Exercise 5 : Classification Tree

Problem 1 : Predicting CentralAir using SalePrice

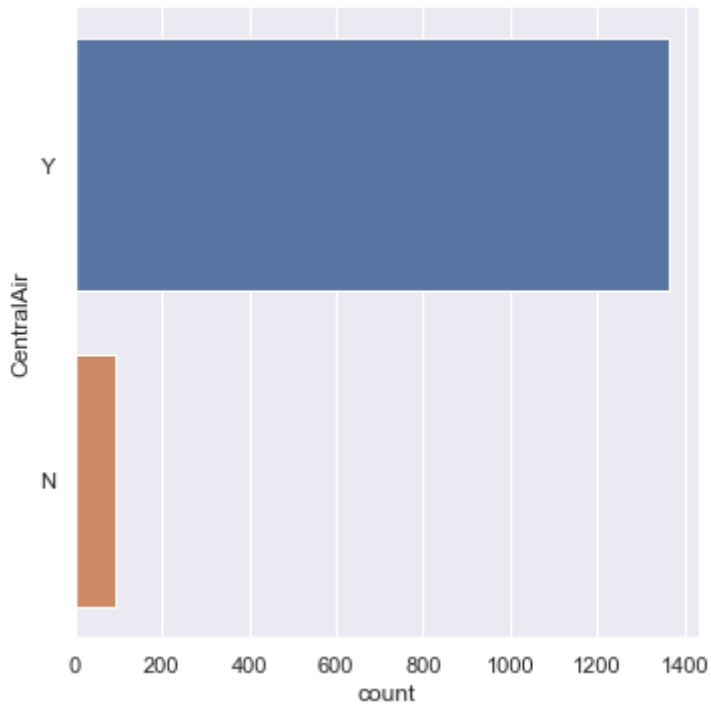
Import the complete dataset “train.csv” in Jupyter, as houseData = pd.read_csv('train.csv')

Note : In this exercise, we will not extract the variables from the dataset, as we did the last time.

a) Plot the binary distribution of houseData['CentralAir'] using catplot to check the ratio of Y against N. Note that the classes Y and N are quite unbalanced; do you think this will create any problem in our Classification?

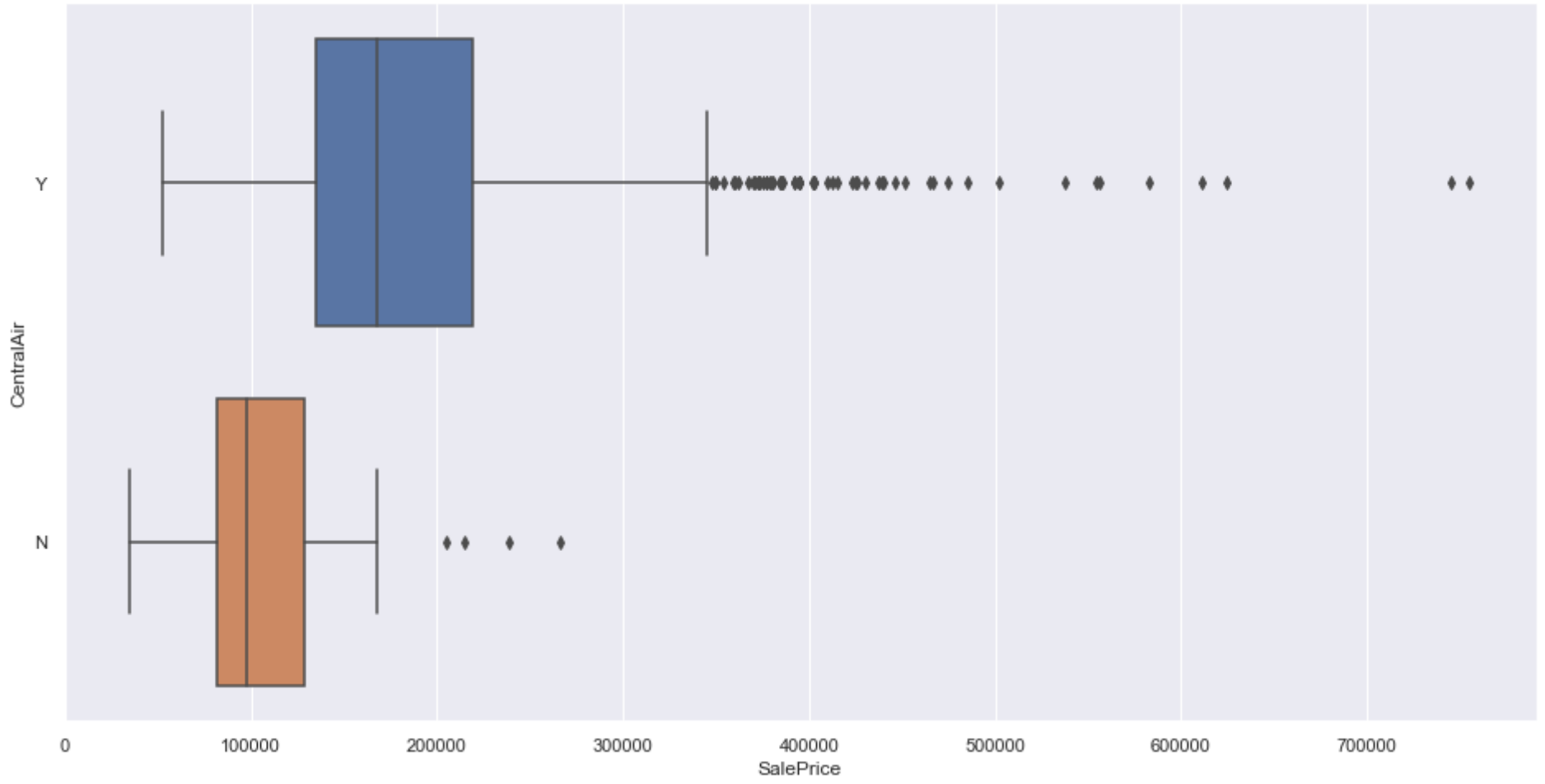
In [3]:
```
# Basic Libraries
import numpy as np
import pandas as pd
import seaborn as sb
import matplotlib.pyplot as plt
from sklearn.tree import DecisionTreeClassifier
from sklearn.tree import plot_tree
from sklearn.metrics import confusion_matrix
sb.set()

houseData = pd.read_csv('C:/Users/pengh/OneDrive/Desktop/Cx1015_MA10/shared folder/train.csv')
```

