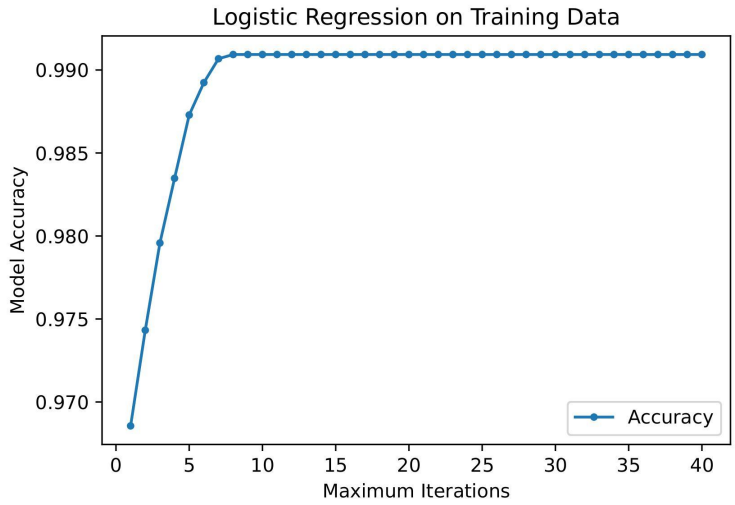
Sharrey Suhendra

Professor Allen

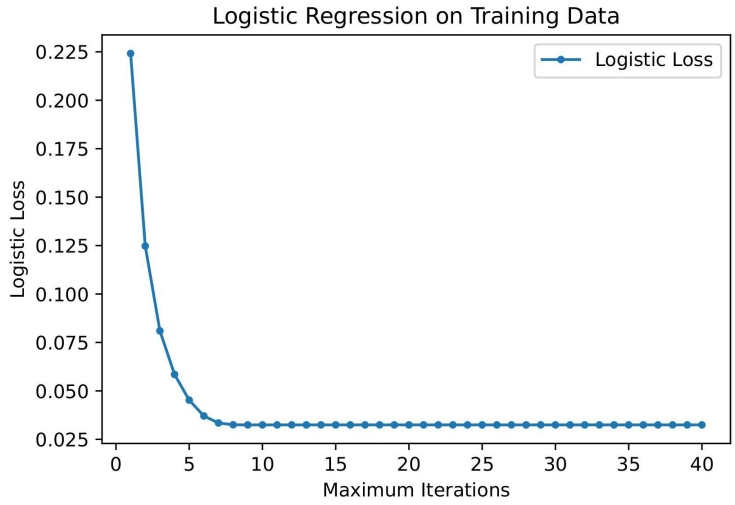
Comp 135: Intro to Machine Learning - Project 01

Part One: Logistic Regression for Digit Classification

1)

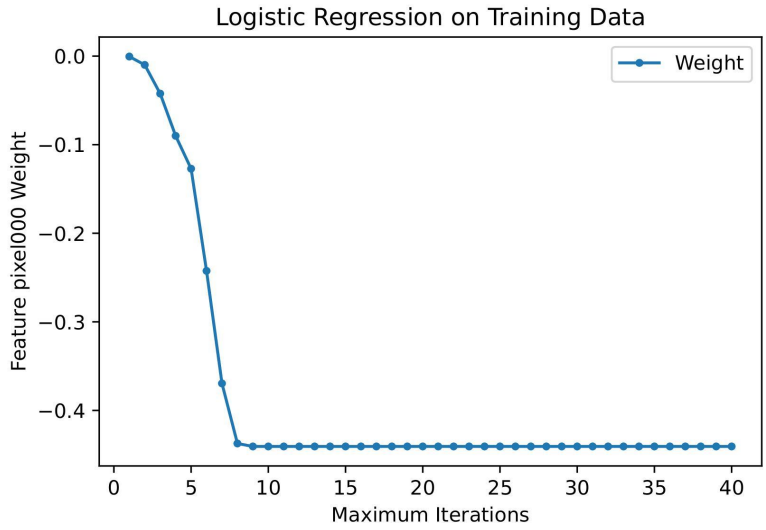


The plot shows that accuracy is lowest when maximum iteration is just set to one, and progressively increases as the iteration given gets higher. Once the number of iterations reaches 7 and above, the model accuracy shows a near-constant trend of 99%; In this range, the near-zero difference between the accuracy of successive iterations implies that the model is/almost converged.



