Name : MIN SOE HTUT

ID : 1631938

Load the dataset

In [73]:
```
import numpy as np
import pandas as pd
url = 'https://raw.githubusercontent.com/bpfa/data_for_compx310_2023/main/covtypeNorm_f1_5_s42.csv'
df = pd.read_csv(url)
df
```

Out[73]:

	Elevation	Aspect	Slope	Horizontal_Distance_To_Hydrology	Vertical_Distance_To_Hy
0	0.556278	0.272222	0.136364	0.107373	0.
1	0.543272	0.944444	0.257576	0.047960	0.
2	0.375188	0.266667	0.212121	0.137437	0.
3	0.552276	0.294444	0.090909	0.434503	0.
4	0.642821	0.069444	0.166667	0.211167	0
...	...	...	...	...	
116198	0.497749	0.919444	0.424242	0.231210	0.
116199	0.565283	0.013889	0.212121	0.000000	0.
116200	0.696348	0.408333	0.242424	0.273443	0.
116201	0.687344	0.147222	0.378788	0.173228	0.
116202	0.545273	0.141667	0.333333	0.121689	0.

116203 rows × 55 columns

Display the dataset information

In [74]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 116203 entries, 0 to 116202
Data columns (total 55 columns):
 #   Column                              Non-Null Count   Dtype  
---  ------                              --------------   -----  
 0   Elevation                           116203 non-null  float64
 1   Aspect                              116203 non-null  float64
 2   Slope                               116203 non-null  float64
 3   Horizontal_Distance_To_Hydrology    116203 non-null  float64
 4   Vertical_Distance_To_Hydrology      116203 non-null  float64
 5   Horizontal_Distance_To_Roadways     116203 non-null  float64
 6   Hillshade_9am                       116203 non-null  float64
 7   Hillshade_Noon                      116203 non-null  float64
 8   Hillshade_3pm                       116203 non-null  float64
 9   Horizontal_Distance_To_Fire_Points  116203 non-null  float64
 10  Wilderness_Area1                    116203 non-null  int64  
 11  Wilderness_Area2                    116203 non-null  int64  
 12  Wilderness_Area3                    116203 non-null  int64  
 13  Wilderness_Area4                    116203 non-null  int64  
 14  Soil_Type1                          116203 non-null  int64  
 15  Soil_Type2                          116203 non-null  int64  
 16  Soil_Type3                          116203 non-null  int64  
 17  Soil_Type4                          116203 non-null  int64  
 18  Soil_Type5                          116203 non-null  int64  
 19  Soil_Type6                          116203 non-null  int64  
 20  Soil_Type7                          116203 non-null  int64  
 21  Soil_Type8                          116203 non-null  int64  
 22  Soil_Type9                          116203 non-null  int64  
 23  Soil_Type10                         116203 non-null  int64  
 24  Soil_Type11                         116203 non-null  int64  
 25  Soil_Type12                         116203 non-null  int64  
 26  Soil_Type13                         116203 non-null  int64  
 27  Soil_Type14                         116203 non-null  int64  
 28  Soil_Type15                         116203 non-null  int64  
 29  Soil_Type16                         116203 non-null  int64  
 30  Soil_Type17                         116203 non-null  int64  
 31  Soil_Type18                         116203 non-null  int64  
 32  Soil_Type19                         116203 non-null  int64  
 33  Soil_Type20                         116203 non-null  int64  
 34  Soil_Type21                         116203 non-null  int64  
 35  Soil_Type22                         116203 non-null  int64  
 36  Soil_Type23                         116203 non-null  int64  
 37  Soil_Type24                         116203 non-null  int64  
 38  Soil_Type25                         116203 non-null  int64  
 39  Soil_Type26                         116203 non-null  int64  
 40  Soil_Type27                         116203 non-null  int64  
 41  Soil_Type28                         116203 non-null  int64  
 42  Soil_Type29                         116203 non-null  int64  
 43  Soil_Type30                         116203 non-null  int64  
 44  Soil_Type31                         116203 non-null  int64  
 45  Soil_Type32                         116203 non-null  int64  
 46  Soil_Type33                         116203 non-null  int64  
 47  Soil_Type34                         116203 non-null  int64  
 48  Soil_Type35                         116203 non-null  int64  
 49  Soil_Type36                         116203 non-null  int64  
 50  Soil_Type37                         116203 non-null  int64  
 51  Soil_Type38                         116203 non-null  int64  
```

