Eksplorasi Library Decision Tree pada Jupyter Notebook

Anggota (K03):

- Benidictus Galih Mahar Putra - 13519159
- Alvin Wilta - 13519163

Load Datasets

Library yang digunakan pada notebook ini adalah:

- pandas (operasi sederhana pada dataset)
- sklearn (modul machine learning)
- graphviz (visualisasi tree id3)
- id3 (melakukan algoritma id3)
- numpy (operasi matematika sederhana)
- matplotlib (visualisasi tree decision tree classification)
- seaborn (visualisasi confusion matriks)
- six, sys (menangani konflik dari library id3)

In [ ]:
```
import pandas as pd
import graphviz
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
from sklearn import datasets, model_selection, tree
from sklearn.cluster import KMeans
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import accuracy_score, f1_score, confusion_matrix
from sklearn.neural_network import MLPClassifier
from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
from sklearn.tree import export_graphviz as export_graphviz_sk
from sklearn.linear_model import LogisticRegression
import six
import sys
sys.modules['sklearn.externals.six'] = six
from id3 import Id3Estimator, export_graphviz
```

In [ ]:
```
# load datasets
cancer = datasets.load_breast_cancer(as_frame=True)
dttennis = pd.read_csv("../data/play_tennis.csv")
```

In [ ]:
```
# copy a dataset
dtcancer = cancer.data.copy()
dtcancer['res'] = cancer.target
```

In [ ]:
```
# Inisialisasi datagrame kosong untuk rekap semua nilai skor
```

```
recap = pd.DataFrame(columns=['Pembelajaran','Dataset','Tipe','Nilai'])
```

In [ ]:
```
# declare function
def addToRecap(pemb, cancer_acc, cancer_f1, tennis_acc, tennis_f1):
    if len(recap.index) != 0:
        if (recap['Pembelajaran'].loc[len(recap.index)-1] == pemb):
            recap.drop(recap.tail(4).index, inplace=True)
    c = 'Cancer'
    t = 'Tennis'
    acc = 'Accuracy Score'
    f1 = 'F1 Score'
    i = len(recap)
    recap.loc[i] = [pemb, c, acc, cancer_acc]
    recap.loc[i+1] = [pemb, c, f1, cancer_f1]
    recap.loc[i+2] = [pemb, t, acc, tennis_acc]
    recap.loc[i+3] = [pemb, t, f1, tennis_f1]

def simpleDescribe(df: datasets):
    print('Jumlah kolom: ' + str(len(df.columns)))
    print('Jumlah baris: ' + str(len(df)))
    print()
    print('Kolom pada tabel:')
    print(df.columns.tolist())

def printScore(acc, f1):
    print('Accuracy   :', acc)
    print('F1 Score   :', f1)

def printSummary(pemb, cacc, cf1, tacc, tf1):
    addToRecap(pemb, cacc, cf1, tacc, tf1)
    print('==============================')
    print('Cancer Dataset:')
    print('Accuracy   :', cacc)
    print('F1 Score   :', cf1)
    print('==============================')
    print('Tennis Dataset')
    print('Accuracy   :', tacc)
    print('F1 Score   :', tf1)
    print('==============================')
```

In [ ]:
```
# Gambar Confusion Matrix
def drawConfusionMatrix(title,pred,truth):
    matriks = confusion_matrix(truth, pred)
    group_names = ['True Neg', 'False Pos', 'False Neg', 'True Pos']
    group_counts = ["{0:0.0f}".format(value) for value in
                    matriks.flatten()]
    group_percentages = ["{0:.2%}".format(value) for value in
                         matriks.flatten()/np.sum(matriks)]
    labels = [f"{v1}\n{v2}\n{v3}" for v1, v2, v3 in
            zip(group_names, group_counts, group_percentages)]
    labels = np.asarray(labels).reshape(2, 2)
    ax = sns.heatmap(matriks, annot=labels, fmt='', cmap='Blues')
    ax.set_title(title+'\n\n')
    ax.set_xlabel('\nPredicted Values')
    ax.set_ylabel('Actual Values ')
    # Ticket labels - List must be in alphabetical order
    ax.xaxis.set_ticklabels(['False', 'True'])
    ax.yaxis.set_ticklabels(['False', 'True'])
    # Display the visualization of the Confusion Matrix.
    plt.show()
```

Data Breast Cancer

Sampel data kanker payudara yang disediakan oleh data internal sklearn:

In [ ]:
```
dtcancer.head()
```

Out[ ]:

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	c
0	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	0.2419	
1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	0.1812	
2	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	0.2069	
3	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	0.2597	
4	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	0.1809	

5 rows × 31 columns

In [ ]:
```
simpleDescribe(dtcancer)
```

```
Jumlah kolom: 31
Jumlah baris: 569

Kolom pada tabel:
['mean radius', 'mean texture', 'mean perimeter', 'mean area', 'mean smoothn
ess', 'mean compactness', 'mean concavity', 'mean concave points', 'mean sym
metry', 'mean fractal dimension', 'radius error', 'texture error', 'perimete
r error', 'area error', 'smoothness error', 'compactness error', 'concavity
error', 'concave points error', 'symmetry error', 'fractal dimension error',
'worst radius', 'worst texture', 'worst perimeter', 'worst area', 'worst smo
othness', 'worst compactness', 'worst concavity', 'worst concave points', 'w
orst symmetry', 'worst fractal dimension', 'res']
```

Data Tennis

Sampel data tennis yang didapatkan dari file eksternal pada folder data:

In [ ]:
```
dttennis.head()
```

Out[ ]:

	day	outlook	temp	humidity	wind	play
0	D1	Sunny	Hot	High	Weak	No
1	D2	Sunny	Hot	High	Strong	No
2	D3	Overcast	Hot	High	Weak	Yes
3	D4	Rain	Mild	High	Weak	Yes
4	D5	Rain	Cool	Normal	Weak	Yes

In [ ]:
```
simpleDescribe(dttennis)
```

```
Jumlah kolom: 6
Jumlah baris: 14

Kolom pada tabel:
['day', 'outlook', 'temp', 'humidity', 'wind', 'play']
```

Terdapat beberapa hal penting yang bisa dilihat disini, yaitu:

1. Kolom 'day' pada dataset merupakan data hari ke-sekian dan merupakan urutan hari dalam minggu. Untuk bisa mengetahui pola atau pengaruh 'urutan hari' dari sebuah tabel terhadap play adalah dengan melakukan operasi modulo 7 untuk melihat pengaruh hari (senin, selasa, dll) terhadap atribut play. Akan tetapi karena data yang dipakai sangat kecil (14 record) maka kolom ini dapat di-drop karena tidak terlalu berpengaruh.
2. Data pada dataset tersebut masih berupa string dan perlu untuk di-encode sesuai dengan kategorinya

In [ ]:
```
dttennis.drop(columns=['day'], inplace=True)
dttennis.head()
```

Out[ ]:

	outlook	temp	humidity	wind	play
0	Sunny	Hot	High	Weak	No
1	Sunny	Hot	High	Strong	No
2	Overcast	Hot	High	Weak	Yes
3	Rain	Mild	High	Weak	Yes
4	Rain	Cool	Normal	Weak	Yes

Encode Play Tennis Column

In [ ]:
```
le = LabelEncoder()
dttennis_old = dttennis.copy()
for col in dttennis_old[1:]:
    dttennis[col] = le.fit_transform(dttennis_old[col])
dttennis.head()
```

Out[ ]:

	outlook	temp	humidity	wind	play
0	2	1	0	1	0
1	2	1	0	0	0
2	0	1	0	1	1
3	1	2	0	1	1
4	1	0	1	1	1

Split Dataset

Split into test and train

Memisahkan dataset menjadi 2 bagian, test dan train. Pembagian dilakukan secara random dengan 80% data train dan 20% data test.

