UTM

UNIVERSITI TEKNOLOGI MALAYSIA

FACULTY OF COMPUTING

SEMESTER 2 2024/2025

SECP2753 Data Mining

SECTION 02

Project 1

LECTURER: DR. ROZILAWATI BINTI DOLLAH @ MD. ZAIN

GROUP MEMBER

Student Name	Matric No.
AHMAD ZIYAAD	A23CS0206
GOE JIE YING	A23CS0224
TEH RU QIAN	A23CS0191

1.0 Introduction

The purpose of this research is to find significant business insights about customer behavior and retention by applying classification algorithms to the Telco Customer Churn dataset. Predicting churn and identifying key elements including payment methods, contract types, bundled products, technical support and optional services are the objective. This project aims to assist strategic decision-making that raises customer satisfaction, boosts service engagement and lowers churn through analyzing these factors.

2.0 Preprocessing Steps

Several types of data preparation methods were used to prepare the dataset before to model building. Figure 2.0 shows that the data is prepared by using Python.

1. **Data Cleaning**
    - Removed rows with missing values in add columns
    - Verified no duplicated entries
2. **Data Transformation**
    - Converted categorical variables to numerical format using *get_dummies*
    - Converted *SeniorCitizen* from numeric (0/1) to a binary categorical variable.
3. **Feature Scaling**
    - Used *StandardScaler()* to normalize numerical features for KNN models.
4. **Target Variable**
    - Converted the *Churn* column to binary (0/1)

Step 1: Load & Prepare the Data

[2]:
```python
import pandas as pd

# Load the dataset
df = pd.read_csv("Telco-Customer-Churn.csv")

# Convert TotalCharges to numeric (handle empty strings)
df["TotalCharges"] = pd.to_numeric(df["TotalCharges"], errors='coerce')
df = df.dropna()  # Drop rows with missing values

# Encode categorical variables
df = pd.get_dummies(df, drop_first=True)

# Separate features and target
X = df.drop("Churn_Yes", axis=1)
y = df["Churn_Yes"]
```

Figure 2.0 dataset was prepared by Python

3.0 Sampling Strategy

Train-Test Split:

The dataset was split into 70% training and 30% testing. Figure 3.0 the method used in Python.

Step 2: Split the Data

```
[6]: from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
```

Figure 3.0 dataset was split by Python

4.0 Business Insights for Classification

4.1 Service Usage – Customers Without Security or Support Services Churn More

The usage of the company's optional services was the focus of a targeted analysis. OnlineSecurity, OnlineBackup, DeviceProtection and TechSupport are optional services which are aimed to give customers convenience, protection and additional features. Customers who actively use one or more of these services are thought to be more committed to the business and less likely to churn. We aim to determine whether service engagement contributes to churn behavior and whether it can guide future customer retention initiatives by exploring the connection between the use of these services and customer retention.

The scikit-learn library was used to initialize these models and the training dataset (*X_train*, *y_train*) was created. Each model was used to generate predictions on the test dataset (*X_test*) when training was completed. Next, four standard classification metrics: Accuracy, Precision, Recall and F1 Score were used to evaluate the predictions (*y_pred*). These metrics offer an in-depth view of each model's performance, including overall accuracy to recall, which balances accurate identification with false positives. For easy comparison and future analysis, the outcomes for every model were saved in a dictionary format. By using key performance metrics, this method enables a fair overview of several models to determine which one is the best for churn prediction. Figure 4.1.1 shows the functions to train classification models.

