Name: Bosheng Li (li2343@purdue.edu)
PUID: 0028946785

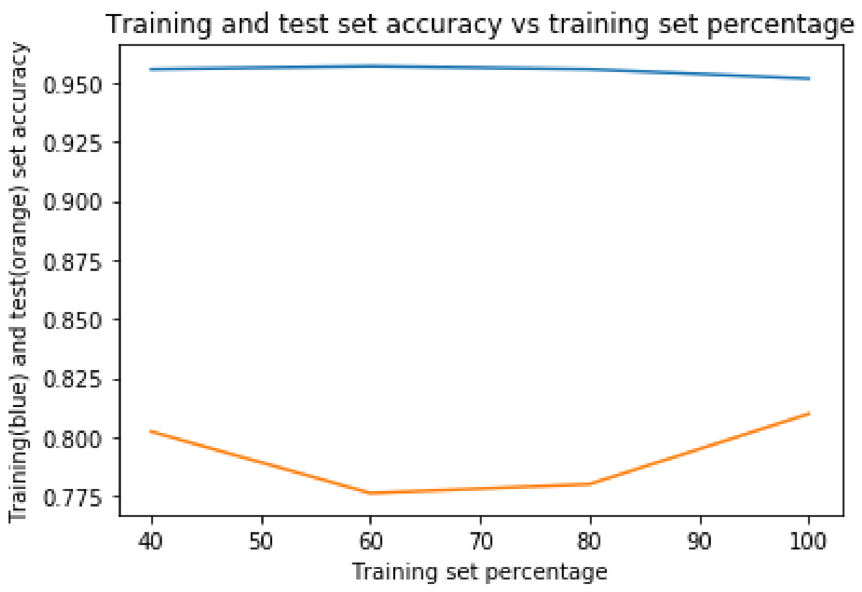
Part 2 Analysis

1. Plot a graph of test set accuracy and training set accuracy against training set percentage on the same plot.

   Script:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plot
xaxis = [40, 60, 80, 100]
train_yaxis = [0.9558, 0.9571, 0.9558, 0.9518]
test_yaxis = [0.8022, 0.7761, 0.7799, 0.8097]
df=pd.DataFrame({'x': xaxis, 'train': train_yaxis, 'test': test_yaxis})
plot.plot('x', 'train', data=df)
plot.plot('x', 'test', data=df)
plot.title("Training and test set accuracy vs training set percentage")
plot.xlabel("Training set percentage")
plot.ylabel("Training(blue) and test(orange) set accuracy")
plot.show()
```

   Graph:

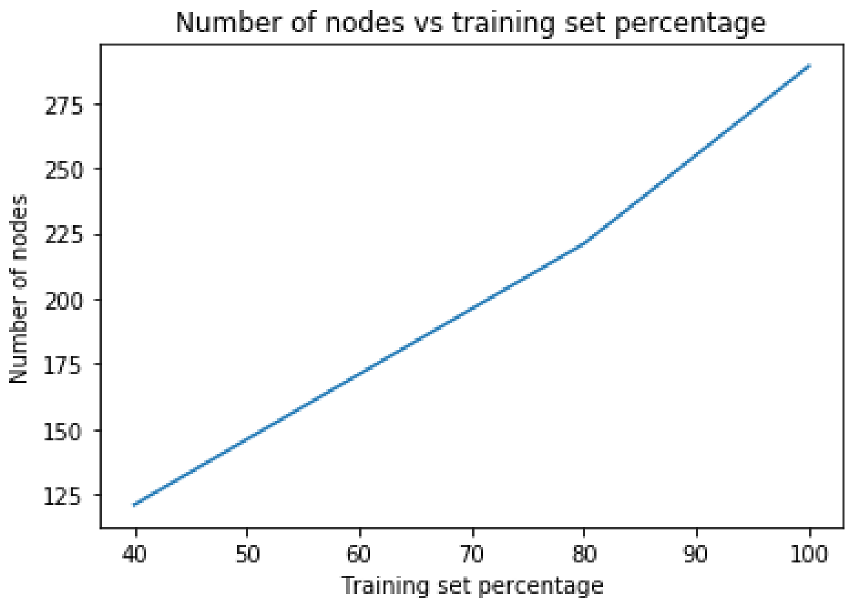


   Plot another graph of number of nodes vs training set percentage.

   Script:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plot
xaxis = [40, 60, 80, 100]
train_yaxis = [121, 171, 221, 289]
df=pd.DataFrame({'x': xaxis, 'train': train_yaxis})
plot.plot('x', 'train', data=df)
plot.title("Number of nodes vs training set percentage")
plot.xlabel("Training set percentage")
plot.ylabel("Number of nodes")
plot.show()
```