In [ ]:
```
# Constant variables
# Random state = memastikan bahwa setiap randomness bisa direproduksi kembali
randState = 10
# Ukuran test dan train
testSize = 0.2
trainSize = 0.8
```

In [ ]:
```
# Dataset cancer
ctrain, ctest = model_selection.train_test_split(dtcancer, test_size=testSize,


# Dataset tennis
tntrain, tntest = model_selection.train_test_split(dttennis, test_size=testSize
```

Kemudian untuk memisahkan atribut yang akan digunakan sebagai parameter dan atribut prediksi, dataset awal akan dibagi menjadi 4 bagian:

- **x train:** feature yang akan digunakan untuk train model
- **y train:** ground truth (hasil asli)
- **x test:** feature yang digunakan untuk diuji oleh model
- **y test:** data yang digunakan untuk memvalidasi hasil prediksi

Dataset Cancer

In [ ]:
```
y = dtcancer['res']
x = dtcancer.drop('res', axis=1)

x_ctrain, x_ctest, y_ctrain, y_ctest = model_selection.train_test_split(x, y,
```

Dataset Tennis

In [ ]:
```
y = dttennis['play']
x = dttennis.drop('play', axis=1)

x_tntrain, x_tntest, y_tntrain, y_tntest = model_selection.train_test_split(x,
```

Pembelajaran Mesin

Pada bagian ini akan ada 2 hal yang dibahas, yaitu akurasi model dan nilai f1 dari model. Berikut adalah definisi dari masing-masing skor:

Akurasi model:

> Perbandingan antara jumlah data yang benar diprediksi dengan total jumlah data secara keseluruhan. Akurasi memiliki nilai maksimal 1 dan nilai minimal 0.

Nilai F1:

> Sebuah pengukuran akurasi model terhadap dataset dan biasanya digunakan dalam klasifikasi biner. Nilai F1 juga merupakan sebuah cara untuk menggabungkan *precision* dan *recall* dari sebuah model. Recall dan precision akan berbanding lurus dengan nilai F1.

Decision Tree Classifier

Decision Tree Classifier adalah sebuah metode supervised learning yang digunakan untuk memberikan model yang akan membagi data bergantung dari parameter tertentu.

Tahap pembuatan DTC adalah:

1. Pilih atribut terbaik menggunakan ASM (Attribute Selection Measures)
2. Buat atribut tersebut menjadi decision node dan pecah dataset menjadi lebih kecil
3. Ulangi secara rekursif hingga kondisi berikut terpenuhi:
    - semua atribut sudah habis
    - tidak ada instansi lain lagi
    - semua tuple terdapat dalam nilai atribut yang sama

Banyak metode ASM yang bisa digunakan, tetapi ASM yang akan dipakai disini adalah *Entropy*.

DTC - Breast Cancer Dataset

Terdapat 30 parameter yang digunakan antara lain:

- mean radius
- mean texture
- mean perimeter
- mean area
- mean smoothness
- mean compactness
- mean concavity
- mean concave points
- mean symmetry
- mean fractal dimension
- radius error
- texture error
- perimeter error
- area error
- smoothness error
- compactness error
- concavity error

- concave points error
- symmetry error
- fractal dimension error
- worst radius
- worst texture
- worst perimeter
- worst area
- worst smoothness
- worst compactness
- worst concavity
- worst concave points
- worst symmetry
- worst fractal dimension

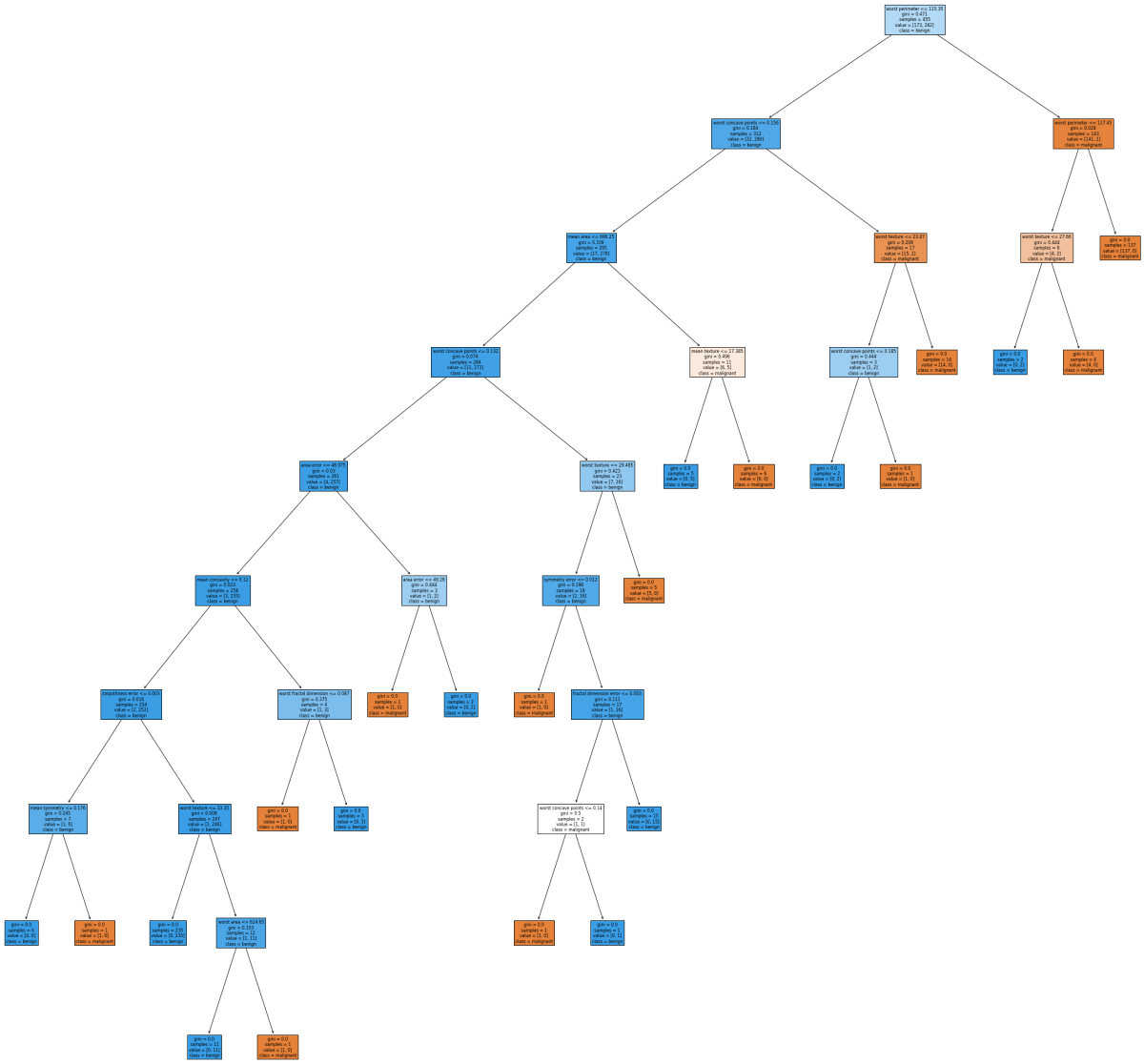
Kemudian dilakukan prediksi dengan criterion berupa entropy pada kode di bawah ini dan didapatkan akurasi beserta F1 dari hasil training model untuk memprediksi kanker.

