CA03 – Decision Tree Algorithm

Data Source and Contents

The dataset is obtained from the Census Bureau and represents salaries of people along with seven demographic variables. The following is a description of our dataset:

- **Number of target classes:** 2 ('>50K' and '<=50K') [Labels: 1, 0]
- **Number of attributes (Columns):** 7
- **Number of instances (Rows):** 48,842

The data is provided in a .csv file at a Github location. Use the following GitHub link in your Python code to "read" the data into a Data-frame.

```
url = "https://github.com/ArinB/MSBA-CA-Data/blob/main/CA03/census_data.csv?raw=true"
data = pd.read_csv(url, encoding = "ISO-8859-1")
```

NOTE: Use the above EXACT URL in your code as data file location

There is a column indicating the rows to be used as "Training Data" and "Testing Data". You can split the files based on this column value. Note that the "continuous" data columns have been "transformed" into certain "data groups" or "data blocks". This technique is called "**binning**" and by this process you can transform "continuous" data to "discrete categories", because for certain predictive applications like this, "discrete categories" makes more sense than the exact numerical value in the continuous scale. It's also called "discretization" of data. **Investigate the data and answer the following question in your assignment** submission. (Probably, you will be able to answer these questions better AFTER you have completed the rest parts of the assignment.)

Q.1.1 Why does it makes sense to discretize columns for this prediction problem?

Q.1.2 What might be the issues (if any) if we DID NOT discretize the columns.

PART 1

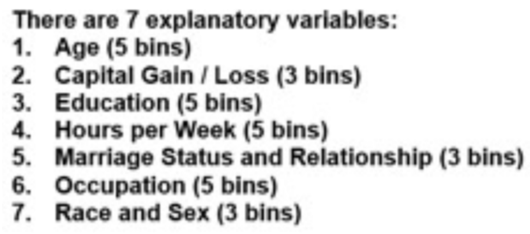
1. Data Quality Analysis (DQA)

Do all of these inside your Notebook:

- Perform a Data Quality Analysis to find missing values, outliers, NaNs etc.
- Display descriptive statistics of each column
- Perform necessary data cleansing and transformation based on your observations from the data quality analysis

2. Exploratory Data Analysis (EDA)

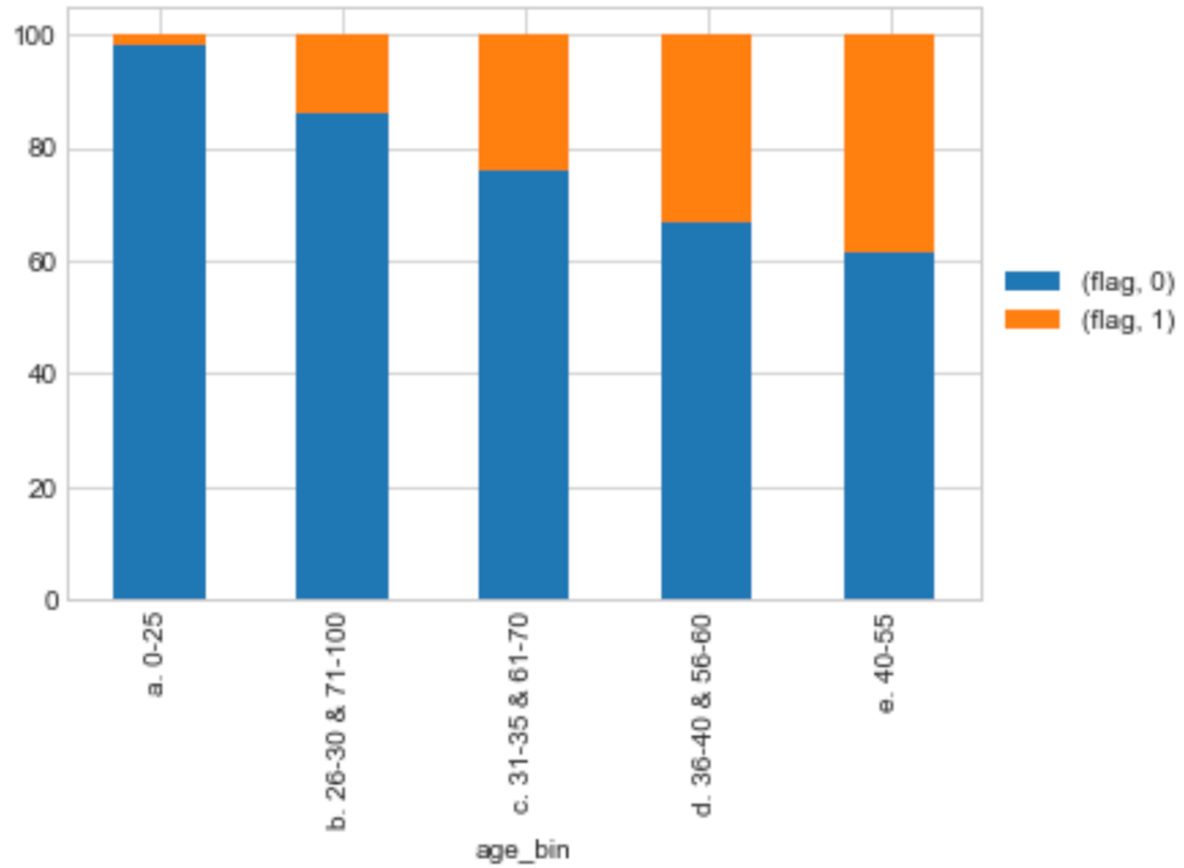
First, understand the data. Open the .csv file in an Excel spreadsheet and study the data columns and the data visually. Then, perform EDA (in your Notebook) of the income group with respect to the seven explanatory variables and display graphical representations as shown below. Before you do the EDA, you need to do transform the data into “categorical” columns by doing the “data binning” with respect to SEVEN explanatory variables and their specified number of “bins”. You will also find the “bin” value ranges for each of the seven variables in the graphs. If your EDA data transformation and graphing code is correct the output of your EDA will look like the graphs displayed below. Do all of these inside your Notebook. (Hint: Use Stacked Bar Chart options in MatPlotLib).



