Task 2  
Saim Ali  
  
**2.1**   
In order to find the proportion of the two class samples (Label = 0, 1) I used the function below to distribute the dataset into two parts each of which contained either Label = 1 or Label = 0 values:  
Graphical user interface, text, application, email

Description automatically generated

When I created the subset df1 (dataset with class 1) I found there was 3400 of the 8400 total variables present. On the other hand, when calling the df0 (dataset with class 0) subset there was 5000 variables. After getting all these values I was able to calculate the proportion of the two class samples using the function below:  
Graphical user interface, text, email

Description automatically generated   
As seen in the image above when I called my probability function for classes 0 and 1 I found that of the 8400 variables 59.52% were of class 0 and the remaining 40.48% were of class 1. This means that class 0 is approximately 20% more likely to get picked if we randomly collected a sample from this whole dataset. As for the ratio Class 0/Class 1 the number reported for this part of the taskset was 1.47.  
  
**2.2 Sampling (Regular)**For this task I created a function (regSamp) which randomly samples “q” number of samples from our given dataset without replacement and returns the new created dataset. When setting q = 1000 in this function the code is show below:   
Text

Description automatically generated  
This code samples the dataset for 1000 values randomly and the results are as follow:  
Table

Description automatically generatedTable

Description automatically generated  
The given values for classes 0 and 1 from the sample set are 436 for class 1 and 564 for class 0. When I called my function to display the probabilities for both classes I noticed that it was not exactly the same as the original dataset’s numbers:  
Graphical user interface, text, application

Description automatically generated  
I believe this is due to the sample variables being picked at random without any perseverance of proportion. For the ratio Class 0/Class 1 I got 1.29 which also differs from the original datasets value.  
  
**2.3 (Stratified)**For this task I created a function which in some ways was similar to regular sampling, but the difficulty lied in the requirement to preserve the proportion of the number of different class samples. The function I created is shown below:  
Text

Description automatically generated with low confidence  
This allowed me to carry out sampling 1000 random variables from the dataset but this time while maintaining the original proportion of classes from the main dataset. The subsets for classes 0 and 1 for dataset 3 are shown below:  
Table

Description automatically generatedTable

Description automatically generated  
We got 595 variables for class 0 and 405 for class 1 which equates to 59.5% and 40.5% which satisfies the original datasets proportion of classes meaning we have successfully carried out stratified sampling. For the ratio Class 0/Class 1 we also get 1.47 which also aligns to the original dataset.  
  
**2.4 Feature Selection**To create a covariance matrix for dataset 3 I used the .cov() function and the results are shown below:  
Text, table

Description automatically generated with medium confidence  
When analyzing this covariance matrix, I can see there is many positive as well as negative strong relations between each of the features and label. I chose Feature 1 and Feature 2 since both features as both of their covariance values were close to zero. I believe these features will help me better discriminate between class 0 and class 1 in the dataset. I created dataset 4 with Features 1 and 2 below:   
Table

Description automatically generated  
 **2.5 Visualization**For this task I must create a supervised scatter plot for dataset 4 which is a subset of dataset 3 with 1000 variables but only contains the attributes Feature 1, Feature 2, and Label. I will also use my personalized colors given for task 2 which will represent the two classes (0 and 1) in the scatter plot. The scatter plot is shown below:  
Chart, scatter chart

Description automatically generated  
In this scatter plot above the clear dark color (color 1) is my class 0 and the light harder to see color given to me by putting in my PSID is class 1 (color 2). I was surprised when I computed this scatterplot as an odd spiral shape was formed. I believe that since such a weird shape was formed that I cannot currently use this scatterplot to determine any relations that would be useful to me for my dataset. There is also no real overlapping between the two classes as I checked by changing the alpha (transparency) value of the plot points in the scatterplot. Although there are a few clusters there is no clear pattern which I can identify the clusters appearing in.  
  
**2.6 Visualization**  
For this task I created four histograms to display the frequency of each feature (1 & 2) with each class (0 & 1). The histograms labeled “0” are of class 0 and the ones labeled “1” are of class 1. The results are shown below for each feature.  
**Feature 1:  
Chart, histogram

Description automatically generated  
Feature 2  
Chart, histogram

Description automatically generated**When looking at these histograms firstly we can see that there are no crazy values in either feature that sticks out as they are pretty even. The values we are paying attention to are the dark colored ones (color 1). It is difficult to separate the two classes based on the features selected because they are similar in plots from histogram to histogram making them unable to be distinguished. If we assume linear boundaries it also makes it hard to distinguish these features due to their overlapping areas between the two classes.  
  
