Pendahuluan

```
df = pd.read_csv('Titanic-Dataset.csv')
df.head()
```

Setelah kita mendapatkan dataset Titanic (import library dan import dataset), kita drop data yang kira-kira tidak perlu digunakan untuk proses train model. Berikut adalah list dari column yang ada di dataset

```
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   PassengerId  891 non-null    int64  
 1   Survived     891 non-null    int64  
 2   Pclass       891 non-null    int64  
 3   Name         891 non-null    object 
 4   Sex          891 non-null    object 
 5   Age          714 non-null    float64
 6   SibSp        891 non-null    int64  
 7   Parch        891 non-null    int64  
 8   Ticket       891 non-null    object 
 9   Fare         891 non-null    float64
 10  Cabin        204 non-null    object 
 11  Embarked     889 non-null    object 
```

Ada PassengerId, Name (nama penumpang), Ticket (nama ticket), dan Cabin (nama cabin) yang tidak perlu diikutsertakan dalam analisis, maka kita drop itu semua.

```
df_dropped = df.drop(['PassengerId', 'Name', 'Ticket', 'Cabin'], axis=1)
df_dropped.head()
```

Setelah melakukan drop feature yang tidak diperlukan, kita mengecek apakah ada missing values.

```
Survived      0
Pclass        0
Sex           0
Age         177
SibSp         0
Parch         0
Fare          0
Embarked      2
dtype: int64
```

Ternyata didapat missing value berada di feature Age sebanyak 177 record fan Embarked sebanyak 2 record. Oleh karena itu, kita perlu melakukan data preprocessing.

```
df_dropped['Age'].fillna(df_dropped['Age'].median(), inplace=True)
```

Untuk age dilakukan imputasi median, imputasi sendiri adalah teknik dalam preprocessing data yang bertujuan untuk mengisi atau mengganti nilai yang mengandung missing values pada dataset agar bisa dilakukan analisis lanjutan apabila datanya sudah lengkap dan konsisten. Sehingga imputasi median berarti memasukkan missing values pada dataset dengan nilai tengah pada satu feature tersebut.

Sementara untuk Embarked karena termasuk column categorical, maka penanganannya nanti saja.

Dikarenakan tujuan kita melakukan analisis ini adalah untuk membuat model dalam memprediksi hasil biner yang selamat untuk penumpang Titanic. Maka dari itu, feature yang menjadi patokan selamat atau tidak adalah feature Survived. Oleh karena itu, kita memisahkan feature Survived dengan feature yang akan dijadikan bahan perbandingannya.

```
y = df_dropped['Survived']
```

Setelah itu kita memisahkan menjadi dua tipe column untuk dilakukan proses preprocessing StandardScaler pada column Numerical dan preprocessing LabelEncoder pada column Categorical. Yang kemudian akan digabung kembali untuk menjadi variable X (selected features). Jangan lupa untuk menghapus Survived karena Survived dipisah menjadi variable y (target).

```
num_col = df_dropped.select_dtypes(exclude='object').columns.tolist()
num_col.remove('Survived')
cat_col = df_dropped.select_dtypes(include='object').columns.tolist()
```

Setelah dipisah, dilakukan preprocessing StandardScaler untuk numerical column, yang dimana StandardScaler adalah metode preprocessing dimana metode tersebut akan melakukan standarisasi fitur dengan menghapus rata-rata dan menskalakan unit varian. Itu semua dilakukan pada numerical column.

```
scaler = StandardScaler()
df_num_scaled = pd.DataFrame(scaler.fit_transform(df_dropped[num_col]), columns=num_col)
```

Kemudian, lakukan proses encoding yang dimana nilai kategorikal diubah menjadi nilai numerical.

```
df_cat_encoded = df_dropped[cat_col].copy()

le = LabelEncoder()
for col in cat_col:
    df_cat_encoded[col] = le.fit_transform(df_cat_encoded[col])
```

Setelah itu digabungkan kembali column numerical dan column kategorikal.

```
X = pd.concat([df_num_scaled, df_cat_encoded], axis=1)
```

Setelah itu, kita lakukan proses train dan test model

Pertanyaan

1. **Split and dataset into a training set and a testing set to evaluate your model's performance (70:30)**

   Kita lakukan split dengan jumlah ratio training data 70 dan test data 30 dengan menggunakan variable X dan y yang tadi telah dibuat.

