Decision Tree Classifier

In [22]:
```
import pandas as pd

data_table= {
    'Cerah': ['Yes', 'Yes', 'No', 'No', 'Yes', 'Yes', 'No' ],
    'Dingin': ['Yes', 'No', 'Yes', 'Yes', 'Yes', 'No', 'No'] ,
    'Kelembaban': [7, 12, 18, 35, 38, 50, 83],
    'Hujan': ['No', 'No', 'Yes', 'Yes', 'Yes', 'No', 'No']
  }

df = pd.DataFrame(data_table)
df
```

Out[22]:

	Cerah	Dingin	Kelembaban	Hujan
0	Yes	Yes	7	No
1	Yes	No	12	No
2	No	Yes	18	Yes
3	No	Yes	35	Yes
4	Yes	Yes	38	Yes
5	Yes	No	50	No
6	No	No	83	No

In [23]:
```
X = df.drop("Hujan",axis=1)
y = df.Hujan

X.head()
```

Out[23]:

	Cerah	Dingin	Kelembaban
0	Yes	Yes	7
1	Yes	No	12
2	No	Yes	18
3	No	Yes	35
4	Yes	Yes	38

In [24]:
```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
X['Cerah'] = le.fit_transform(X['Cerah'])
X['Dingin'] = le.fit_transform(X['Dingin'])
X
```

Out[24]:

	Cerah	Dingin	Kelembaban
0	1	1	7
1	1	0	12

	Cerah	Dingin	Kelembaban
2	0	1	18
3	0	1	35
4	1	1	38
5	1	0	50
6	0	0	83

In [25]:
```
le = LabelEncoder()
y = le.fit_transform(y)
y
```

Out[25]:
```
array([0, 0, 1, 1, 1, 0, 0])
```

In [26]:
```
from sklearn.tree import DecisionTreeClassifier

classifier = DecisionTreeClassifier()
classifier.fit(X,y)
```

Out[26]:
```
DecisionTreeClassifier()
```

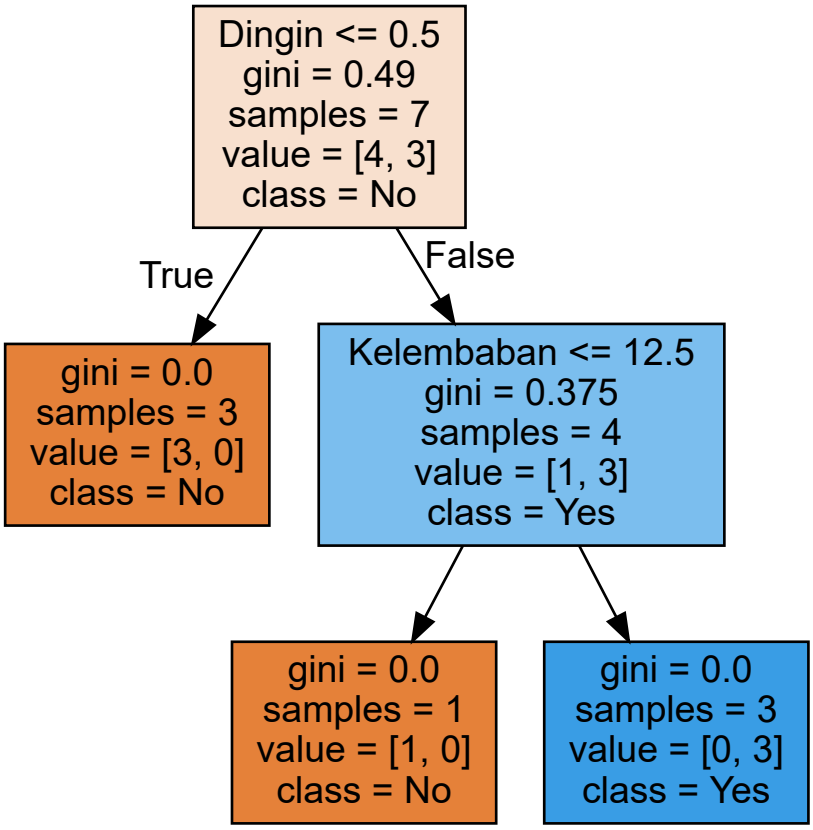
In [27]:
```
from sklearn import tree
import graphviz


f_names = ['Cerah', 'Dingin', 'Kelembaban']
t_names = ['No','Yes']

# DOT data
dot_data = tree.export_graphviz(classifier, out_file=None, feature_names = f_names,

# Draw graph
graph = graphviz.Source(dot_data, format="png")
graph
```

