Problem A

For this question I used the supervised learning classifier logistic regression form sklearn to train a model and then test data to predict whether a buyer used a credit card or not. The performance from the 'Logistic Regression' section below is from using the default values in the model and a StandardScaler to normalise the data. I then tried to optimise the data by changing the hyperparameter intercept_scaling to 0.3 and 0.5. This value creates a bias in the intercept weights by multiplying by the provided float. The default value is 1, meaning there is no bias in the intercept weights. I then did data generation using the MinMaxScaler which converts the data between 0 and 1 as a normalization technique, meaning each data value is dependent on one another. For my optimal model I decided to select the 'Logistic Regression, intercept scaling = 0.3', shown below. Not only does this model have the highest accuracy score at 59%, but precision is 77% meaning the model is less likely to label true positives as false negatives with a recall of 39% meaning the models ability to correctly label positive values. The above code can be found in kanestor_hw2a.py.

```
Logistic Regression
  Recall: 0.29
  Precision: 0.77
  F1 score: 0.42
  Accuracy: 0.55

Logistic Regression, intercept scaling = 0.3
  Recall: 0.39
  Precision: 0.77
  F1 score: 0.52
  Accuracy: 0.59

Logistic Regression, intercept scaling = 0.5
  Recall: 0.35
  Precision: 0.75
  F1 score: 0.48
  Accuracy: 0.57

Logistic Regression with MinMaxScaler
  Recall: 0.35
  Precision: 0.8
  F1 score: 0.48
  Accuracy: 0.58
```

I then saved the model using joblib and created a new file titled kanestor_hw2a_lrmod.py, which takes the saved model logreg_model.joblib as input. I then applied this saved model to the test data and obtained the same performance values as above. Additionally I applied the saved model to the new data instance: Card 3, Category 1, Amount $135.86. For the new instance I obtained a class value of 1, meaning this buyer likely used a credit card to conduct their transaction.

```
Saved Log Res, intercept scaling = 0.3
  Recall: 0.39
  Precision: 0.77
  F1 score: 0.52
  Accuracy: 0.59

The new buyer would be: 1
```

Problem B

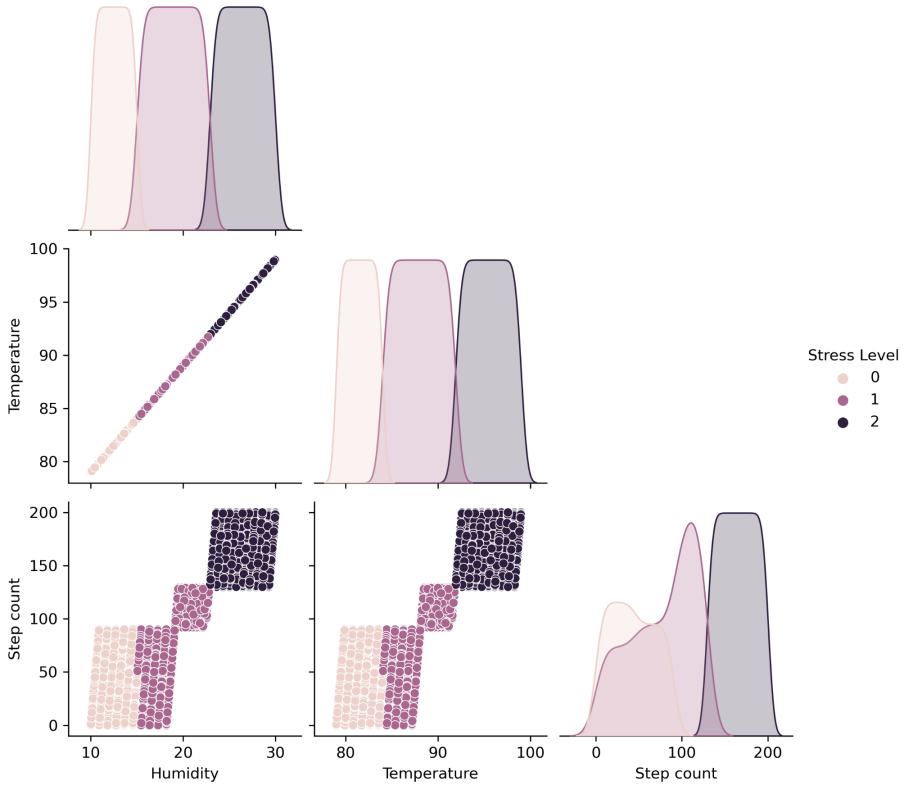
The code to create the plots below can be found in kanestor_hw2b.py, which takes as input /stress_data/Stress-Lysis.csv. The pairwise plot below with kernel density estimation (kde) on the diagonal axis shows the distribution of the Human Stress Detection dataset. In the second pairwise plot below the data is shown with a histogram on the diagonal axis. The pairplot in my opinion allows us to determine what would be the best model to analyse the data to achieve our specific goal. For eg we can determine there is a strong linear relationship between the humidity and temperature variables that would lend appropriate analysis with linear regression. This data would do particularly well with regression since there is such a strong positive relationship, the residuals which would ultimately allow us to have strong predictions.