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

2. **Develop a Support Vector Machine model using your selected features.**
   Model pertama yang ingin kita test adalah Support Vector Machine (SVM). SVM merupakan salah satu model machine learning yang digunakan dalam tujuan melakukan klasifikasi dengan memisahkan dua buah class pada input space, dengan menggunakan hipotesis berupa fungsi linear dalam sebuah ruang fitur berdimensi tinggi.

```
classifier = SVC(kernel='linear', random_state=42)
classifier.fit(X_train, y_train)
```

Menggunakan fungsi SVC yang berasal dari library sklearn.svm. Linear sendiri itu memiliki tujuan memisahkan data dengan fungsi linear (garis lurus).
Seletlah selesai kita melakukan prediksi

```
y_pred = classifier.predict(X_test)
```

3. **Evaluate the performance of your model using metrics such as accuracy, precision, recall, and the ROC-AUC performance**

   Setelah melakukan prediksi kita melakukan evaluasi model untuk menghitung accuracy, precision, recall, dan ROC-AUC. Accuracy, Precision, dan Recall sebenarnya bisa dihitung dari Confusion Matrix. Hanya saja untuk menilai metric ini saya pakai langsung.

```
classifier = SVC(kernel='linear', probability=True, random_state=42)
classifier.fit(X_train, y_train)
y_prob = classifier.predict_proba(X_test)[:, 1]
```

Accuracy sendiri merupakan nilai seberapa besar prediksi tersebut benar dari model yang dibuat.
Accuracy = (TP + TN) / (TP + FP + TN + FN)
Precision adalah nilai seberapa besar ratio nilai prediksi positif yang benar terhadap hasil yang positif (True Positive or False Positive)
Precision = TP / (TP + FP)
Recall adalah nilai seberapa besar kasus positif dari semua data yang akuratnya dengan kasus yang positif (True Positive or False Negative)
Recall= TP / (TP + FN).

```
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred)
recall = recall_score(y_test, y_pred)
conf_mat = confusion_matrix(y_test, y_pred)
roc_auc = roc_auc_score(y_test, y_prob)

print(f"Accuracy  : {accuracy:.2f}")
print(f"Precision : {precision:.2f}")
print(f"Recall    : {recall:.2f}")
print(f"ROC AUC   : {roc_auc:.2f}")
```

ROC (Receiver Operating Characteristic) dan AUC (Area Under Curve) biasanya diimplementasikan ke dalam curve (kurva) yang dimana kurva ROC divisualisasikan berdasarkan nilai yang didapatkan dari hasil perhitungan Confusion Matrix yang didasari oleh False Positive Rate (FPR) dan True Positive Rate (TPR). Cara melihat perbandingannya sendiri juga bisa dilihat dari luas di bawah kurva (AUC). Penggunaan kurva ROC sendiri ini bisa memvisualiasikan semua threshold klasifikasi yang memungkinkan.

4. **Compare it with Decision Tree and Random Forest. Analyze the results**
   a. **Decision Tree**

   Melakukan training model

```
dt_model = DecisionTreeClassifier(random_state=42)
dt_model.fit(X_train, y_train)
```

   Melakukan test model

```
y_pred_dt = dt_model.predict(X_test)
```

```
y_prob_dt = dt_model.predict_proba(X_test)[:, 1]
```

```
accuracy_dt = accuracy_score(y_test, y_pred_dt)
precision_dt = precision_score(y_test, y_pred_dt)
recall_dt = recall_score(y_test, y_pred_dt)
conf_mat_dt = confusion_matrix(y_test, y_pred_dt)
roc_auc_dt = roc_auc_score(y_test, y_prob_dt)

print(f"Accuracy  : {accuracy_dt:.2f}")
print(f"Precision : {precision_dt:.2f}")
print(f"Recall    : {recall_dt:.2f}")
print(f"ROC AUC   : {roc_auc_dt:.2f}")
```

   b. **Random Forest**

   Melakukan training model

```
rf_model = RandomForestClassifier(n_estimators=100, random_state=42)
rf_model.fit(X_train, y_train)
```

