A1: Research Question

I am interested in investigating whether a decision tree model with a boosting method can identify patients with anxiety using the provided data. I have previously tackled similar research questions in D208 Task 2 (Logistic Regression) and D209 Task 1 (K-Nearest Neighbors Classification, or KNN), where I developed models to predict or identify patients with anxiety.

A2: Objectives and Goals of Analysis

For D209 Task 2, I selected a decision tree classification model. The objective is to develop a prediction model with an accuracy and AUC score of 0.50. Recognizing patients at risk for chronic anxiety is key for managing their healthcare, addressing existing issues, and preventing future problems. This goal is vital for the hospital system, as it involves dealing with current challenges and mitigating future ones. Effective and proactive management can enhance patient outcomes, shorten hospital stays, prevent rehospitalizations, and avert potential complications.

B1: Justification of Classification Method

A decision tree works by constructing a hierarchical structure with nodes, each node posing a binary question to classify the data. Depending on the answer, the data is directed either left or right to a subsequent node, which asks another question. This sequence continues until reaching a terminal node, or leaf, which provides the final classification based on previous responses. At the leaf, the tree predicts either 0 (patient does not have anxiety) or 1 (patient does have anxiety). For example, one question might be, "Is the data point's age above the average age of the dataset?"

In comparison to my previous experience with KNN classification, decision trees offer another method for sorting data into different categories. A graph of the data might reveal several decision regions marked as "1" or "0," indicating that decision trees can model the data more accurately than linear models. Decision trees manage imbalanced data well, do not require scaling, and are resistant to outliers (Milena Afeworki, Towards Data Science, 2021). Unlike KNN, decision trees do not heavily depend on clustering, providing a different classification approach.

Furthermore, I can improve my decision tree using Adaboost, which enables the creation of multiple models that learn from one another. I am confident that a boosted decision tree will effectively tackle the classification tasks I have been working on in various projects, assuming this method is viable.

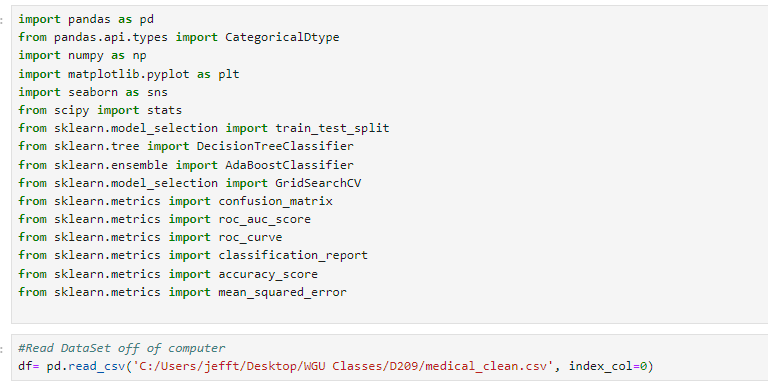
B2: Assumptions of a Decision Tree

Compared to other predictive models, decision trees require few assumptions about their input data. Nonetheless, they inherently assume that the connections between response variables and explanatory variables are simple. Decision trees analyze data points by moving step-by-step through nodes, which split into two possible directions (left or right). As a result, decision trees are limited in their ability to understand or learn complex relationships among variables.

B3: Benefit of Chosen Tools

For my project, I'm relying on Python, a language that's effective for data science due to its libraries..

* **pandas**: Manages the dataset like a large table or spreadsheet.
* **CategoricalDtype (pandas)**: Creates columns that are categorical and ordinal.
* **NumPy**: Performs mathematical operations and assigns values within the dataset.
* **Seaborn and Matplotlib**: Used for generating graphs.
* **sklearn**: Provides essential functions for this analysis.
  + **train\_test\_split**: Splits the dataset into training and testing parts.
  + **DecisionTreeClassifier**: The foundation of our model, used to classify patients as 1 (back pain) or 0 (none).
  + **AdaBoostClassifier**: An ensemble method to enhance the classifier's performance.
  + **GridSearchCV**: Tunes the hyperparameters of the DecisionTreeClassifier.
  + **confusion\_matrix**: Outputs a confusion matrix for the classification model.
  + **roc\_auc\_score**: Calculates the Area Under the Curve (AUC) score for the model.
  + **roc\_curve**: Plots the Receiver Operating Characteristic (ROC) curve of the model.
  + **classification\_report**: Provides a summary of the classification model's metrics.
  + **accuracy\_score**: Measures the model's accuracy.
  + **mean\_squared\_error**: An evaluation method required by the project rubric.



