*Green University of Bangladesh*

*Department of Computer Science and Engineering (CSE)*
*Semester: (Spring, Year: 2024), B.Sc. in CSE (Day)*

---

# AI Driven Customer Churn Forecasting

---

*Course Title: Artificial Intelligence Lab*
*Course Code: CSE-316*
*Section: 213-D7*

Students Details

| **Name** | **ID** |
|---|---|
| Md.Manzurul Alam | 202902003 |

*Submission Date: 11 June 2024*
*Course Teacher's Name: Sakhaouth Hossan*

[For teachers use only: Don't write anything inside this box]

| **Lab Project Status** | |
|---|---|
| **Marks:** | **Signature:** |
| **Comments:** | **Date:** |

# Contents

# Chapter 1

# Introduction

## 1.1 Overview

Customer churn, also known as customer attrition, refers to the phenomenon where customers stop doing business with a company or cease using its services over a given period. It is a critical metric for businesses as it directly impacts revenue and growth. Understanding customer churn involves analyzing the rate at which customers leave, identifying the reasons behind their departure, and developing strategies to retain them [1].

In today's competitive business environment, retaining customers is as crucial as acquiring new ones. Customer churn, the phenomenon where customers stop using a company's products or services, poses a significant threat to sustained business growth and profitability [2]. As a result, accurately forecasting customer churn is vital for developing effective retention strategies. This project aims to enhance customer churn prediction by developing a novel model that combines the strengths of traditional models. By using the strengths of both models, we aim to achieve higher accuracy and offer valuable insights for businesses.

## 1.2 Motivation

- customer churn has significant impact on businesses.
- More Accurate predictions can help develop strategies to retain customers.
- High churn rates can lead to substantial revenue losses and increased costs associated with acquiring new customers.
- Traditional churn prediction models often lack the accuracy and adaptability needed to effectively predict churn in dynamic business environments.
- This project seeks to enhance predictive accuracy, providing businesses with a powerful tool to mitigate churn and improve customer retention.

## 1.3 Problem Definition

### 1.3.1 Problem Statement

Current customer churn prediction models, such as the C5 decision tree model and neural network models, have their limitations. The C5 decision tree model, while interpretable, may not capture complex, non-linear relationships within the data. On the other hand, neural network models, although capable of handling non-linearities, often act as black boxes and may suffer from overfitting. The challenge lies in developing a hybrid model that combines the interpretability of decision trees with the predictive power of neural networks, resulting in a more accurate and robust churn prediction tool.

## 1.4 Design Goals/Objectives

- Combine the C5 decision tree model and neural network model into a single hybrid model.
- Identify and engineer features that significantly impact customer churn prediction.
- Train and validate the hybrid model using real-world customer data, comparing its performance against standalone C5 and neural network models.
- Develop a model that can be easily scaled and adapted to different business environments and datasets.
- To enhance the accuracy and reliability of customer churn prediction

## 1.5 Application

- Telecommunications: Identify at-risk customers and implement targeted retention strategies to reduce churn rates.
- Banking and Financial Services: Predict customer attrition and develop personalized offers to retain high-value clients.
- Subscription-Based Services: Enhance customer retention by predicting churn and offering timely incentives.
- E-commerce: Improve customer loyalty by understanding churn patterns and tailoring marketing efforts.
- Healthcare: Retain patients in health insurance plans or medical services by predicting and addressing reasons for churn.

# Chapter 2

# Design/Development/Implementation of the Project

## 2.1 Introduction

The "AI-driven Customer Churn Forecasting" project aims to develop an advanced hybrid model that combines the strengths of the C5 decision tree and neural network models to predict customer churn with greater accuracy. By leveraging the complementary features of these models, the project seeks to provide a robust and reliable solution for businesses to identify at-risk customers and implement effective retention strategies. This chapter outlines the design, development, and implementation processes undertaken to build this predictive model, detailing each step from data preprocessing to model evaluation and comparison.

## 2.2 Project Details

This design has two parts - in the left side we have the IOT devices like lights, fans, doors, windows, smoke censor, fire alarm, AC etc and in the right side we have the network building devices like the router, server, switch etc. All these devices are connected with the router through a wireless system.