   Melakukan test model

```
y_pred_rf = rf_model.predict(X_test)
```

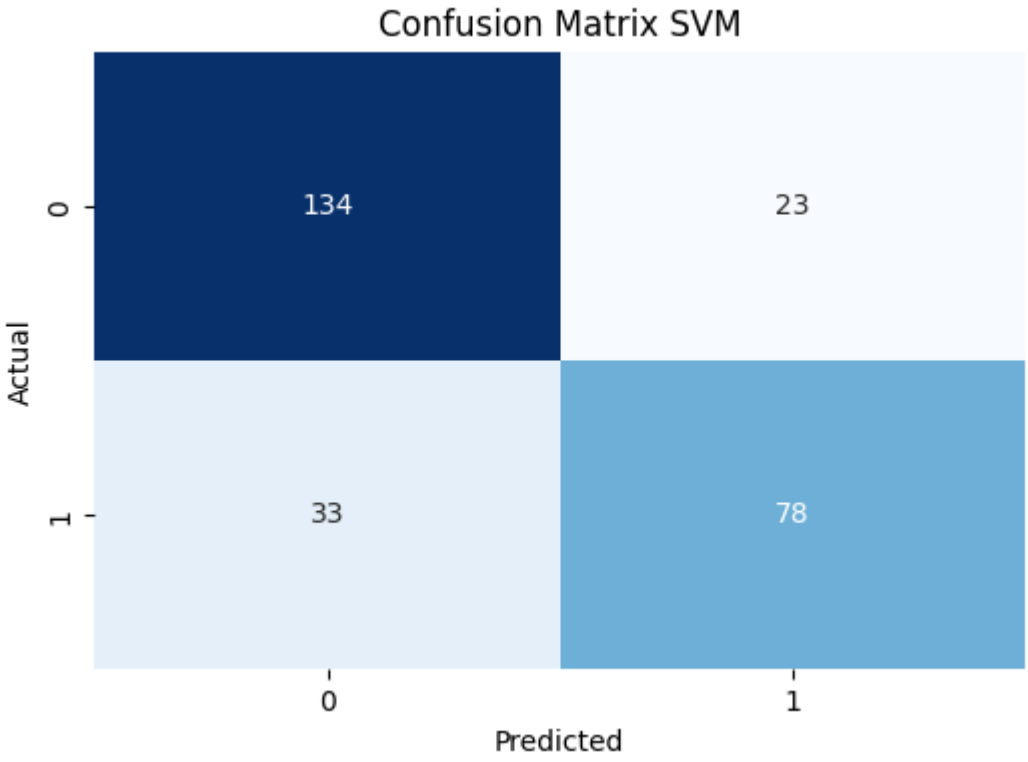
```
 y_prob_rf = rf_model.predict_proba(X_test)[:, 1]
```