Step 3: Train Classification Models

[9]:
```python
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier

models = {
    "Decision Tree": DecisionTreeClassifier(),
    "Random Forest": RandomForestClassifier(),
    "K-Nearest Neighbors": KNeighborsClassifier()
}

results = {}

from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score

for name, model in models.items():
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    results[name] = {
        "Accuracy": accuracy_score(y_test, y_pred),
        "Precision": precision_score(y_test, y_pred),
        "Recall": recall_score(y_test, y_pred),
        "F1 Score": f1_score(y_test, y_pred)
    }
```

Figure 4.1.1 dataset was trained by Python

The Random Forest classifier performed better than the other methods, with the maximum accuracy of 79.2% and the best precision score of 0.658, according to the model evaluation results (Figure 4.1.2) and shown in the bar chart (Figure 4.1.3). It indicates that, among the predicted churn situations, Random Forest was the most effective in accurately identifying actual churners. With an accuracy of 75.7% and relatively similar precision and recall, the Decision Tree model showed a balanced performance, indicating that it provided predictions that were more constant across classes. Although it came at the expense of overall precision and F1 score, the K-Nearest Neighbors (KNN) model bettered Random Forest in capturing actual churners, with an accuracy of 76.9% and a recall of 0.449. In summary, Random Forest performed the best in this dataset at predicting customer attrition.

Step 4: Display Results as Table

[11]:
```
results_df = pd.DataFrame(results).T
print(results_df)
```

```
                     Accuracy  Precision    Recall  F1 Score
Decision Tree        0.772512   0.577735  0.536542  0.556377
Random Forest        0.790995   0.657895  0.445633  0.531350
K-Nearest Neighbors  0.771564   0.593381  0.447415  0.510163
```

Figure 4.1.2 result in table

Step 5: Visualize with Bar Charts

[13]:
```
import seaborn as sns
import matplotlib.pyplot as plt

results_df.plot(kind='bar', figsize=(10, 6))
plt.title("Model Comparison")
plt.ylabel("Score")
plt.ylim(0, 1)
plt.xticks(rotation=45)
plt.legend(loc="lower right")
plt.tight_layout()
plt.show()
```