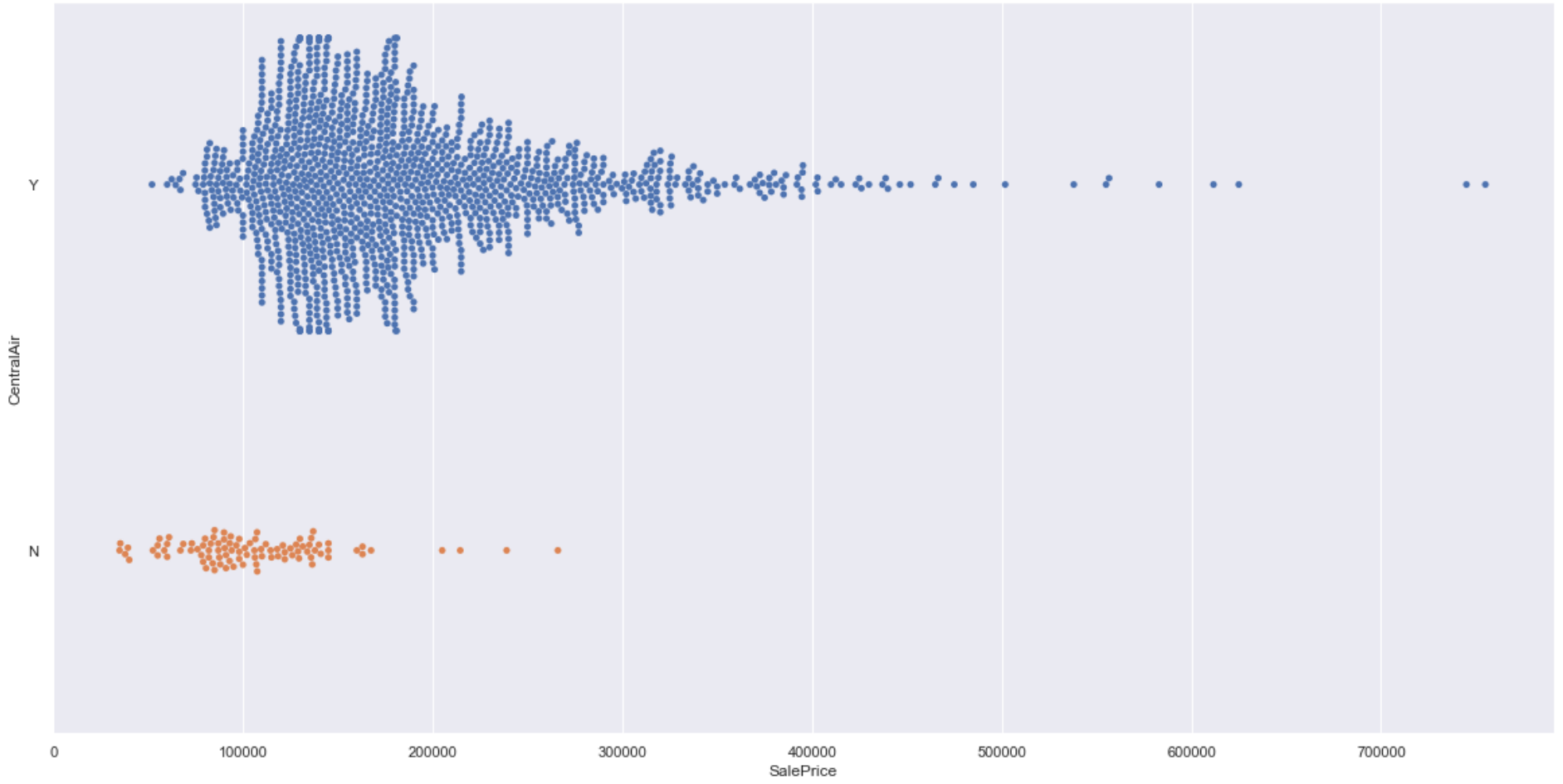
```
sb.catplot(data=houseData, y='CentralAir', kind="count");
```



b) Plot houseData['CentralAir'] vs houseData['SalePrice'] using boxplot, and note the strong relationship. Also check the mutual relationship by plotting the two variables using a swarmplot, and note the difference.

In [4]:
```
f, axes = plt.subplots(1, 1, figsize=(16, 8))
sb.boxplot(data=houseData, x='SalePrice', y='CentralAir')


f, axes = plt.subplots(1, 1, figsize=(20, 10))
sb.swarmplot(data=houseData, x='SalePrice', y='CentralAir');
```





c) Import Classification Tree model from Scikit-Learn : from sklearn.tree import DecisionTreeClassifier

In [5]:
```
from sklearn.tree import DecisionTreeClassifier
```

d) Partition the complete dataset houseData into houseData_train (1100 rows) and houseData_test (360 rows).