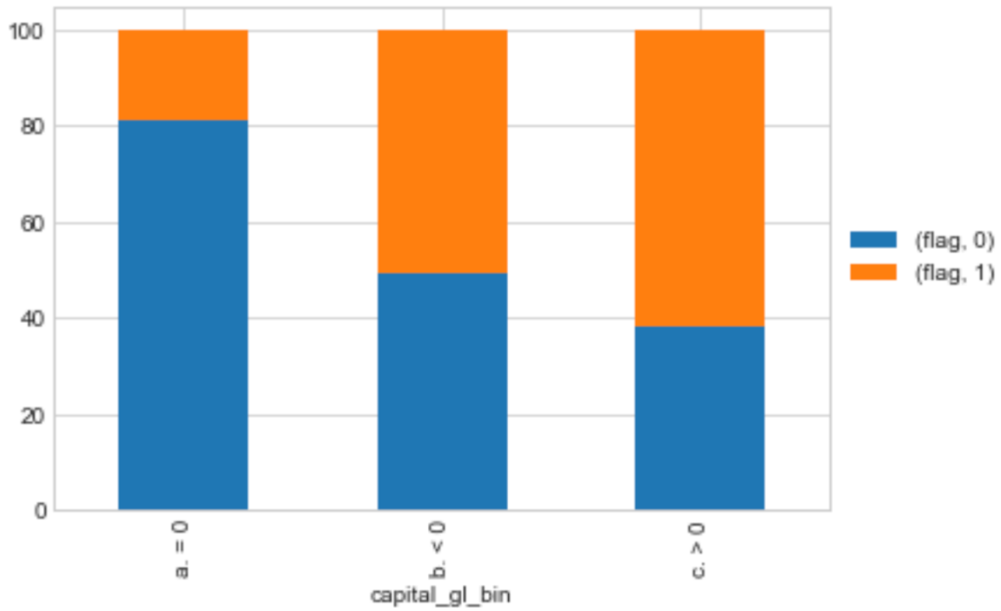
There are 7 explanatory variables:

1. **Age (5 bins)**
2. **Capital Gain / Loss (3 bins)**
3. **Education (5 bins)**
4. **Hours per Week (5 bins)**
5. **Marriage Status and Relationship (3 bins)**
6. **Occupation (5 bins)**
7. **Race and Sex (3 bins)**