In [ ]:
```
clf_c = DecisionTreeClassifier()
clf_c = clf_c.fit(x_ctrain,y_ctrain)
y_dtc_cpred = clf_c.predict(x_ctest)
c_dtc_acc = accuracy_score(y_ctest,y_dtc_cpred)
c_dtc_f1 = f1_score(y_ctest,y_dtc_cpred)
printScore(c_dtc_acc, c_dtc_f1)
```

```
Accuracy   : 0.9210526315789473
F1 Score   : 0.9379310344827586
```

Berikut di bawah ini adalah ilustrasi decision tree yang dihasilkan dari pembelajaran mesin Decision Tree Classifier menggunakan library matplotlib. warna biru pada leaf node menandakan **benign**, sedangkan warna orange pada leaf node menandakan **malignant**

In [ ]:
```
# Export text (textual)
# model = tree.export_text(clf_c)
# print(model)
fig = plt.figure(figsize=(40,40))
_ = tree.plot_tree(clf_c,feature_names=cancer.feature_names, class_names=cancer
```



Jalankan kode di bawah ini untuk menyimpan gambar figure:

In [ ]:
```
fig.savefig("../img/cancer_dtc.png")
```

DTC - Tennis Dataset

Terdapat 4 parameter yang digunakan, antara lain:

- outlook
- temp
- humidity
- wind

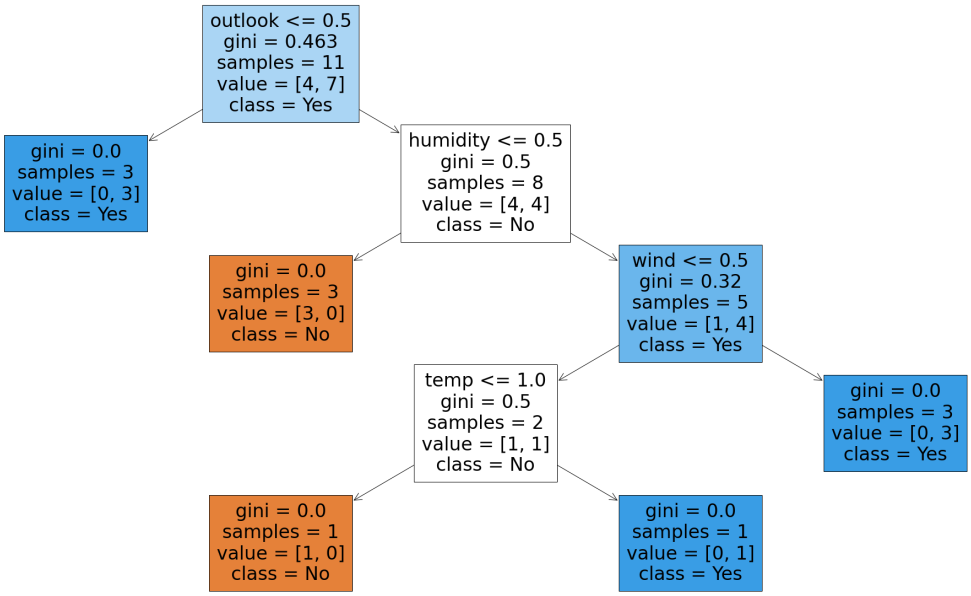
In [ ]:
```
clf_t = DecisionTreeClassifier()
clf_t = clf_t.fit(x_tntrain, y_tntrain)
y_dtc_tnpred = clf_t.predict(x_tntest)
t_dtc_acc = accuracy_score(y_tntest,y_dtc_tnpred)
```

```
t_dtc_f1 = f1_score(y_tntest,y_dtc_tnpred)
printScore(t_dtc_acc, t_dtc_f1)
```

```
Accuracy   : 0.6666666666666666
F1 Score   : 0.6666666666666666
```

Berikut di bawah ini adalah ilustrasi decision tree yang dihasilkan dari pembelajaran mesin Decision Tree Classifier menggunakan library matplotlib pada dateset tennis. warna biru pada leaf node menandakan **No**, sedangkan warna orange pada leaf node menandakan **Yes**

In [ ]:
```
fig = plt.figure(figsize=(40,20))
_ = tree.plot_tree(clf_t,feature_names=dttennis.columns[:-1], class_names=['No
```



In [ ]:
```
fig.savefig("../img/tennis_dtc.png")
```

DTC - Analisis Prediksi

In [ ]:
```
printSummary('DTC',c_dtc_acc, c_dtc_f1, t_dtc_acc, t_dtc_f1)
```

```
================================
Cancer Dataset:
Accuracy   : 0.9210526315789473
F1 Score   : 0.9379310344827586
================================
Tennis Dataset
Accuracy   : 0.6666666666666666
F1 Score   : 0.6666666666666666
================================
```

Dataset kanker:

- Dataset kanker memiliki akurasi yang cukup tinggi dengan skor akurasi >0.9 yang artinya memiliki akurasi mendekati 90%.
- Dataset kanker memiliki nilai F1 yang cukup tinggi dengan skor akurasi >0.9 yang artinya data cenderung masih seimbang sehingga false negative ataupun ataupun false positive bisa dihindari dengan menggunakan dataset saat ini untuk training.

Dataset tennis:

- Dataset tennis memiliki akurasi yang cukup rendah dengan skor akurasi ~0.66. Hal ini juga bisa disebabkan karena sedikitnya dataset yang dimiliki sehingga kesalahan kecil akan sangat berdampak pada akurasi.
- Dataset tennis memiliki nilai F1 yang cukup rendah juga karena jumlah data pada dataset yang relatif sedikit sehingga perbedaan 1 record akan berpengaruh besar terhadap *outcome* dari hasil training.
- Hal ini banyak dipengaruhi oleh jumlah dataset yang digunakan saat ini, yaitu hanya 11 data dengan 4 parameter yang harus di fine-tune. 3 data lainnya digunakan untuk menguji model. Meskipun umumnya decision tree bisa ditrain menggunakan training set yang kecil, tetapi jumlah yang sangat kecil akan membuat model menjadi tidak akurat.