Out[27]:



In [28]:
```
X_test =[[1,1,15]]

classifier.predict(X_test)
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/base.py:451: UserWarning: X does not have valid feature names, but DecisionTreeClassifier was fitted with feature names
  "X does not have valid feature names, but"
```

Out[28]:
```
array([1])
```

Decision Trees in Machine Learning

Pada contoh kali ini, kita akan memprediksi apakah seseorang menderita penyakit diabetes atau tidak menggunakan dua fitur yaitu umur dan nilai tekanan darah.

Memanggil semua library yang digunakan

In [29]:
```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import classification_report, confusion_matrix
```

In [30]:
```
url = 'https://raw.githubusercontent.com/dhirajk100/DT-Classification/master/Decisio

data = pd.read_csv(url)

data
```

Out[30]:

	age	bp	diabetes
0	65	65	1
1	45	82	0
2	35	73	1
3	45	90	0
4	50	68	1
...	...	...	...
982	45	87	0
983	40	83	0
984	40	83	0
985	40	60	1
986	45	82	0

987 rows × 3 columns

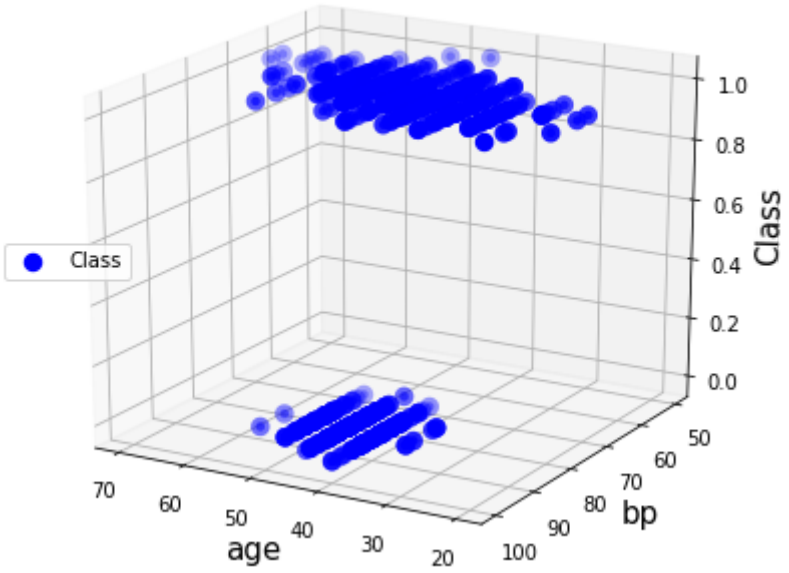
Cek missing values

In [31]:
```
data.isna().sum()
```

Out[31]:
```
age         0
bp          0
diabetes    0
dtype: int64
```

In [32]:
```
from mpl_toolkits.mplot3d import Axes3D
import matplotlib.pyplot as plt

fig = plt.figure(figsize=(7,6))
ax = fig.add_subplot(111, projection='3d')
ax.view_init(20, 120)
ax.scatter(data.age, data.bp, data.diabetes,  color='blue',lw=5)
plt.legend(('Class',), loc='center left')
ax.set_xlabel('age',fontsize=15)
ax.set_ylabel('bp',fontsize=15)
ax.set_zlabel('Class', fontsize=15)
#ax.set_title('Data Scatter', fontsize=15)
plt.show()
```



Splitting Data

Menentukan data X (response) dan y (class)