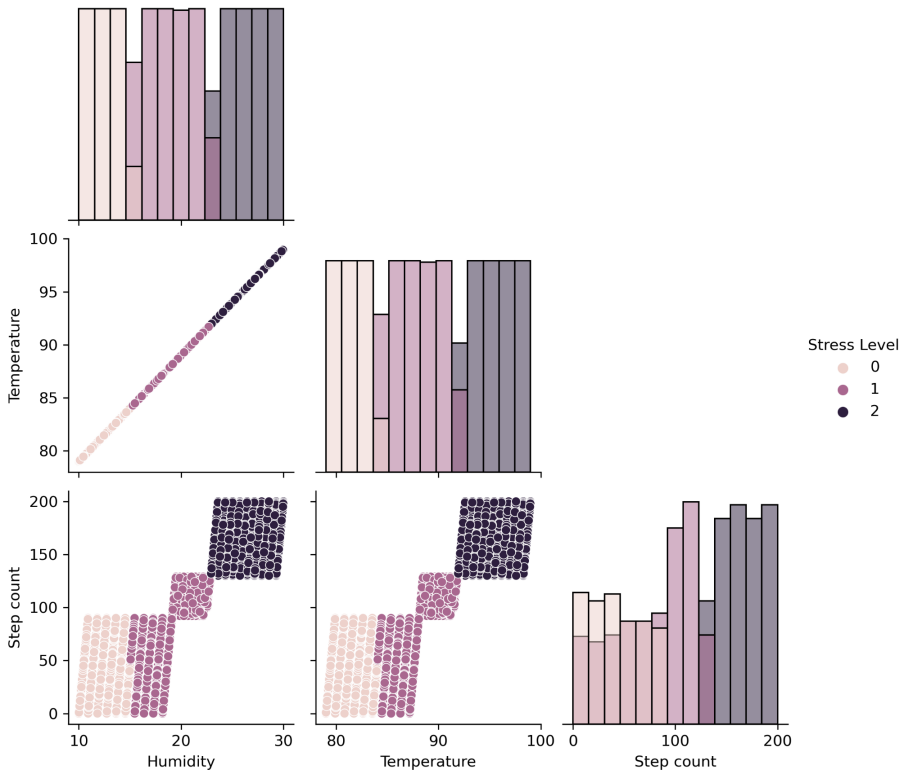
The scatter plots for humidity and temperature on the x axis and step count on the y axis show there is separability in the data into three clusters consistent with the three stress level classes. However there is high overlap in class 0 and 1 - as shown in the step count kde - that would make clustering with an algorithm like k means difficult. SVM could potentially be a good algorithm to use in this case given the overlap. Though if we only look at the humidity or temperature variable we could easily get good separation of the data using something like k means since there is minimal overlap in the classes.