ID3 Estimator

ID3 Algorithm - Iterative Dichotomiser 3

> Algoritma klasitikasi yang menggunakan pendekatan *greedy algorithm* untuk membentuk *decision tree* dengan mencari atribut yang memiliki *information gain* terbesar (atau entropi terkecil)

Tahapan dalam algoritma ID3:

1. Kalkulasi entropi dari dataset
2. Untuk setiap atribut/*feature*:
   A. Hitung entropi dari semua nilai kategorikalnya
   B. Hitung *information gain* dari featurenya
3. Cari *feature* dengan *information gain* terbesar
4. Ulangi hingga habis dan terbentuk decision tree yang diinginkan

Untuk ID3 seharusnya bisa menggunakan sklearn DecisionTreeClassifier dengan criterion='Entropy' dan akan menghasilkan hasil yang sama dengan menggunakan library pada svaante/decision-tree-id3. Akan tetapi dengan graphviz

ID3 - Breast Cancer Dataset

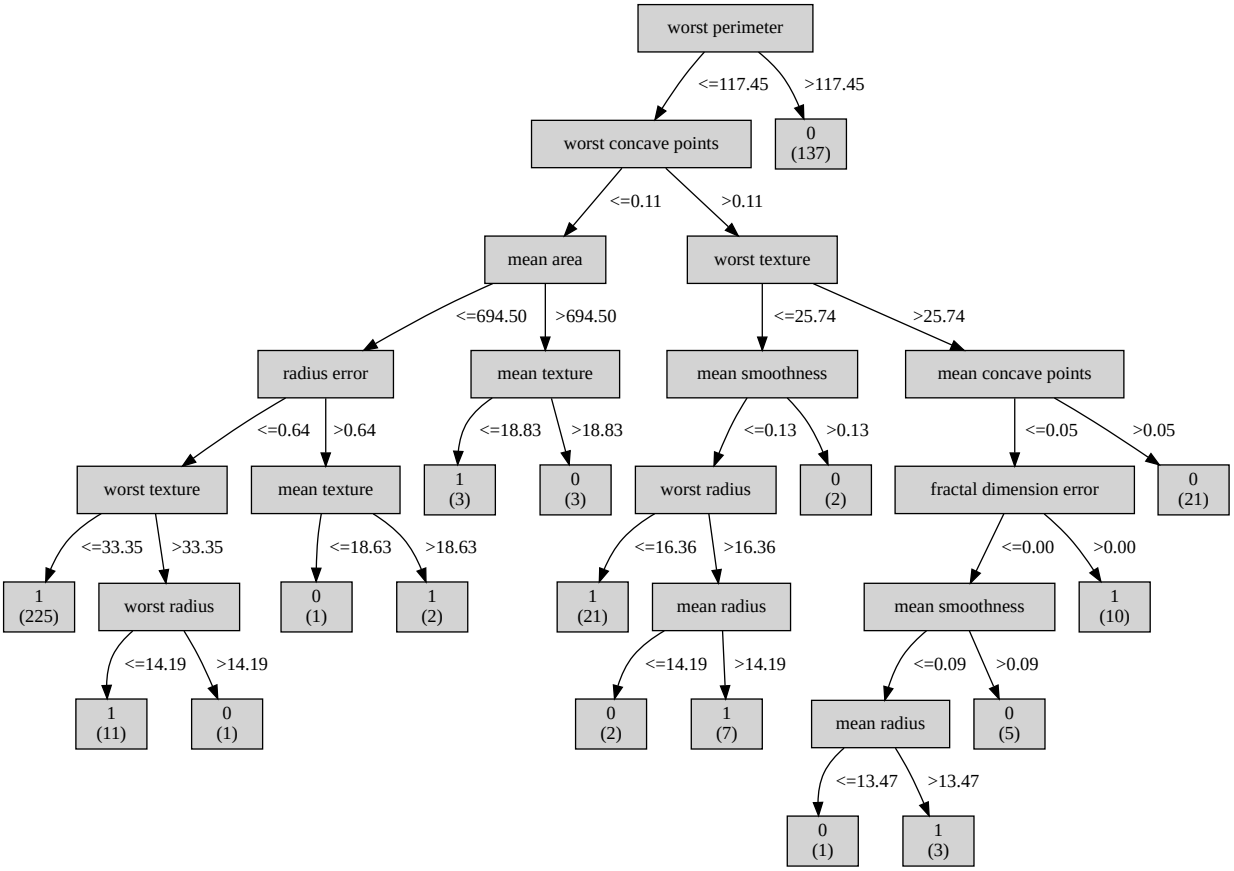
In [ ]:
```
estimator = Id3Estimator()
estimator = estimator.fit(x_ctrain, y_ctrain)
y_id3_cpred = estimator.predict(x_ctest)
c_id3_acc = accuracy_score(y_ctest, y_id3_cpred)
c_id3_f1 = f1_score(y_ctest, y_id3_cpred)
printScore(c_id3_acc, c_id3_f1)
```

```
Accuracy   : 0.9122807017543859
F1 Score   : 0.9295774647887323
```

Berikut di bawah ini adalah ilustrasi decision tree yang dihasilkan dari pembelajaran mesin IM3 menggunakan library graphviz.

In [ ]:
```
tree = export_graphviz(estimator.tree_, '../graph/c_id3.dot', cancer.feature_na
graphviz.Source.from_file('../graph/c_id3.dot')
```

Out[ ]:



ID3 - Tennis Dataset

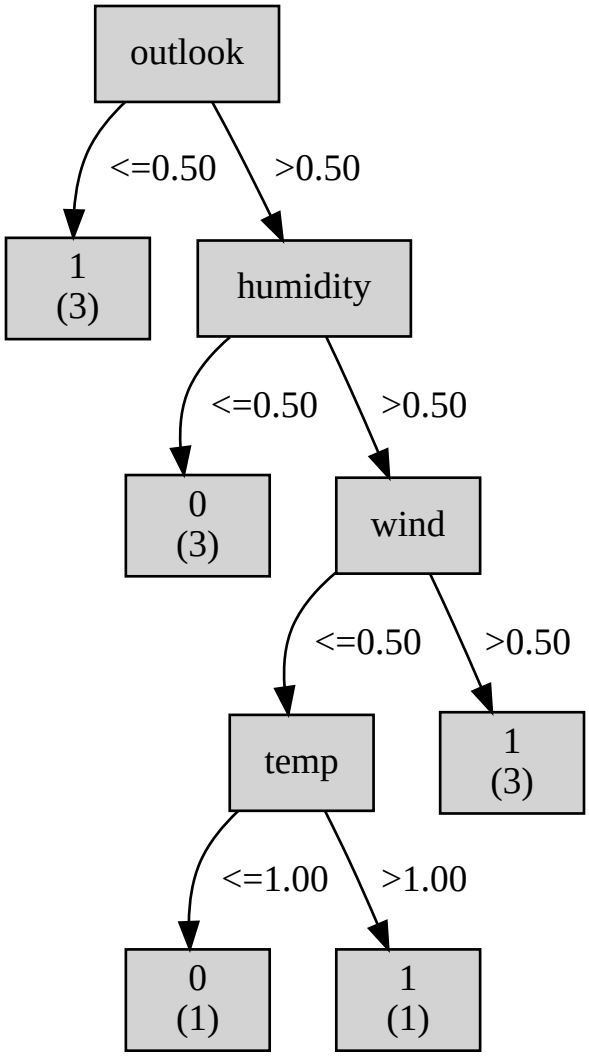
In [ ]:
```
estimator = Id3Estimator()
estimator = estimator.fit(x_tntrain, y_tntrain)
y_id3_tnpred = estimator.predict(x_tntest)
t_id3_acc = accuracy_score(y_tntest, y_id3_tnpred)
t_id3_f1 = f1_score(y_tntest, y_id3_tnpred)
printScore(t_id3_acc, t_id3_f1)
```

```
Accuracy   : 0.6666666666666666
F1 Score   : 0.6666666666666666
```