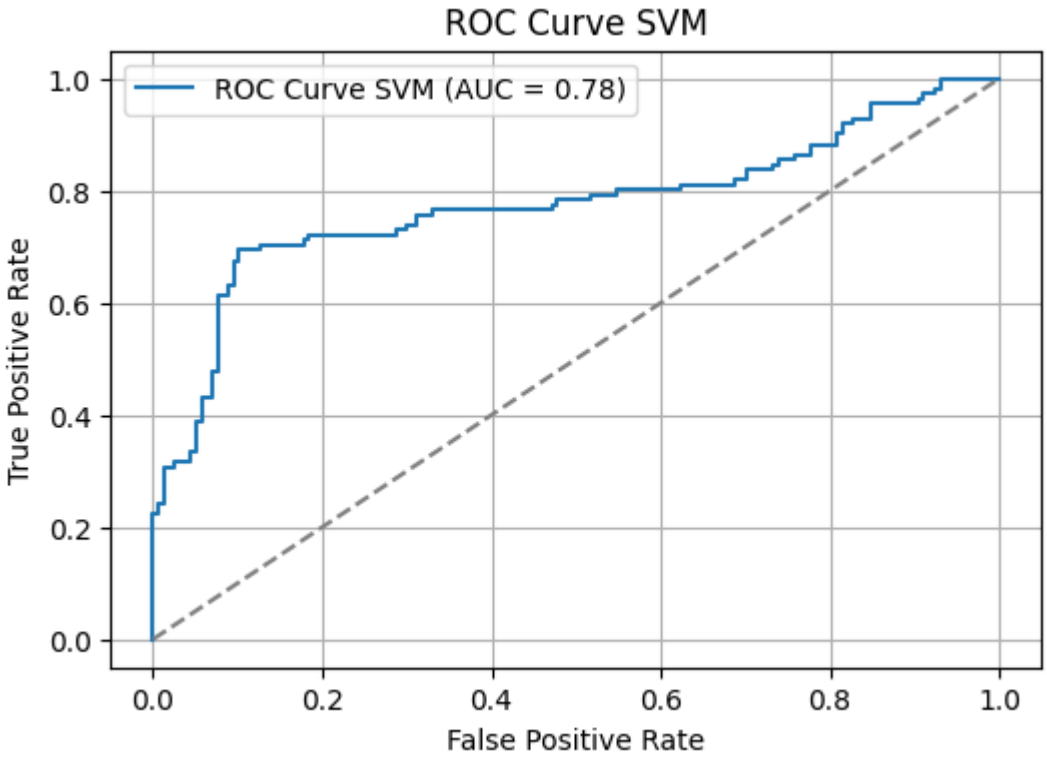
```
accuracy_rf = accuracy_score(y_test, y_pred_rf)
precision_rf = precision_score(y_test, y_pred_rf)
recall_rf = recall_score(y_test, y_pred_rf)
conf_mat_rf = confusion_matrix(y_test, y_pred_rf)
roc_auc_rf = roc_auc_score(y_test, y_prob_rf)