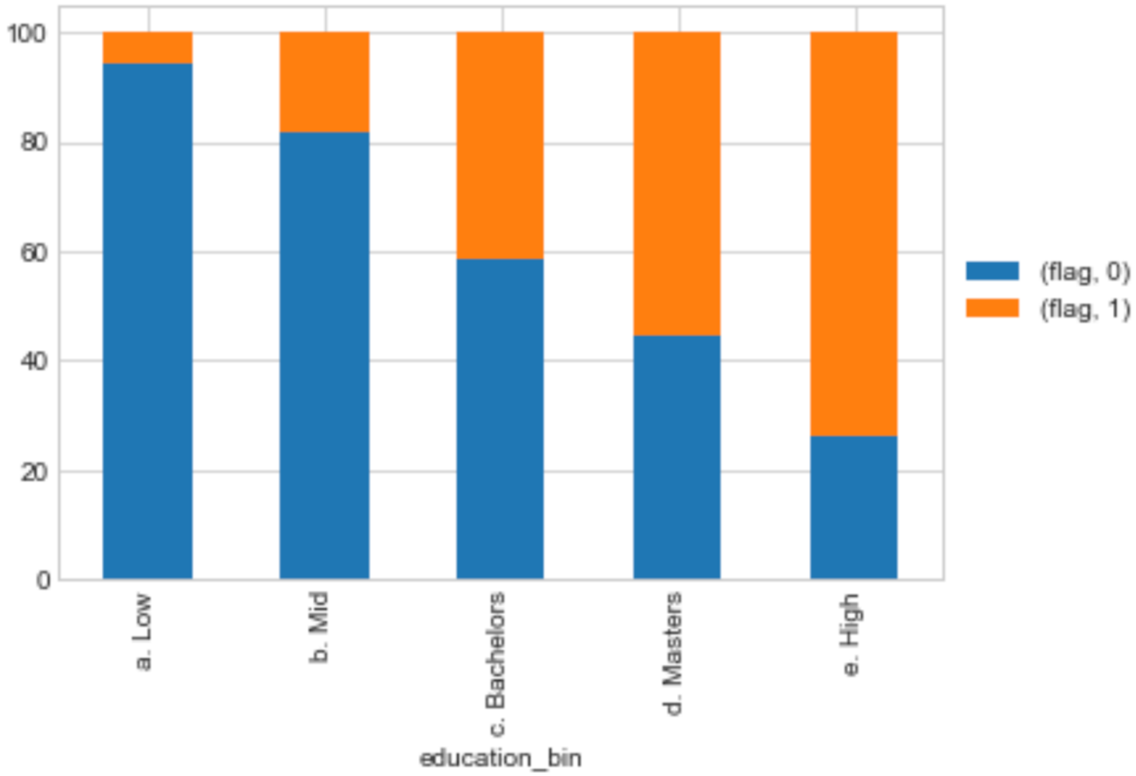
Age Bin:



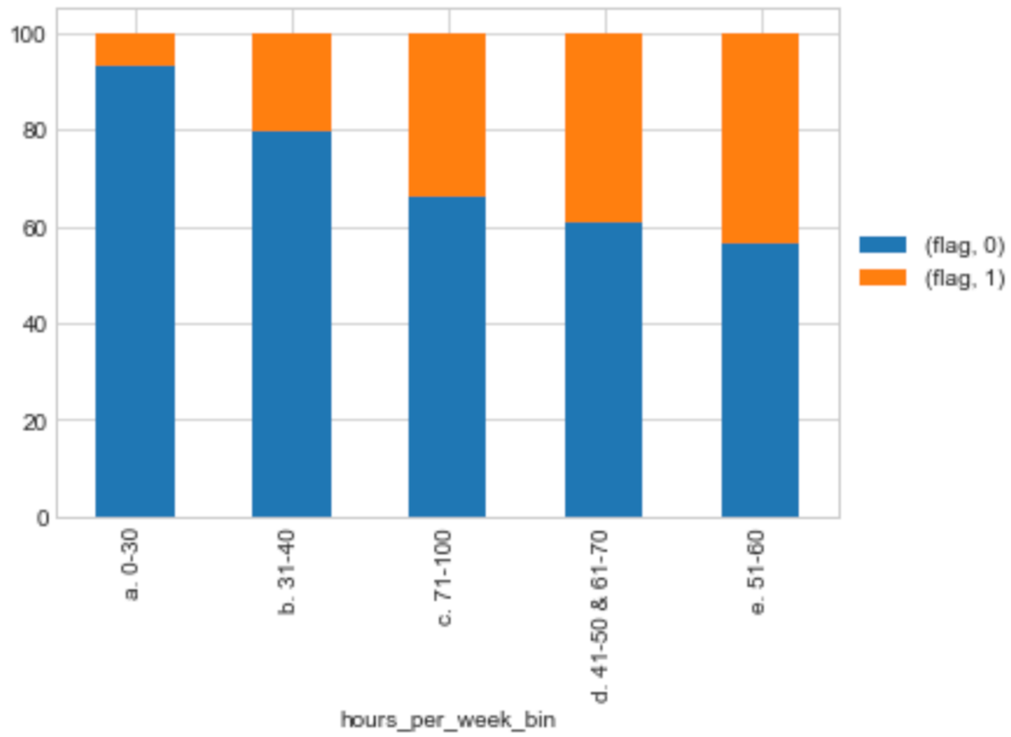
Capital Gain Bin:



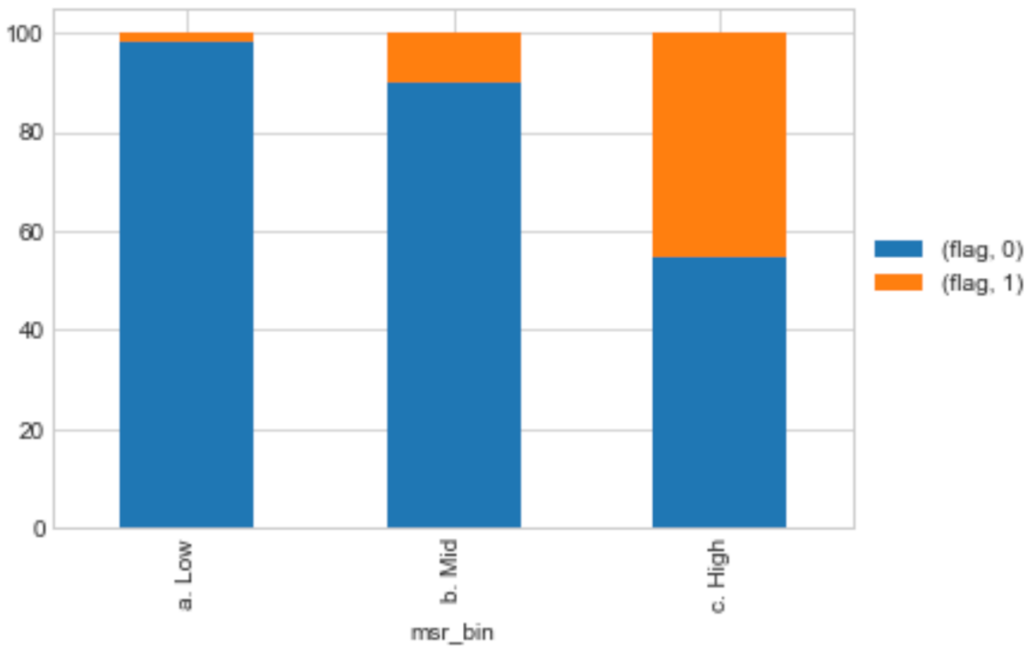
Education Bin (Use the “Education” column and NOT the “Education_Number” column):