Berikut di bawah ini adalah ilustrasi decision tree yang dihasilkan dari pembelajaran mesin IM3 menggunakan library graphviz.

In [ ]:
```
tree = export_graphviz(estimator.tree_, '../graph/t_id3.dot', dttennis.columns
graphviz.Source.from_file('../graph/t_id3.dot')
```

Out[ ]:



ID3 - Analisis Prediksi

In [ ]:
```
printSummary('ID3',c_id3_acc, c_id3_f1, t_id3_acc, t_id3_f1)
```

```
==============================
Cancer Dataset:
Accuracy   : 0.9122807017543859
F1 Score   : 0.9295774647887323
==============================
Tennis Dataset
Accuracy   : 0.6666666666666666
F1 Score   : 0.6666666666666666
==============================
```

Dataset kanker:

- Dataset kanker memiliki akurasi yang cukup tinggi dengan skor akurasi sekitar 0.9 yang artinya model yang dibentuk berhasil memprediksi ~90% dari dataset yang diberikan dengan benar.
- Dataset kanker memiliki nilai F1 yang cukup tinggi dengan skor akurasi sekitar 0.9 yang artinya data cenderung masih seimbang sehingga false negative ataupun ataupun false positive bisa dihindari dengan menggunakan dataset saat ini untuk training.

Dataset tennis:

- Dataset tennis memiliki akurasi yang cukup rendah dengan skor akurasi ~0.66. Ini artinya hanya 66% data saja yang benar diprediksi.
- Dataset tennis memiliki nilai F1 yang cukup rendah juga karena jumlah data pada dataset yang relatif sedikit sehingga perbedaan 1 record akan berpengaruh besar terhadap *outcome* dari hasil training.

KMeans

> Algoritma unsupervised learning yang digunakan untuk memberikan model yang mengelompokan nilai (data point) yang mirip atau serupa ke dalam kelompok yang sama.

Tahapan dalam KMeans:

1. Inisiasi jumlah kluster secara random.
2. Tentukan posisi centroid secara random.
3. Hitung jarak antara data point ke centroid.
4. Assign setiap data ke centroid terdekat.
5. Hitung ulang centroid lainnya berdasarkan label data sebelumnya.
6. Assign kembali setiap data ke centroid terdekat.

Dibuat model cluster KMeans dengan jumlah cluster sama dengan 2 dan dilakukan pengacakan data training pada saat dilakukan training.

In [ ]:
```
# create model
c_kmeans = KMeans(n_clusters=2, random_state=42).fit(x_ctrain)
c_kmeanspredict = c_kmeans.predict(x_ctest)

tn_kmeans = KMeans(n_clusters=2, random_state=42).fit(x_tntrain)
tn_kmeanspredict = tn_kmeans.predict(x_tntest)
```

KMeans - Breast Cancer Dataset

In [ ]:
```
# accuracy and f1 score
c_kmeansAccScore = accuracy_score(y_ctest, c_kmeanspredict)
c_kmeansF1Score = f1_score(y_ctest, c_kmeanspredict, average='weighted')
```

In [ ]:
```
printScore(c_kmeansAccScore, c_kmeansF1Score)
```

```
Accuracy   : 0.16666666666666666
F1 Score   : 0.1075785000393484
```

Berikut merupakan hasil visualisasi dari model Breast Cancer

In [ ]:
```
# visualize Breast Cancer model
```

KMeans - Tennis Dataset

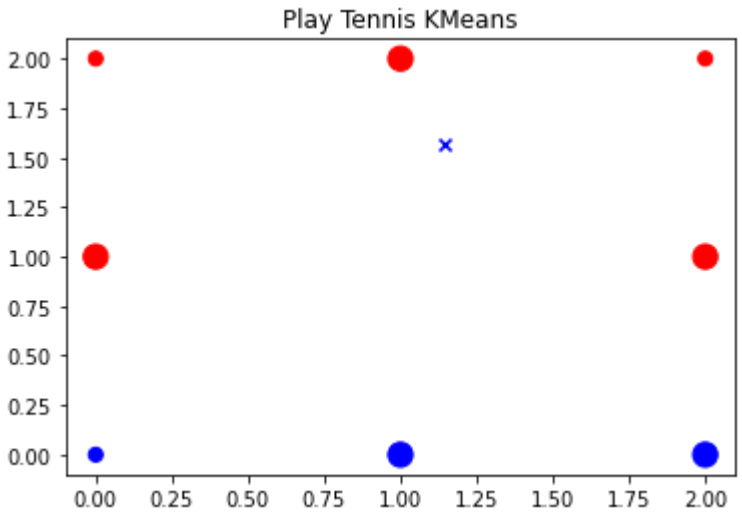
In [ ]:
```
tn_kmeansAccScore = accuracy_score(y_tntest, tn_kmeanspredict)
tn_kmeansF1Score = f1_score(y_tntest, tn_kmeanspredict)
```

In [ ]:
```
printScore(tn_kmeansAccScore, tn_kmeansF1Score)
```

```
Accuracy   : 0.6666666666666666
F1 Score   : 0.6666666666666666
```

Berikut merupakan hasil visualisasi dari model Tennis

In [ ]:
```
# visualize Tennis model
tn_centroids = np.array(tn_kmeans.cluster_centers_)

tn_x_cen = [i[0] for i in tn_centroids]
tn_y_cen = [i[1] for i in tn_centroids]
arr_color = ['#ff0000', '#0000ff']

# copy a dataset
x_new_tntrain = x_tntrain.copy()
x_new_tntrain['cluster'] = tn_kmeans.labels_

x_new_tntrain['x_cen'] = x_new_tntrain.cluster.map({0:tn_x_cen[0], 1:tn_x_cen[1
x_new_tntrain['y_cen'] = x_new_tntrain.cluster.map({0:tn_y_cen[0], 1:tn_y_cen[1
x_new_tntrain['color'] = x_new_tntrain.cluster.map({0:arr_color[0], 1:arr_color

plt.title('Play Tennis KMeans')
plt.scatter(x_new_tntrain.outlook, x_new_tntrain.temp, c=x_new_tntrain['color']

plt.scatter(tn_centroids[:,0], tn_centroids[:,1], marker='x', color='b')

plt.show()
```



KMeans - Analisis Prediksi

In [ ]:
```
printSummary('KMeans',c_kmeansAccScore, c_kmeansF1Score, tn_kmeansAccScore, tn_
```

```
==============================
Cancer Dataset:
Accuracy   : 0.16666666666666666
F1 Score   : 0.1075785000393484
==============================
Tennis Dataset
Accuracy   : 0.6666666666666666
F1 Score   : 0.6666666666666666
==============================
```

Dataset kanker:

- Dataset kanker memiliki akurasi yang **cukup rendah** dengan skor akurasi **0.19**. Hal ini dapat disebabkan karena data cancer memiliki outliers dan KMeans sangat sensitif dengan data outliers.
- Dataset kanker memiliki nilai F1 yang **cukup rendah** dengan skor akurasi **0.12** yang artinya salah satu dari recall atau precision memiliki rentang nilai yang jauh.