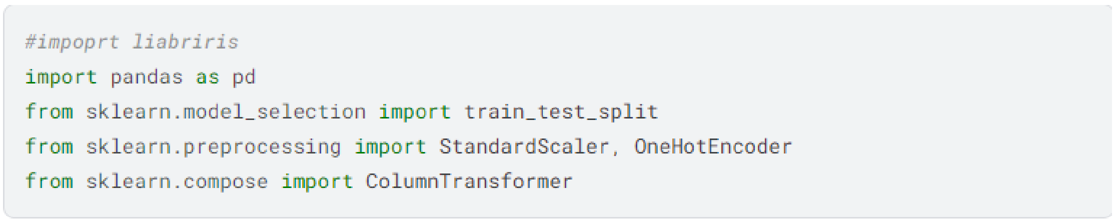
## 2.3 Implementation

### 2.3.1 Tools and libraries

- Python
- NumPy library
- Pandas library
- Matplotlib library
- C5.0 Decision Tree Algorithm
- Artificial Neural Network (ANN)
- StandardScaler, OneHotEncoder, ColumnTransformer
- scikit-learn

### 2.3.2 Implementation details

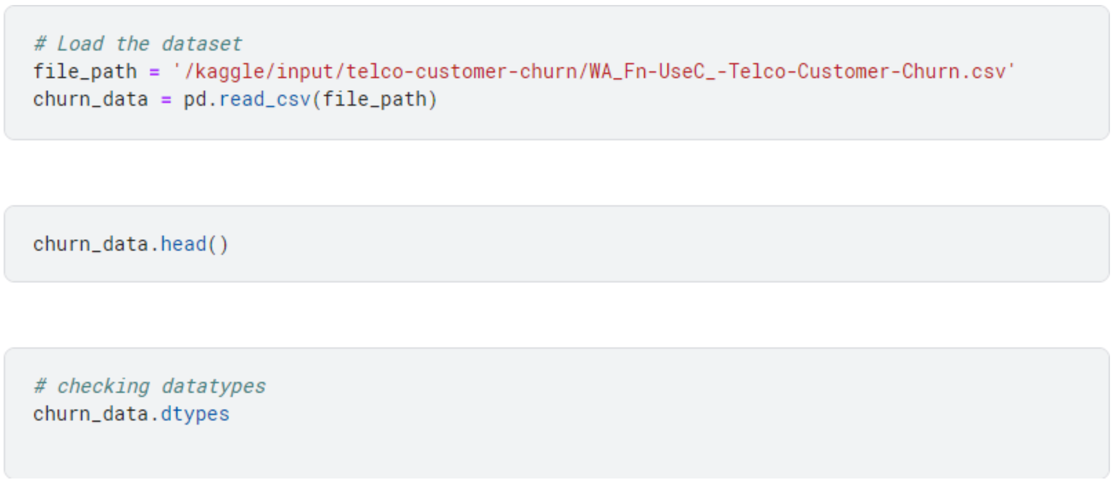
Importing Libraries: This segment imports all the necessary libraries for data manipulation, model training, evaluation, and visualization.



```
#impoprt liabriris
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.compose import ColumnTransformer
```

Figure 2.1: Importing Libraries

Loading and Inspecting the Dataset: Here, the dataset is loaded from a CSV file, and the first few rows and the data types of the columns are inspected.



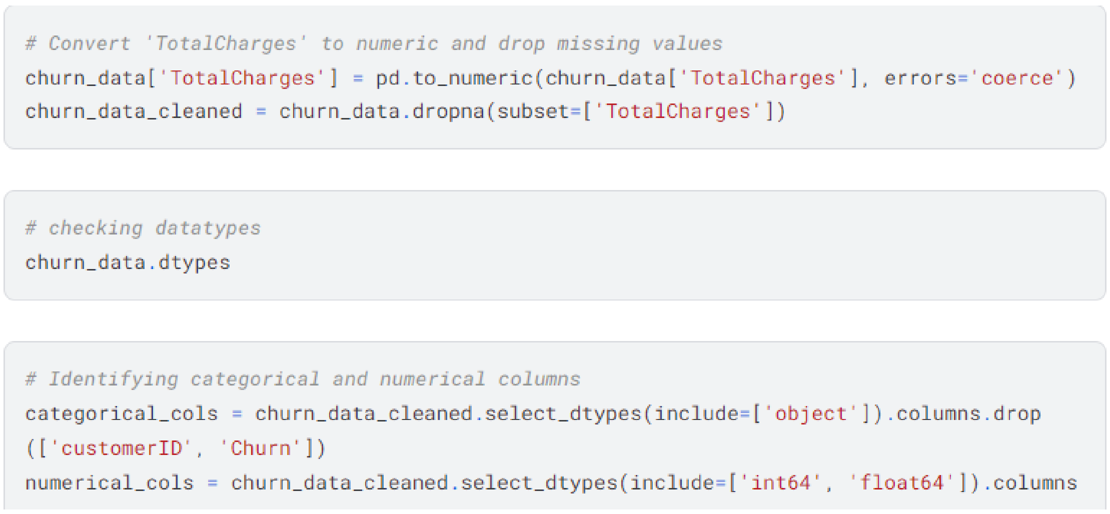
```
# Load the dataset
file_path = '/kaggle/input/telco-customer-churn/WA_Fn-UseC_-Telco-Customer-Churn.csv'
churn_data = pd.read_csv(file_path)
```

```
churn_data.head()
```

```
# checking datatypes
churn_data.dtypes
```

