

# Technical Project Report: Telecommunication Customer Churn Prediction

**Developer:** Chinene J. Orji

**Project Type:** Independent Technical Project

**Tools & Libraries:** Python (Scikit-Learn, Pandas, NumPy)

**Model Used:** Logistic Regression

## 1. Project Objective

In the telecommunications industry, retaining existing customers is significantly more cost-effective than acquiring new ones. The goal of this project was to build a classification model that predicts whether a customer will "churn" (leave the service) based on their usage patterns and demographic data.

## 2. Technical Workflow

### Phase 1: Data Preprocessing

Raw data often contains categorical text and varied numerical scales that machine learning models cannot process directly. My preprocessing pipeline included:

- **Label Encoding:** Converted categorical "object" types into numerical values to allow mathematical computation.
- **Feature Scaling:** Utilised `StandardScaler` to normalise numerical features, ensuring that variables with larger ranges did not influence the Logistic Regression model.
- **Data Splitting:** Applied a stratified `train_test_split` (80/20) to maintain the distribution of the target variable across both training and testing sets.

```

## Preview target variable distribution
print(df['Churn Value'].value_counts())

## Convert categorical columns to numerical using Label Encoding for simplicity
categorical_cols = df.select_dtypes(include=['object']).columns.tolist()

label_encoders = {}
for col in categorical_cols:
    le = LabelEncoder()
    df[col] = le.fit_transform(df[col])
    label_encoders[col] = le

## Separate features and target label
X = df.drop('Churn Value', axis=1)
y = df['Churn Value']

## Standardize features
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

## Split data into train and test sets
X_train, X_test, y_train, y_test = train_test_split(
    X_scaled, y, test_size=0.2, random_state=42, stratify=y)

```

*Figure 1: Screenshot of Data Preprocessing Code Snippet on Telecommunication Customer Churn Data*

## Phase 2: Model Training

I selected **Logistic Regression** as the primary classifier due to its high interpretability in binary classification tasks.

- **Initialization:** The model was configured with `max_iter=1000` to ensure convergence during the optimization process.
- **Training:** The model was fit to the scaled training data to learn the coefficients associated with churn indicators.

```

## Initialize logistic regression model
model = LogisticRegression(max_iter=1000, random_state=42)

## Train the model
model.fit(X_train, y_train)

## Predict on test set
y_pred = model.predict(X_test)

## Evaluation metrics
print("Accuracy:", accuracy_score(y_test, y_pred))
print("\nConfusion Matrix:\n", confusion_matrix(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred))

```

*Figure 2: Screenshot of Model Training Code Snippet on Telecommunication Customer Churn Data*

## 3. Evaluation & Results

The model achieved an **Accuracy of 79.49%**, demonstrating a solid baseline for predicting customer behavior.

### Performance Metrics:

- **Confusion Matrix:** Successfully identified **913** customers who would stay and **207** who were likely to churn.
- **Precision (Class 1):** 63% - When the model predicts churn, it is correct nearly two-thirds of the time.
- **Recall (Class 1):** 55% - The model successfully identified 55% of all actual churners in the dataset.

```
Churn Value
0    5174
1    1869
Name: count, dtype: int64
Accuracy: 0.794889992902768

Confusion Matrix:
[[913 122]
 [167 207]]

Classification Report:
precision    recall    f1-score   support
          0      0.85     0.88      0.86     1035
          1      0.63     0.55      0.59      374

accuracy                           0.79      1409
macro avg       0.74     0.72      0.73     1409
weighted avg    0.79     0.79      0.79     1409
```

Figure 3: Churn Prediction Confusion Matrix and Classification Report

## 4. Reflection & Learning

This project deepened my understanding of the relationship between feature scaling and model performance. In Logistic Regression, scaling is vital because the model relies on the magnitude of weights.

**Key Takeaway:** While the overall accuracy is high (79%), the recall for churners (Class 1) shows room for improvement. My next step for this project would be to explore **Random Forest** or **SMOTE (Synthetic Minority Over-sampling Technique)** to better handle the class imbalance and improve the identification of at-risk customers.