**Predicting Customer Churn in a Telecommunications Company**

**Objective**

The primary objective of this project is to develop a predictive model that can identify customers at risk of churning, enabling the company to take proactive measures to retain them.

**Introduction**

Customer churn is a critical issue for businesses, especially in highly competitive industries such as telecommunications. Churn occurs when customers stop using a company's services, leading to a loss of revenue. Understanding the factors that contribute to churn and being able to predict which customers are at risk of churning can help companies implement targeted retention strategies, ultimately improving customer satisfaction and loyalty.

The primary objective of this project is to develop a predictive model that can identify customers at risk of churning, enabling the telecommunications company to take proactive measures to retain them. This project involves data preprocessing, exploratory data analysis (EDA), feature engineering, model building, and evaluation to achieve this objective.

**Data Collection and Preprocessing**

**Data Collection**

We used the dataset provided by Kaggle: [Telco Customer Churn](https://www.kaggle.com/datasets/blastchar/telco-customer-churn). The dataset includes information about customers, their demographics, services they have signed up for, and whether or not they have churned.

**Data Inspection**

The dataset contains 7043 rows and 21 columns. The columns include various customer attributes such as `gender`, `SeniorCitizen`, `Partner`, `Dependents`, `tenure`, `PhoneService`, `MultipleLines`, `InternetService`, `OnlineSecurity`, `OnlineBackup`, `DeviceProtection`, `TechSupport`, `StreamingTV`, `StreamingMovies`, `Contract`, `PaperlessBilling`, `PaymentMethod`, `MonthlyCharges`, `TotalCharges`, and `Churn`.

**Handling Missing Values**

The `TotalCharges` column was found to have some non-numeric values which were converted to NaN and subsequently filled with the median value of the column.

*data['TotalCharges'] = pd.to\_numeric(data['TotalCharges'], errors='coerce')*

*data['TotalCharges'].fillna(data['TotalCharges'].median(), inplace=True*)

**Encoding Categorical Variables**

Categorical variables were encoded using binary encoding for binary features and one-hot encoding for multi-category features.

*# Binary encoding*

*data['gender'] = data['gender'].map({'Female': 1, 'Male': 0})*

*data['Partner'] = data['Partner'].map({'Yes': 1, 'No': 0})*

*data['Dependents'] = data['Dependents'].map({'Yes': 1, 'No': 0})*

*data['PhoneService'] = data['PhoneService'].map({'Yes': 1, 'No': 0})*

*data['PaperlessBilling'] = data['PaperlessBilling'].map({'Yes': 1, 'No': 0})*

*data['Churn'] = data['Churn'].map({'Yes': 1, 'No': 0})*

*# One-hot encoding for other categorical variables*

*data = pd.get\_dummies(data, columns=[*

*'MultipleLines', 'InternetService', 'OnlineSecurity', 'OnlineBackup',*

*'DeviceProtection', 'TechSupport', 'StreamingTV', 'StreamingMovies',*

*'Contract', 'PaymentMethod'*

*]*

**Exploratory Data Analysis (EDA)**

**Distribution of Churn**

The dataset shows a class imbalance with a higher proportion of customers who did not churn.

*sns.countplot(x='Churn', data=data)*

*plt.show()*

**Tenure vs. Churn**

Customers with a shorter tenure are more likely to churn.

*sns.boxplot(x='Churn', y='tenure', data=data)*

*plt.show()*

**Monthly Charges vs. Churn**

Higher monthly charges are associated with a higher churn rate.

*sns.boxplot(x='Churn', y='MonthlyCharges', data=data)*

*plt.show()*

**Feature Engineering**

A new feature `TotalServices` was created to represent the total number of services subscribed by a customer.

*data['TotalServices'] = data[['PhoneService', 'MultipleLines\_Yes', 'InternetService\_DSL',*

*'InternetService\_Fiber optic', 'OnlineSecurity\_Yes',*

*'OnlineBackup\_Yes', 'DeviceProtection\_Yes', 'TechSupport\_Yes',*

*'StreamingTV\_Yes', 'StreamingMovies\_Yes']].sum(axis=1)*

**Building the Churn Prediction Model**

**Data Splitting**

The data was split into training and testing sets.