In [33]:
```
X = data.drop('diabetes',axis=1)
y = data.diabetes
```

Teknik splitting data dengan Scikit-Learn

Training 80%, testing 20%

In [34]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2)
```

In [35]:
```
print(X_test)
```

```
     age  bp
551   50  90
397   50  85
682   45  90
242   45  58
159   55  73
..   ...  ..
778   60  62
343   60  85
760   55  62
928   40  65
99    50  73

[198 rows x 2 columns]
```

Training Model

In [36]:
```
classifier = DecisionTreeClassifier()
classifier.fit(X_train,y_train)
```

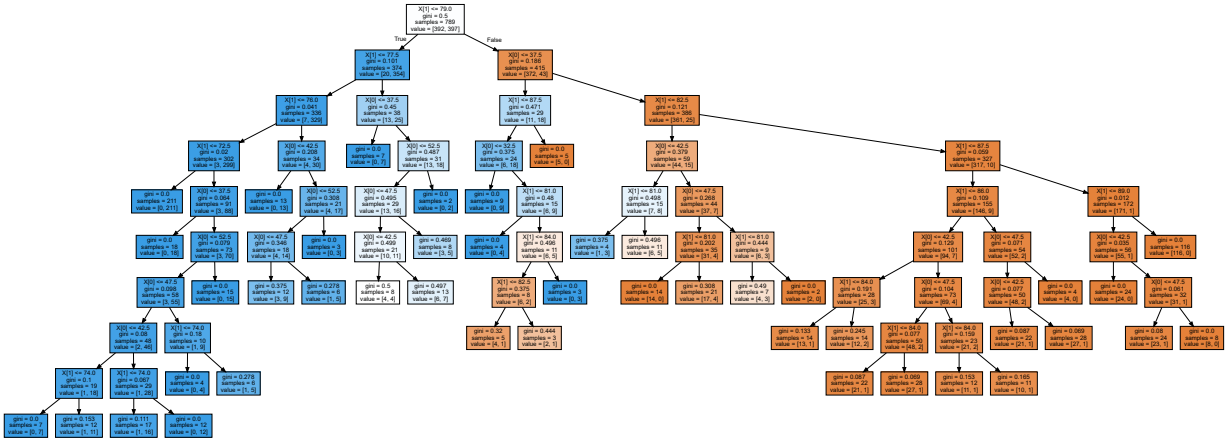
```
DecisionTreeClassifier()
```

Out[36]:

In [37]:
```
import graphviz
# DOT data
dot_data = tree.export_graphviz(classifier, out_file=None,
                                filled=True)

# Draw graph
graph = graphviz.Source(dot_data, format="png")
graph
```

Out[37]:



In [38]:
```
y_pred = classifier.predict(X_test)
y_pred
```

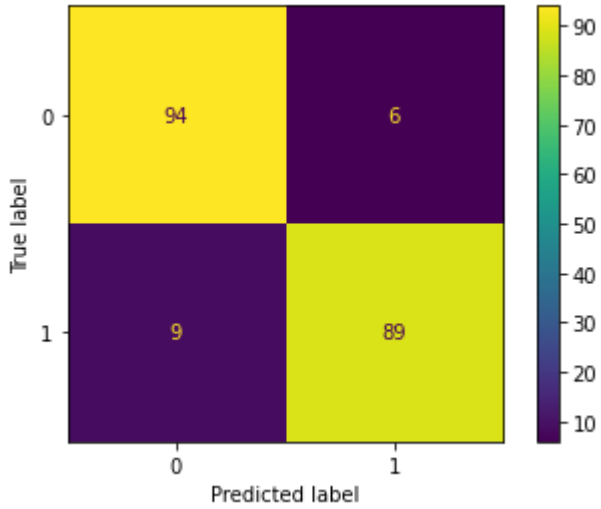
Out[38]:
```
array([0, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 1, 0, 1, 0, 0, 1, 1, 0, 0, 0,
       0, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0,
       1, 1, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0,
       1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 0, 0, 0, 1, 1, 1, 0, 1, 0, 0, 1, 0,
       0, 0, 0, 1, 0, 0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 0, 1, 0, 0,
       0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 1, 1, 1, 0, 1, 1, 0, 0,
       0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1,
       0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 1, 1, 1, 0, 1, 0, 1, 0, 0, 1, 1, 1,
       0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 1])
```

Evaluation

In [39]:
```
from sklearn.metrics import plot_confusion_matrix
plot_confusion_matrix(classifier, X_test, y_test)
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/utils/deprecation.py:87: FutureWarnin
g: Function plot_confusion_matrix is deprecated; Function `plot_confusion_matrix` is
deprecated in 1.0 and will be removed in 1.2. Use one of the class methods: Confusio
nMatrixDisplay.from_predictions or ConfusionMatrixDisplay.from_estimator.
  warnings.warn(msg, category=FutureWarning)
```

Out[39]:
```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7f009f6e7f90>
```



In [40]:
```
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       0.91      0.94      0.93       100
           1       0.94      0.91      0.92        98