Dataset tennis:

- Dataset tennis memiliki akurasi yang **cukup tinggi** dengan skor akurasi **1** yang artinya akurasi sangat tinggi 100%.
- Dataset tennis memiliki nilai F1 yang **cukup tinggi** dengan skor akurasi **1** yang artinya recall dan precision memiliki nilai yang sama antara bernilai 0 atau 1.

> KMeans tidak cocok digunakan untuk data yang memiliki outliers.

Logistic Regression

> Logistic Regression adalah teknik analisis regresi yang digunakan untuk melakukan analisis prediksi terhadap variabel dependen yang biner (dikotomi) berdasarkan konsep probabilitas.

Probabilitas dari kelas: $$

p = P(y=1|x,b) = \dfrac{1}{1+e^{-b^Tx}}

$$

Logistic Regression - Breast Cancer Dataset

In [ ]:
```
logreg = LogisticRegression(max_iter=10000)
logreg = logreg.fit(x_ctrain,y_ctrain)
y_log_cpred = logreg.predict(x_ctest)
c_logreg_acc = accuracy_score(y_ctest, y_log_cpred)
c_logreg_f1 = f1_score(y_ctest,y_log_cpred)
```

In [ ]:
```
printScore(c_logreg_acc, c_logreg_f1)
```

```
Accuracy   : 0.956140350877193
F1 Score   : 0.9659863945578231
```

Logistic Regression - Tennis Dataset

In [ ]:
```
logreg = LogisticRegression(max_iter=10000)
logreg = logreg.fit(x_tntrain,y_tntrain)
y_log_tnpred = logreg.predict(x_tntest)
t_logreg_acc = accuracy_score(y_tntest, y_log_tnpred)
t_logreg_f1 = f1_score(y_tntest,y_log_tnpred)
```

In [ ]:
```
printScore(t_logreg_acc, t_logreg_f1)
```

```
Accuracy   : 0.6666666666666666
F1 Score   : 0.8
```

Logistic Regression - Analisis Prediksi

In [ ]:
```
printSummary('Logistc Regression',c_logreg_acc, c_logreg_f1, t_logreg_acc, t_lo
```

```
================================
Cancer Dataset:
Accuracy   : 0.956140350877193
F1 Score   : 0.9659863945578231
================================
Tennis Dataset
Accuracy   : 0.6666666666666666
F1 Score   : 0.8
================================
```

Dataset kanker:

- Dataset kanker memiliki akurasi yang cukup tinggi dengan skor akurasi sekitar 0.9 yang artinya model yang dibentuk berhasil memprediksi ~90% dari dataset yang diberikan dengan benar.
- Dataset kanker memiliki nilai F1 yang cukup tinggi dengan skor akurasi sekitar 0.9 yang artinya data cenderung masih seimbang sehingga false negative ataupun ataupun false positive bisa dihindari dengan menggunakan dataset saat ini untuk training.

Dataset tennis:

- Dataset tennis memiliki akurasi yang cukup rendah dengan skor akurasi ~0.66. Ini artinya hanya 66% data saja yang benar diprediksi.
- Dataset tennis memiliki nilai F1 yang cukup rendah juga karena jumlah data pada dataset yang relatif sedikit sehingga perbedaan 1 record akan berpengaruh besar terhadap *outcome* dari hasil training. Akan tetapi skor F1 juga akan terpengaruh terhadap variasi data yang terdapat pada training set.

Neural Network

> Algoritma yang memanfaatkan pendekatan bentuk *net* (jaring) untuk memberikan model dengan membuat setiap *layer* (*input layer*, *hidden layer* (dapat lebih dari

> 1), dan *output layer*) yang dapat menerjemahkan *input* data menjadi sebuah *output* yang diinginkan.

Tahapan dalam Neural Network:

Forward Propagation

1. Lakukan perhitungan total beban / *weight* dari setiap node input layer lalu simpan dalam sebuah variabel.
2. Lakukan penghitungan fungsi aktivasi untuk setiap node dari layer saat ini.
3. Lakukan penghitungan kembali (seperti langkah 1) dengan layer saat ini sampai sebelum fungsi aktivasi pada layer output.

Back Propagation

1. Lakukan perhitungan error yang didapatkan dari forward propagation.
2. Mengalikan error yang dihitung tadi dengan menggunakan turunan untuk mencari nilai delta.
3. Kalikan delta dengan layer input lalu jumlahkan semua sampai seluruh node telah selesai dihitung.

Dibuat model Neural Network klasifikasi dengan dilakukan maksimum iterasi sampai dengan 2000 dan pengambilan data training secara acak.

In [ ]:
```
# create model
clf = MLPClassifier(random_state=1, max_iter=2000, )

c_neural = clf.fit(x_ctrain, y_ctrain)
c_neuralpredict = c_neural.predict(x_ctest)

tn_neural = clf.fit(x_tntrain, y_tntrain)
tn_neuralpredict = tn_neural.predict(x_tntest)
```

Neural Network - Breast Cancer Dataset

In [ ]:
```
# accuracy and f1 score
c_neuralAccScore = accuracy_score(y_ctest, c_neuralpredict)
c_neuralF1Score = f1_score(y_ctest, c_neuralpredict)
```

In [ ]:
```
printScore(c_neuralAccScore, c_neuralF1Score)
```

```
Accuracy   : 0.8859649122807017
F1 Score   : 0.9182389937106918
```

Berikut merupakan hasil visualisasi dari model Breast Cancer

In [ ]:
```
# visualize Breast Cancer model
```

Neural Network - Tennis Dataset

In [ ]:
```
# accuracy and f1 score
tn_neuralAccScore = accuracy_score(y_tntest, tn_neuralpredict)
tn_neuralF1Score = f1_score(y_tntest, tn_neuralpredict)
```

In [ ]:
```
printScore(tn_neuralAccScore, tn_neuralF1Score)
```

```
Accuracy   : 1.0
F1 Score   : 1.0
```

Berikut merupakan hasil visualisasi dari model Tennis

In [ ]:
```
# visualize Tennis model
```

Neural Network - Analisis Prediksi

In [ ]:
```
printSummary('Neural Network', c_neuralAccScore, c_neuralF1Score, tn_neuralAccS
```

```
==============================
Cancer Dataset:
Accuracy   : 0.8859649122807017
F1 Score   : 0.9182389937106918
==============================
Tennis Dataset
Accuracy   : 1.0
F1 Score   : 1.0
==============================
```

Dataset kanker:

- Dataset kanker memiliki akurasi yang **sangat tinggi** dengan skor akurasi **0.95** yang artinya data model prediksi benar tinggi.
- Dataset kanker memiliki nilai F1 yang **sangat tinggi** dengan skor akurasi **0.96** yang artinya NN dapat melakukan training dengan meminimalkan false positive dan false negatif.