print(f"Accuracy  : {accuracy_rf:.2f}")
print(f"Precision : {precision_rf:.2f}")
print(f"Recall    : {recall_rf:.2f}")
print(f"ROC AUC   : {roc_auc_rf:.2f}")
```

Result

SVM

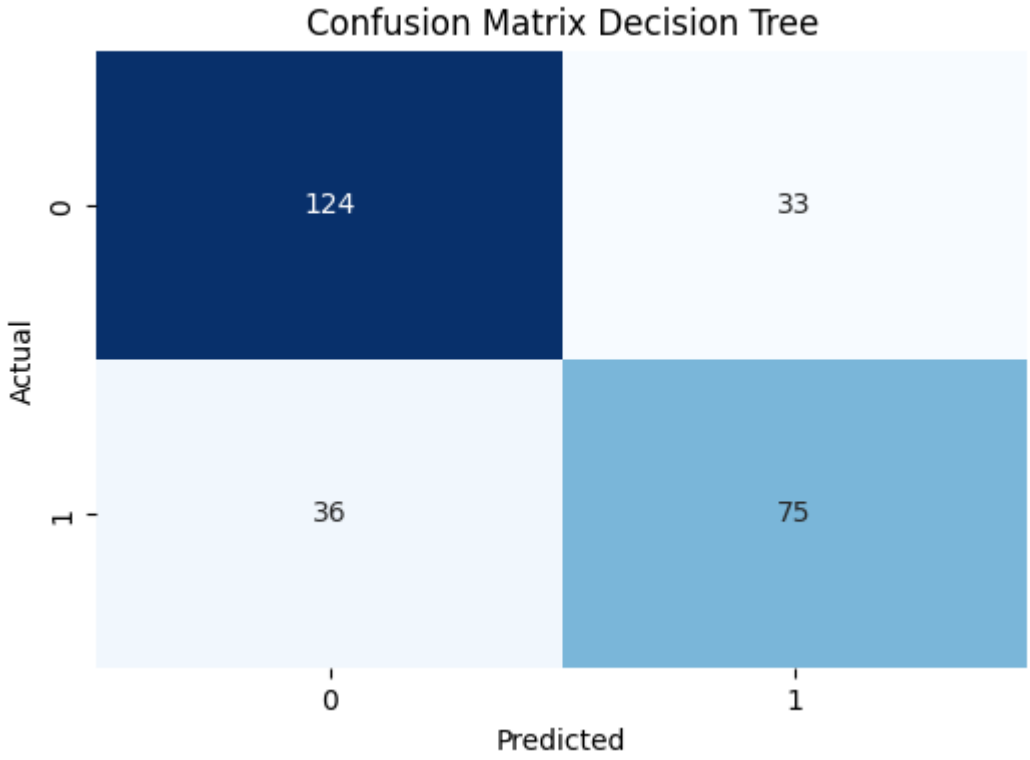
```
SVM Classifier

Accuracy  : 0.79
Precision : 0.77
Recall    : 0.70
ROC AUC   : 0.78
```

