In this computer assignment, we observe a few interesting things.

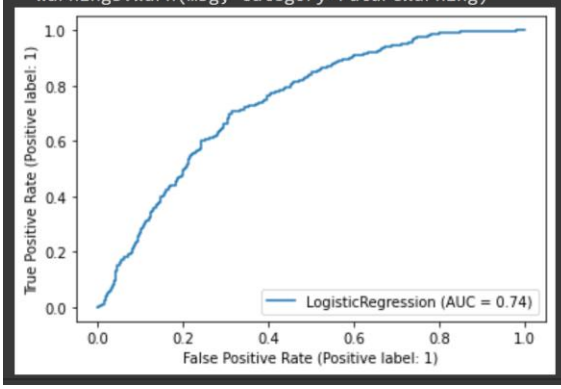
First, we observe an unbalance in our dataset, where we have more positive than negative cases. For reliable results, we need to balance the sample. However, I decided to try to use the unbalanced data to train the logistic regression model. The metrics were quite surprising.

```
[[189 210]
 [ 78 496]]
--------------------------------------------------
              precision    recall  f1-score   support

           0       0.71      0.47      0.57       399
           1       0.70      0.86      0.77       574

    accuracy                           0.70       973
   macro avg       0.71      0.67      0.67       973
weighted avg       0.70      0.70      0.69       973

/usr/local/lib/python3.7/dist-packages/sklearn/utils/d
  warnings.warn(msg, category=FutureWarning)
```



With a sample where there are more positive cases than negative ones, one would expect the machine will predict a lot more positives than negatives. However, the metrics for this model show otherwise. Somehow, the model made a lot of negative predictions than positive ones. This resulted in the model having high accuracy, positive recall, and precision. But if we look at recall for the negative predictions, we can see that the model makes a lot of false negatives predictions.

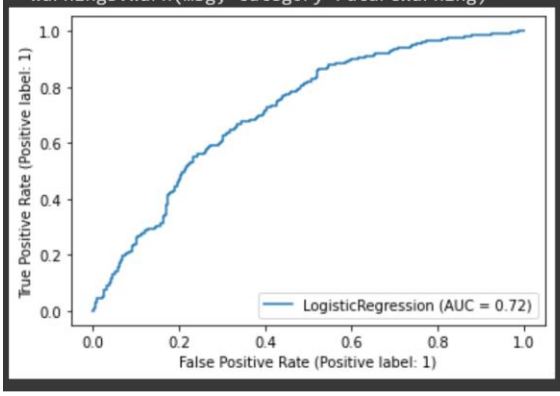
After Model 0, I did an under-sampling, and trained Model 1 using that dataset. The results were not impressive.

```
[[254 166]
 [112 266]]
--------------------------------------------------
              precision    recall  f1-score   support

           0       0.69      0.60      0.65       420
           1       0.62      0.70      0.66       378

    accuracy                           0.65       798
   macro avg       0.65      0.65      0.65       798
weighted avg       0.66      0.65      0.65       798

/usr/local/lib/python3.7/dist-packages/sklearn/utils/d
  warnings.warn(msg, category=FutureWarning)
```



Although the model is making fewer false negative predictions, the other metrics have also declined. But this would be a more reliable model than Model 0.

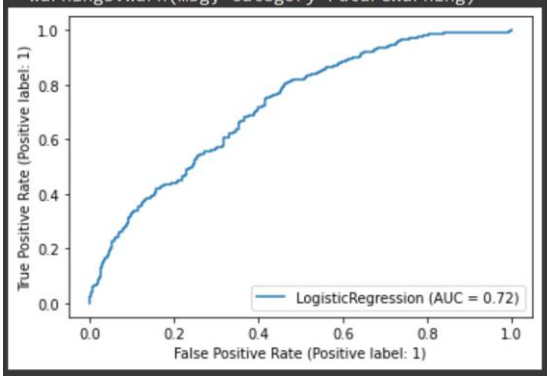
Lastly, I used the SMOTE technique to over-sample the dataset, and then use the RFE technique to select my features. With this dataset, I trained Model 3. The model's metrics did not quite improve.

```
[[253 146]
 [137 277]]
--------------------------------------------------
              precision    recall  f1-score   support

           0       0.65      0.63      0.64       399
           1       0.65      0.67      0.66       414

    accuracy                           0.65       813
   macro avg       0.65      0.65      0.65       813
weighted avg       0.65      0.65      0.65       813

/usr/local/lib/python3.7/dist-packages/sklearn/utils/d
  warnings.warn(msg, category=FutureWarning)
```



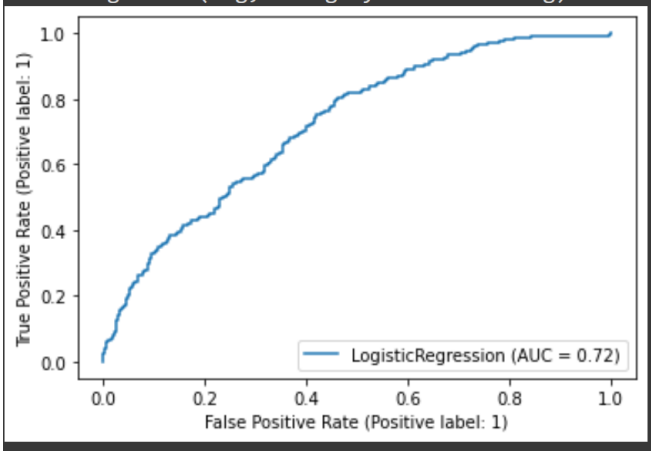
Even with a GridSearchCV looking for the best hyperparameters, the model's performance has only barely improved.

```
[[253 146]
 [135 279]]
-----------------------------------------------
              precision    recall  f1-score   support

           0       0.65      0.63      0.64       399
           1       0.66      0.67      0.67       414

    accuracy                           0.65       813
   macro avg       0.65      0.65      0.65       813
weighted avg       0.65      0.65      0.65       813

/usr/local/lib/python3.7/dist-packages/sklearn/utils/d
  warnings.warn(msg, category=FutureWarning)
```



After hours of effect, I came to this conclusion: logistic regression might not be the best machine learning model for this specific dataset. So the next step would be trying some other models such as random forest, KNN, etc.