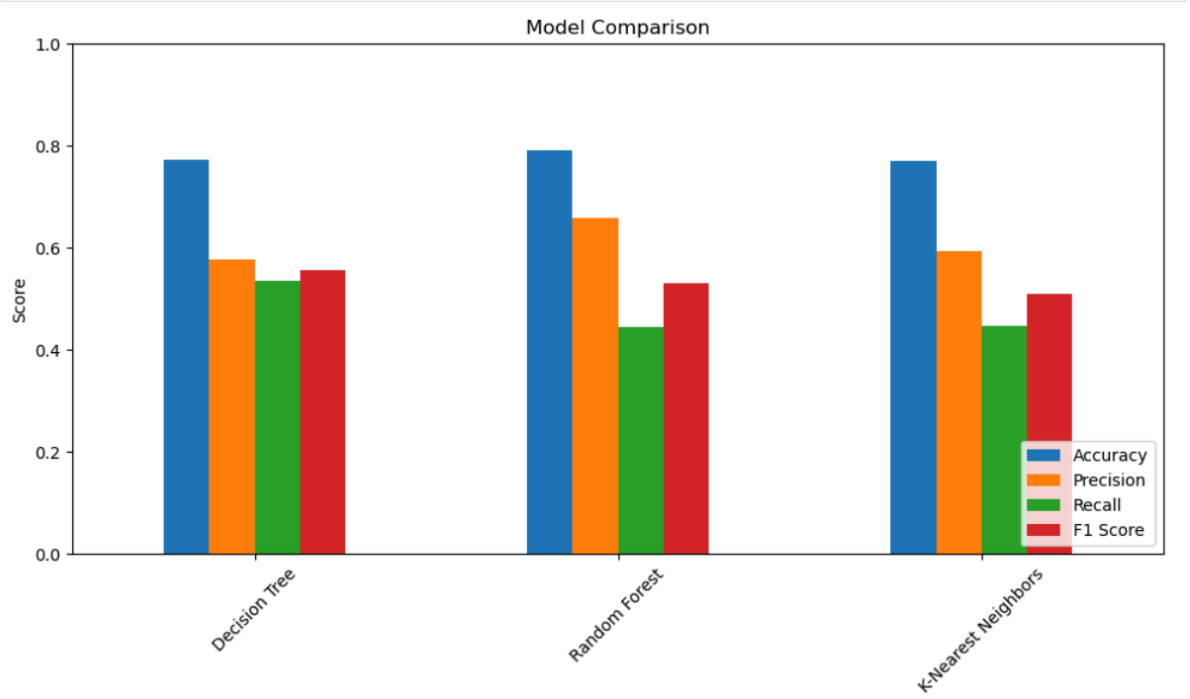

Figure 4.1.3 result in bar chart

4.2 Account & Billing – Month-to-Month Contracts and High Charges Drive Churn

In order to determine how contract types and payment methods affect churn rates, we analyze clients in this insight. Customers that have month-to-month contracts, paperless billing, and electronic check payment methods are more likely to churn, according to our findings. This pattern is particularly noticeable for those with limited service tenure and high monthly rates. These clients are more likely to withdraw from the service quickly after joining up and frequently show lower levels of loyalty. People with month-to-month contracts, in instance, have greater freedom and are therefore simpler to churn.

Using the scikit-learn toolkit, we implemented three machine learning models—Decision Tree, Random Forest, and Support Vector Machine (SVM)—to investigate this. Each model was trained to predict customer churn based on certain features after the dataset was split into training and testing sets. Following training, the models produced predictions that were assessed using F1-Score, Accuracy, Precision, and Recall. These performance metrics aid in gauging each model's ability to balance false positives and false negatives while detecting churn. We were able to identify the most dependable algorithm for this specific churn scenario by compiling the results from each model into a structured style for convenient comparison.

```
[7]: # Initialize models
dt = DecisionTreeClassifier(random_state=42)
rf = RandomForestClassifier(random_state=42)
svm = SVC()

# Train models
dt.fit(X_train, y_train)
rf.fit(X_train, y_train)
svm.fit(X_train_scaled, y_train)

# Predictions
y_pred_dt = dt.predict(X_test)
y_pred_rf = rf.predict(X_test)
y_pred_svm = svm.predict(X_test_scaled)
```

Figure 4.2.1 dataset was trained by Python using 3 classification models

```
[8]: # Confusion matrices (Decision Tree)
cm = confusion_matrix(y_test, y_pred_dt)

sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix')
plt.show()
```

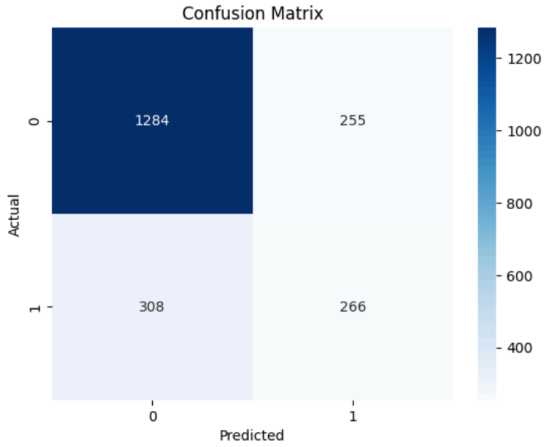

Figure 4.2.2 & 4.2.3 Confusion matrix for Decision Tree

```
[9]: # Confusion matrices (Random Forest)
cm = confusion_matrix(y_test, y_pred_rf)

sns.heatmap(cm, annot=True, fmt='d', cmap='Purples')
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix')
plt.show()
```