Graph:



QUESTION 2 IS ON THE NEXT PAGE

2. For the training set percentage, I have the following results:

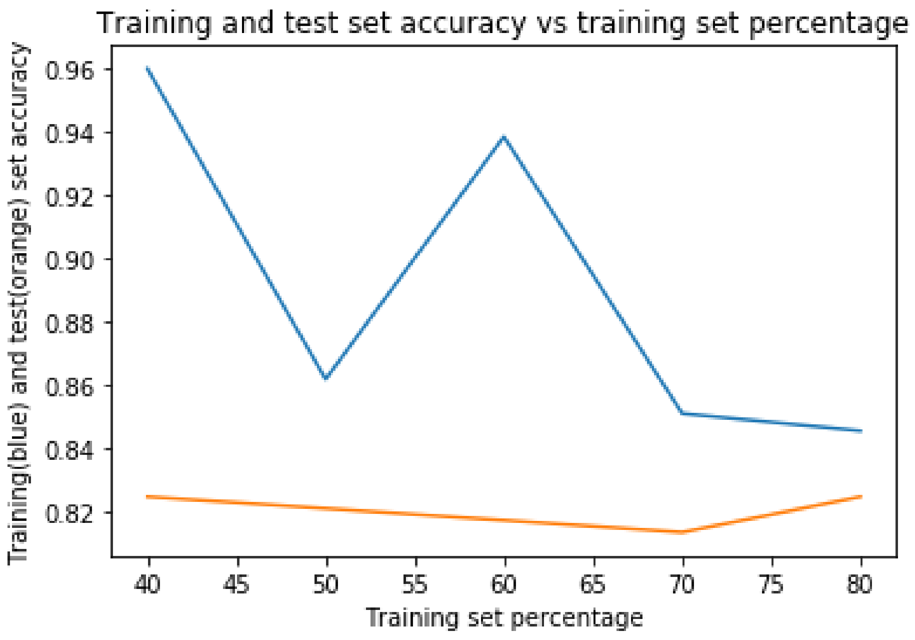
If the training set percentage is 40%, the best maximum depth is 10 (0.7840, 0.7920, 0.7440, 0.7440)
If the training set percentage is 50%, the best maximum depth is 5 (0.8160, 0.8080, 0.7680, 0.7600)
If the training set percentage is 60%, the best maximum depth is 10 (0.8000, 0.8160, 0.7840, 0.7840)
If the training set percentage is 70%, the best maximum depth is 5 (0.8000, 0.7680, 0.7520, 0.7200)
If the training set percentage is 80%, the best maximum depth is 5 (0.7920, 0.7440, 0.7360, 0.7360)

Plot a graph of test set accuracy and training set accuracy against training set percentage on the same plot.