In [133…
```
houseData_train = pd.DataFrame(houseData[:1100])
houseData_test = pd.DataFrame(houseData[-360:])


# To split them randomly
# houseData_train, houseData_test = train_test_split(houseData, test_size = 360)

print("Train Set\t:", houseData_train.shape)
print("Test Set\t:", houseData_test.shape)
```

```
Train Set       : (1100, 81)
Test Set        : (360, 81)
```

e) Training : Fit a Decision Tree model for classification of CentralAir using SalePrice using the following variables.

y_train = pd.DataFrame(houseData_train['CentralAir'])

X_train = pd.DataFrame(houseData_train['SalePrice'])

In [139…

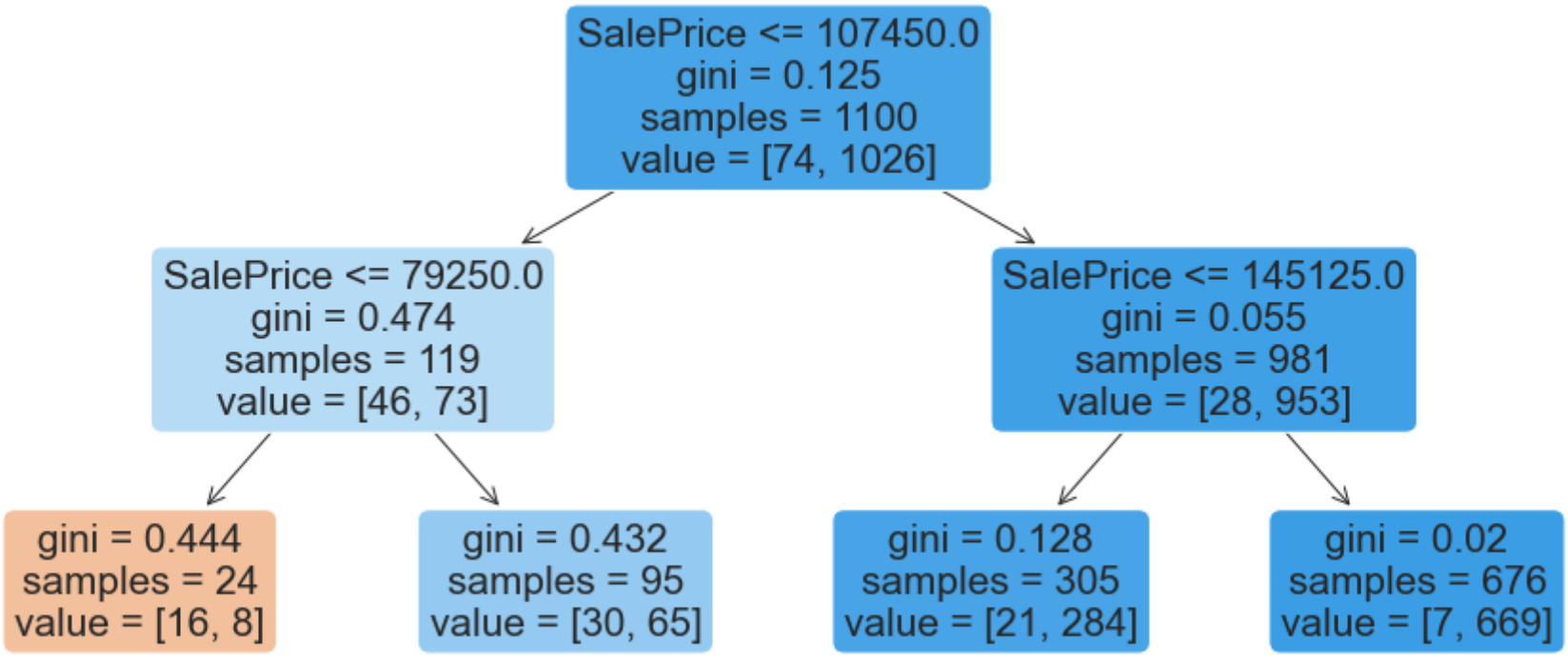
```
from sklearn.tree import DecisionTreeClassifier

# Must first initialise the decision tree before the fit
dectree = DecisionTreeClassifier(max_depth=2)

y_train = pd.DataFrame(houseData_train['CentralAir'])
X_train = pd.DataFrame(houseData_train['SalePrice'])

dectree.fit(X_train, y_train)
```

Out[139… DecisionTreeClassifier(max_depth=2)

f) Visualize the Decision Tree model using graphviz (needs the packages to be installed; check if they are installed).

In [140…

```
from sklearn.tree import plot_tree

f, axes = plt.subplots(1, 1, figsize=(16, 7))
plot_tree(dectree,
          feature_names = X_train.columns,
          filled = True,
          rounded = True)
plt.show()
```

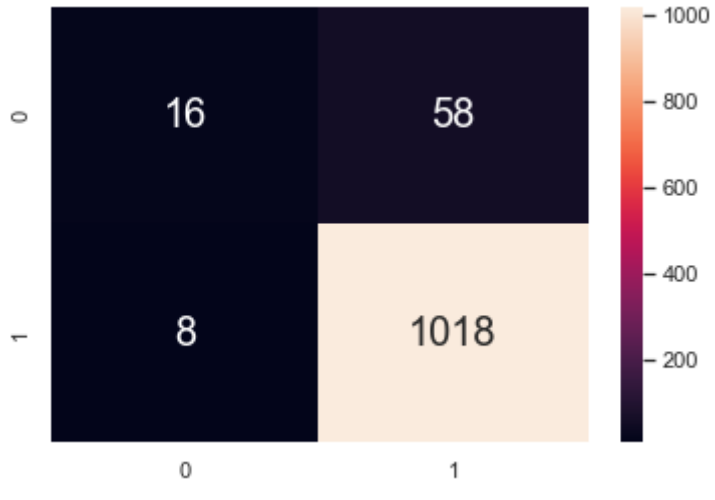


g) Predict CentralAir for the train dataset using the Decision Tree model, and plot the Two-Way Confusion Matrix. Predict CentralAir for the test dataset using the Decision Tree model, and plot the Two-Way Confusion Matrix.