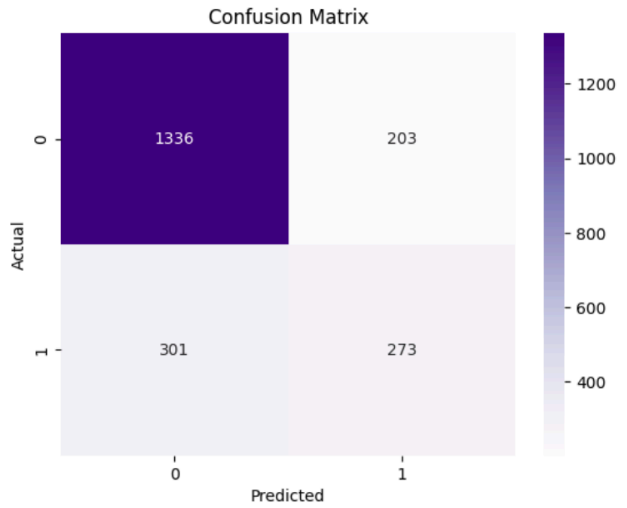

Figure 4.2.4 & 4.2.5 Confusion matrix for Random Forest

```
[13]: # Confusion matrices (SVM)
cm = confusion_matrix(y_test, y_pred_svm)

sns.heatmap(cm, annot=True, fmt='d', cmap='Greens')
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix')
plt.show()
```

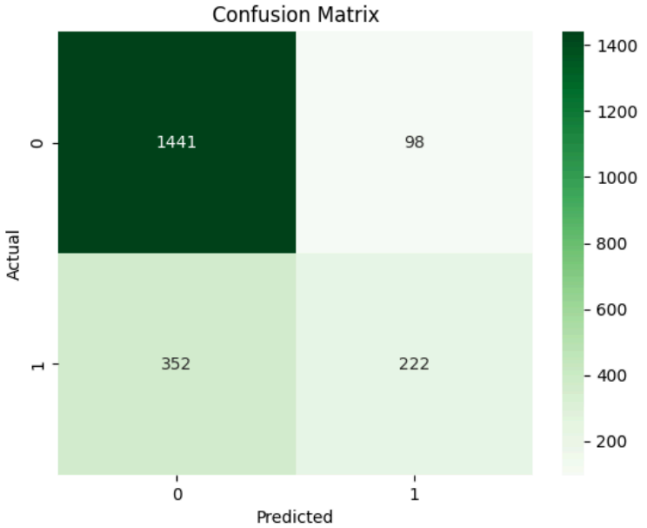

Figure 4.2.6 & 4.2.7 Confusion matrix for SVM

By generating the confusion matrix for each classification model, we obtained the values of true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN). Accuracy, precision, recall, and F1-score were then computed using these data. Each of these metrics offers a different perspective on the model's performance.

Model	Accuracy	Precision	Recall	F1 Score
Decision Tree	0.733554	0.510557	0.463415	0.485845
Random Forest	0.761477	0.573529	0.475610	0.520000
SVM	0.787033	0.693750	0.386760	0.496644

