da-lab-9

April 28, 2023

1 DECISION TREE

[119]:
```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call
drive.mount("/content/drive", force_remount=True).
```

[120]:
```
# Package import
import pandas as pd
import numpy as np
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
```

[121]:
```
# Importing dataset
df = pd.read_csv('/content/drive/MyDrive/Project/amazon(DA).csv')
```

[122]:
```
df.head(5)
```

[122]:
```
   discounted_price  actual_price  discount_percentage  rating  rating_count
0             399.0        1099.0               8755.0     4.2       24269.0
1             199.0         349.0               4300.0     4.0       43994.0
2             199.0        1899.0               9000.0     3.9        7928.0
3             329.0         699.0               5300.0     4.2       94363.0
4             154.0         399.0               6100.0     4.2       16905.0
```

[123]:
```
# Filling missing values
df.fillna(df.mean(), inplace = True)
```

[124]:
```
bin = [0, 5000, 20000, 50000, max(df['rating_count'])]
label = ['Low', 'Medium', 'High', 'Very High']
df['new_bin'] = pd.cut(df['rating_count'], bins=bin,labels=label)
```

[125]:
```python
# Splitting dataset into features and target and also imputing missing values
Df=df.values
X = Df[:,:-1]
imputer = SimpleImputer(strategy = 'mean')
imputer.fit(X)
X = imputer.transform(X)
y= (Df[:,-1])
```

[126]:
```python
# Feature subset selection
scaler = StandardScaler()
X_norm = scaler.fit_transform(X)
X_norm.shape
y.shape
```

[126]:
```
(1465,)
```

[127]:
```python
# Selecting top 5 features using ANOVA F-Test
fs = SelectKBest(score_func=f_classif, k=4)
X_new = fs.fit_transform(X_norm, y)
```

[128]:
```python
# 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)
```

[129]:
```python
# Building decision tree classifier model
dtc = DecisionTreeClassifier(random_state = 42)
```

[130]:
```python
# Training classifier
dtc.fit(X_train, y_train)
```

[130]:
```
DecisionTreeClassifier(random_state=42)
```

[131]:
```python
# Classifier evaluation
y_pred = dtc.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
df.columns
```

```
Accuracy: 0.9965870307167235
```

[131]:
```
Index(['discounted_price', 'actual_price', 'discount_percentage', 'rating',
       'rating_count', 'new_bin'],
      dtype='object')
```