In [141…

```
from sklearn.metrics import confusion_matrix


# Using SalePrice train Set to predict CentralAir train set
y_train_pred = dectree.predict(X_train)

# Print the Classification Accuracy
print("Classification Accuracy \t:", dectree.score(X_train, y_train))

# Plot the two-way Confusion Matrix
sb.heatmap(confusion_matrix(y_train, y_train_pred),
           annot = True,
           fmt=".0f",
           annot_kws={"size": 20});
```

```
Classification Accuracy         : 0.94
```

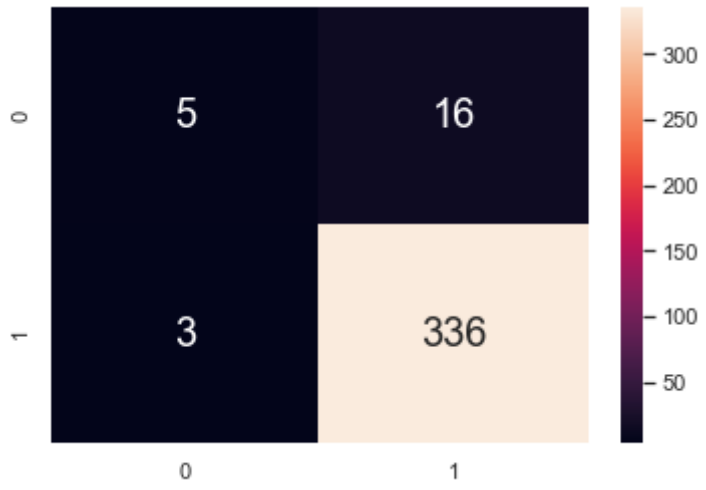


In [142…

```
# Extract the two variables X_test and y_test
y_test = pd.DataFrame(houseData_test['CentralAir'])
X_test = pd.DataFrame(houseData_test['SalePrice'])
```

```
# Predict the Response corresponding to Predictors
y_test_pred = dectree.predict(X_test)

# Print the Classification Accuracy
print("Classification Accuracy \t:", dectree.score(X_test, y_test))

# Plot the two-way Confusion Matrix
sb.heatmap(confusion_matrix(y_test, y_test_pred),
           annot = True,
           fmt=".0f",
           annot_kws={"size": 20});
```

```
Classification Accuracy         : 0.9472222222222222
```

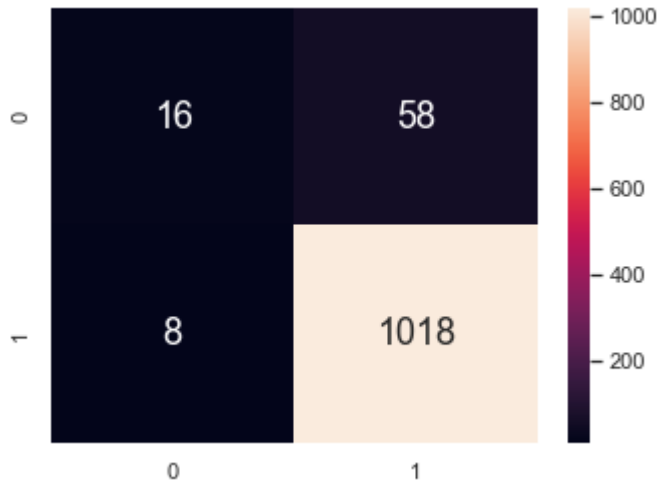


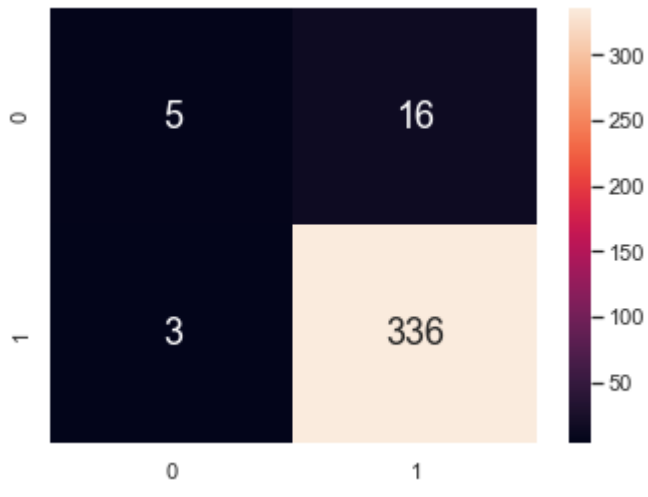
h) Print all the accuracy parameters of the decision tree model, including its Classification Accuracy, True Positive Rate, True Negative Rate, False Positive Rate and False Negative Rate, based on the aforesaid confusion matrix.