*from sklearn.model\_selection import train\_test\_split*

*X = data.drop(['customerID', 'Churn'], axis=1)*

*y = data['Churn']*

*X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)*

**Model Training and Evaluation**

Two models were trained: Logistic Regression and Random Forest.

**Logistic Regression**

*from sklearn.linear\_model import LogisticRegression*

*lr = LogisticRegression(max\_iter=1000)*

*lr.fit(X\_train, y\_train)*

*y\_pred\_lr = lr.predict(X\_test)*

**Random Forest**

*from sklearn.ensemble import RandomForestClassifier*

*rf = RandomForestClassifier(n\_estimators=100, random\_state=42)*

*rf.fit(X\_train, y\_train)*

*y\_pred\_rf = rf.predict(X\_test)*

**Model Evaluation**

The models were evaluated using accuracy, precision, recall, and F1-score.

*from sklearn.metrics import accuracy\_score, precision\_score, recall\_score, f1\_score*

*def evaluate\_model(y\_test, y\_pred):*

*accuracy = accuracy\_score(y\_test, y\_pred)*

*precision = precision\_score(y\_test, y\_pred)*

*recall = recall\_score(y\_test, y\_pred)*

*f1 = f1\_score(y\_test, y\_pred)*

*return accuracy, precision, recall, f1*

*lr\_metrics = evaluate\_model(y\_test, y\_pred\_lr)*

*rf\_metrics = evaluate\_model(y\_test, y\_pred\_rf)*

*print(f'Logistic Regression: Accuracy={lr\_metrics[0]}, Precision={lr\_metrics[1]}, Recall={lr\_metrics[2]}, F1-Score={lr\_metrics[3]}')*

*print(f'Random Forest: Accuracy={rf\_metrics[0]}, Precision={rf\_metrics[1]}, Recall={rf\_metrics[2]}, F1-Score={rf\_metrics[3]}')*

**Results**

**Logistic Regression**

The Logistic Regression model achieved the following performance metrics:

* **Accuracy**: 0.81
* **Precision**: 0.68
* **Recall**: 0.50
* **F1-Score**: 0.58

**Interpretation**:

* **Accuracy** of 0.81 indicates that the model correctly predicted 81% of the cases.
* **Precision** of 0.68 means that 68% of the customers predicted to churn actually did churn.
* **Recall** of 0.50 shows that the model identified 50% of all actual churn cases.
* **F1-Score** of 0.58, a harmonic mean of precision and recall, indicates a balance between the two, although there is room for improvement in recall.

The Logistic Regression model's performance reflects a good balance between precision and recall, but the recall value suggests it misses a significant number of actual churn cases. This might be acceptable in scenarios where false positives are more costly than false negatives, but for a customer retention strategy, improving recall is crucial.

**Random Forest**

The Random Forest model achieved the following performance metrics:

* **Accuracy**: 0.79
* **Precision**: 0.67
* **Recall**: 0.49
* **F1-Score**: 0.57

**Interpretation**:

* **Accuracy** of 0.79 indicates that the model correctly predicted 79% of the cases, slightly lower than Logistic Regression.
* **Precision** of 0.67 shows that 67% of the customers predicted to churn did churn, comparable to Logistic Regression.
* **Recall** of 0.49 demonstrates that the model captured 49% of actual churn cases, again slightly lower than Logistic Regression.
* **F1-Score** of 0.57, close to that of Logistic Regression, indicates a similar balance between precision and recall.

The Random Forest model, while powerful and capable of capturing complex interactions between features, did not outperform the Logistic Regression model in this scenario. Its slightly lower recall suggests it also misses a notable portion of actual churn cases. However, its performance is robust and might improve significantly with hyperparameter tuning.

**Summary**

Both models exhibit comparable performance with some trade-offs:

* **Logistic Regression** is slightly better in terms of accuracy and recall, making it a good choice for initial implementation and understanding the factors influencing churn due to its interpretability.
* **Random Forest** provides a marginally lower performance but offers robustness and the ability to capture non-linear relationships.

**Key Insights**

* **Class Imbalance**: Both models struggled with the class imbalance in the dataset, evident from their recall scores. Addressing this imbalance could significantly improve model performance.
* **Model Interpretability**: Logistic Regression offers better interpretability, allowing the company to understand and act on specific factors contributing to churn.
* **Feature Importance**: Random Forest can provide insights into feature importance, which can be valuable for feature selection and understanding the most influential variables in predicting churn.