Script:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plot
xaxis = [40, 50, 60, 70, 80]
train_yaxis = [0.9598, 0.8617, 0.9383, 0.8509, 0.8454]
test_yaxis = [0.8246, 0.8209, 0.8172, 0.8134, 0.8246]
df=pd.DataFrame({'x': xaxis, 'train': train_yaxis, 'test': test_yaxis})
plot.plot('x', 'train', data=df)
plot.plot('x', 'test', data=df)
plot.title("Training and test set accuracy vs training set percentage")
plot.xlabel("Training set percentage")
plot.ylabel("Training(blue) and test(orange) set accuracy")
plot.show()
```

Graph:

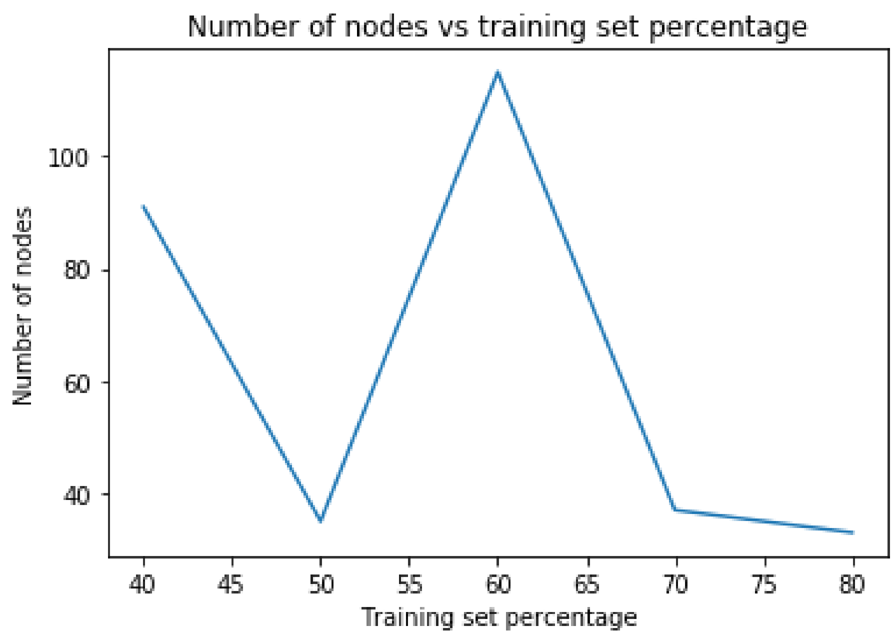


Plot another graph of number of nodes vs training set percentage.

Script:

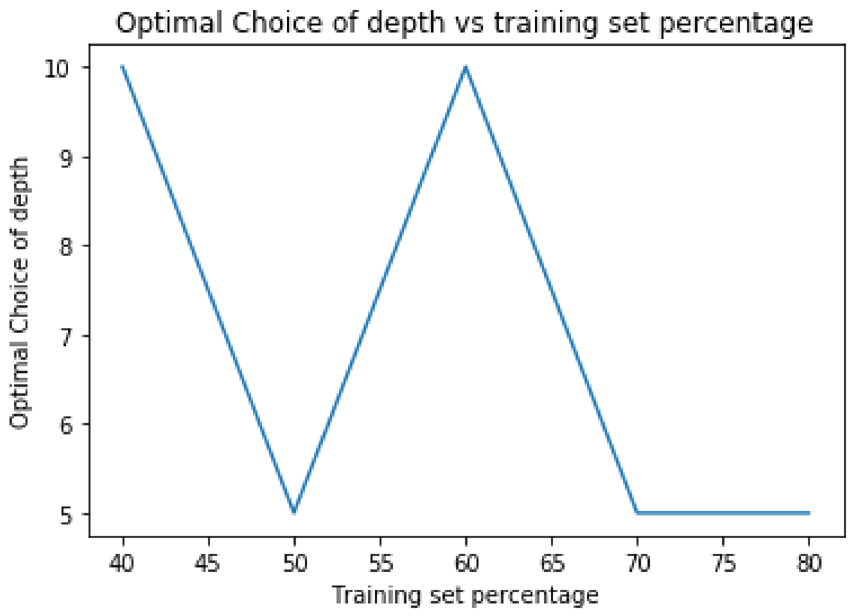
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plot
xaxis = [91, 35, 115, 37, 33]
train_yaxis = [40, 50, 60, 70, 80]
df=pd.DataFrame({'train': xaxis, 'x': train_yaxis})
plot.plot('x', 'train', data=df)
plot.title("Number of nodes vs training set percentage")
```

```
        plot.xlabel("Training set percentage")
        plot.ylabel("Number of nodes")
        plot.show()
```
Graph:



Finally, plot the optimal choice of depth against the training set percentage.

Script:
```
        import numpy as np
        import pandas as pd
        import matplotlib.pyplot as plot
        xaxis = [10, 5, 10, 5, 5]
        train_yaxis = [40, 50, 60, 70, 80]
        df=pd.DataFrame({'train': xaxis, 'x': train_yaxis})
        plot.plot('x', 'train', data=df)
        plot.title("Optimal Choice of depth vs training set percentage")
        plot.xlabel("Training set percentage")
        plot.ylabel("Optimal Choice of depth")
        plot.show()
```
Graph:

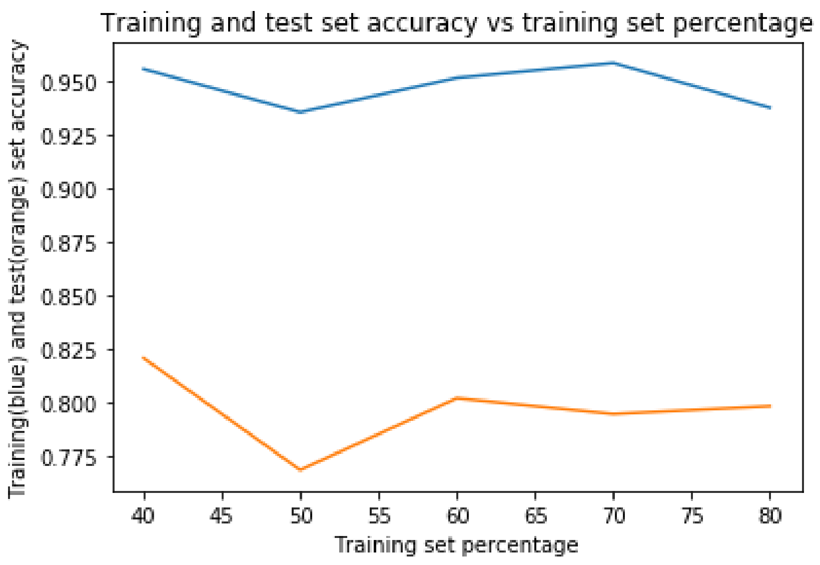


QUESTION 3 IS ON THE NEXT PAGE

3. Plot a graph of test set accuracy and training set accuracy against training set percentage on the same plot.
   Script:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plot
xaxis = [40, 50, 60, 70, 80]
train_yaxis = [0.9558, 0.9357, 0.9517, 0.9587, 0.9378]
test_yaxis = [0.8209, 0.7687, 0.8022, 0.7948, 0.7985]
df=pd.DataFrame({'x': xaxis, 'train': train_yaxis, 'test': test_yaxis})
plot.plot('x', 'train', data=df)
plot.plot('x', 'test', data=df)
plot.title("Training and test set accuracy vs training set percentage")
plot.xlabel("Training set percentage")
plot.ylabel("Training(blue) and test(orange) set accuracy")
plot.show()
```

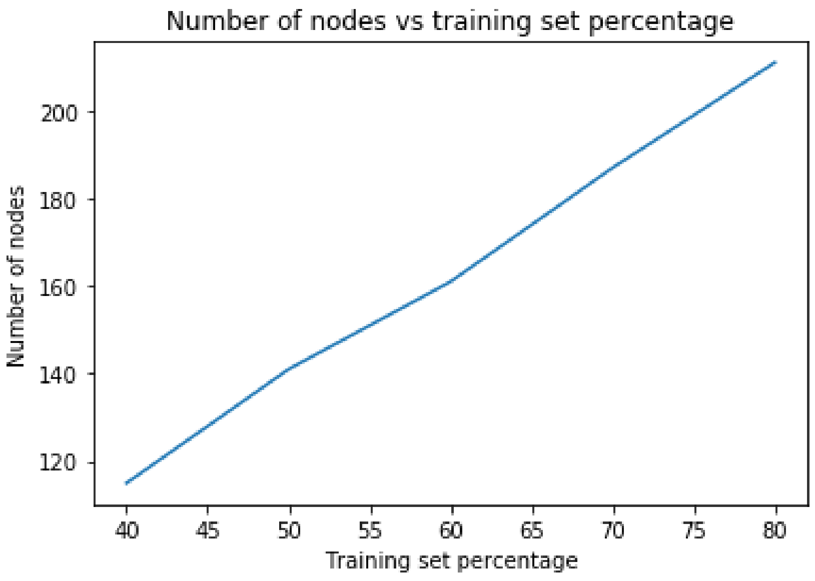
   Graph:



   Plot another graph of number of nodes vs training set percentage.
   Script:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plot
xaxis = [40, 50, 60, 70, 80]
train_yaxis = [115, 141, 161, 187, 211]
df=pd.DataFrame({'x': xaxis, 'train': train_yaxis})
plot.plot('x', 'train', data=df)
plot.title("Number of nodes vs training set percentage")
plot.xlabel("Training set percentage")
plot.ylabel("Number of nodes")
plot.show()
```

   Graph:



<u>QUESTION 4 AND 5 IS ON THE NEXT PAGE</u>

4. The goal of pruning is to reduce the overfitting problem caused by the noise in the data set during the learning process. The test set is the data for testing, and we shouldn’t be able to use it during the learning process. If we use pruning on the data set, then we would mistakenly fit our model directly to the test data and we won’t get a good measure of our model performance on the testing data.

5. In my current model, the new label in the pruning process is determined by majority vote. If the list of label consists multi-values, we can use the classified labels (as different ranks) instead of using the majority vote. If the data is labeled with continuous variables, we can divide them in to several intervals as the rank.