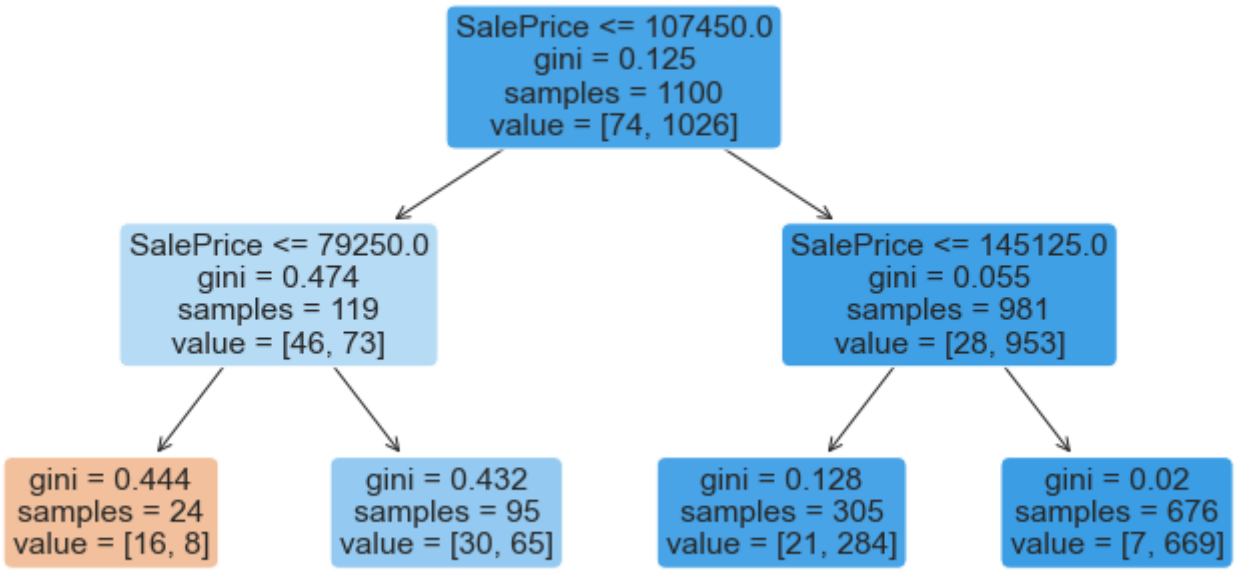
In [143…

```
dectree = DecisionTreeClassifier(max_depth = 2)
dectree.fit(X_train, y_train)
y_train_pred = dectree.predict(X_train)
y_test_pred = dectree.predict(X_test)

print("Goodness of Fit of Model \tTrain Dataset")
print("Classification Accuracy \t:", dectree.score(X_train, y_train),'\n')

print("Goodness of Fit of Model \tTest Dataset")
print("Classification Accuracy \t:", dectree.score(X_test, y_test))


f, axes = plt.subplots(1, 2, figsize=(12, 4))
sb.heatmap(confusion_matrix(y_train, y_train_pred), annot = True, fmt=".0f", annot_kws={"size": 18}, ax = axes[0])
sb.heatmap(confusion_matrix(y_test, y_test_pred), annot = True, fmt=".0f", annot_kws={"size": 18}, ax = axes[1])

f, axes = plt.subplots(1, 1, figsize=(12, 6))
plot_tree(dectree, filled=True, rounded = True, feature_names=X_train.columns)
plt.show()
```

```
Goodness of Fit of Model        Train Dataset
Classification Accuracy         : 0.94

Goodness of Fit of Model        Test Dataset
Classification Accuracy         : 0.9472222222222222
```







Problem 2 : Predicting CentralAir using Other Variables

Perform all the above steps on 'CentralAir' against each of the variables 'GrLivArea', 'LotArea', 'TotalBsmtSF' one-by-one to obtain individual Decision Trees. Discuss with your Friends about the models, compare the Classification Accuracy, check the True Positives and False Positives, and determine which model is the best to predict 'CentralAir'.