Future work should focus on improving recall through techniques such as resampling, exploring additional features, and extensive hyperparameter tuning. Additionally, evaluating other models like Gradient Boosting or neural networks could further enhance predictive performance.

**Challenges**

**Handling Non-Numeric Values in TotalCharges**

The TotalCharges column was supposed to be numeric, but it contained non-numeric values that had to be handled. Converting these values to NaN and then filling them with the median was crucial to ensure the data's integrity. This step was essential to avoid errors during model training but also highlighted the need for robust data validation processes in data collection systems.

**Class Imbalance**

The dataset had a significant class imbalance, with a much higher proportion of non-churning customers compared to churning ones. This imbalance can lead to models that are biased towards predicting non-churn, as they are trained on a majority of non-churning cases. Techniques such as resampling, using appropriate evaluation metrics, or applying more advanced methods like SMOTE (Synthetic Minority Over-sampling Technique) could be explored to address this issue.

**Feature Selection and Encoding**

Selecting the right features and encoding categorical variables appropriately was another challenge. With a mix of binary, multi-category, and numerical features, it was important to ensure that the encoding methods preserved the information without introducing noise. One-hot encoding for multi-category features, while effective, can lead to high dimensionality, which needs to be managed carefully to prevent overfitting.

**Model Selection and Hyperparameter Tuning**

Choosing the right model and tuning its hyperparameters was a critical step. While Logistic Regression and Random Forest were used in this project, exploring other models such as Gradient Boosting, XGBoost, or neural networks might yield better performance. Hyperparameter tuning using grid search or randomized search could further enhance the model's accuracy and generalization ability.

**Evaluation Metrics**

Evaluating the models using appropriate metrics was necessary to understand their performance comprehensively. Relying solely on accuracy was not sufficient due to the class imbalance. Precision, recall, and F1-score provided more insight into the model's ability to correctly predict churn cases, guiding improvements in model training and feature engineering.

**Conclusion**

The development of the predictive models for customer churn in this telecommunications company project provided valuable insights into customer behavior and the factors contributing to churn. Both the Logistic Regression and Random Forest models demonstrated reasonable performance, with the Logistic Regression model slightly outperforming the Random Forest model in terms of accuracy and F1-score.

**Model Comparison and Performance**

* **Logistic Regression** showed an accuracy of 0.81, precision of 0.68, recall of 0.50, and F1-score of 0.58. This model is straightforward and interpretable, making it useful for understanding the impact of each feature on the probability of churn.
* **Random Forest** exhibited an accuracy of 0.79, precision of 0.67, recall of 0.49, and F1-score of 0.57. This model can capture complex relationships between features and is robust to overfitting, although it did not outperform Logistic Regression in this case.

**Future Work**

1. **Hyperparameter Tuning**: Future iterations should involve extensive hyperparameter tuning for both Logistic Regression and Random Forest models to optimize their performance. Techniques such as Grid Search and Random Search can be employed to find the best parameters.
2. **Addressing Class Imbalance**: The class imbalance issue can be mitigated using methods such as SMOTE, ADASYN, or class weighting in the models. This will help improve the recall and F1-score, ensuring better identification of customers at risk of churning.
3. **Exploring Additional Models**: Other machine learning models like Gradient Boosting, XGBoost, LightGBM, and neural networks should be explored. These models often provide superior performance for classification tasks and can capture non-linear relationships more effectively.
4. **Feature Engineering**: Additional feature engineering could uncover more significant predictors of churn. This could include interaction terms between features, polynomial features, or incorporating external data sources.
5. **Cross-validation**: Implementing cross-validation techniques can provide a more reliable estimate of model performance by ensuring that the model is validated on multiple subsets of the data.
6. **Deploying the Model**: Finally, integrating the model into a production environment where it can make real-time predictions and provide actionable insights to the company's customer retention team.

By addressing these areas, the predictive performance of the churn model can be further improved, leading to more effective retention strategies and better customer management.

## Repository

The complete code and report are available on GitHub: [GitHub Repository Link]