Saood Usmani's report
Project 1

What did you do to prepare the data?

- To prepare the data, I first studied the columns of the data from the .csv file to get a sense of what I was working with. I noticed that this .csv file did have a header row, which meant that I could import this data into a pandas dataframe. After importing, I looked at a few key properties, such as the head of the data, the shape, the size, and the info of the columns. After getting a better understanding of my dataset, I wanted to check for duplicates. I used the duplicated method, chained with the sum method to see if there were any duplicated rows, and found that there were 11 duplicates. I dropped the duplicates using the drop_duplicates method, as duplicates don't provide valuable information to our model. Then, I used the duplicate and sum methods again to verify that there were 0 duplicates, which there were. After checking and getting rid of duplicates, I wanted to check for null values. I used the isnull method chained with the sum method to see if there were any null values, and found that the tumor-size column had 1 null value, and the inv-nodes column also had 1 null value. I visualized and confirmed these values by chaining the isnull method with the any method, specifying the column axis, and looking up the resulting rows in my dataframe. To treat these null values, I used the .fillna method along with the .mode method to fill the null values with the mode of that column. Then, I ran the same visualization and got no rows back, confirming that these nulls had been treated. Now that I had prepared my data, I was ready to perform univariate analysis, which I will talk about in my answer to the next question. After I had performed univariate analysis, I also performed one hot encoding on all of my categorical columns using the get_dummies method so that my model could train on these columns. Finally, I renamed the class_recurrence-events to class, for simplicity. I confirmed that the one hot encoding had worked using the info method, and I was now ready to begin training my model.

What insights did you get from your data preparation?

- From the histogram I constructed for deg-malig, I saw there were more 2s and 3s than 1s. The box plot I constructed for deg-malig was also heavily skewed. This makes sense as there was a higher amount of 2s and 3s than 1s, which would pull the mean and median towards the right.
- I also noticed that there were more Falses in our class than Trues. This is interesting because it might make our model more susceptible to labeling things as False just because there is a higher volume of falses. Although it isn't as bad as the Olympic example where it is a million to one, there seems to be around two to three times more falses than trues.
- Looking at the age countplot, it seems that this study has more older people, which may skew the rate of cancer, as different age groups may be more susceptible.
- From the tumor-size countplot, I noticed that the tumors tend to be moderately sized, most grouped in the 20-35 range.
- For the inv-nodes 0-2 dominated the count as the rest of the categories combined did not come close to the count of 0-2.
- Looking at the irradiat countplot, it seems that not as many people got radiation treatment which might be significant for cancer.

What procedure did you use to train the model?

- To train the model I first split up my data into X and y variables. For X, I simply dropped the class column using the drop method, specifying column axis. For y, I simply set it to the class column. Then, I further split my data into X_train, X_test, y_train, y_test, using the train_test_split method from sklearn.model_selection. I made sure to stratify y so that it can

roughly maintain the same proportion of class, and I also made sure to make my split reproducible by setting the random_state to 1.

- Then, to train the three models, I followed these steps:
    - First, I instantiated the model. For linear classification and K-Nearest Neighbor Classifier without using Grid Search CV, this was just using the relevant class's constructor. However, for the K-Nearest Neighbor Classifier using Grid Search CV, I also had to define the param_grid and pass that to the Grid Search CV constructor, along with the instantied K-Nearest Neighbor object, along with setting scoring to recall.
    - Once I had instantied the model, I fit the model to our X_train and y_train using the fit method.
    - Finally, using the classification_report method from sklearn.metrics and predict method from the model, I printed out the performance of the model on both the train and test data. This prints out the precision, recall, f1-score, and accuracy, which is exactly what we need.

How does the model perform to predict the class?

- The linear classification model performed the best in terms of accuracy, as it had a .68 accuracy vs the K-Nearest Neighbor Classifier with n = 5 at .6 and K-Nearest Neighbor Classifier using Grid Search CV at .58. However, the K-Nearest Neighbor Classifier using Grid Search CV performed the best in terms of recall with .33, vs the K-Nearest Neighbor Classifier with n = 5 at .14 and the linear classification model also with .14. I believe recall is the most important statistic for this problem as we want to minimize the amount of false negatives. This is because a false negative means that a patient was falsely diagnosed clear of cancer, when in reality they had cancer. This means that they may not get the life saving treatment that they need which could potentially save their lives.

How confident are you in the model?

- I am not very confident in my model, as a recall of .33 is quite low. Furthermore, the f-score was also .33 which further shows that the model's performance is not great and could definitely be improved. Further improvements may require a larger training set, better data, or a different type of model to improve the performance of the model.

I also used ChatGPT to help me achieve a deeper understanding of how certain methods worked and to debug any errors I ran across. For example, I ran across this error when attempting to use the K-Nearest Neighbor Classifier using Grid Search CV
usr/local/lib/python3.11/site-packages/sklearn/model_selection/_validation.py:821: UserWarning: Scoring failed. The score on this train-test partition for these parameters will be set to nan. Details:
…
and GPT was able to help me identify that I had forgotten to use one hot encoding on the class variable. Once I encoded it, this error went away. I did not use GPT to generate large chunks of code.