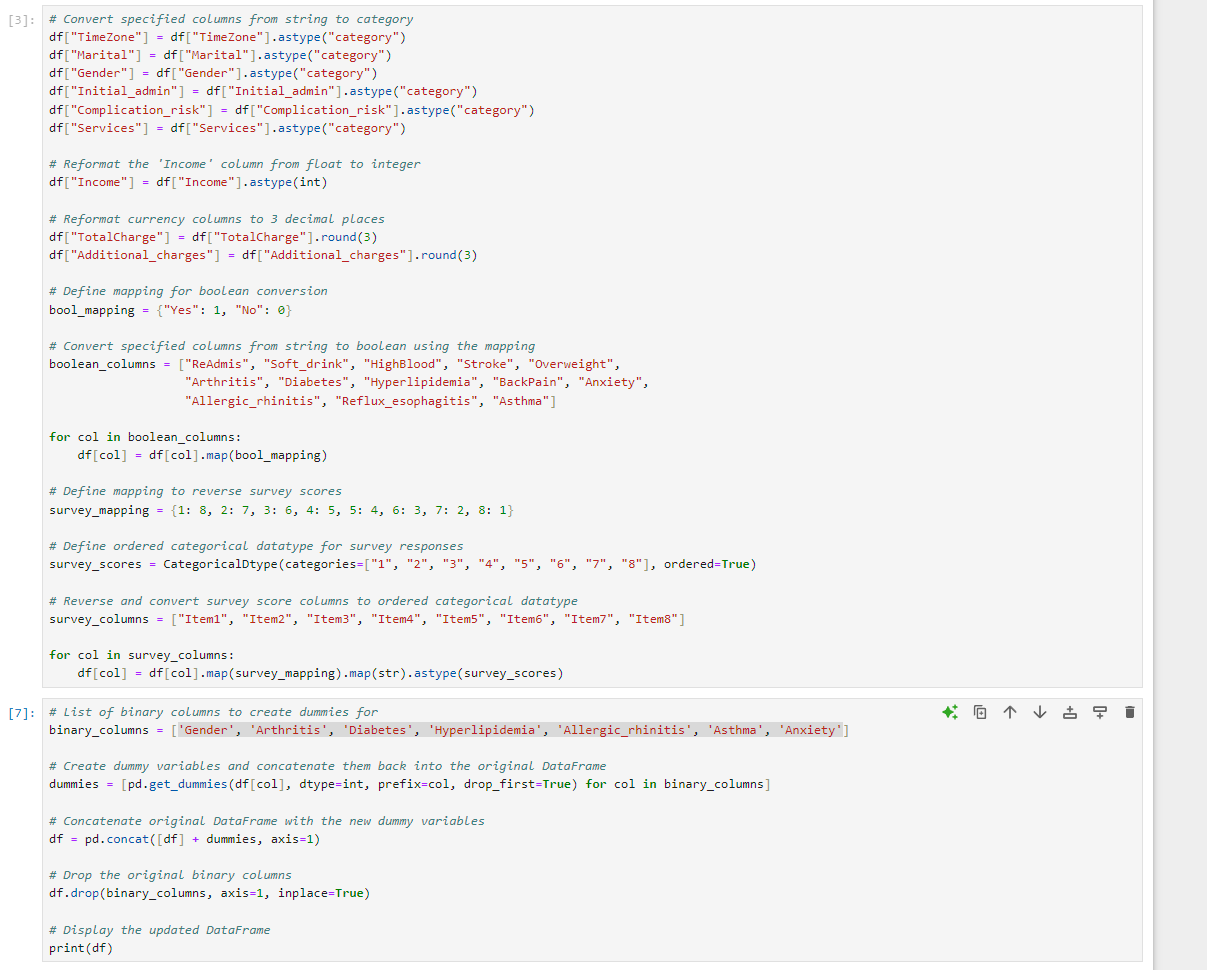
C1 Data Preparation Goals and necessary Manipulation

Preparing data for decision tree classification involves one hot encoding among other steps. This method translates categorical data into numerical values by creating separate columns filled with 1s and 0s. For example, a gender column with Female, Male, and Nonbinary can be split into two new columns: one for Male and one for Nonbinary. If a 1 is in the first column, the patient is male; a 1 in the second means the patient is nonbinary. If both are 0, the patient is female. This can be done using pandas' get\_dummies().