```
 52  Soil_Type39                             116203 non-null  int64
 53  Soil_Type40                             116203 non-null  int64
 54  class                                   116203 non-null  int64
dtypes: float64(10), int64(45)
memory usage: 48.8 MB
```

Selecting all features except "class" as X, selecting "class" as y, and splitting into three parts: 60% train, 20% validation and 20% test; remember to use stratification, and "random_state=YOUR_ID"

In [75]:
```
from sklearn.model_selection import train_test_split
df = df.dropna()
X = df.iloc[:, :-1]
y = df.iloc[:, -1]
ID = 1631938
# Split the data into 60% train and 40% temporary set using stratification and
random state
X_train, X_temp, y_train, y_temp = train_test_split(X, y, test_size=0.4, rando
m_state=ID, stratify=y)
# Split the temporary set into 50% validation and 50% test using stratificatio
n and random state
X_test, X_val, y_test, y_val = train_test_split(X_temp, y_temp, test_size=0.5,
random_state=ID, stratify=y_temp)
```

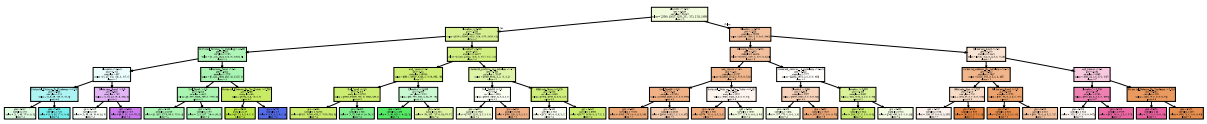
Building twenty trees, by varying the max_depth from 5 up to 100, in steps of 5; remember to use "random_state=YOUR_ID" for the tree

In [76]:
```
from sklearn.tree import DecisionTreeClassifier
# Initialize a list to store models
models = []
# Define the range of depths
depths = range(5, 101, 5)
# Build and evaluate the decision trees
for depth in depths:
    tree = DecisionTreeClassifier(max_depth=depth, random_state=ID)
    tree.fit(X_train, y_train)
    models.append(tree)
```

Plot the smallest tree

In [87]:
```
from sklearn import tree
import graphviz
# Visualize the smallest tree (max_depth=5)
smallest_tree = models[0]
dot_data = tree.export_graphviz(smallest_tree, out_file=None,feature_names=X.c
olumns,class_names=[str(cls) for cls in smallest_tree.classes_], filled=True)
graph = graphviz.Source(dot_data, format="png")
graph
```

Out[87]:

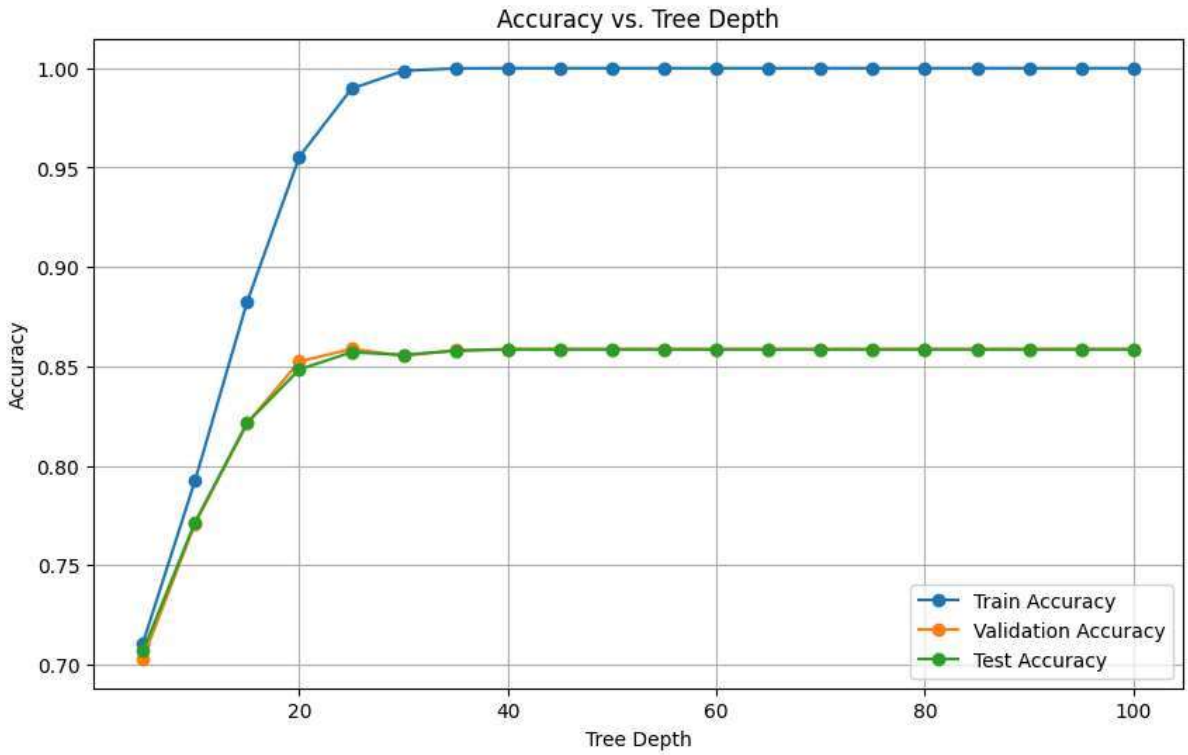


Collect and Plot Accuracies

In [78]:
```
import matplotlib.pyplot as plt
from sklearn.metrics import accuracy_score
# Initialize lists to store accuracies
train_accuracies = []
val_accuracies = []
test_accuracies = []