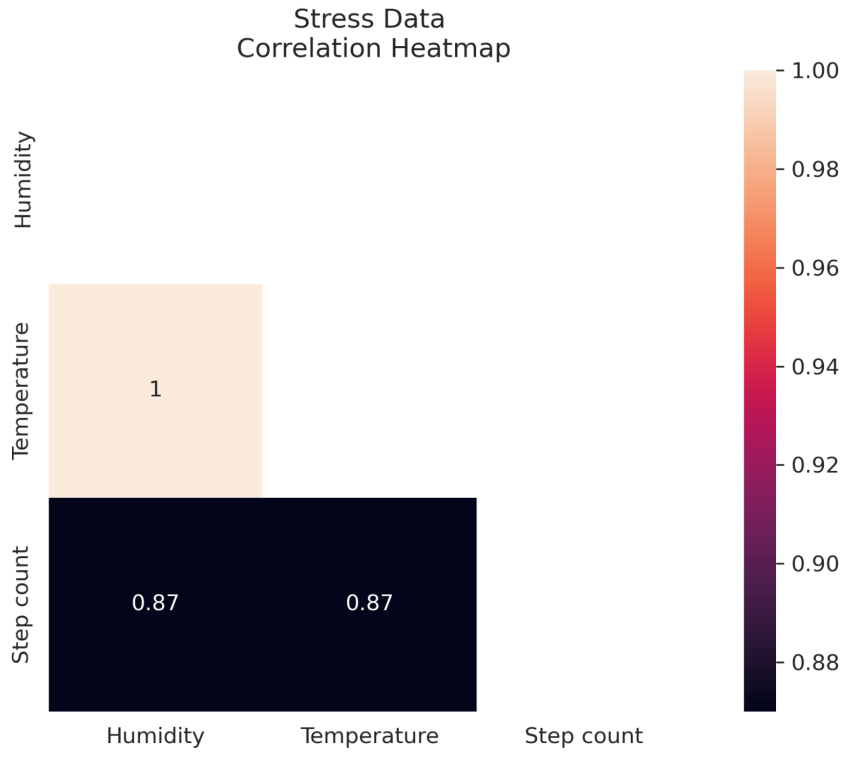
The correlation heatmap, uses correlation values of the pairs of variables to confirm what we visualized determined using the pairwise plots. The raincloud plots again show us the distribution of the data so we can ensure we don't have a heavy skew or many outliers. Finally the donut plot also helps us ensure we don't have a skew in the classes available in the dataset as this would inhibit our ability to build effective models.

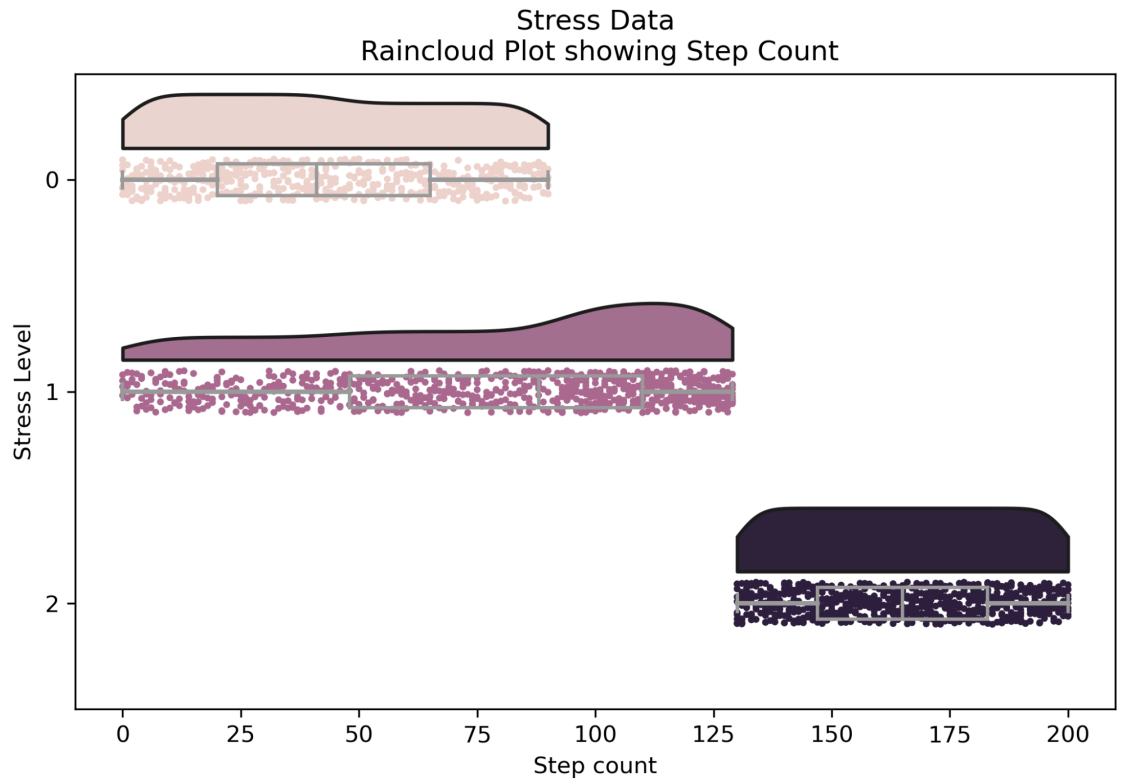
In terms of the principles for data visualization for the five plots, I followed these principles for all of them: **1)** "Does everything in the visualization have a purpose?", **2)** "Acknowledge that default settings are not always best when creating visualizations", **3)** "Many plots benefit from providing titles, axis labels and legends." and **4)** "Consider whether you have selected good colours for your plots.". I particularly wanted to highlight how I followed rule 2; for