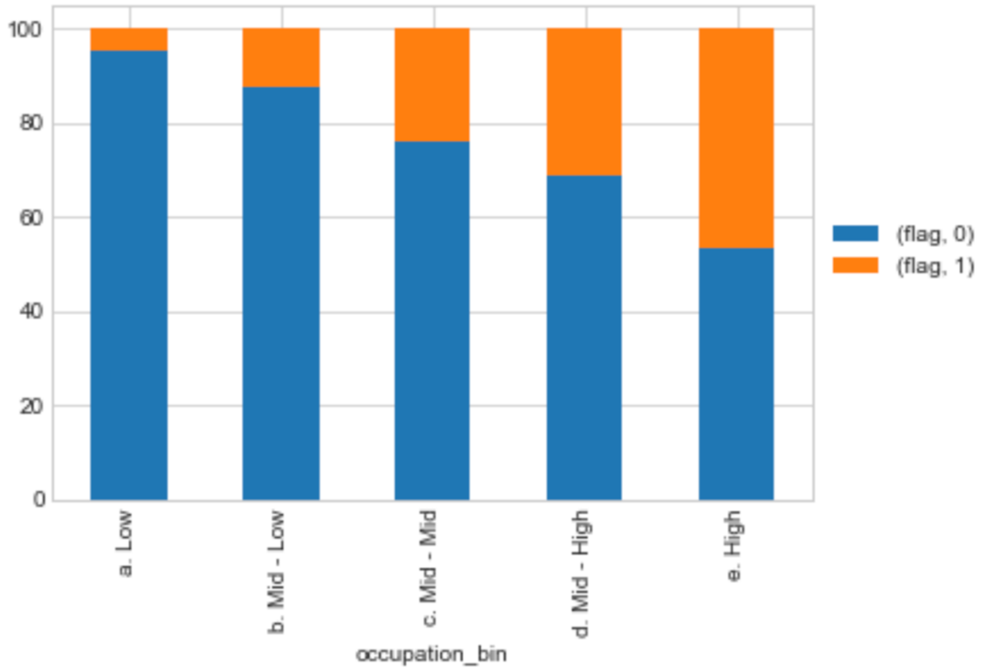
Hours per Week Bin:



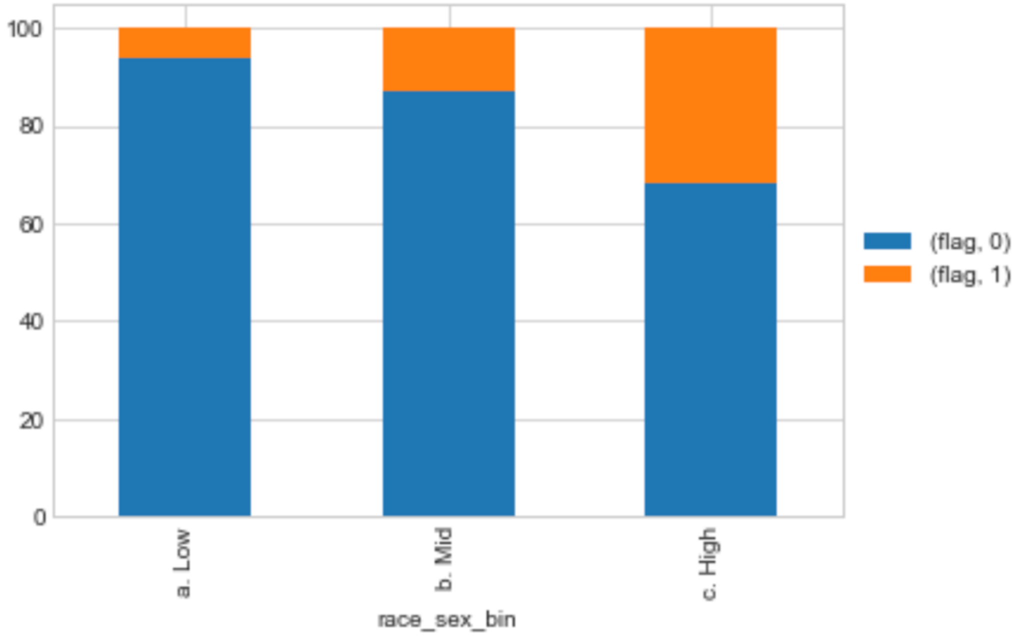
Marriage Status and Relationship Bin:



Occupation Bin



Race and Sex Bin:



PART 2

3. Build Decision Tree Classifier Models

Definition: Given a data of attributes together with its classes, a decision tree produces a sequence of rules that can be used to classify the data.