Dataset tennis:

- Dataset tennis memiliki akurasi yang **sangat tinggi** dengan skor akurasi **1** yang artinya data model prediksi benar tinggi.
- Dataset tennis memiliki nilai F1 yang **sangat tinggi** dengan skor akurasi **1** yang artinya NN dapat melakukan training dengan meminimalkan false positive dan false negatif.

> Neural Network cocok digunakan untuk data klasifikasi atau pun regresi

SVM - Support Vector Machine

> Algoritma supervised learning yang dapat digunakan untuk klasifikasi ataupun regresi dengan memanfaatkan param **kernel** atau sebuah fungsi yang dapat mengubah non-linearly separable data menjadi linearly separable data dengan menambahkan dimensi ruang.

Tahap dalam SVM:

Linearly Separable

1. Menghitung jarak dari inputan data menjadi sebuah hyperplane (2D menjadi sebuah garis lurus) yang dapat memisahkan dua atau lebih label data.

Non-Linearly Separable

1. Menambahkan satu dimensi baru ke dalam ruang (x, y, and z dimension).
2. Melakukan penghitungan jarak dari dimensi baru tadi.
3. Lakukan assignment label dari hasil hyperplane pada dimensi baru.

Dibuat model SVM dengan regulasi parameter sama dengan 1 dan tipe kernel linear.

In [ ]:
```
# create model
svc = SVC(C=1, kernel='linear')

c_svc = svc.fit(x_ctrain, y_ctrain)
c_svcpredict = c_svc.predict(x_ctest)

tn_svc = svc.fit(x_tntrain, y_tntrain)
tn_svcpredict = tn_svc.predict(x_tntest)
```

SVM - Breast Cancer Dataset

In [ ]:
```
# accuracy and f1 score
c_svcAccScore = accuracy_score(y_ctest, c_svcpredict)
c_svcF1Score = f1_score(y_ctest, c_svcpredict)
```

In [ ]:
```
printScore(c_svcAccScore, c_svcF1Score)
```

```
Accuracy   : 0.9473684210526315
F1 Score   : 0.9594594594594594
```

Berikut merupakan hasil visualisasi dari model Breast Cancer

In [ ]:
```
# visualize Breast Cancer model
```

SVM - Tennis Dataset

In [ ]:
```
tn_svcvAccScore = accuracy_score(y_tntest, tn_svcpredict)
tn_svcF1Score = f1_score(y_tntest, tn_svcpredict)
```

In [ ]:
```
printScore(tn_svcvAccScore, tn_svcF1Score)
```

```
Accuracy   : 0.6666666666666666
F1 Score   : 0.8
```

Berikut merupakan hasil visualisasi dari model Tennis

In [ ]:
```
# visualize Tennis model
```

SVM - Analisis Prediksi

In [ ]:
```
printSummary('SVM',c_svcAccScore, c_svcF1Score, tn_svcvAccScore, tn_svcF1Score
```

```
==============================
Cancer Dataset:
Accuracy   : 0.9473684210526315
F1 Score   : 0.9594594594594594
==============================
Tennis Dataset
Accuracy   : 0.6666666666666666
F1 Score   : 0.8
==============================
```

Dataset kanker:

- Dataset kanker memiliki akurasi yang **sangat tinggi** dengan skor akurasi **>0.95** yang artinya hasil prediksi dari model yang dibuat memiliki kesamaan dengan model data.
- Dataset kanker memiliki nilai F1 yang **sangat tinggi** dengan skor akurasi **>0.96** yang artinya model dapat meminimalkan kesalahan training data false negatif atau false positif.

Dataset tennis:

- Dataset tennis memiliki akurasi yang **cukup tinggi** dengan skor akurasi **>~0.66** yang artinya hasil prediksi model memiliki hasil prediksi yang salah.
- Dataset tennis memiliki nilai F1 yang **tinggi** dengan skor akurasi **xxx** yang artinya model yang dibuat masih mendeteksi false negatif atau false positif dalam dilakukannya training.

> SVM cocok digunakan untuk algoritma klasifikasi meskipun SVM juga dapat digunakan untuk membuat model regresi namun tidak terlalu akurat.

Analisis dan Kesimpulan

Breast Cancer - Accuracy Score

In [ ]:
```
cancer_recap = recap.loc[(recap['Dataset']=='Cancer') & (recap['Tipe']=='Accura
cancer_recap.sort_values(by=['Nilai'], ascending=False)
```

Out[ ]:

	Pembelajaran	Dataset	Tipe	Nilai
12	Logistc Regression	Cancer	Accuracy Score	0.956140
20	SVM	Cancer	Accuracy Score	0.947368
0	DTC	Cancer	Accuracy Score	0.921053
4	ID3	Cancer	Accuracy Score	0.912281
16	Neural Network	Cancer	Accuracy Score	0.885965
8	KMeans	Cancer	Accuracy Score	0.166667

Breast Cancer - F1 Score

In [ ]:
```
cancer_recap = recap.loc[(recap['Dataset']=='Cancer') & (recap['Tipe']=='F1 Sc
cancer_recap.sort_values(by=['Nilai'], ascending=False)
```

Out[ ]:

	Pembelajaran	Dataset	Tipe	Nilai
13	Logistc Regression	Cancer	F1 Score	0.965986
21	SVM	Cancer	F1 Score	0.959459
1	DTC	Cancer	F1 Score	0.937931
5	ID3	Cancer	F1 Score	0.929577
17	Neural Network	Cancer	F1 Score	0.918239
9	KMeans	Cancer	F1 Score	0.107579

Tennis - Accuracy Score

In [ ]:
```
tennis_recap = recap.loc[(recap['Dataset']=='Tennis') & (recap['Tipe']=='Accura
tennis_recap.sort_values(by=['Nilai'], ascending=False)
```

Out[ ]:

	Pembelajaran	Dataset	Tipe	Nilai
18	Neural Network	Tennis	Accuracy Score	1.000000
2	DTC	Tennis	Accuracy Score	0.666667
6	ID3	Tennis	Accuracy Score	0.666667
10	KMeans	Tennis	Accuracy Score	0.666667
14	Logistc Regression	Tennis	Accuracy Score	0.666667
22	SVM	Tennis	Accuracy Score	0.666667