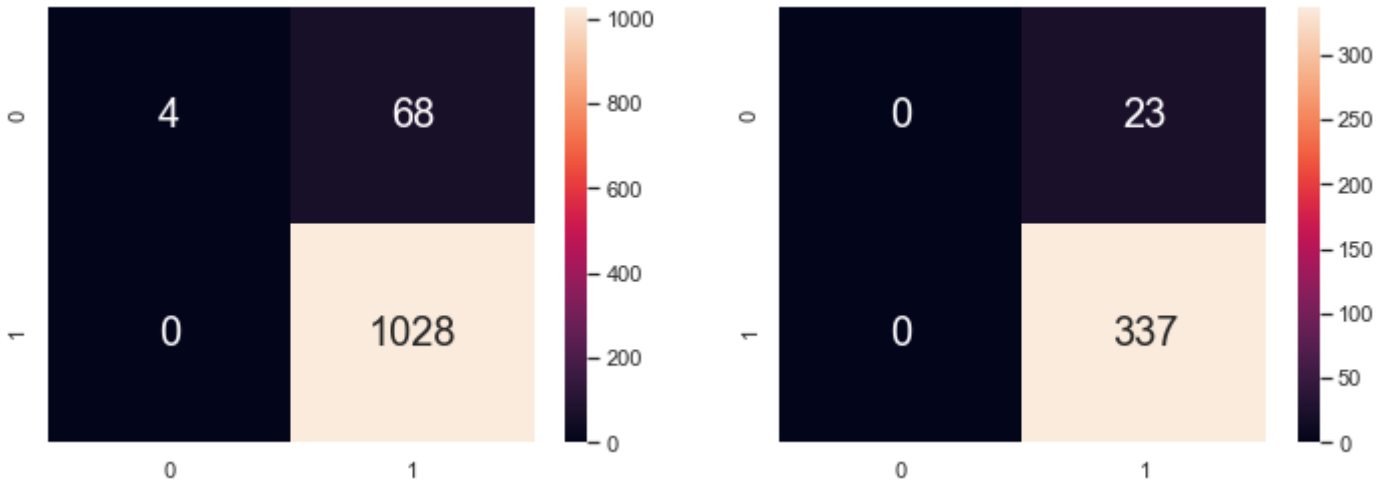
In [106…

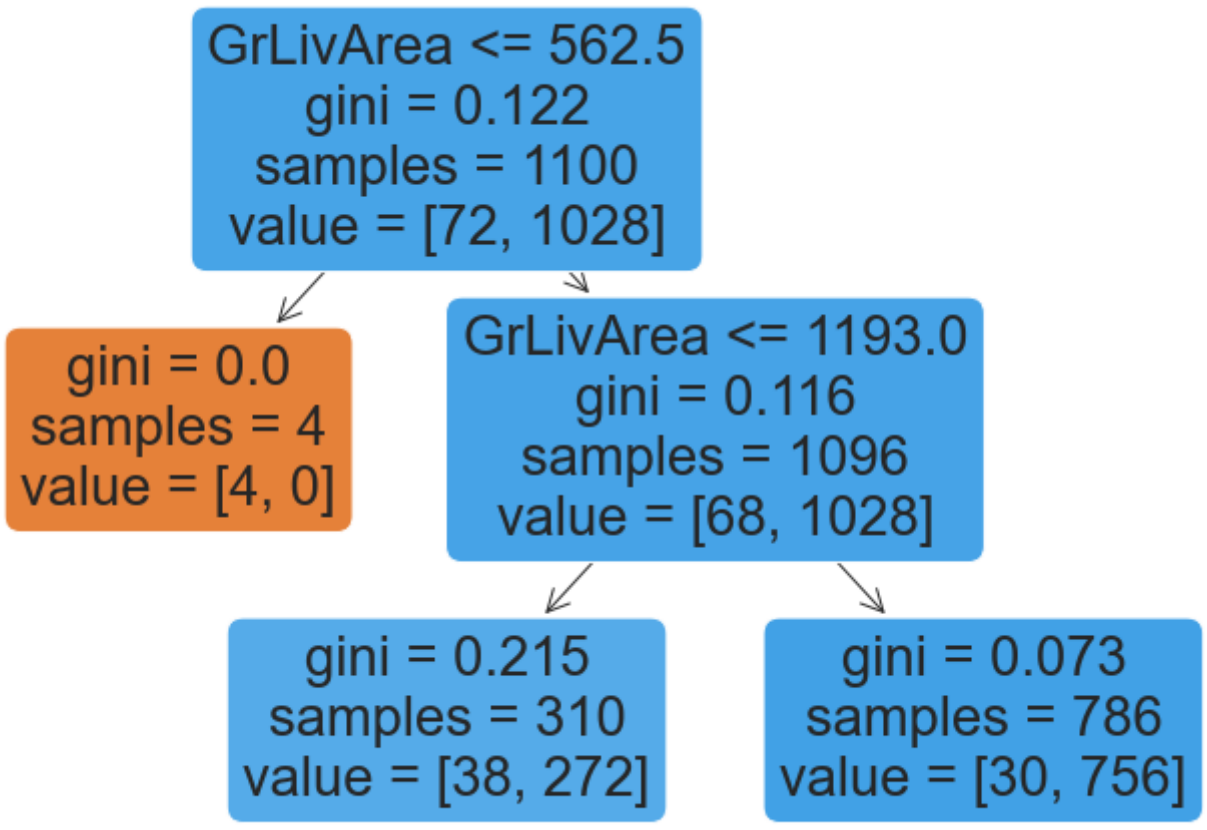
```
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix
from sklearn.tree import plot_tree

def ClassProcedure(predictor_any, response_str = 'CentralAir'):
    """
    Procedure. Given the name (string or list) of the predictor in a string, perform
    the classification and visualise the relevant visuals.
    """
    y = pd.DataFrame(houseData[response_str])
    X = pd.DataFrame(houseData[predictor_any])
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 360)       # Splitting data randomly
    dectree = DecisionTreeClassifier(max_depth = 2)
    dectree.fit(X_train, y_train)
    y_train_pred = dectree.predict(X_train)
    y_test_pred = dectree.predict(X_test)

    print("Goodness of Fit of Model \tTrain Dataset")
    print("Classification Accuracy \t:", dectree.score(X_train, y_train),'\n')

    print("Goodness of Fit of Model \tTest Dataset")
    print("Classification Accuracy \t:", dectree.score(X_test, y_test))


    f, axes = plt.subplots(1, 2, figsize=(12, 4))
    sb.heatmap(confusion_matrix(y_train, y_train_pred), annot = True, fmt=".0f", annot_kws={"size": 20}, ax = axes[0])
    sb.heatmap(confusion_matrix(y_test, y_test_pred), annot = True, fmt=".0f", annot_kws={"size": 20}, ax = axes[1])

    f, axes = plt.subplots(1, 1, figsize=(12, 8))
    plot_tree(dectree, filled=True, rounded = True, feature_names=X_train.columns)
    plt.show()
```

In [146…

```
ClassProcedure('GrLivArea')
```

```
Goodness of Fit of Model        Train Dataset
Classification Accuracy         : 0.9381818181818182

Goodness of Fit of Model        Test Dataset
Classification Accuracy         : 0.9361111111111111
```