    accuracy                           0.92       198
   macro avg       0.92      0.92      0.92       198
weighted avg       0.92      0.92      0.92       198
```

EXERCISE/ HOMEWORK

Buatlah program klasifikasi diabetes menggunakan model Decision Tree Clasifier. Data dapat diunduh pada tautan berikut ini https://www.kaggle.com/uciml/pima-indians-diabetes-database

Fitur yang digunakan untuk memprediksi Label adalah:

1. glucose
2. skin
3. insulin
4. age

In [41]:
```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import classification_report, confusion_matrix
import graphviz
from sklearn import tree
from sklearn.metrics import plot_confusion_matrix
```

In [42]:
```
data = pd.read_csv('diabetes.csv')
data
```

Out[42]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	A
0	6	148	72	35	0	33.6	0.627	
1	1	85	66	29	0	26.6	0.351	

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	A
2	8	183	64	0	0	23.3	0.672	
3	1	89	66	23	94	28.1	0.167	
4	0	137	40	35	168	43.1	2.288	
...	...	...	...	...	...	...	...	
763	10	101	76	48	180	32.9	0.171	
764	2	122	70	27	0	36.8	0.340	
765	5	121	72	23	112	26.2	0.245	
766	1	126	60	0	0	30.1	0.349	
767	1	93	70	31	0	30.4	0.315	

768 rows × 9 columns

Cek missing values

In [43]:
```
data.isna().sum()
```

Out[43]:
```
Pregnancies                 0
Glucose                     0
BloodPressure               0
SkinThickness               0
Insulin                     0
BMI                         0
DiabetesPedigreeFunction    0
Age                         0
Outcome                     0
dtype: int64
```

Splitting Data

Menentukan data X (response) dan y (class)

In [44]:
```
X=data[["Glucose","SkinThickness","Insulin","Age"]]
y=data.Outcome
print(X.head())
print(y.head())
```

```
   Glucose  SkinThickness  Insulin  Age
0      148             35        0   50
1       85             29        0   31
2      183              0        0   32
3       89             23       94   21
4      137             35      168   33
0    1
1    0
2    1
3    0
4    1
Name: Outcome, dtype: int64
```

Teknik splitting data dengan Scikit-Learn

Training 80%, testing 20%

In [45]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2)
```

In [46]:
```
print(X_test)
```

```
     Glucose  SkinThickness  Insulin  Age
99       122             51      220   31
418       83              0        0   27
302       77             41       42   35
84       137              0        0   37
701      125             31        0   49
..       ...            ...      ...  ...
174       75             24       55   33
10       110              0        0   30
160      151             38        0   36
630      114              0        0   34
360      189             33      325   29

[154 rows x 4 columns]
```