Tennis - F1 Score

In [ ]:
```
tennis_recap = recap.loc[(recap['Dataset']=='Tennis') & (recap['Tipe']=='F1 Sc
tennis_recap.sort_values(by=['Nilai'], ascending=False)
```

Out[ ]:

	Pembelajaran	Dataset	Tipe	Nilai
19	Neural Network	Tennis	F1 Score	1.000000
15	Logistc Regression	Tennis	F1 Score	0.800000
23	SVM	Tennis	F1 Score	0.800000
3	DTC	Tennis	F1 Score	0.666667
7	ID3	Tennis	F1 Score	0.666667
11	KMeans	Tennis	F1 Score	0.666667

Pembahasan

Berdasarkan hasil rekap nilai skor akurasi dan F1 dari kedua tabel di atas, didapati bahwa DTC dan ID3 memiliki nilai tertinggi dalam akurasi sehingga dapat disimpulkan (dengan informasi nilai akurasi dan F1) bahwa untuk dataset breast cancer, **decision tree classifier dan ID3 adalah teknik pembelajaran mesin yang lebih cocok** dibandingkan Logistic Regression, SVM, Neural Network, dan KMeans.

Hal ini dipengaruhi oleh berbagai faktor

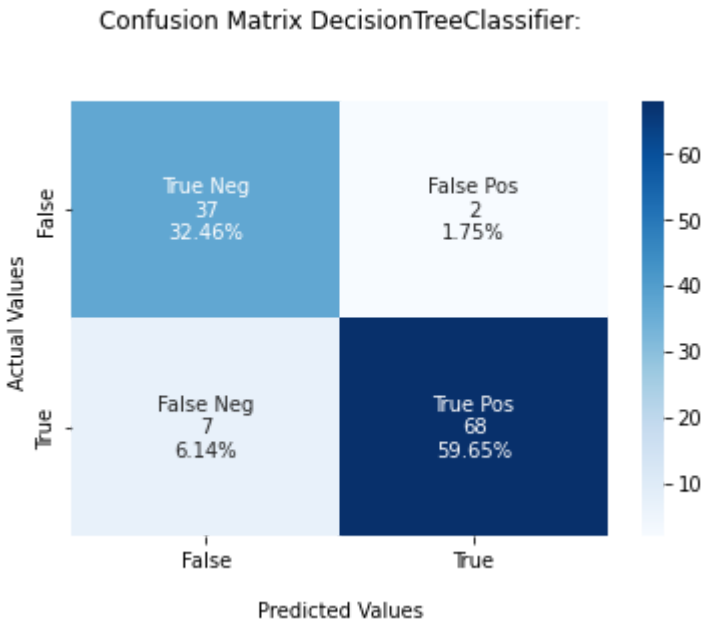
- decision tree bekerja (classifier maupun ID3) dengan membagi 2 kelas yang berbeda dengan membagi-bagi dimensi secara berkali-kali sehingga lebih bisa mendekati pembagian antara kelas 1 dengan kelas lainnya dibandingkan dengan logistic regression yang terbatas hanya 1 garis saja.
- SVM tidak buruk dalam menangani klasifikasi, akan tetapi untuk klasifikasi umumnya decision tree akan lebih baik dalam menangani data kategorikal khususnya pada dataset ini. Pada dataset tennis, SVM tampak lebih baik dalam nilai F1 karena F1 mengoptimalkan klasifikasi.
- Neural Network umumnya lebih cocok untuk dataset berukuran besar, pada kasus ini dapat dilihat perbedaan drastis antara dataset besar (breast cancer) dengan dataset kecil (tennis) dimana Neural Network tampaknya mendominasi dataset kecil, akan tetapi model neural network ini hanya akurat pada spesifik dataset yang digunakan saat ini saja dan tidak akan akurat lagi jika diberikan dataset test baru.
- KMeans adalah unsupervised learning dan tidak cocok untuk melakukan KMeans terhadap data yang berlabel seperti dataset tennis dan breast cancer.

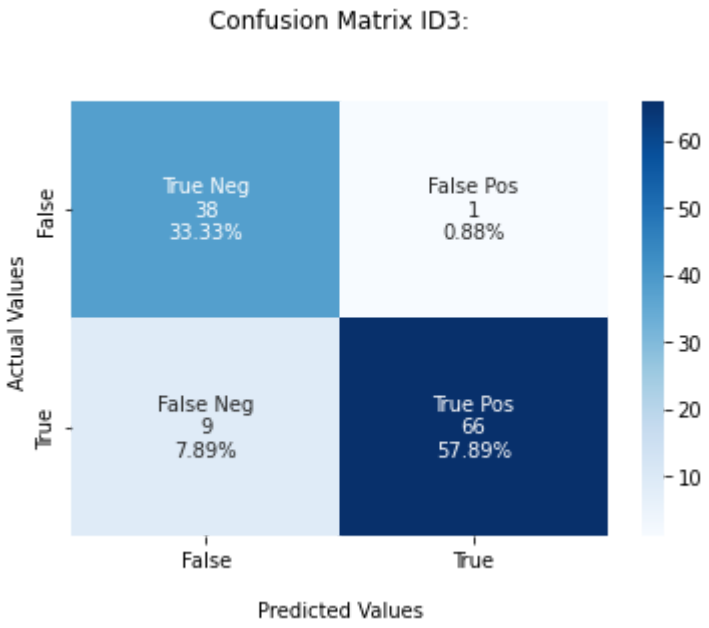
ID3 dan DecisionTree pada tabel skor ini terlihat sama persis, akan tetapi untuk pembelajaran mesin pada dataset yang lebih banyak data dan lebih banyak variabel, ID3 akan sangat menghemat waktu dalam penyusunan decision treenya karena menggunakan teknik *Information Gain* untuk menentukan atribut mana yang sebaiknya dijadikan node lebih awal berdasarkan *Gain* yang bisa didapat dari atribut tersebut. Pada dataset lain, DecisionTreeClassifier dan ID3 bisa jadi akan menghasilkan hasil yang berbeda.

Selain itu dapat dilihat juga bahwa model yang dihasilkan oleh DecisionTreeClassifier dan ID3 berbeda dari hasil prediksinya:

In [ ]:
```
def printMatrix(truth, pred):
    print(confusion_matrix(truth, pred))
```

In [ ]:
```
drawConfusionMatrix('Confusion Matrix DecisionTreeClassifier:',y_dtc_cpred, y_c
drawConfusionMatrix('Confusion Matrix ID3:',y_id3_cpred, y_ctest)
```





Kesimpulan

- Dataset breast cancer lebih baik untuk membandingkan kecocokan dari masing-masing teknik pembelajaran mesin dengan data kategorikal karena memiliki jumlah data yang lebih banyak.
- Dataset tennis seharusnya digunakan untuk membandingkan kecocokan teknik pembelajaran mesin terhadap dataset yang kecil, akan tetapi dataset tennis terlalu kecil untuk dapat mengevaluasi kecocokan dari masing-masing teknik.
- DecisionTreeClassifier dan ID3 merupakan teknik pembelajaran mesin yang paling cocok untuk melakukan prediksi terhadap Breast Cancer dan Tennis