

decision-tree-22mcb1002

April 28, 2023

1 DECISION TREE

```
[18]: # Package import
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier, plot_tree
from sklearn.preprocessing import StandardScaler
from sklearn.feature_selection import SelectKBest, f_classif
from sklearn.pipeline import Pipeline
from sklearn.metrics import accuracy_score
from sklearn.impute import SimpleImputer
import graphviz
```

```
[6]: # Importing dataset
df = pd.read_csv('/content/drive/MyDrive/Data Analytics/heart1.csv')
```

```
[7]: df.head(5)
```

```
[7]:
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	\
0	63	1	3	145	233	1	0	150	0	2.3	0	
1	37	1	2	130	250	0	1	187	0	3.5	0	
2	41	0	1	130	204	0	0	172	0	1.4	2	
3	56	1	1	120	236	0	1	178	0	0.8	2	
4	57	0	0	120	354	0	1	163	1	0.6	2	

	ca	thal	target	Unnamed: 14
0	0	1	1	NaN
1	0	2	1	NaN
2	0	2	1	NaN
3	0	2	1	NaN
4	0	2	1	NaN

```
[8]: # Filling missing values
df.fillna(df.mean(), inplace = True)
```

```
[19]: # Splitting dataset into features and target and also imputing missing values
X = df.drop(columns = ['target'])
```

```

imputer = SimpleImputer(strategy = 'mean')
imputer.fit(X)
X = imputer.transform(X)
y = df['target']

```

```

[20]: # Feature subset selection
scaler = StandardScaler()
X_norm = scaler.fit_transform(X)

```

```

[21]: # Selecting top 5 features using ANOVA F-Test
fs = SelectKBest(score_func=f_classif, k=5)
X_new = fs.fit_transform(X_norm, y)

```

```

[22]: # Splitting data into training and testing dataset
X_train, X_test, y_train, y_test = train_test_split(X_new, y, test_size = 0.2,
↳ random_state = 42)

```

```

[23]: # Building decision tree classifier model
dtc = DecisionTreeClassifier(random_state = 42)

```

```

[24]: # Training classifier
dtc.fit(X_train, y_train)

```

```

[24]: DecisionTreeClassifier(random_state=42)

```

```

[25]: # Classifier evaluation
y_pred = dtc.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)

```

Accuracy: 0.819672131147541

```

[32]: # Tree visualization
dot_data = plot_tree(dtc, feature_names = ['age', 'sex', 'cp', 'trestbps',
↳ 'chol',
                                     'fbs', 'restecg', 'thalach', 'exang',
                                     'foldpeak', 'slope', 'ca', 'thal'],
                    class_names = ['class1', 'class2'], filled = True, rounded
↳ = True)
graph = graphviz.Source(dot_data)
graph.render("decision_tree")

graph

```