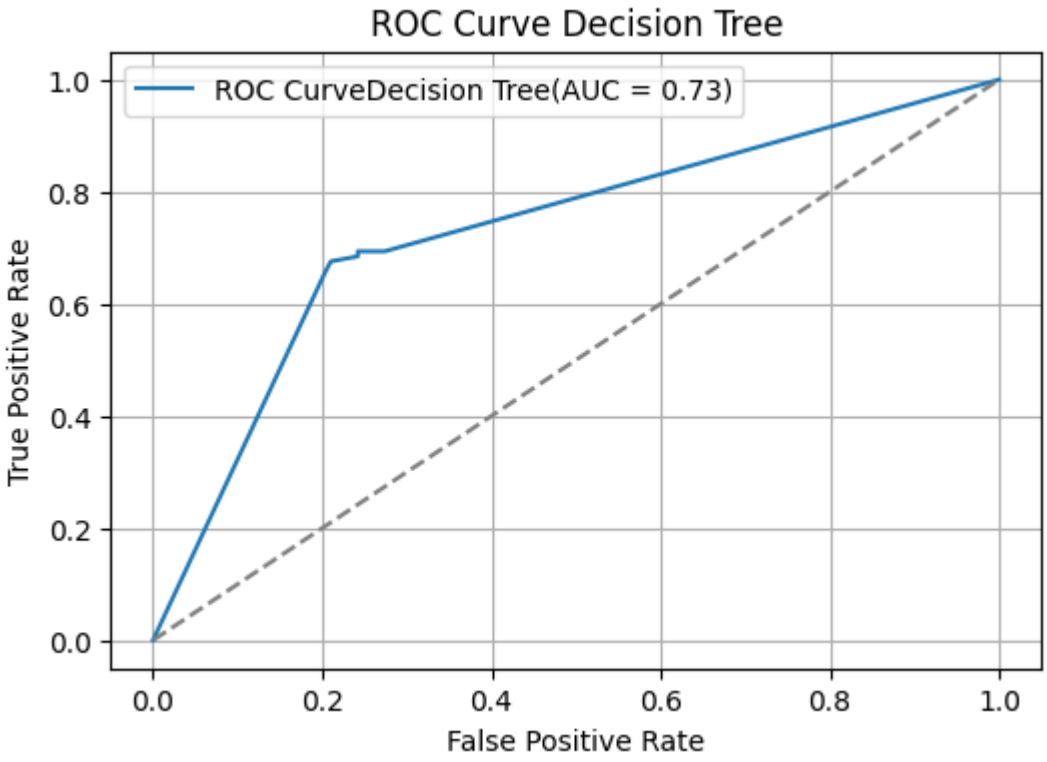




Decision Tree

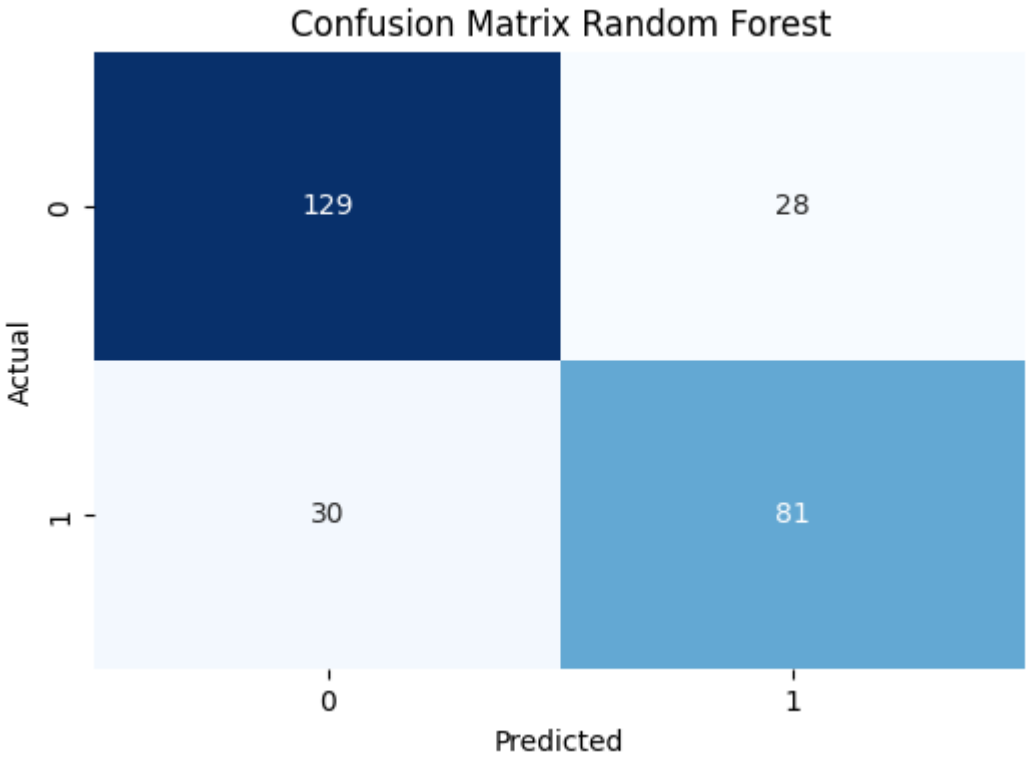
```
Decision Tree Classifier

Accuracy  : 0.74
Precision : 0.69
Recall    : 0.68
ROC AUC   : 0.73
```