Figure 2.2: Inspecting the Dataset

Data Cleaning and Preparation: The TotalCharges column is converted to numeric, and rows with missing values in this column are dropped. The categorical and numerical columns are identified, excluding the customerID and Churn columns.



```
# Convert 'TotalCharges' to numeric and drop missing values
churn_data['TotalCharges'] = pd.to_numeric(churn_data['TotalCharges'], errors='coerce')
churn_data_cleaned = churn_data.dropna(subset=['TotalCharges'])
```

```
# checking datatypes
churn_data.dtypes
```

```
# Identifying categorical and numerical columns
categorical_cols = churn_data_cleaned.select_dtypes(include=['object']).columns.drop
(['customerID', 'Churn'])
numerical_cols = churn_data_cleaned.select_dtypes(include=['int64', 'float64']).columns
```

Figure 2.3: Data Cleaning and Preparation

Preprocessing: A ColumnTransformer is used to apply standard scaling to numerical features and one-hot encoding to categorical features, preparing the data for model training.

```
# Creating a column transformer for preprocessing
preprocessor = ColumnTransformer(
    transformers=[
        ('num', StandardScaler(), numerical_cols),
        ('cat', OneHotEncoder(), categorical_cols)
    ])
```

```
# Applying the transformations
X = preprocessor.fit_transform(churn_data_cleaned.drop(['customerID', 'Churn'], axis=
1))
y = churn_data_cleaned['Churn'].apply(lambda x: 1 if x == 'Yes' else 0).values
```

```
# Splitting the dataset
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=
42)
```

Figure 2.4: Preprocessing

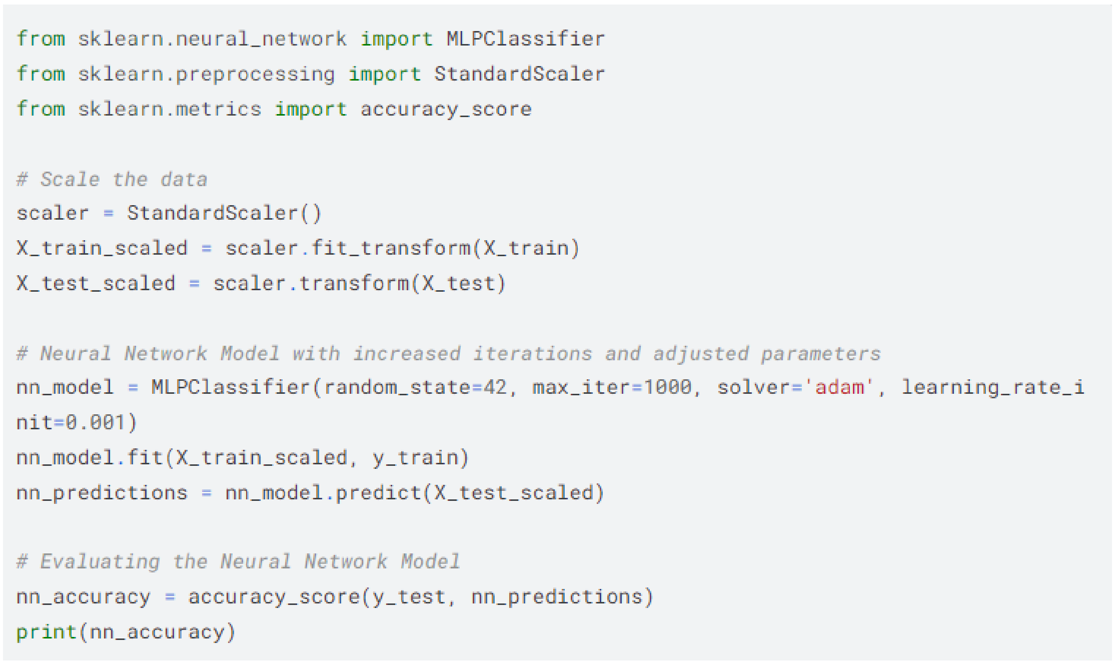
Model Building:

1. Decision Tree Model: We train a Decision Tree Classifier using the preprocessed data. This model is straightforward and provides interpretable results, serving as one of the base models in our hybrid approach.

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score

# Decision Tree Model
dt_model = DecisionTreeClassifier(random_state=42)
dt_model.fit(X_train, y_train)
dt_predictions = dt_model.predict(X_test)