# Collect accuracies
for tree in models:
    train_pred = tree.predict(X_train)
    val_pred = tree.predict(X_val)
    test_pred = tree.predict(X_test)

    train_acc = accuracy_score(y_train, train_pred)
    val_acc = accuracy_score(y_val, val_pred)
    test_acc = accuracy_score(y_test, test_pred)

    train_accuracies.append(train_acc)
    val_accuracies.append(val_acc)
    test_accuracies.append(test_acc)
```

Generate a single plot

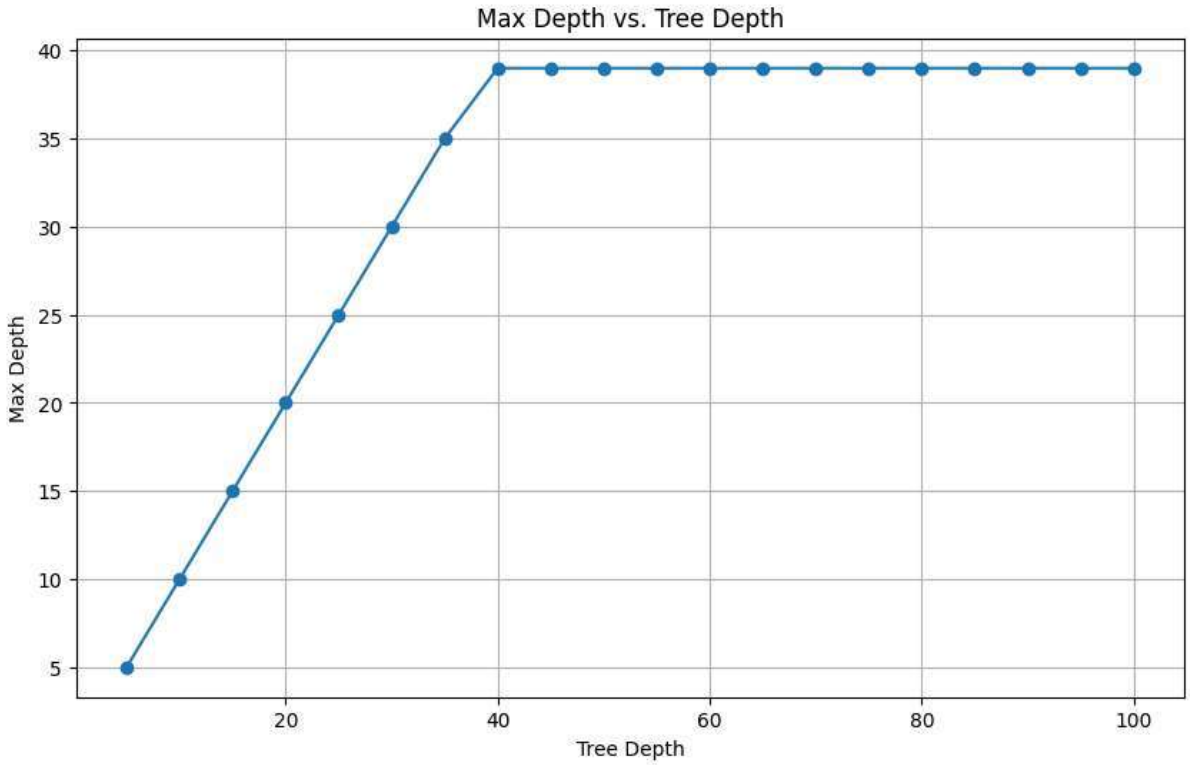
In [79]:
```
tree_depths = range(5, 101, 5)
plt.figure(figsize=(10, 6))
plt.plot(tree_depths, train_accuracies, label='Train Accuracy', marker = 'o')
plt.plot(tree_depths, val_accuracies, label='Validation Accuracy', marker =
'o')
plt.plot(tree_depths, test_accuracies, label='Test Accuracy', marker = 'o')
plt.xlabel('Tree Depth')
plt.ylabel('Accuracy')
plt.title('Accuracy vs. Tree Depth')
plt.legend()
plt.grid()
plt.show()
```



According to the plot Yes, the depth of the tree with the maximum validation accuracy coincides with the depth of the tree for the maximum test accuracy. Both the validation and test accuracies peak at a tree depth of approximately 15-20.

