Q1

We perform data preprocessing in 3 ways in this assignment.

Initially, we utilized one hot encoding to convert the string-based sex and region columns into float values. Converting each categorical value into a binary vector--one in which every element represents a possible value of the categorical variable--occurs through one hot encoding. This process enables the decision tree to operate with categorical variables as though they were numerical ones; consequently, it enhances model accuracy.

Next, we employ the function `encode\_output()` to transform the output variable 'smoker' from string values ('yes' and 'no') into binary digits (specifically 1 and 0). We can predict whether an individual is a smoker or not based on their distinct characteristics in a decision tree; this becomes possible when we convert our output variable into binary values.

Finally, consider the ensuing lines of code:

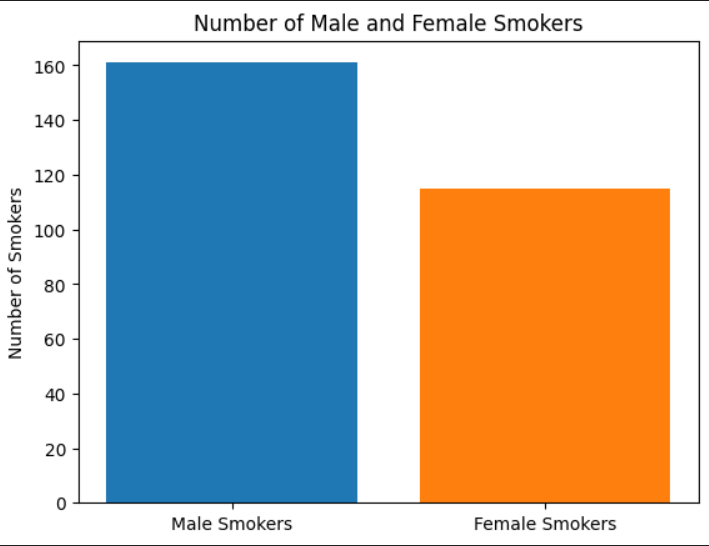
encoded\_df2.dropna(inplace=True) #Drop NA Vals

missing\_values\_count = encoded\_df2.isnull().sum()

missing\_values\_count[0:10]

Here, we are handling missing values in the dataset, and the lines of code execute data preprocessing. The initial line--`encoded\_df2.dropna(inplace=True)`--eliminates all rows with missing values: a step that guarantees completeness and accuracy of the dataset. Subsequently; through `missing\_values\_count = encoded\_df2.isnull().sum()`, it quantifies each column's unavailability within our set—this assists us in discerning which columns necessitate additional processing or imputation. The third line of code: `missing\_values\_count[0:10]` actively prints the number of missing values in the dataset's initial ten columns; this action swiftly identifies any potentially problematic columns--those with a significant number of missing values.

Q2

First, we create a bar plot illustrating the count of male and female smokers within our dataset. This process involves filtering the DataFrame to exclusively incorporate rows with 'male' or 'female' under the sex column; subsequently, it counts how many rows have 'yes' in their respective smoker columns. Finally--on completion: an x-axis displaying gender distribution amongst smokers (both male and female), while a y-axis represents overall number—is used for generating this graphical representation via Matplotlib's bar chart function. Upon examination--the number of male smokers appears as 160, while around 120 represent their female counterparts. However, we cannot definitively establish a strong correlation here: factors such as potential dataset skewing and insignificant numerical disparities may limit our conclusions.

Next, we make a bar plot that shows the number of smokers in each dataset region. This again shows similar values, with the southeast showing the highest number of smokers. Both the East coasts have a high number of smokers, showing some correlation.

A graph of smokers by region

Description automatically generated

Finally, we plot the number of children against the number of smokers. This shows a very strong correlation that the more kids a person has, the less likely they are to be a smoker. This is visible very strongly as the people without kids are nearly double in number than the people with no kids.