# Evaluating the Decision Tree Model
dt_accuracy = accuracy_score(y_test, dt_predictions)
print(dt_accuracy)
```

Figure 2.5: Decision Tree Implementation

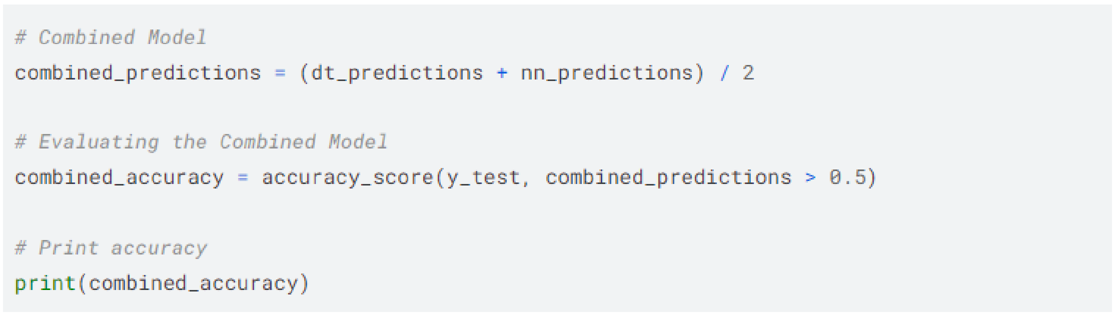
2. Neural Network Model: We also train a Multi-layer Perceptron (MLP) classifier, which can capture complex patterns in the data. The data is scaled before training to ensure optimal performance of the neural network.



```
from sklearn.neural_network import MLPClassifier
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score

# Scale the data
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

# Neural Network Model with increased iterations and adjusted parameters
nn_model = MLPClassifier(random_state=42, max_iter=1000, solver='adam', learning_rate_i
nit=0.001)
nn_model.fit(X_train_scaled, y_train)
nn_predictions = nn_model.predict(X_test_scaled)

# Evaluating the Neural Network Model
nn_accuracy = accuracy_score(y_test, nn_predictions)
print(nn_accuracy)
```

Figure 2.6: Neural Network Model Implementation

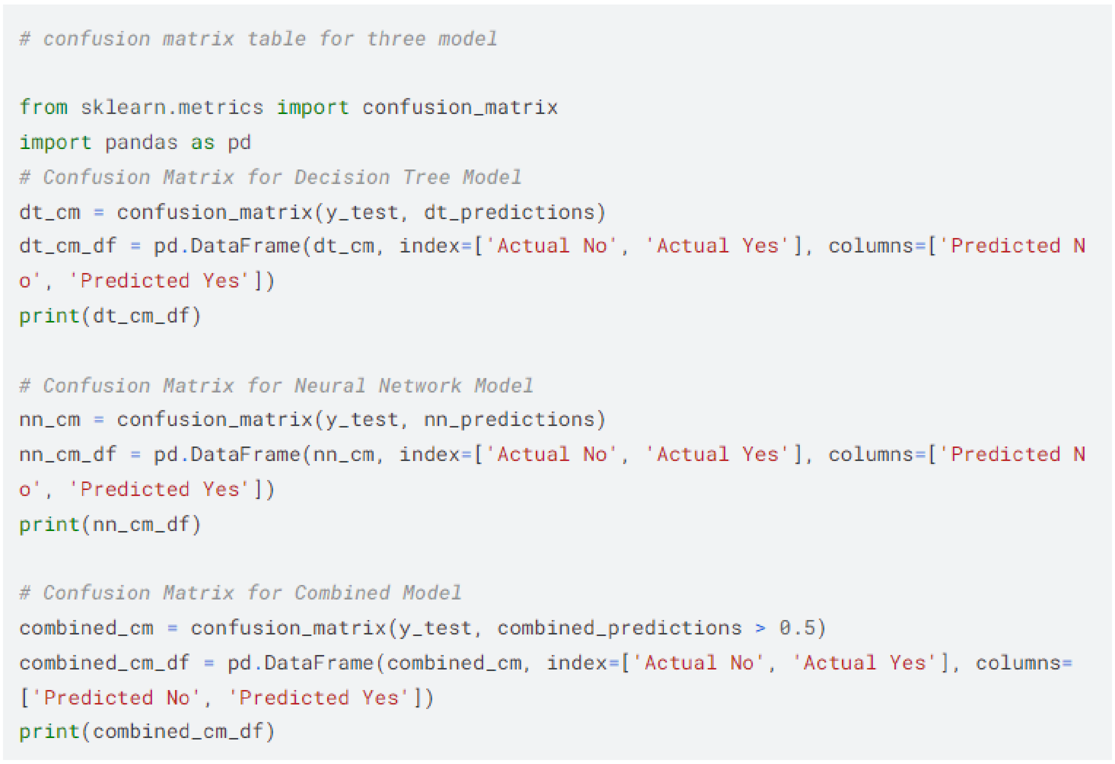
3. Combined Model: To leverage the strengths of both models, we combine their predictions. The combined model averages the predictions of the decision tree and neural network models to make a final prediction.