Log loss is a probabilistic measure of accuracy; A score close to 0 means there is a high probability that the classifier has made the right decision (that it's more accurate) and vise versa. So, the higher the accuracy, the lower the logistic loss score. Thus, it makes sense that the trend of this plot is the opposite as to the Accuracy Plot; the Logistic loss score is greatest at 1 iteration, and after 7 iterations, the loss seems constant.

2)



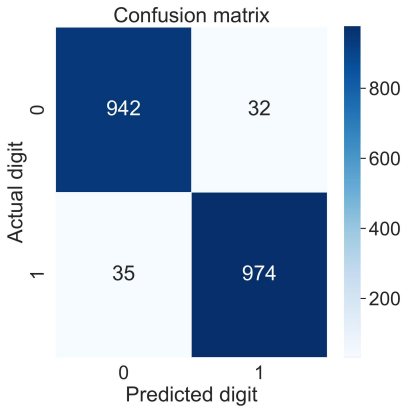
This plot shows the relationship between the first weight the model applies to feature pixel000. When the maximum iteration is 1, pixel000 has a weight of 0, indicating that this feature does not have an effect on the overall 'decision' of whether or not the input data is an 8 or 9. However as the iterations increase and the model begins to converge, the weight is not set at about -0.47; This negative weight implies that the feature pixel000 corresponds to an 8.

3)

Model with least loss on test data:

- Regularization Parameter C = 0.03162277660168379
- Accuracy = 0.9662128088754413

Confusion matrix:

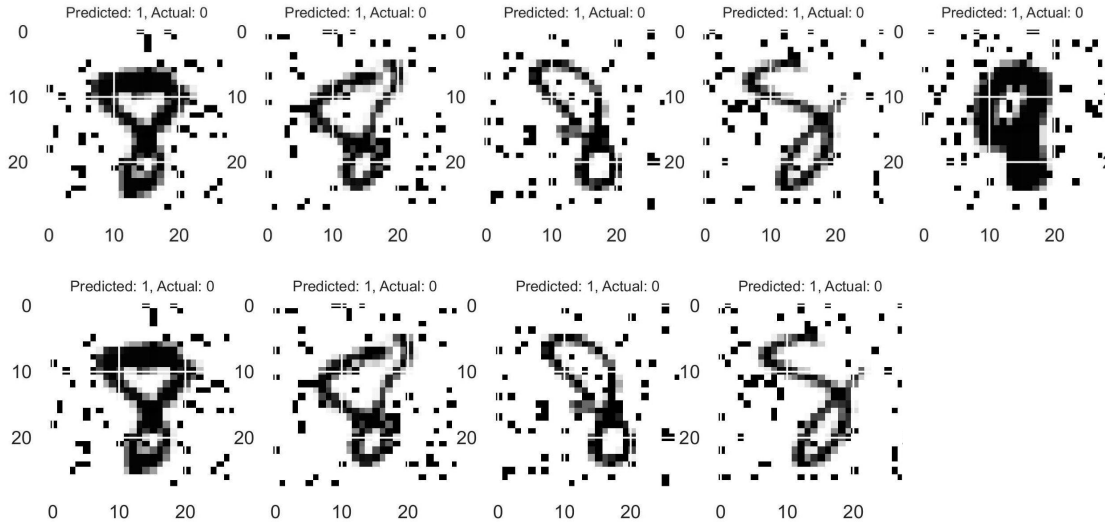


Predicted	0	1
Actual		
0	942	32
1	35	974

C is an inverse of regularization strength/penalty, so smaller C specifies stronger regularization. The C that gives the best model is low, about 0.03; It is telling the model to give more weight to complexity penalty at the expense of fitting to the training data, that the training data may not be fully representative of the real-world data. Thus, it makes sense that this model gives the least logistic loss and high accuracy (96.6%) on the test data. Based on the confusion matrix, the model was able to classify most of the test data right, with about 96.7% accuracy on negative data (8) and 96.5% on positive data (9).