C2:Variable Selection & Identification

* Numerical: Number of Children
* Numerical: Age
* Numericall: Gender
* Numerical: Vitamin D levels
* Numerical: Number of Dr Visits
* Numerical: Full meals
* Categorical: Categorical: Arthritis
* Categorical: Diabetes
* Categorical: Hyperlipidemia
* Categorical: Anxiety
* Categorical: Allergic Rhinitis
* Categorical: Asthma
* Numerical: Initial Days

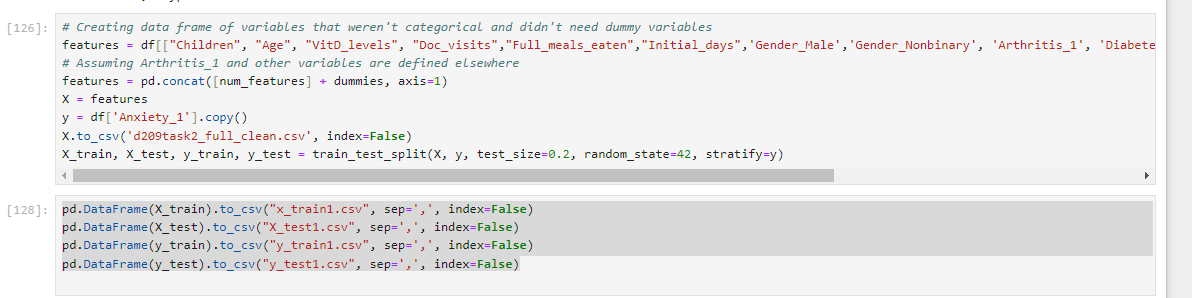
C3:Preparation of Data



C4:Copy of Prepared data set + D1: Data Splitting, Copy of Split Data

A copy of the variables will be in the attached file d209task2\_full\_clean.csv

The Splits also will have their own individual files attached to the submission



D2+D3 :Analysis Description + Analysis Code

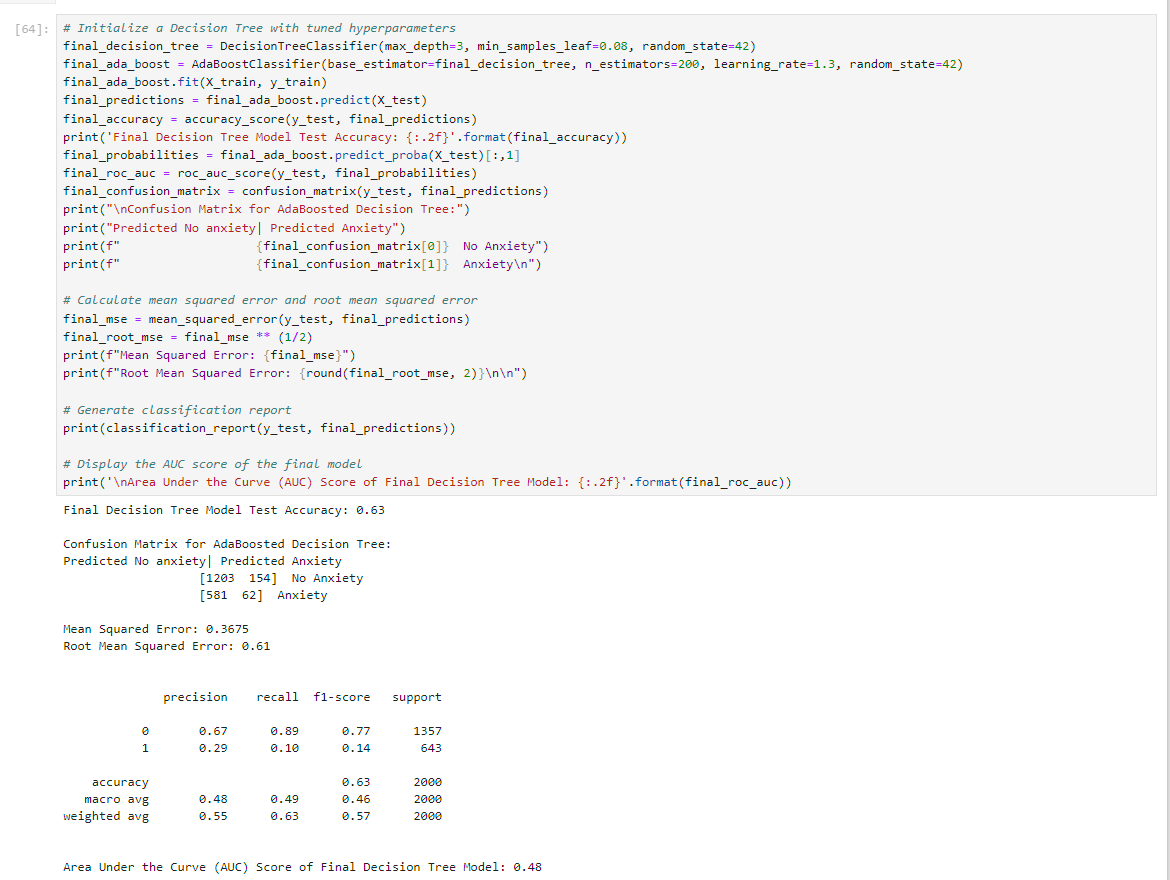
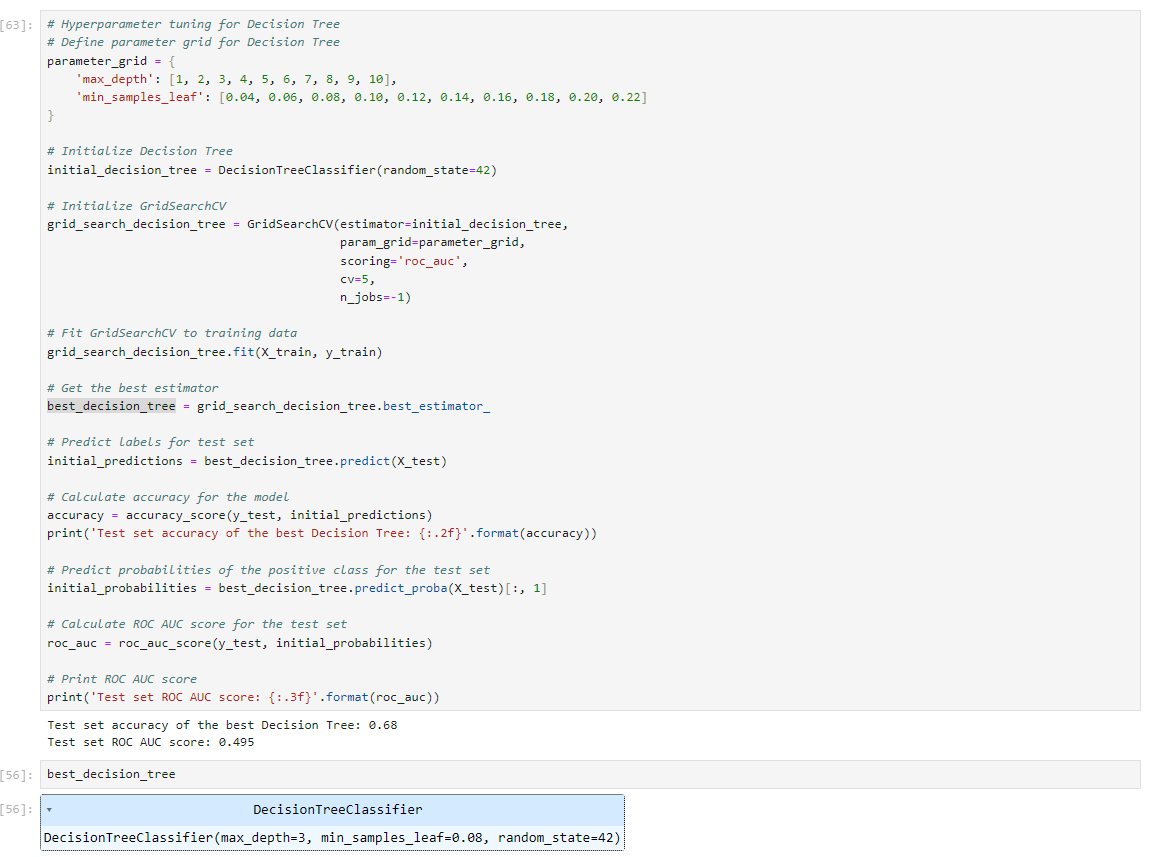
To build an optimal decision tree classification model with the DecisionTreeClassifier, I'll need to carefully choose the best settings, especially for max\_depth and min\_samples\_leaf. The max\_depth parameter controls how deep the tree can grow. While deeper trees can capture more complexity, they also run the risk of overfitting. The min\_samples\_leaf parameter sets the minimum number of samples that must be present in a leaf node, influencing how the nodes can be split.

I'll start by defining an initial DecisionTreeClassifier and listing potential values for max\_depth and min\_samples\_leaf. Then, I'll use GridSearchCV to test the model with all the specified parameters. This grid search will evaluate every possible combination to find the best-performing model. I'll report the accuracy and AUC score for this optimized model as a reference point for further improvements.