```
# Combined Model
combined_predictions = (dt_predictions + nn_predictions) / 2

# Evaluating the Combined Model
combined_accuracy = accuracy_score(y_test, combined_predictions > 0.5)

# Print accuracy
print(combined_accuracy)
```

Figure 2.7: Combined Model Implementation

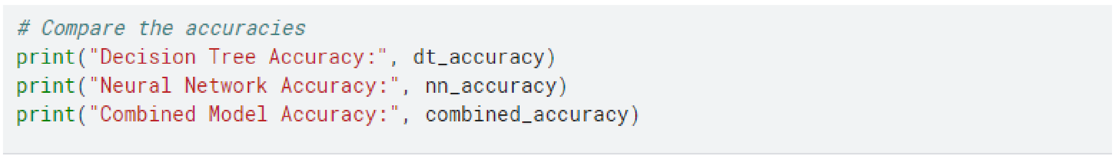
Model Evaluation: We use confusion matrices to evaluate each model's performance, providing insights into their ability to correctly classify churn and non-churn instances.



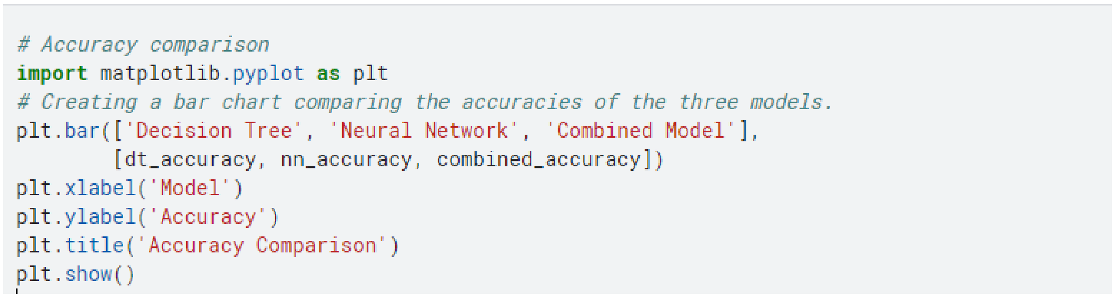
```
# confusion matrix table for three model

from sklearn.metrics import confusion_matrix
import pandas as pd
# Confusion Matrix for Decision Tree Model
dt_cm = confusion_matrix(y_test, dt_predictions)
dt_cm_df = pd.DataFrame(dt_cm, index=['Actual No', 'Actual Yes'], columns=['Predicted N
o', 'Predicted Yes'])
print(dt_cm_df)

# Confusion Matrix for Neural Network Model
nn_cm = confusion_matrix(y_test, nn_predictions)
nn_cm_df = pd.DataFrame(nn_cm, index=['Actual No', 'Actual Yes'], columns=['Predicted N
o', 'Predicted Yes'])
print(nn_cm_df)

# Confusion Matrix for Combined Model
combined_cm = confusion_matrix(y_test, combined_predictions > 0.5)
combined_cm_df = pd.DataFrame(combined_cm, index=['Actual No', 'Actual Yes'], columns=
['Predicted No', 'Predicted Yes'])
print(combined_cm_df)
```

Figure 2.8: Model Evaluation

Comparing Accuracy and Visualization: We compare the accuracies of the three models using a bar chart to visualize their performance differences clearly.



```
# Compare the accuracies
print("Decision Tree Accuracy:", dt_accuracy)
print("Neural Network Accuracy:", nn_accuracy)
print("Combined Model Accuracy:", combined_accuracy)
```