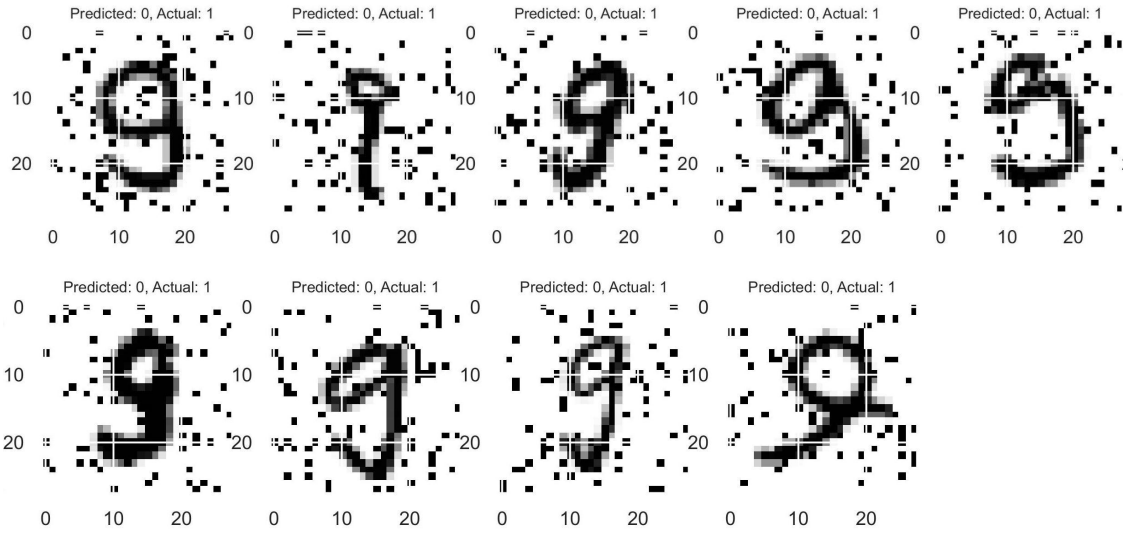
4)

9 Samples of False Positives



All these nine input test data's actual value is 0 (negative), indicating the number 8. However, the classifier mistakenly classifies them all as having a value of 1 (positive), indicating that it is the number 9. Looking at how all the eights are displayed, some of the lower circles of the eight are found slightly on the lower right and not that center, which may lead to the false classification

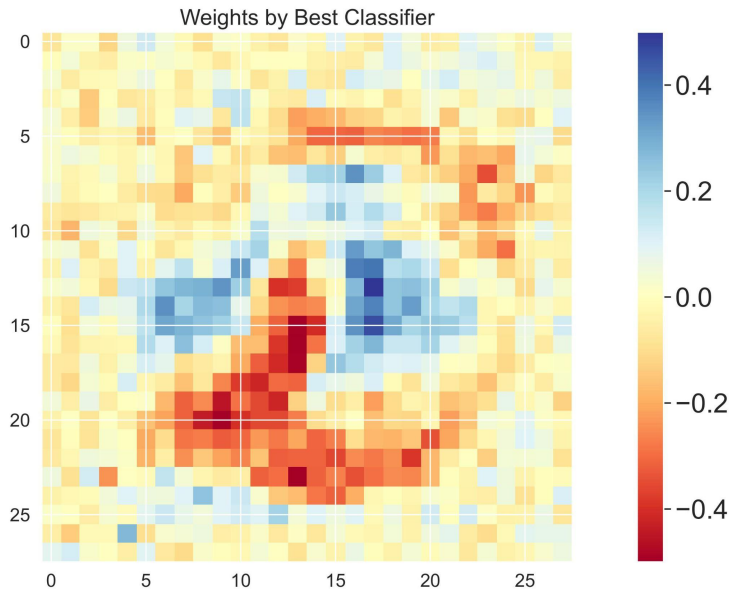
9 Samples of False Negatives



All these nine input test data's actual value is 1 (positive), indicating the number 9. However, the classifier mistakenly classifies them all as having a value of 0 (negative), indicating that it is the number 8. Looking at all how all the nines are displayed, the tail of the 9 seemed more curved and slanted, which may indicate that the model has a higher negative weight on these parts of the pixels correspondingly to an 8.

Overall, the linear weighted model may have put higher magnitude weights on specific pixels that correspond to 8 and 9 respectively, which explains why some images can be falsely classified.