Table 4.2.1 Classification Model Performance Comparison Table

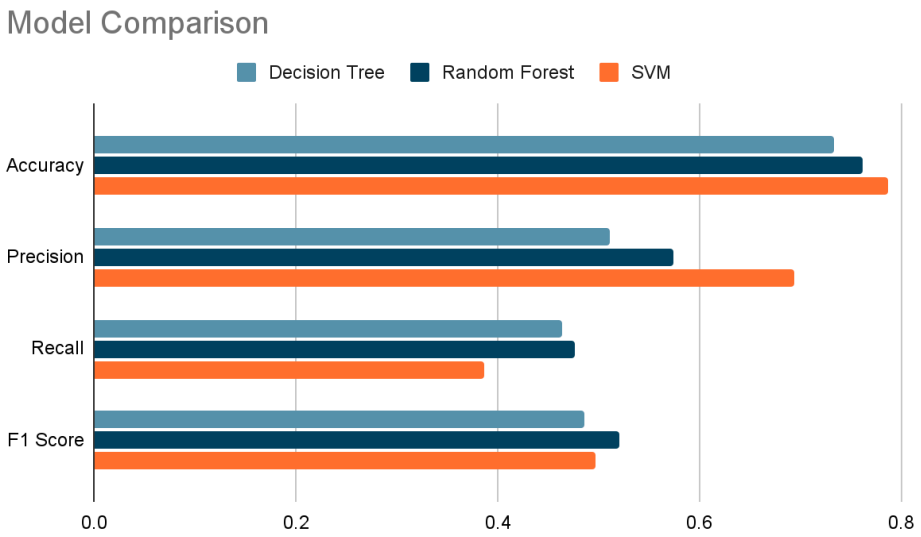

Figure 4.2.8 Model Performance Comparison in Bar Chart

In this analysis, Decision Tree, Random Forest, and Support Vector Machine (SVM) were applied to assess customer churn based on contract type, tenure, and monthly charges. Among those it predicts to churn, the SVM model showed the best overall accuracy (78.7%) and precision (69.4%), indicating that it is especially good at predicting churners. However, its recall (38.7%) indicates that it may miss a significant number of actual churners. In contrast, the Random Forest model performed SVM in terms of capturing a greater percentage of real churners, which is important in churn prediction tasks. Its accuracy was 76.1%, and its recall was better at 47.6%. Although its low complexity, the Decision Tree model provided a more balanced precision and recall as well as a respectable accuracy of 73.4%. These findings are consistent with the dataset's nature, which shows that a large number of consumers with high monthly fees and month-to-month contracts are more likely to leave. In these situations, models that can handle complicated patterns and feature interactions, like Random Forest, typically perform better. Overall, Random Forest provides a strong balance between precision and recall, making it a suitable choice for churn prediction in this context.

4.3 Demographics – Senior Citizens and Singles Are More Likely to Churn

This section examines the impact of demographic factors on customer churn behavior. The analysis highlights that senior citizens, along with individuals who are single or without dependents, tend to leave the service at higher rates. These customers may be more cost-conscious, less reliant on the service, or less engaged overall, which can lead to lower retention. This pattern is especially visible among users who may not perceive long-term value in remaining subscribed.

To investigate this further, three machine learning models—Decision Tree, Random Forest, and Support Vector Machine (SVM)—were applied using demographic attributes as key input features. The dataset was split into training and testing portions, and model predictions were evaluated using metrics such as Accuracy, Precision, Recall, and F1 Score. These indicators help assess how well each model detects churn while minimizing errors in classification. The side-by-side performance comparison allowed us to identify the most suitable model for predicting churn among different demographic groups.

```
# Train/test split
X_train, X_test, y_train, y_test =
train_test_split(X_scaled, y,
test_size=0.2, random_state=42)

# Initialize models
models = {
    "Random Forest":
RandomForestClassifier(random_state=42),
    "Decision Tree":
DecisionTreeClassifier(random_state=42),
    "SVM": SVC(kernel='rbf',
random_state=42)
}

# Train and evaluate all models
for name, model in models.items():
    model.fit(X_train, y_train)
    evaluate_model(name, model, X_test,
y_test)
```

Figure 4.3.1 dataset was trained by Python using 3 classification models

```
# Helperfunction to plot confusion matrix
def plot_confusion_matrix(cm, title):
    plt.figure(figsize=(5, 4))
    sns.heatmap(cm, annot=True, fmt='d',
cmap='Blues', xticklabels=['No Churn',
'Churn'], yticklabels=['No Churn',
'Churn'])
    plt.xlabel('Predicted')
    plt.ylabel('Actual')
    plt.title(f'Confusion Matrix -
{title}')
    plt.tight_layout()
    plt.show()

# Confusion matrices for RF, DT and SVM
for name, model in models.items():
    y_pred = model.predict(X_test)
    cm = confusion_matrix(y_test, y_pred)
    plot_confusion_matrix(cm, name)
```