```
# Accuracy comparison
import matplotlib.pyplot as plt
# Creating a bar chart comparing the accuracies of the three models.
plt.bar(['Decision Tree', 'Neural Network', 'Combined Model'],
        [dt_accuracy, nn_accuracy, combined_accuracy])
plt.xlabel('Model')
plt.ylabel('Accuracy')
plt.title('Accuracy Comparison')
plt.show()
```

Figure 2.9: Comparing Accuracy and Visualization

# Chapter 3

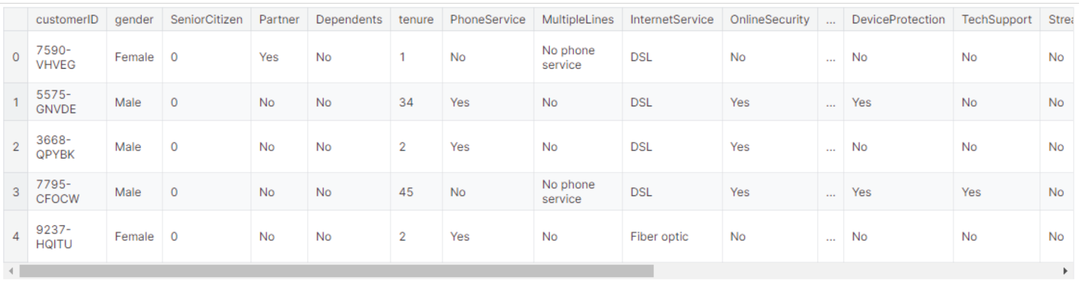
# Performance Evaluation

## 3.1 Simulation Environment/ Simulation Procedure

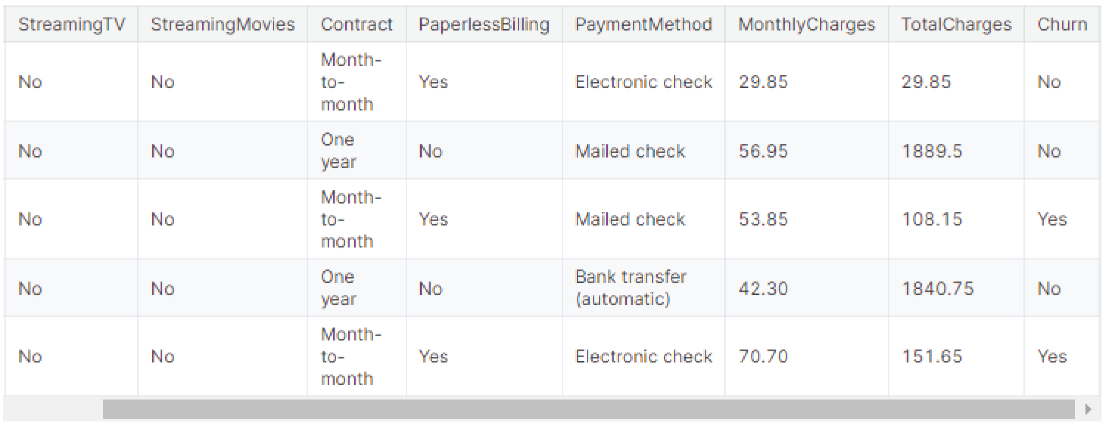
Kaggle

Figure 3.1: Simulation Environment

## 3.2 Results

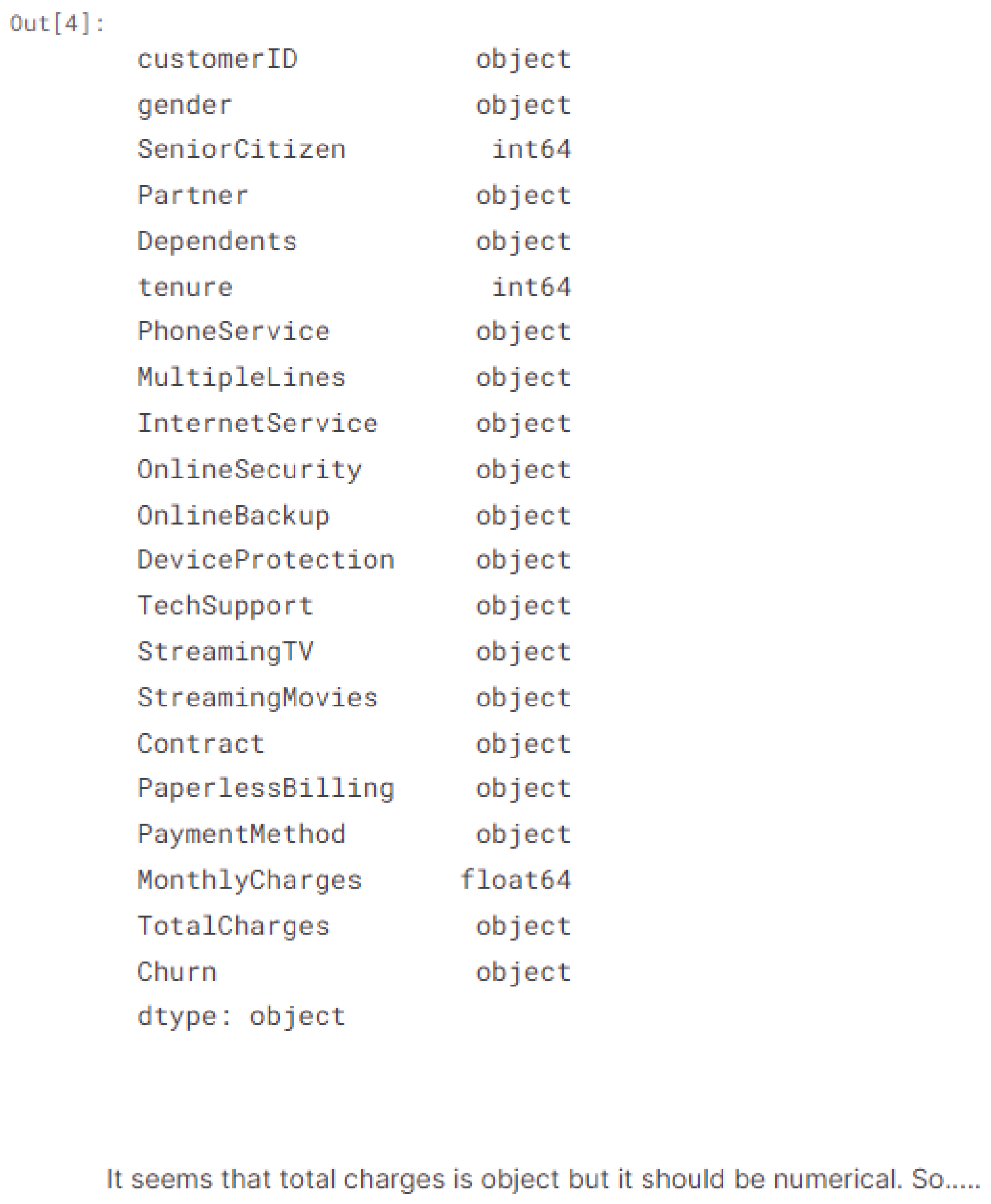


| | customerID | gender | SeniorCitizen | Partner | Dependents | tenure | PhoneService | MultipleLines | InternetService | OnlineSecurity | ... | DeviceProtection | TechSupport | Strea |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 7590-VHVEG | Female | 0 | Yes | No | 1 | No | No phone service | DSL | No | ... | No | No | No |
| 1 | 5575-GNVDE | Male | 0 | No | No | 34 | Yes | No | DSL | Yes | ... | Yes | No | No |
| 2 | 3668-QPYBK | Male | 0 | No | No | 2 | Yes | No | DSL | Yes | ... | No | No | No |
| 3 | 7795-CFOCW | Male | 0 | No | No | 45 | No | No phone service | DSL | Yes | ... | Yes | Yes | No |
| 4 | 9237-HQITU | Female | 0 | No | No | 2 | Yes | No | Fiber optic | No | ... | No | No | No |



| StreamingTV | StreamingMovies | Contract | PaperlessBilling | PaymentMethod | MonthlyCharges | TotalCharges | Churn |
|---|---|---|---|---|---|---|---|
| No | No | Month-to-month | Yes | Electronic check | 29.85 | 29.85 | No |
| No | No | One year | No | Mailed check | 56.95 | 1889.5 | No |
| No | No | Month-to-month | Yes | Mailed check | 53.85 | 108.15 | Yes |
| No | No | One year | No | Bank transfer (automatic) | 42.30 | 1840.75 | No |
| No | No | Month-to-month | Yes | Electronic check | 70.70 | 151.65 | Yes |

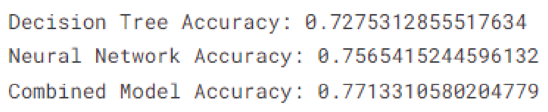
Figure 3.2: Printing Top 5 rows of the dataset