Training Model

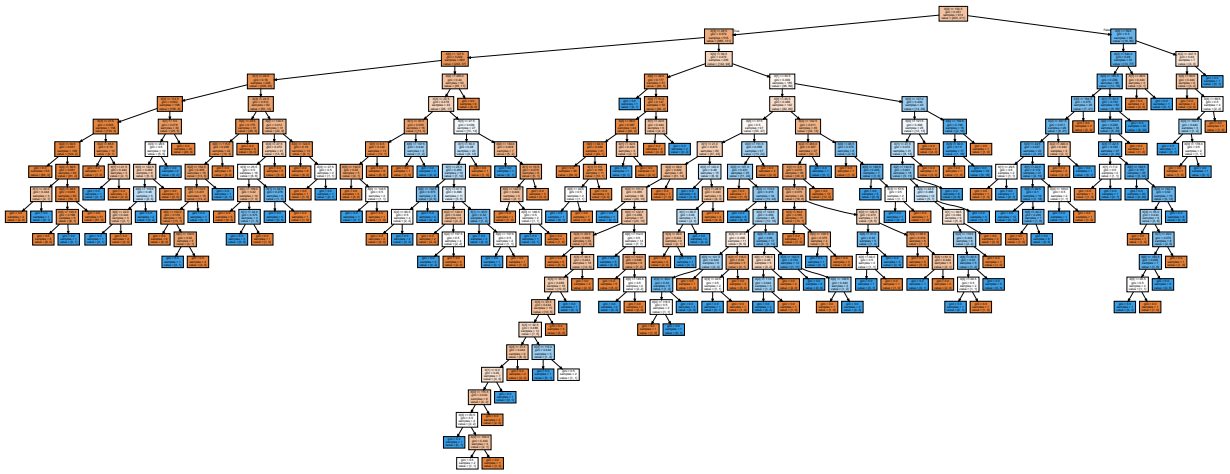
In [47]:
```
classifier = DecisionTreeClassifier()
classifier.fit(X_train,y_train)
```

Out[47]:
```
DecisionTreeClassifier()
```

In [48]:
```
# DOT data
dot_data = tree.export_graphviz(classifier, out_file=None,
                                filled=True)

# Draw graph
graph = graphviz.Source(dot_data, format="png")
graph
```

Out[48]:



In [49]:
```
y_pred = classifier.predict(X_test)
y_pred
```

Out[49]:
```
array([0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0,
       1, 1, 1, 0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 1, 0, 1,
       1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0,
```

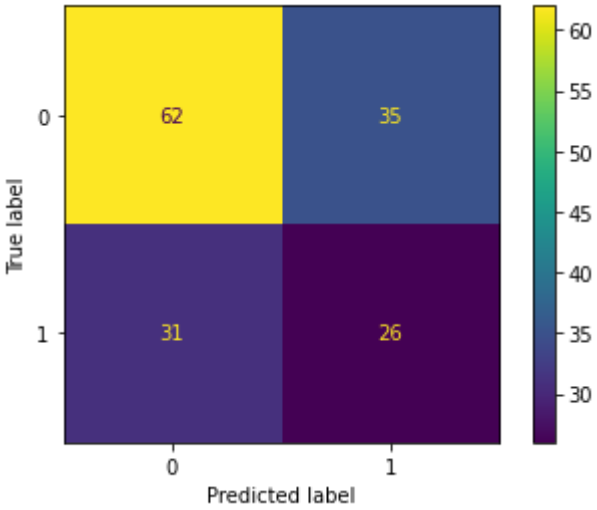
```
0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 1, 1, 1, 0, 0, 1, 0, 0, 0, 1, 1,
0, 0, 1, 1, 1, 1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
0, 1, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 1])
```

Evaluation

In [50]:
```
plot_confusion_matrix(classifier, X_test, y_test)
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/utils/deprecation.py:87: FutureWarnin
g: Function plot_confusion_matrix is deprecated; Function `plot_confusion_matrix` is
deprecated in 1.0 and will be removed in 1.2. Use one of the class methods: Confusio
nMatrixDisplay.from_predictions or ConfusionMatrixDisplay.from_estimator.
  warnings.warn(msg, category=FutureWarning)
```

Out[50]:
```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7f00a4028550>
```



In [51]:
```
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       0.67      0.64      0.65        97
           1       0.43      0.46      0.44        57

    accuracy                           0.57       154
   macro avg       0.55      0.55      0.55       154
weighted avg       0.58      0.57      0.57       154
```