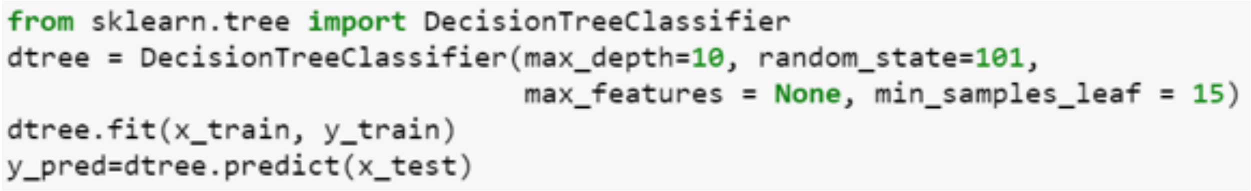
Advantages: Decision Tree is simple to understand and visualize, requires little data preparation, and can handle both numerical and categorical data.

Disadvantages: Decision tree can create complex trees that do not generalize well, and decision trees can be unstable because small variations in the data might result in a completely different tree being generated.

Use “DecisionTreeClassifier” algorithm from scikit learn. Find details of sklearn tree algorithm below. Scitkit Learn implements an optimized version of CART algorithm and can be used for binary class as well as multi-class classifications. It can be used for classification, as well as regression. Study the following link thoroughly, including Section 1.10.5 (Tips on Practical Use).

https://scikit-learn.org/stable/modules/tree.html#tree-algorithms-id3-c4-5-c5-0-and-cart

Syntax to use the classifier is shown below:



```
from sklearn.tree import DecisionTreeClassifier
dtree = DecisionTreeClassifier(max_depth=10, random_state=101,
                               max_features = None, min_samples_leaf = 15)
dtree.fit(x_train, y_train)
y_pred=dtree.predict(x_test)
```

random_state: This is a number you choose arbitrarily. It’s also called “Random Seed”. If you use a number for this parameter (any number), it ensures that if you run the program multiple times, it will generate the same randomness. Hence, the solution becomes more “reproduceable”.

4. Visualize Your Decision Tree using GraphViz

Get the detail of how to do this from the following link:

https://medium.com/@rnbrown/creating-and-visualizing-decision-trees-with-python-f8e8fa394176

5. Evaluate Decision Tree Performance

Calculate and display the following. Do all of these inside your Notebook.

- Confusion Matrix (TP, TN, FP, FN … etc.)
- Accuracy, Precision, Recall, F1 Score, AUC Value, ROC Curve (graph)

PART 3

6. Tune Decision Tree Performance

Learn about all hyper-parameters and methods of Scikit Learn DecisionTreeClassifier algorithm at:
https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html#sklearn.tree.DecisionTreeClassifier

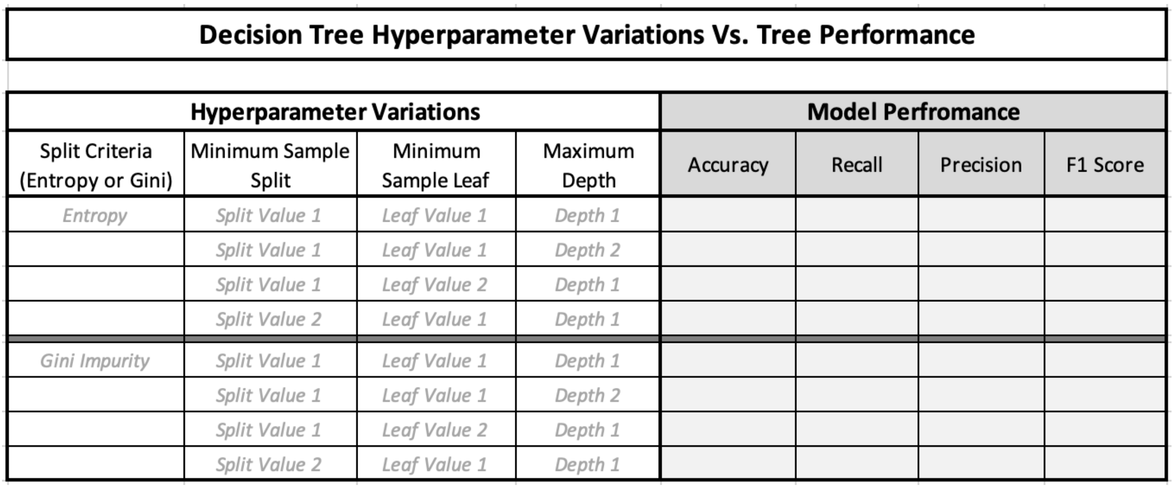
Try varying FOUR of the hyperparameters **manually** (hard coded), as per the following table, and train / score your model for each set of these hyperparameters. Record your Tree's performance with respect to each of these sets of hyperparameters in the Model Performance section of the following table.