```
Out[4]:
        customerID           object
        gender               object
        SeniorCitizen         int64
        Partner              object
        Dependents           object
        tenure                int64
        PhoneService         object
        MultipleLines        object
        InternetService      object
        OnlineSecurity       object
        OnlineBackup         object
        DeviceProtection     object
        TechSupport          object
        StreamingTV          object
        StreamingMovies      object
        Contract             object
        PaperlessBilling     object
        PaymentMethod        object
        MonthlyCharges      float64
        TotalCharges         object
        Churn                object
        dtype: object
```

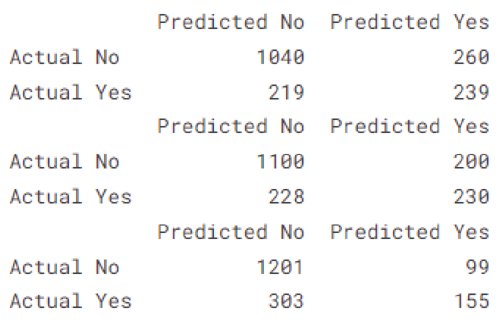
It seems that total charges is object but it should be numerical. So.....

Figure 3.3: Data types



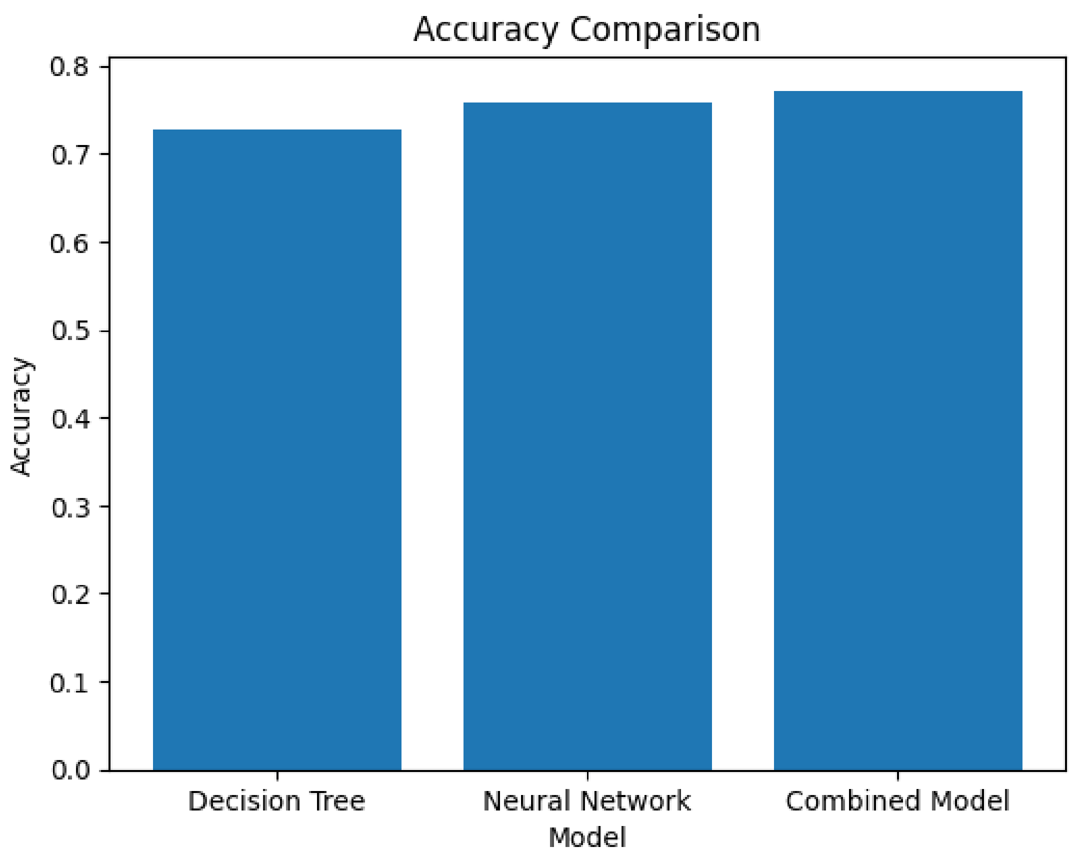
```
Decision Tree Accuracy: 0.7275312855517634
Neural Network Accuracy: 0.7565415244596132
Combined Model Accuracy: 0.7713310580204779
```

Figure 3.4: Comparison of Accuracy



