WRITTEN REPORT

OVERVIEW: The assignment dealt with the implementation of logistic regression using gradient descent. Tasks included running the implemented model on different iterations, learning rates, and reporting the cross-entropy error and the classification error. Lastly, the custom regression model result was compared with sklearn's logistic regression model. Steps, methods, and details are mentioned below.

STEPS:

- Data Pre-processing
  - Model wasn't performing good, so I removed features with low correlation using SelectKBest. This increased the efficiency as well as training time.
  - Standardized the feature vectors using sklearn's Standard Scaler.
- Split the train and test data into a 30:70 split.
- Created the Gradient Descent
  - Initialized the weight and bias.
  - Calculated the cost using cross-entropy error.
  - Calculated the gradient and updated the weights and bias.
  - Created a predict function to predict the target variables of the training data.
  - Used the newly created class to predict and check the accuracy of the model.

Model was run on different number of iterations. The results are in the table below.

	ITERATIONS	CROSSENTROPY ERROR	CLASSIFICATION ERROR – TRAINING DATA	CLASSIFICATION ERROR – TEST DATA	TIME
1	10000	0.642	0.045	0.15	2s
2	100000	0.403	0.043	0.14	4s
3	1000000	0.107	0	0.21	52s

INSIGHTS

- As expected, the cross-entropy error is inversely proportional to the number of iterations. As the number of iterations increased, the model's cost also decreased.

- Same can be said for the classification error on the training data and the test data, it decreases as the number of iterations increases.
- However, there was an outlier in the results as the classification error on the test data actually increased when the model ran one million iterations. This was a great learning moment for me as I realized that this outlier result was a classic example of overfitting. The huge size of iterations allowed the model to learn on the training set to the point where it started to adversely affect the performance on the unseen test data.
- As expected, the higher the number of iterations the longer time it took for the model to learn.
- The closer the cross-entropy error is to 0, the better the model is, and hence the lower the classification error will be. This was evident on the classification error on the training data but it didn't follow on the test data when the iterations were increased to one million. There is a directly proportional relationship between cross entropy error and classification error which can be seen from the table above.

COMPARISON WITH LIBRARY LOGISITC REGRESSION

- My best accuracy on the training model was with 100000 iterations was 95.6% whereas sklearn's logistic regression gives an accuracy close to a 100.
- Time taken by sklearn's model was almost the same as my model.
- The high accuracy of the model shows that my model predicted very effectively.

EFFECT OF LEARNING RATE ON CROSS-ENTROPY

- It is crucial to pick a good learning rate because if too low then training will progress very slow and if it's too fast, we might miss the global minimum. I tried running the model on several different learning rates and results are summarized below.

	LEARNING RATE	CROSS-ENTROPY ERROR	TIME
1	0.001	0.011	6s
2	0.0001	0.107	7s
3	0.00001	0.403	8s

As seen from the table above, a learning rate of 0.001 keeps the cost to a minimum. As the learning rate was lowered, the cost of the function increased.

References:

Loss function and gradient formula adapted from:
https://www.youtube.com/watch?v=0VMK18nphpg