Process Overview

1. Data Collection

Objective: Gather a comprehensive dataset that includes customer information and churn status.

Steps Taken:

- **Source**: Used the Telecom Customer Churn dataset available on Kaggle.
- **Data Details**: The dataset includes customer features such as demographics, service usage, and contract details, along with a target variable indicating whether the customer churned.

Reference: Kaggle Telecom Customer Churn Dataset

2. Data Preprocessing

Objective: Prepare the raw data for analysis and modeling.

Steps Taken:

- **Data Cleaning**:
  - **Removed** unnecessary columns (e.g., CustomerID).
  - **Handled** missing values by filling them with appropriate values or using imputation techniques.
- **Feature Engineering**:
  - **Encoded** categorical variables (e.g., Gender, InternetService) using one-hot encoding.
  - **Standardized** numerical features (e.g., MonthlyCharges, TotalCharges) to bring them to the same scale.
- **Data Splitting**:
  - Split the data into training and test sets (80% train, 20% test) for model building and evaluation.

Code Example:

```
# Feature Engineering
dataset = pd.get_dummies(tele_churn_data, columns=['gender', 'Partner',
'Dependents','PhoneService', 'MultipleLines',
'InternetService','OnlineSecurity', 'OnlineBackup',
'DeviceProtection','TechSupport', 'StreamingTV', 'StreamingMovies',
'Contract','PaperlessBilling', 'PaymentMethod',
'MonthlyCharges','TotalCharges', 'Churn'], dtype = int)

# Data Splitting
from sklearn.model_selection import train_test_split
```

```
X = dataset.iloc[:, 1:-2].values
y = dataset.iloc[:, -1].values
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2,
random_state = 0)
```

3. Exploratory Data Analysis (EDA)

Objective: Understand the dataset and identify patterns or anomalies.

Steps Taken:

- **Visualized** feature distributions and correlations using plots (histograms, heatmaps).
- **Analyzed** feature importance using various models.
- **Investigated** churn patterns based on different features.

Code Example:

```
import seaborn as sns
import matplotlib.pyplot as plt

# Plot the distribution of the target variable
plt.figure(figsize=(8, 6))
sns.countplot(data=tele_churn_data, x='Churn')
plt.title('Distribution of Churn')
plt.show()
```

4. Model Building

Objective: Create and evaluate different machine learning models to predict churn.

Models Used:

1. **Logistic Regression**
    - **Objective:** Baseline model for binary classification.
    - **Code Example:**

```
from sklearn.linear_model import LogisticRegression
classifier = LogisticRegression()
classifier.fit(X_train, y_train)
```

2. **K-Nearest Neighbors (KNN)**
    - **Objective:** Classify churn based on the proximity of data points.
    - **Code Example:**

```
from sklearn.neighbors import KNeighborsClassifier
classifier = KNeighborsClassifier(n_neighbors=5, metric='minkowski',
p=2)
classifier.fit(X_train, y_train)
```

3. **Random Forest Classifier**
    - **Objective:** Use ensemble methods to improve classification accuracy.
    - **Code Example:**

```
from sklearn.ensemble import RandomForestClassifier
classifier = RandomForestClassifier(n_estimators=10,
criterion='entropy', random_state=0)
classifier.fit(X_train, y_train)
```

4. **Artificial Neural Network (ANN)**
    - **Objective:** Apply a deep learning model to capture complex patterns.
    - **Code Example:**

```
import tensorflow as tf
ann = tf.keras.models.Sequential()
ann.add(tf.keras.layers.Dense(units=6, activation='relu'))
ann.add(tf.keras.layers.Dense(units=6, activation='relu'))
ann.add(tf.keras.layers.Dense(units=1, activation='sigmoid'))
ann.compile(loss="binary_crossentropy", optimizer="adam",
metrics=["accuracy"])
ann.fit(X_train, y_train, batch_size=32, epochs=10,
validation_split=0.1)
```

5. Hyperparameter Tuning

Objective: Optimize model performance through hyperparameter adjustments.

Steps Taken:

- **Randomized Search** and **Grid Search** were used to find the best hyperparameters for KNN and Random Forest models.

Code Example:

```
from sklearn.model_selection import RandomizedSearchCV, GridSearchCV

# KNN Randomized Search
param_grid_knn = {'n_neighbors': [3, 5, 7, 9, 11], 'weights': ['uniform',
'distance']}
random_search_knn = RandomizedSearchCV(KNeighborsClassifier(),
param_distributions=param_grid_knn, n_iter=10, cv=5, scoring='accuracy',
n_jobs=-1, random_state=0)
random_search_knn.fit(X_train, y_train)

# Random Forest Grid Search
param_grid_rf = {'n_estimators': [100, 500, 1000], 'max_depth': [10, 50,
None], 'min_samples_split': [2, 5, 10], 'min_samples_leaf': [1, 2, 4]}
grid_search_rf = GridSearchCV(RandomForestClassifier(), param_grid_rf, cv=5,
scoring='accuracy', n_jobs=-1)
grid_search_rf.fit(X_train, y_train)
```

6. Model Evaluation

Objective: Assess the performance of the models using various metrics.

Steps Taken:

- **Confusion Matrix**: Analyzed true positives, true negatives, false positives, and false negatives.
- **Evaluation Metrics**: Calculated accuracy, precision, recall, F1-score, and AUC-ROC for each model.

Code Example:

```
from sklearn.metrics import confusion_matrix, accuracy_score,
precision_score, recall_score, f1_score, roc_auc_score

# For Random Forest
y_pred = classifier.predict(X_test)
cm = confusion_matrix(y_test, y_pred)
ac = accuracy_score(y_test, y_pred)
pr = precision_score(y_test, y_pred)
re = recall_score(y_test, y_pred)
f = f1_score(y_test, y_pred)
ra = roc_auc_score(y_test, y_pred)

print("Confusion Matrix:", cm)
print(f"Accuracy: {ac:.2f}")
print(f"Precision: {pr:.2f}")
print(f"Recall: {re:.2f}")
print(f"F1-score: {f:.2f}")
print(f"AUC-ROC: {ra:.2f}")
```

7. Recommendations

Objective: Provide actionable strategies based on the analysis.

Recommendations:

- **Enhance Contract Offerings**: Promote long-term contracts with incentives.
- **Improve Customer Support Services**: Strengthen online security and tech support.
- **Review Pricing Strategy**: Adjust pricing and offer promotions.
- **Evaluate Internet Service Options**: Assess and enhance fiber optic services.
- **Optimize Payment Methods**: Encourage stable payment methods over electronic checks.

Implementation:

- **Marketing Strategies**: Create campaigns for contract upgrades.
- **Service Enhancements**: Invest in customer support infrastructure.

- **Pricing Adjustments**: Review and adjust pricing structures.
- **Service Quality Improvement**: Analyze customer feedback for fiber optic services.
- **Payment Method Optimization**: Offer incentives for stable payment methods.