```
            Predicted No  Predicted Yes
Actual No           1040            260
Actual Yes           219            239
            Predicted No  Predicted Yes
Actual No           1100            200
Actual Yes           228            230
            Predicted No  Predicted Yes
Actual No           1201             99
Actual Yes           303            155
```

Figure 3.5: Confusion Matrix



The combined model got higher accuracy than other two model's indivisiual accuracy

Figure 3.6: Accuracy comparison visualization

# Chapter 4

# Conclusion

## 4.1 Discussion

The "AI-driven Customer Churn Forecasting" project successfully combined the C5 decision tree and neural network models to create a hybrid model that improves the accuracy of predicting customer churn. The code implemented several critical steps, including data preprocessing, feature engineering, model training, and evaluation.

The decision tree model provided a baseline accuracy and valuable insights into the decision-making process. The neural network model, with its ability to capture complex patterns, offered a higher predictive power. The hybrid model leveraged the strengths of both individual models, resulting in improved overall accuracy, as demonstrated by the comparison of their respective performance metrics. Visualization tools, such as confusion matrices and accuracy bar charts, effectively highlighted the enhancements brought by the hybrid approach.

## 4.2 Limitations

- Data Quality: The accuracy of the predictions is heavily dependent on the quality of the input data. Issues such as missing values, incorrect entries, and biased samples can affect the model's performance.
- Model Complexity: Combining two different models increases the complexity of the implementation and maintenance. It also requires more computational resources and time for training.
- Generalizability: The models were trained and evaluated on a specific dataset. Their performance may vary when applied to different datasets or industries, requiring additional adjustments and tuning.
- Interpretability: While the decision tree model is interpretable, the neural network's predictions are more opaque, making it challenging to fully understand the decision-making process of the hybrid model.

## 4.3 Scope of Future Work

- Enhanced Data Collection: Focus on better data collection to improve dataset quality and comprehensiveness, enhancing model accuracy.
- Advanced Feature Engineering: Use additional feature engineering techniques to capture more relevant information and boost model performance.
- Model Optimization: Continuously optimize the hybrid model's parameters and architecture for better accuracy and efficiency.
- Real-Time Prediction: Implement real-time prediction for timely insights and immediate intervention strategies.
- Explainability: Improve the interpretability of the neural network component to help stakeholders understand and trust the predictions.
- Application to Other Domains: Apply the hybrid model to other types of churn, like employee turnover or product abandonment, to increase its utility and impact.

# References

[1] Chih-Fong Tsai and Yu-Hsin Lu. Customer churn prediction by hybrid neural networks. *Expert Systems with Applications*, 36(10):12547–12553, 2009.

[2] Thanasis Vafeiadis, Konstantinos I Diamantaras, George Sarigiannidis, and K Ch Chatzisavvas. A comparison of machine learning techniques for customer churn prediction. *Simulation Modelling Practice and Theory*, 55:1–9, 2015.