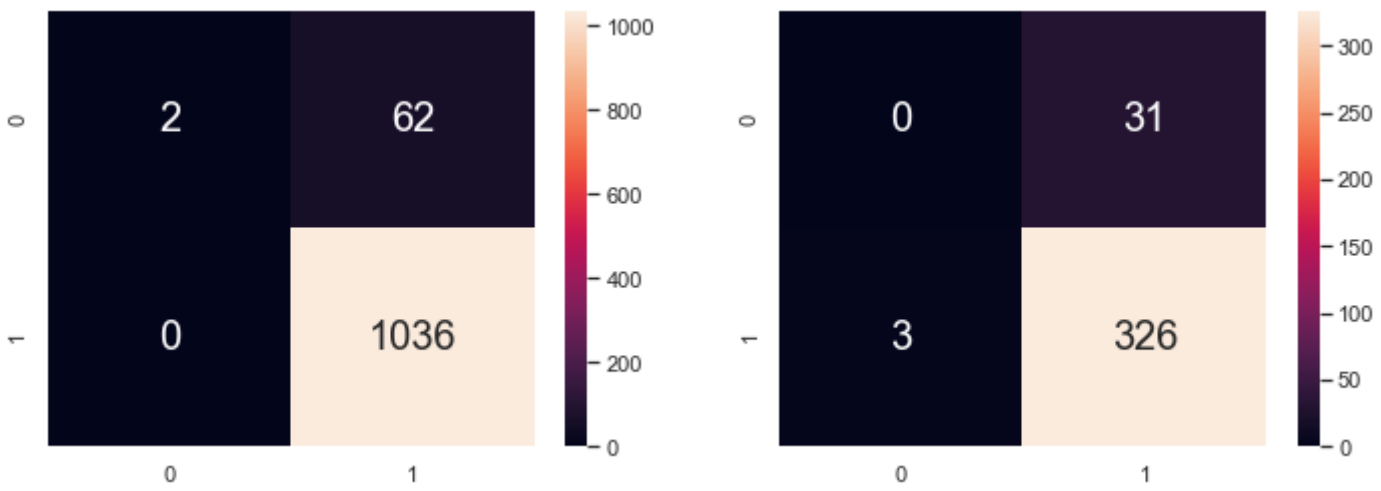


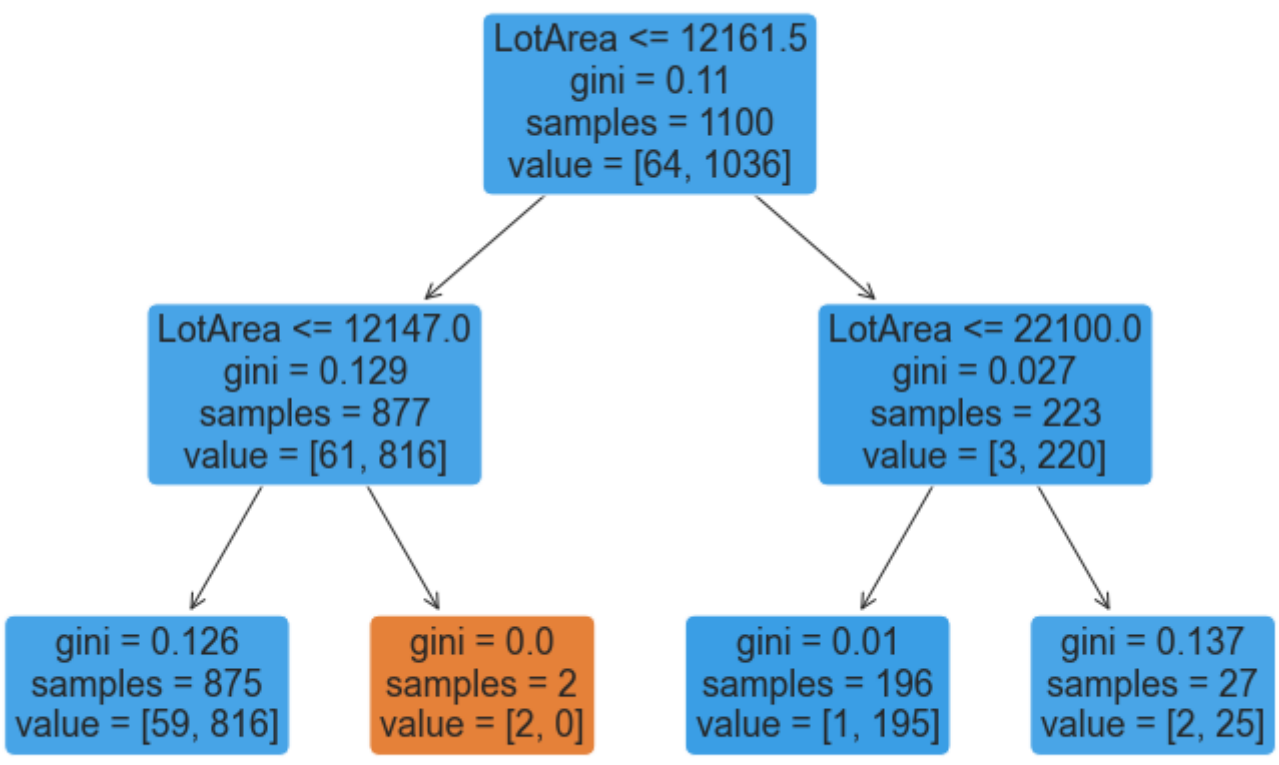


In [147…

```
ClassProcedure('LotArea')
```

```
Goodness of Fit of Model        Train Dataset
Classification Accuracy         : 0.9436363636363636

Goodness of Fit of Model        Test Dataset
Classification Accuracy         : 0.9055555555555556
```