A graph of smokers

Description automatically generated

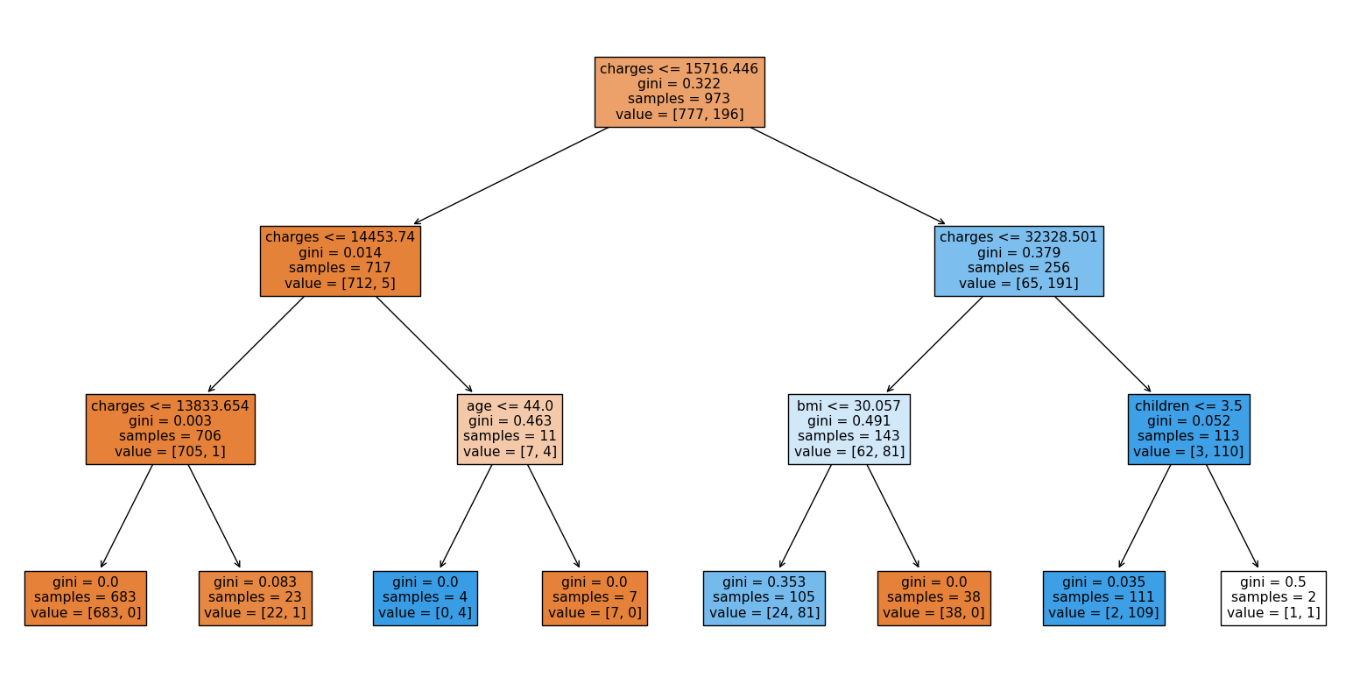
Q3

We followed a few steps to explore the dataset in this particular file using a decision tree: First, we leveraged the `read\_csv()` function; through it, we loaded preprocessed data into a pandas DataFrame. Subsequently – and with further precision – utilizing another function known as `one\_hot\_encode()`, we performed one-hot encoding of two specific columns ('sex' and 'region') within that DataFrame.

Split the data into training and testing sets using the `train\_test\_split()` function from the `sklearn.model\_selection` module. Then, utilize the 'DecisionTreeClassifier()' class from the 'sklearn.tree' module to construct a decision tree on our training set.

After building the decision tree, we evaluated its performance on the testing set; our assessment employed metrics including accuracy, precision, recall and F1 score. We employed `export\_graphviz()` function from 'sklearn.tree' module to generate a Graphviz representation of the tree for visualization: this allowed us to view–in clear detail–the constructed decision tree.

Question 4:



The classification report suggests that the hyperparameters `max\_depth=3`, `min\_samples\_leaf=2`, and `min\_samples\_split=2` perform optimally for this case. The decision tree classifier, with an accuracy score of 0.95, accurately classified 95% of the testing set samples; its high precision and recall rates further validate its ability to identify positive and negative samples with exceptional accuracy.

The charges perform the broadest characterization, considering the BMI, children and age. Both the root and first child nodes solely base their considerations on these charges, thereby presenting a general indicator of an individual's smoking status.

Question 5:

Decision trees serve as a good tool for exploratory data analysis; they visualize and interpret complex relationships between variables as we can see in the plotted tree. Moreover, their utility lies in identifying the most pivotal features to predict the target variable along with subgrouping data that share similar characteristics. We can gain insights into relationship patterns by examining nodes and branches of the decision tree: it reveals associations between input variables–output variables and unobvious data correlations from a simple analysis perspective.

A good example of this is the charges column here, as at first glance it might not seem like a very strong indicator, but after observing the plot, we can see that it is infact the root of the tree. These interesting relationships only came to light after employing the use of the tree itself.

Through employing decision trees for exploratory data analysis; we not only identify potential areas warranting further analysis but also cultivate a more comprehensive grasp of the data.