Four Hyperparameters to vary:

1. Split Criteria – 'Entropy' or 'Gini Impurity'
2. Minimum Sample Split – Minimum number of records required in any node for a further split to be attempted
3. Minimum Sample Leaf – Minimum of samples in a leaf node to stop further splitting (becomes a leaf node)
4. Maximum Depth – Maximum depth of the tree allowed

Based on the ranges of values you find in your Decision Tree for these hyperparameters, vary them by manually choosing your own reasonable values and manually running the program for each of the Hyper-parameter combinations. An Excel file for this table is provided (Tree Tuning Cases.xlsx). You can enter the values in the given Excel File (Tree Tuning Cases.xlsx) and cut/paste the table in your submission document.

Q.7.1 Decision Tree Hyper-parameter variation vs. performance (run your program manually for the following eight cases and enter the Model Performance values manually in the table)



Decision Tree Hyperparameter Variations Vs. Tree Performance

Hyperparameter Variations				Model Perfromance			
Split Criteria (Entropy or Gini)	Minimum Sample Split	Minimum Sample Leaf	Maximum Depth	Accuracy	Recall	Precision	F1 Score
Entropy	*Split Value 1*	*Leaf Value 1*	*Depth 1*				
	Split Value 1	*Leaf Value 1*	*Depth 2*				
	Split Value 1	*Leaf Value 2*	*Depth 1*				
	Split Value 2	*Leaf Value 1*	*Depth 1*				
Gini Impurity	*Split Value 1*	*Leaf Value 1*	*Depth 1*				
	Split Value 1	*Leaf Value 1*	*Depth 2*				
	Split Value 1	*Leaf Value 2*	*Depth 1*				
	Split Value 2	*Leaf Value 1*	*Depth 1*				

7. Conclusion

Explain your observations from the above performance tuning effort.

Q.8.1 How long was your total run time to train the model?
Q.8.2 Did you find the BEST TREE?
Q.8.3 Draw the Graph of the BEST TREE Using GraphViz
Q.8.4 What makes it the best tree?

PART 4

8. Automation of Performance Tuning

Instead of running your program manually eight times for eight cases, can you "automate" generation of the above "Hyperparameter Vs. Model Performance Table" in your code itself, so that the completed performance table can be displayed with all eight cases by running it only once?

(**Hint**: *You can enter the "Hyperparameter Variation" part of the table manually in an Excel file, save it as CSV with header and read it in a dataframe in Python to repeatedly train/score the model for each of the lines of hyperparameters dataframe.)*

PART 5

9. Prediction using your "trained" Decision Tree Model

Based on the Performance Tuning effort in the previous section, pick your BEST PERFORMING TREE. Now make prediction of a "new" individual's Income Category (<=50K, or >50K) with the following information. Do this in your Notebook.

- Hours Worked per Week = 48
- Occupation Category = Mid - Low
- Marriage Status & Relationships = High
- Capital Gain = Yes
- Race-Sex Group = Mid
- Number of Years of Education = 12
- Education Category = High
- Work Class = Income
- Age = 58

Q.10.1 What is the probability of the outcome of the prediction for this ? What is your decision probability threshold and what is your predicted decision based on that?

Q. 10.2 What is the probability that your outcome prediction is accurate?

10. Deliverables

Your assignment outputs will have the following components:

(1) Fully functional Notebook, Data, Readme file in a single folder at GitHub (do you know how to create folders under your repository in GitHub?)
(2) Upload your fully working .ipynb file at BrightSpace CA03 submission folder
(3) Upload your Word / PDF document with answers to all questions marked Q.1, Q.2 …. etc. at BrightSpace CA03 submission folder