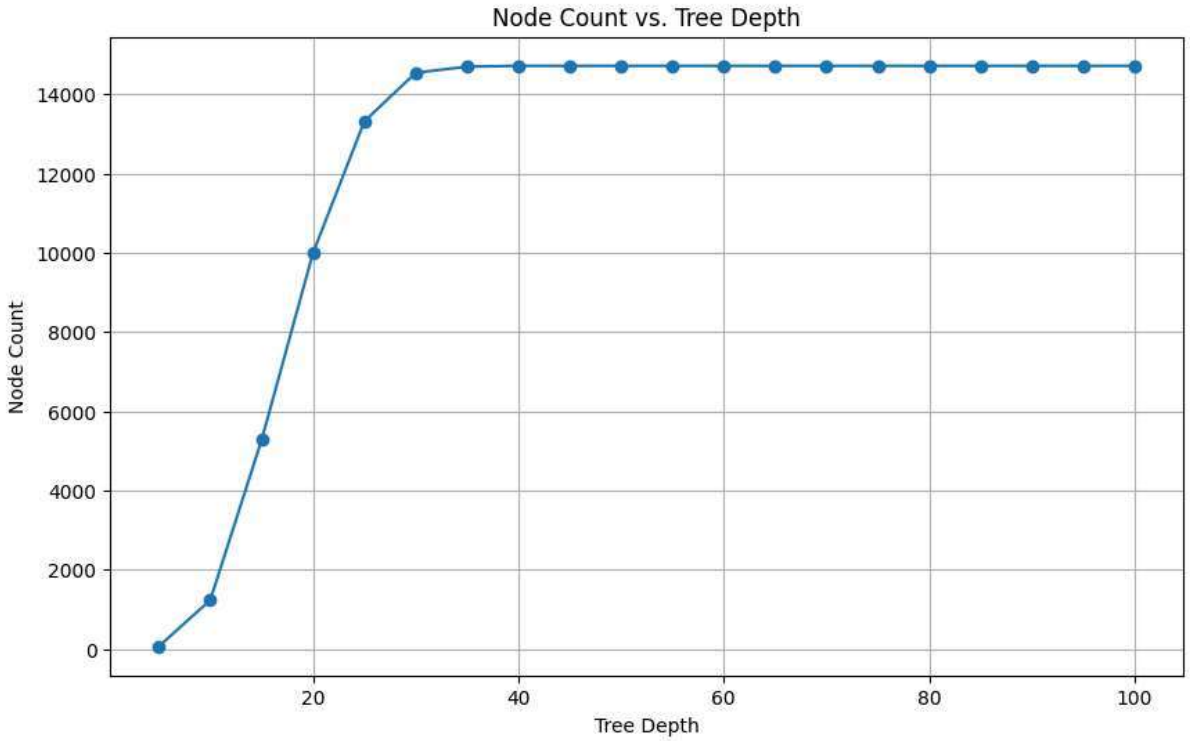
Generate Plots for max_depth

In [86]:
```
# Initialize lists to store tree attributes
max_depths = []
node_counts = []
# Extract attributes for each tree
for tree in models:
  max_depths.append(tree.tree_.max_depth)
  node_counts.append(tree.tree_.node_count)

# Plot node count of each tree
plt.figure(figsize=(10, 6))
plt.plot(tree_depths, max_depths, marker='o')
plt.xlabel('Tree Depth')
plt.ylabel('Max Depth')
plt.title('Max Depth vs. Tree Depth')
plt.grid()
plt.show()
```



Generate Plots for node_count

In [81]:
```
# Plot node count of each tree
plt.figure(figsize=(10, 6))
plt.plot(tree_depths, node_counts, marker='o')
plt.xlabel('Tree Depth')
plt.ylabel('Node Count')
plt.title('Node Count vs. Tree Depth')
plt.grid()
plt.show()
```



It is unlikely that growing trees with a max-depth greater than 100 will improve test accuracy. Both validation and test accuracies peaked around a depth of 15-20 and did not improve with deeper trees.