**2.7 Splitting Dataset**To split the dataset4 into 700 samples for the training set and 300 samples for the testing set we must first consider the function created for subtask 2.3 and modify it so it returns us values for the training and testing sets as well as get us ready to carry out a decision tree formation. The function I created is below:  
Text

Description automatically generated  
Instead of copying the subtask function for 2.3 I came up with an easier function which uses the train\_test\_split function in the sklearn model selection which made this task a lot simpler. I will now display the training and testing sets which were created below:  
**X\_train (left) // X\_test (right)  
Table

Description automatically generated with medium confidenceText

Description automatically generated with medium confidence  
y\_train (left) // Y\_test (right)  
Graphical user interface

Description automatically generated with medium confidenceText

Description automatically generated with low confidence**As we can see we have successfully created a dataset for training and testing with the training set size being 700 and the testing set size being 300.  
  
**2.8 Classification**For this task I must create a decision tree of depth = 3 and analyze it to see if we are able to come up with an accurate decision tree for the dataset. I will be using training and testing sets X and y from the previous task. The code to create the decision tree is below:  
A screenshot of a computer

Description automatically generated with medium confidence  
Text

Description automatically generated  
Diagram

Description automatically generated  
Once I successfully obtained the needed decision tree, I started analyzing it and found that this tree was quite accurate for our dataset. I computed the accuracy score to be .84 which is relatively high for a decision tree. I will also share my classification report below which supports the data I found:  
Table

Description automatically generated  
As for the importance of each feature I calculated that using feature\_imporatances\_ function and got the results below showing feature 1 to be far more important in my decision tree to determine which class to assign:  
Chart, bar chart

Description automatically generated  
Nonetheless, this decision tree was a success and I can state that it was very accurate for the dataset to predict if the values of features 1 and 2 will result in the label of class 0 or 1.  
  
**2.9 Feature creation**We must now write a function which accepts a given dataset with two features (f1, f2) and build a new feature with the computation : sqrt(f1^2 + f2^2). Using the given computed feature, we create a new training and testing set named c\_training\_set and c\_testing\_set respectively. The function and results are shown below: **Text

Description automatically generated**  
This function gives the following output for the new c training and testing sets:  
**c\_training\_set (left) // c\_testing\_set (right  
Table

Description automatically generated with medium confidence**Text, table

Description automatically generated **2.10 Visualization**Now I create two histograms for c\_training\_set for the newly created feature from subtask 9. One histogram will be for class 0 and the other for class 1. The histograms will be labeled “0” or “1” for their respective classes. The histograms are shown below:  
Chart, histogram

Description automatically generated

When analyzing these histograms, we must compare them to the original training set histograms. What I noticed for these newly made histograms is that the newly made feature makes these histograms more easily distinguishable. I believe since the feature’s values were squared and added then square rooted that caused the data to be more easily distinguishable.  
  
**2.11 Classification**I will now create another decision tree with depth = 3 but this time for the c\_training\_set. The code to create the decision tree is below:   
Diagram

Description automatically generated  
This decision tree appears to be far simpler than the one we got for the original testing and sampling sets. I believe this is due to the data now being more easily distinguishable to each class (0 & 1) and easier to determine which one it is.   
  
**2.11 Building a Pipeline**I was unable to create a working pipeline with my functions, but I can analyze how a pipeline would help us in this situation. Some of the advantages of using a pipeline is to conduct data analysis more efficiently as it cuts down time where you have to call all functions to perform a task and also ensures a consistent data quality being carried out as the same functions will be used and called every time the pipeline is in use. In our case, this pipeline if constructed would be a tremendous help as we could analyze this data just by calling the dataset, specifying the sampling method “rgl” (regular) or “stf” stratified, and specifying the number of samples in the training set. We could basically conduct every task we have done previous to this one through this pipeline simultaneously and test different values in a relatively short amount of time.  
  
**2.13 Conclusion**To conclude this study of this dataset I will first discuss my findings. To start off my first notable discovery was the original scatterplot created by using Features 1 and 2. The scatterplot was shocking as it created an almost galaxy-like shape with a spiral and I could not interpret it to establish an increasing or decreasing pattern between the two features. Next I created a decision tree with the first computed training and test samples and found that it was a pretty accurate tree to determine the outcome of either class 0 or class 1. This showed me that there is hope to separate the two classes and determine which features correlate with which feature. I found that Feature 1 had an importance value of .8 which means it was far more detrimental to determining the class at least for the first decision tree. When I started creating the new feature for subtask 2.9 I was pretty confused but after a bit I managed to figure it out and created the new subset with the new feature to conduct a new decision tree/histogram analysis. When looking at this new feature’s data I saw it was easier to distinguish between the values since the numbers were now exaggerated. This was helpful to me because it showed me that it can sometimes be helpful to manipulate your original data in a way so that you can further create/support a theory without losing the original data’s parameters. When it came to creating the pipeline, I was unable to fully create a working pipeline with my functions I wrote and did not have enough remaining time to learn exactly how to achieve this subtask. I understand what a pipeline is as it helps us carry out multiple tasks altogether for our data analysis. This was my first-time using Python for any data analysis and it was quite difficult to learn these functions in a few days’ notice since I had many other projects during this week. Task 2 was difficult because I am used to R studio, but I believe Task 3 will be more forgiving now that I have a grasp on the functions available to me through Python.