Once the grid search has fine-tuned the hyperparameters, I'll create a DecisionTreeClassifier instance using the best parameters. To further enhance the model, I'll apply Adaptive Boosting (AdaBoost). AdaBoost works better with a well-optimized base model, so fine-tuning the decision tree's hyperparameters is crucial for achieving the best possible performance on this dataset.

After optimizing my decision tree with GridSearchCV, I found that it works best with a max depth of 3 and a minimum samples per leaf of 0.08. Despite this, its AUC score of 0.48 is only slightly worse than random,, indicating poor performance. To enhance the model, I turned to AdaBoost, which builds a robust model by combining multiple classifiers that learn from each other's mistakes.

AdaBoost relies on two main parameters: learning rate, which influences each estimator's impact, and the number of estimators, which affects the model's overall complexity and performance. There's a trade-off between these parameters; a higher learning rate assigns more weight to subsequent estimators, while increasing the number of estimators can boost the model's accuracy. I'll use GridSearchCV again to find the best combination of these parameters to maximize the AUC score.



This final tuned and boosted model provides a worse accuracy (0.68) and AUC score (0.63) than the original decision tree did. The result is a classification model that is less effective than the models that I'd previously generated.

E1: Accuracy of the Classification Model

The model reached an accuracy level of 0.63. The confusion matrix shows that out of 2000 observations in the test set, the model correctly predicted 1265 times and incorrectly predicted 735 times. The classification model's mean squared error (MSE) is 0.3675, which is the average squared difference between actual and predicted values. A lower MSE indicates smaller errors, which is better. Nonetheless, MSE isn't suitable for binary classification evaluation. According to Rafay Khan on Towards Data Science, MSE presumes a normal distribution and expects a range from negative to positive infinity, while binary models only produce 0s and 1s. Despite these issues, the MSE is included here for reference.

E2:Model Results

In this graph, the diagonal line in the middle shows a 50% classification rate, which is the result of purely random guessing. A classifier's performance is depicted on this graph, where any curve above the diagonal demonstrates good prediction accuracy, and any curve below it shows poor accuracy, going down to 0% correct at the bottom right corner. The Area Under the Curve (AUC) score is the area under the classifier's curve. A model with perfect accuracy would achieve an AUC of 1.0, while a model with no accuracy would score an AUC of 0.0. This model has an AUC of 0.48, meaning its performance is just below that of random guessing.

E3:Classification Limits

The primary constraint of this analysis lies in the relatively small sample size of 10,000 observations, especially when compared to the overall number of hospitalized individuals in the United States. Predictive models rely on the critical assumption that the sample data is representative of the entire population, but it is unclear whether this dataset meets that requirement. This issue is significant because the model utilizes decision trees, enhanced through adaptive boosting (AdaBoost) to improve performance. Decision trees can be inherently unstable since each data point passing through them might yield different results if data values change (Corporate Finance Institute, 2022). While AdaBoost helps stabilize the model, it requires high-quality data, which means avoiding noisy datasets (Analytics Vidya, 2021). Previous evaluations of this dataset in earlier classes revealed considerable noise, largely due to the varied nature of the data encompassing financial information, survey results, healthcare data, and census data.

E4:Recommended Action

I suggest utilizing this model to assess hospital patients and predict potential anxiety cases. While the model will accurately identify most patients with or without anxiety, attention should be given to those whom the model predicts to have anxiety despite lacking a formal diagnosis. This group, positioned in the top right of our confusion matrix, is crucial as they are marked by the model as likely to have anxiety even if they currently do not. The model’s general accuracy implies that these patients could be at risk of developing anxiety. Recognizing these patients as at-risk presents an opportunity for early intervention and preventative care, potentially averting or reducing the onset of anxiety. Though such proactive measures may incur higher immediate costs, they are likely to result in long-term savings by preventing the need for extensive care if these at-risk patients develop anxiety in the future.

G: Code References

William Townsend's D208 Task 2 Performance Assessment Submission and D209 Task 1 Performance Assessment Submission provided the code used to clean and prepare the dataset. These submissions were also referenced for accuracy scores and other results as a point of comparison in this analysis.

H: Source References

* **Milena Afeworki @ Towards Data Science (2021)**: Provided rationale for using the Decision Tree over the KNN classifier from D209 Task 1.
* **Rafay Khan @ Towards Data Science (2019)**: Addressed why mean squared error is a poor choice for evaluating binary classification models.
* **Corporate Finance Institute (2022)**: Discussed the limitations of decision trees as they pertain to this analysis.
* **Analytics Vidya (2021)**: Discussed the limitations of Adaboosting as it pertains to this analysis.