tools** (like SHAP or LIME) and continuously **monitoring** the model for fairness and accuracy over time.

7. Reflections and Lessons Learned

Working on this churn prediction project was a great learning experience. We really got a feel for how machine learning tackles real business problems.

Looking back, we learned a few key things. **Cleaning the data** and exploring it first (EDA) was super important. Finding things like missing values or weird entries early on made a big difference later. We also saw how just using **accuracy** isn't always the best measure, especially when one group (like non-churners) is much bigger than the other. Switching to the **F1 score** gave us a better sense of which model actually did the best job finding the churners. Trying out **different models** showed us that there's no one-size-fits-all answer, and **tuning** them can really boost performance. Using **PCA** helped simplify the data, but we realized it makes it harder to explain *why* the model makes a certain prediction. Finally, the project reminded us to always think about **ethics and fairness** when working with customer data.

We definitely hit a few bumps. The **initial data wasn't perfect**, needing quite a bit of cleanup. Dealing with the **imbalance** between churners and non-churners was probably the biggest challenge. Finding the right **balance between a model that performs well and one we can easily explain** was also tricky.

Next time, we could try a few things differently. Maybe use techniques like **SMOTE** specifically for the class imbalance *before* PCA. We could also spend more time **tuning all the models** more systematically and explore **creating new features**. Adding **explainability tools** like SHAP would also be a great step to understand the final model better.

8.References

1. Mehrabi, N., Morstatter, F., Saxena, N., Lerman, K., & Galstyan, A. (2021). *A Survey on Bias and Fairness in Machine Learning*. ACM Computing Surveys.
2. Kaggle. (2025). *Telco Customer Churn Prediction Dataset*. https://www.kaggle.com/datasets/blastchar/telco-customer-churn
3. Pedregosa, F., et al. (2011). *Scikit-learn: Machine Learning in Python*. Journal of Machine Learning Research, 12, 2825–2830.
4. Chen, T., & Guestrin, C. (2016). *XGBoost: A Scalable Tree Boosting System*.