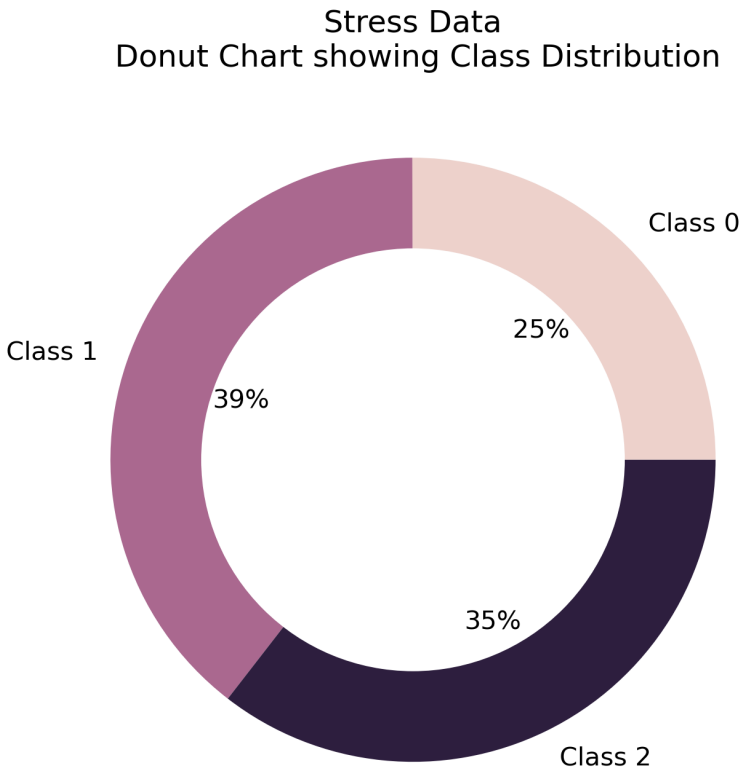
the pairwise plots and the correlation heatmap I changed the visualization from a matrix which would repeat information into only a triangle, as I wanted to ensure best readability and ability to quickly gain information. For the donut chart I decided not to agree with the following rule: “Pie charts are well recognized. Most data visualization experts do not have a positive opinion of them.”. I’m disagreeing with this rule specifically for this context in which it is used, which is to show the distribution of the stress level classes. This chart does not show critical information for the data but simply allows us to confirm we have a good distribution of all classes before building our models. It’s also my opinion that making the chart a donut instead of a pie makes it easier to tell the difference in slices.




Stress Level
0
1
2
Temperature
Step count
Humidity
Temperature
Step count


Stress Data
Correlation Heatmap
Humidity
Temperature
Step count
1
0.87
0.87
1.00
0.98
0.96
0.94
0.92
0.90
0.88
Humidity
Temperature
Step count


Stress Data
Raincloud Plot showing Step Count
Stress Level
0
1
2
0
25
50
75
100
125
150
175
200
Step count


Stress Data
Donut Chart showing Class Distribution
Class 0
25%
Class 1
39%
35%
Class 2