Figure 4.3.2 Function designed to plot each confusion matrix

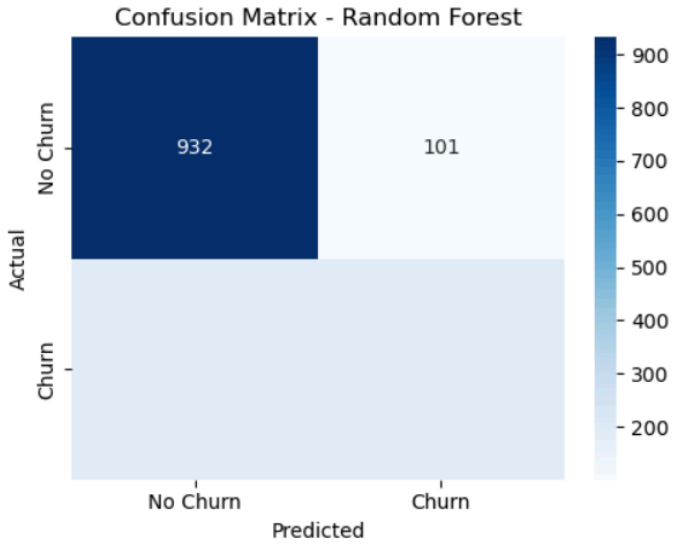

Figure 4.3.3 Confusion matrix for Random Forest

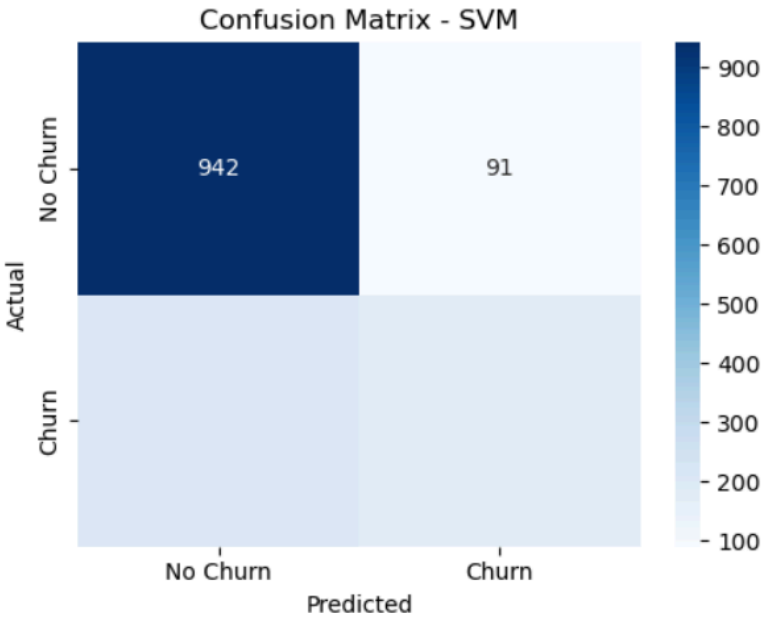

Figure 4.3.4 Confusion matrix for SVM

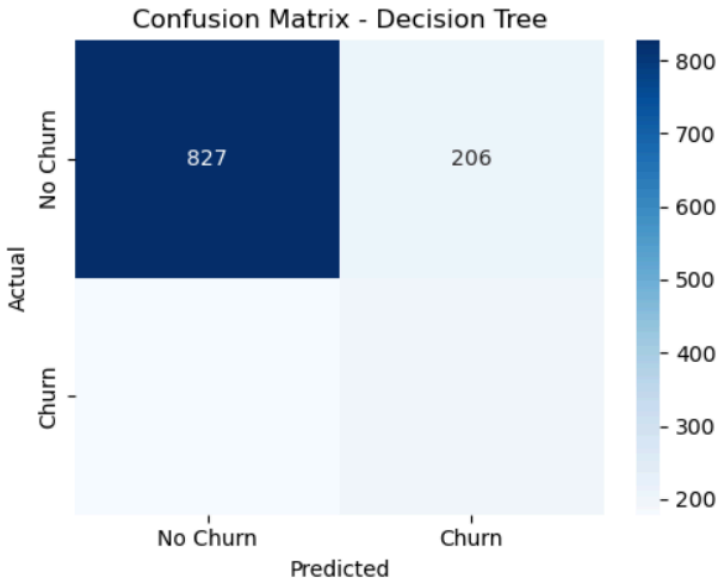

Figure 4.3.5 Confusion matrix for Decision Tree

To analyze churn behavior based on demographics, three classification models—Decision Tree, Random Forest, and Support Vector Machine (SVM)—were trained using selected features such as SeniorCitizen, Partner, and Dependents. Each model's predictions were evaluated using four standard metrics: Accuracy, Precision, Recall, and F1 Score. The performance results are summarized in Table 4.3.1 below:

Model	Accuracy	Precision	Recall	F1 Score
Decision Tree	0.726368	0.486284	0.521390	0.503225
Random Forest	0.791755	0.643109	0.486631	0.554033
SVM	0.791044	0.652671	0.457219	0.537735

Table 4.3.1 Classification Model Performance Comparison Table

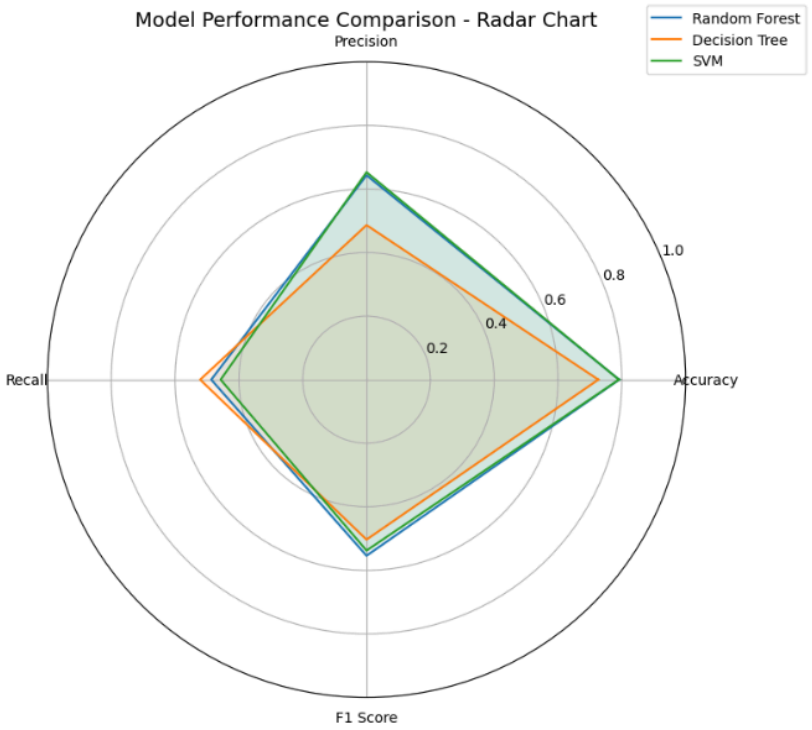

Figure 4.3.6 Model Performance Comparison in Radar Chart

This analysis utilized Decision Tree, Random Forest, and Support Vector Machine (SVM) models to evaluate customer churn in relation to demographic factors, including age group, marital status, and presence of dependents. The SVM model recorded the highest accuracy (79.1%) and precision (65.3%), showing strong capability in correctly predicting churners. Nonetheless, its recall (45.7%) indicates it may fail to detect many actual churn cases. On the other hand, the Random Forest model demonstrated the most balanced performance, achieving a slightly higher accuracy (79.2%) and the top F1 score (0.554), making it more dependable in identifying true churners. Although the Decision Tree model is simpler, it still delivered fairly consistent precision and recall, with a reasonable accuracy of 72.6%. Overall, the results suggest that demographic data is a valuable predictor of churn, and that Random Forest stands out as the most effective model for this use case, thanks to its strength in managing complex data relationships.