5)



The plot shows the weight coefficient corresponding to the pixels of the original images. This plot is done by setting it to a divergent colormap 'RdYlBl' (Red, Yellow, Blue), such that when the color is yellow it indicates a coefficient of 0, and increases up to 0.5 as the color gets closer to blue, and decreases up to -0.5 as the color gets closer to red; This can be seen from the color bar plotted on the right side.

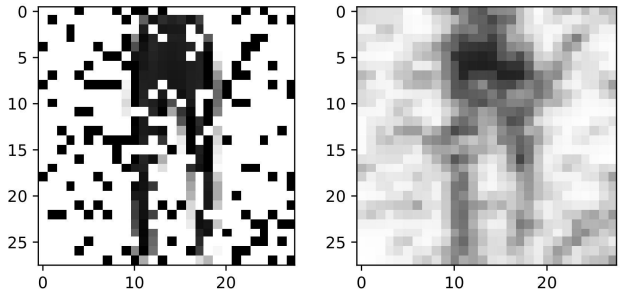
So, orange-red pixels have negative weights and correspond to an 8 whereas blue pixels have positive weights and correspond to a 9. Looking at the plot, there are many light yellows on the outer portions, indicating the weight of 0/close to 0 and does not correspond to 8 or 9, and this makes sense because based on some sample images seen in the previous questions, the writing of both numbers 8 and 9 generally occupy the central areas, and the outer portions are redundant. There is also an area of bolder reds on the lower left-hand side, which makes sense as it most likely indicates the lower left figure of number 8.

Part Two: Trousers v. Dresses

Feature Transformation: Noise Reduction

Since input images have significant noises injected, noise reduction was performed on the inputs x train and test data. This is done by blurring, where each pixel of an image is computed equal to an average of 3x3 pixels around it. Below is a sample of the feature transformation.

Original Transformed

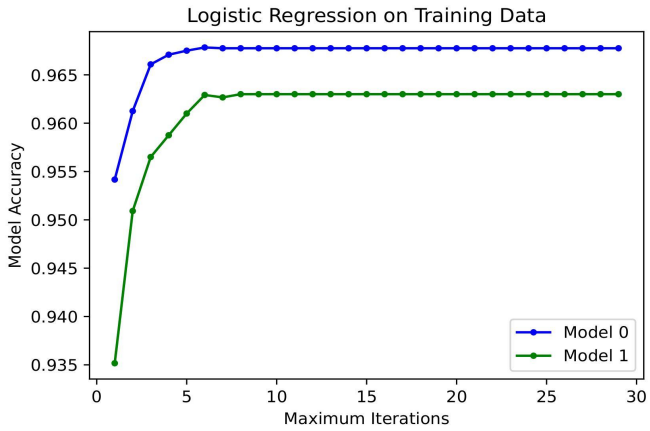


Observations on sample: The black pixels around the sides that are noises and are irrelevant to the image of the trouser are now more light grey and blended. The white pixels in the middle section of the trouser that should have been darker are now darker, so the trouser is more fully colored. However, the darker pixels of the trouser ended up being lighter since some of their 3x3 surrounding pixels are light.

There are pros and cons to this feature transformation. The grey-scale values on the side where the irrelevant pixels are are more blended, so these transformed black pixels will not put too much weight on the model later on. Also, the trouser/dress image is more complete which will help improve the model, although the darker pixels may turn lighter which may be a disadvantage. These are observed through several plotted original and transformed images and since there are 12000 of them, it is difficult to tell whether all the images follow the same observation. Overall, the observations given are not a fully accurate representation of the noise reduction implemented and it is not a guarantee that the transformed features will result in a better model compared to one trained from the unmodified features.

Max-Iteration

Logistic regression models were fit to the whole training data; This was done with sklearn and the liblinear solver. The effects of iteration for the models were explored by leaving all parameters as default except the max_iter value that limits the iterations allowed for the solver to converge on its solution.