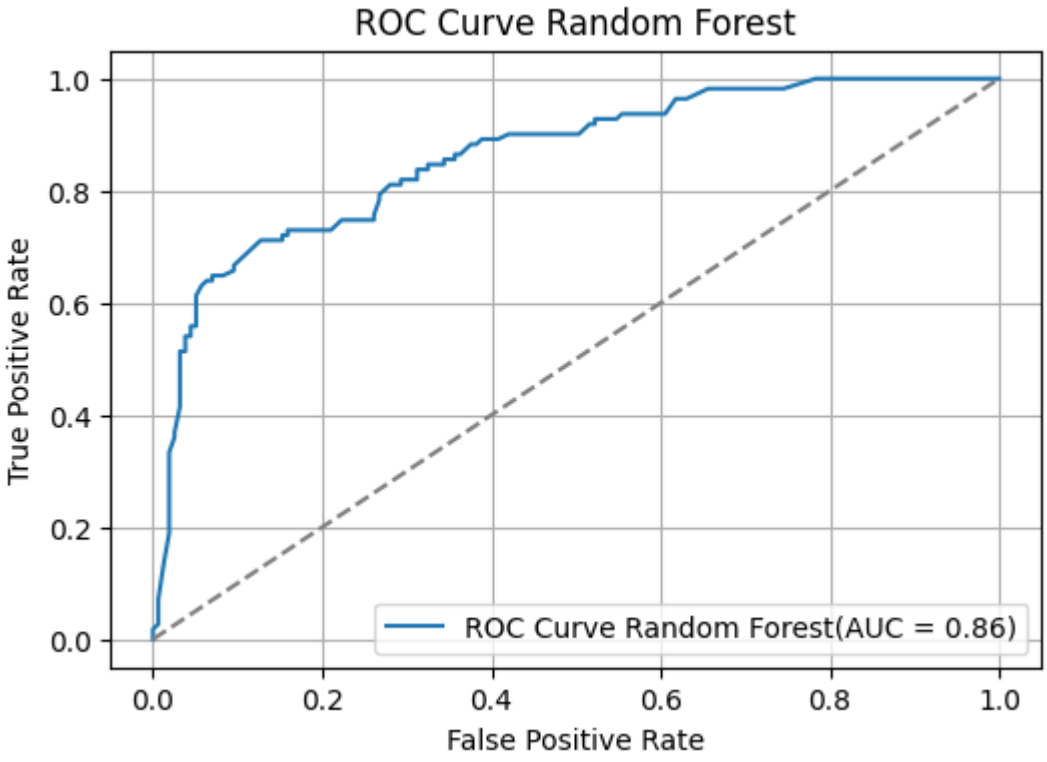




Random Forest

```
Random Forest Classifier

Accuracy  : 0.78
Precision : 0.74
Recall    : 0.73
ROC AUC   : 0.86
```

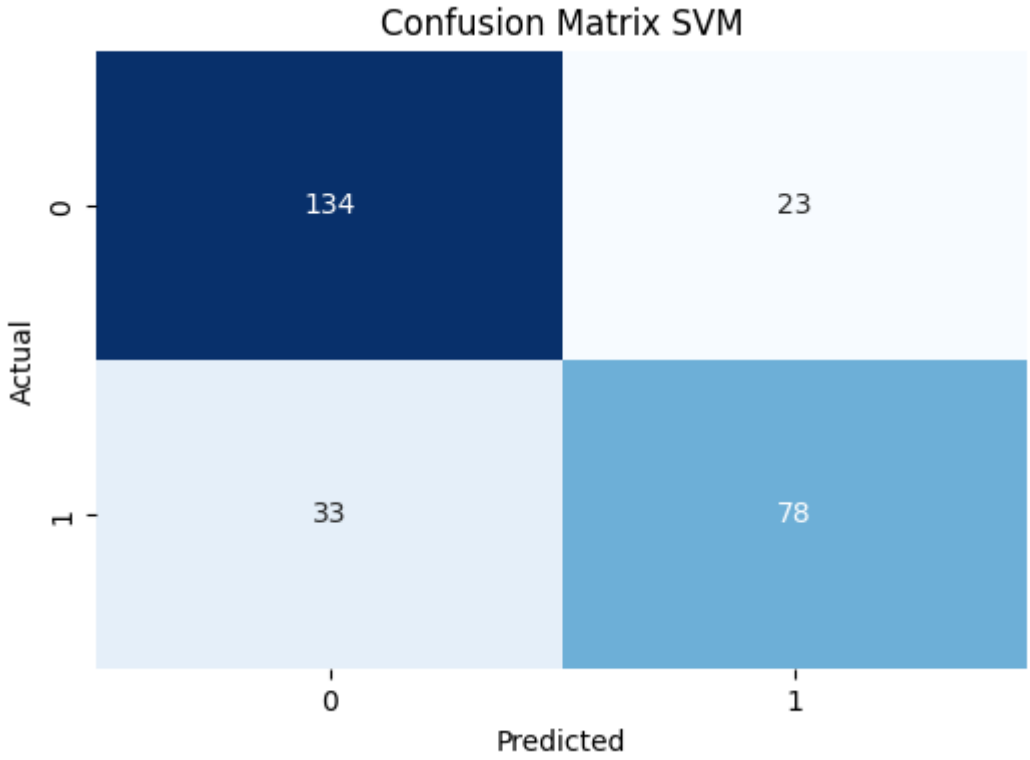




Model	Accuracy	AUC	False Positive
SVM	0.79	0.78	23
Decision Tree	0.74	0.73	33
Random Forest	0.78	0.86	28