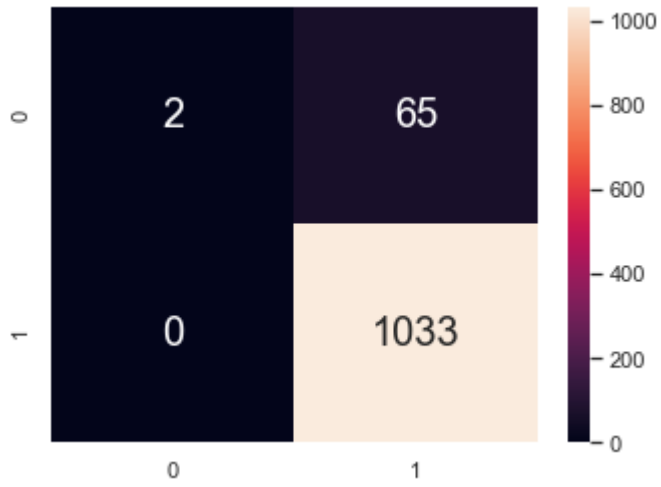


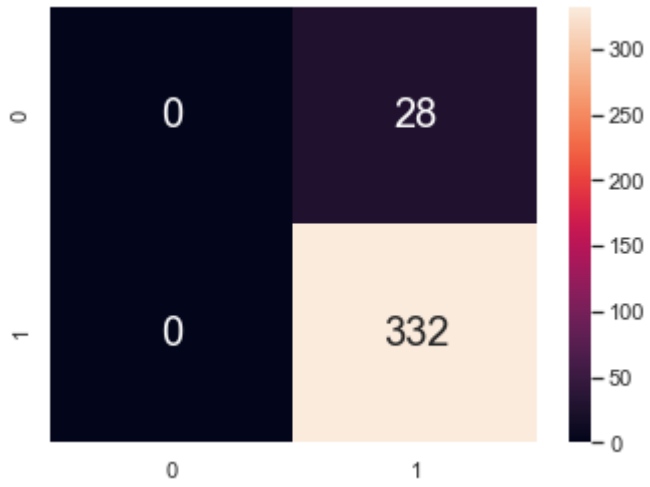


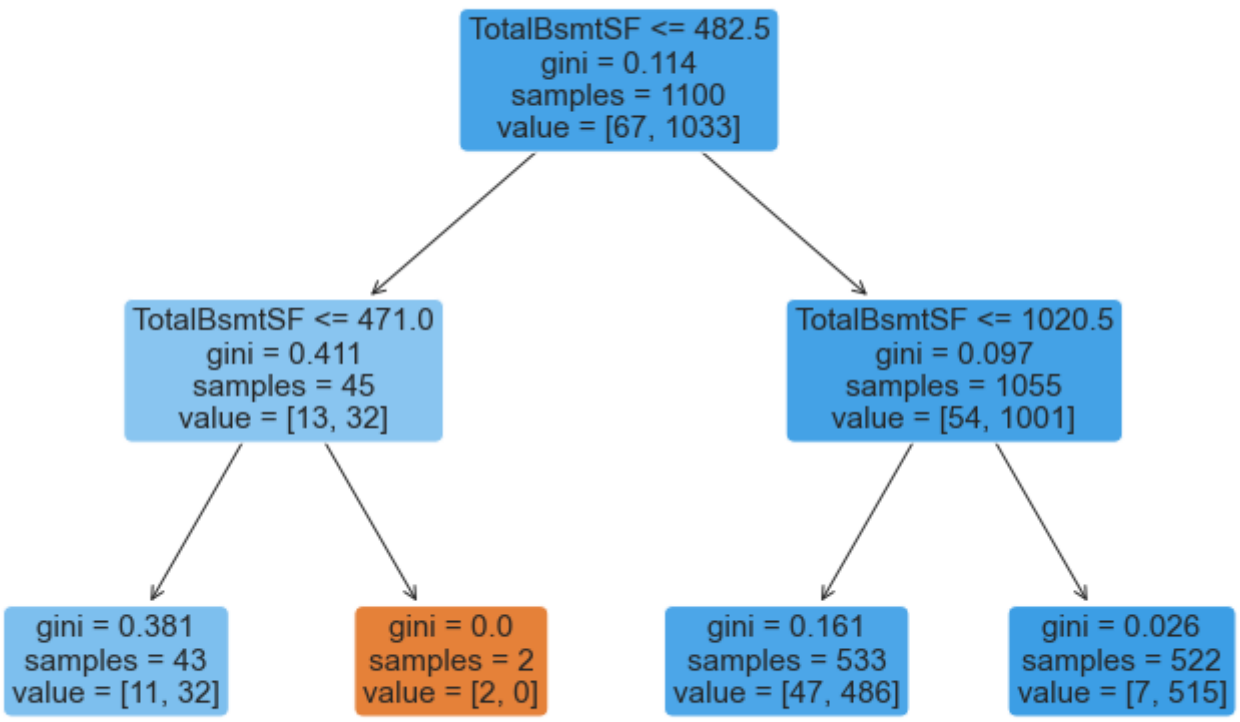
In [148…

```
ClassProcedure('TotalBsmtSF')
```

```
Goodness of Fit of Model        Train Dataset
Classification Accuracy         : 0.9409090909090909

Goodness of Fit of Model        Test Dataset
Classification Accuracy         : 0.9222222222222223
```


2
65
0
1033
0
1
0
1
1000
800
600
400
200
0


0
28
0
332
0
1
0
1
300
250
200
150
100
50
0


TotalBsmtSF <= 482.5
gini = 0.114
samples = 1100
value = [67, 1033]
TotalBsmtSF <= 471.0
gini = 0.411
samples = 45
value = [13, 32]
TotalBsmtSF <= 1020.5
gini = 0.097
samples = 1055
value = [54, 1001]
gini = 0.381
samples = 43
value = [11, 32]
gini = 0.0
samples = 2
value = [2, 0]
gini = 0.161
samples = 533
value = [47, 486]
gini = 0.026
samples = 522
value = [7, 515]