Dari hasil metrics yang paling bagus dalam segi akurasi adalah SVM (0.79), disusul oleh Random Forest (0.78), dan terakhir Decision Tree (0.74). Tapi dari segi nilai ROC-AUC yang paling bagus adalah Random Forest (0.86), disusul oleh SVM (0.78), dan terakhir Decision Tree (0.73).
Untuk disini keunggulan pada SVM di segi akurasi dan ROC-AUC di segi ROC-AUC disini bisa ditentukan kembali pada false positive (kondisi dimana prediksinya mengatakan dia hidup tapi aslinya tidak = kesalahan prediksi). Tapi ketentuannya ternyata SVM yang paling sedikit kesalahannya. Sehingga SVM merupakan model terbaik untuk kasus ini.

5. **Provide a detailed analysis of the confusion matrix to understand the true positives, false positives, true negatives, and false negatives**



Saya mengambil contoh confusion matrix dari SVM. Disini dapat dilihat bahwa ada dibagi menjadi 4 kotak tapi cara melihatnya dari x dan y nya. X dan y disini adalah labelnya. Dari x itu predicted dimana model tersebut memprediksi, sedangkan y itu actual itu menyatakan dimana hasil kenyataan tersebut. Lalu untuk 0 dan 1 itu sebuah klasifikasi untuk column Survived. 0 itu berarti Not Survived, sedangkan nilai 1 itu Survived.

Untuk confusion Matrix disini karena dibagi menjadi 4 kotak, maka masing-masing memiliki nilai yaitu ada TP, FP, FN, dan TN
True Positive : Kondisi dimana model memprediksi selamat dan orang tersebut memang selamat (Predicted: 1, Actual: 1) = 78
False Positive : Kondisi dimana model memprediksi selamat tapi orang tersebut tidak selamat (Predicted: 1, Actual: 0) = 23

False Negative : Kondisi dimana model memprediksi tidak selamat dan orang tersebu memang tidak selamat (Predicted: 0, Actual: 0) = 134
True Negative : Kondisi dimana model memprediksi tidak selamat tapi orang tersebut selamat (Predicted: 0, Actual: 1) = 33
Dari sini, cara melihatnya adalah True Positive dan False Negative adalah contoh model yang berhasil prediksi dengan benar, sedangkan False Positive dan True Negative adalah contoh model yang tidak berhasil prediksi dengan benar (bias). Di SVM sendiri TP dan FN jumlahnya lebih banyak dibandingkan FP dan TN, untuk semua model yang diuji juga seperti itu (Decision Tree dan Random Forest) tapi jumlah kesalahannya (FP dan TN) yang paling kecil adalah SVM (SVM : 56, Decision Tree : 69, Random Forest : 58), maka dari itu SVM adalah model yang paling terbaik untuk kasus ini.

Referensi

Lusmana, P. H., Paramudita, A. (2021). Pemanfaatan Machine Learning untuk Imputasi Data Penerapan pada Data Output Survei Tahunan Industri Manufaktur Tahun 2021. p513

Prasetyo, V. R., Mercifia, M., Averina, A., Lauren, Suntoyo, Budiarjo. (2022). PREDIKSI RATING FILM PADA WEBSITE IMDB MENGGUNAKAN METODE NEURAL NETWORK. NERO, 7(1), p3

Parapat, I. M., Furqon, M. T., Sutrisno. (2018). Penerapan Metode Support Vector Machine (SVM) Pada Klasifikasi Penyimpangan Tumbuh Kembang Anak. Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer, 2(10), p3165

Kristiawan, Andreas Widjaja. (2021). Perbandingan Algoritma Machine Learning dalam Menilai Sebuah Lokasi Toko Ritel. Jurnal Teknik Informatika dan Sistem Informasi, 7(1), p41-46

Sathyanarayanan, S., Tantri, B. R.(2024). Confusion Matrix-Based Performance Evaluation Metrics. African Journal of Biomedical Research,27(4), p4025

https://www.geeksforgeeks.org/auc-roc-curve/