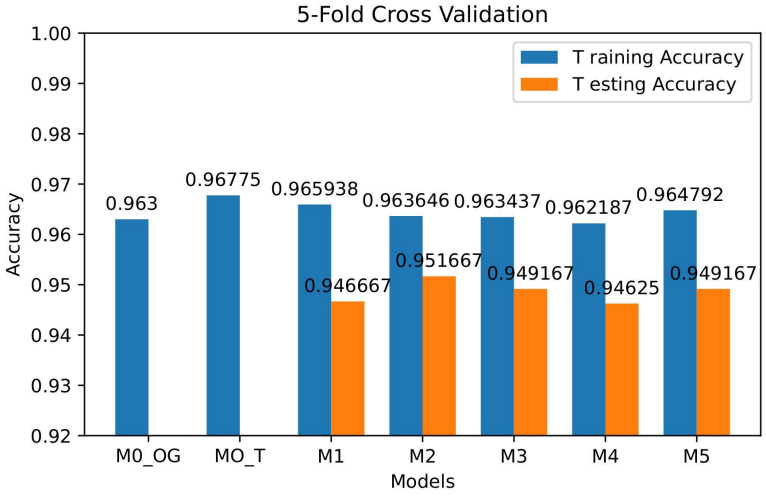
x-axis: Maximum Iterations
y-axis: Accuracy, measured on training data
Model 0: Untransformed input features
Model 1: Transformed input features

Based on the plot, it is clear that Model 0 does a better job of identifying the training data compared to Model 1. Both models seem to converge roughly at around the same iteration, $i = 9$ and showed similar increasing trends from $i = 1$.

These two models were trained on all the untransformed and transformed input features, and since the accuracies were calculated on the training set, it solely depended on which model fits best on the training data; So, if the model fits better and even overfits the training data, the better the accuracy of the model calculated. Here, model 0 has the better performance and in the explorations, models trained from the transformed features will be further explored and focused on. A maximum iteration of 7 was found to give the best model accuracy for both models and this iteration number will be used from now on.

5-Fold Cross-Validation:

Earlier on, logistic regression models were both trained and tested on the training data, and this is not a good way to build a model as it is subjected to overfitting the training data and lowering accuracy on unknown testing data that may not be as similar to the training data. Since no testing data was given, building and comparing multiple models from a k-fold cross validation will ensure that every observation on the input features has the chance of appearing in training and testing data. The figure below shows 5-fold cross-validation of the transformed input data:



x-axis: Type of Models
y-axis: Accuracy
M0_OG: Original Features, No Cross Val
M0_T: Transformed Features, No Cross Val
M1-5: Transformed features, Each number represents fold# used for testing. (e.g. M1 Training = folds 2-5, M1 Testing = fold 1)

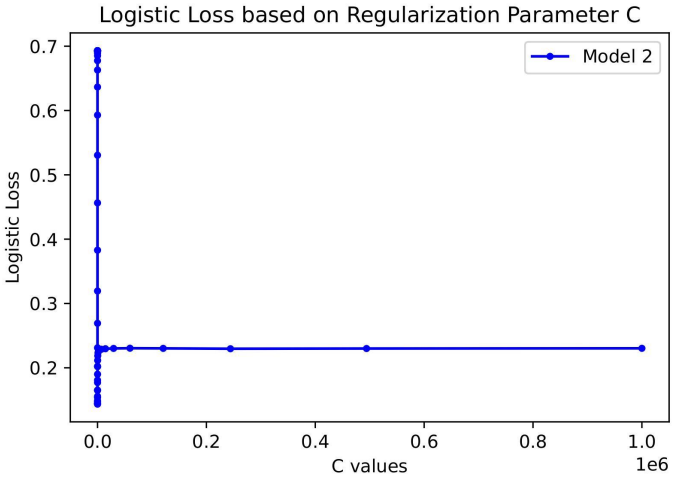
M2 = Best model based on testing accuracy among all 5 models of cross-validation.

The benefit of cross-validation is that we can 'create' new testing data from the training data and help reduce the likelihood of overfitting by finding the best model. Based on the bar chart, even though M0_OG and M0_T models have similar and slightly better training accuracy, there is no reason to guarantee that it will do well on unknown testing data. Looking at the 5 models produced from the 5 fold cross-validation, these models have similar training accuracy and testing accuracy respectively, though the testing accuracy differs more. The result indicates that model 2 has the best performance on unknown testing data, and this may just be down to the luck of the model finding a fit given the folds it received and fold to test on.

This project holds the assumption that model 2 is the best model for unknown testing, and will be used from now on onwards. Models will be fit on training data folds 1, 3, 4, 5, and tested on fold 2.

Parameter C Regularization

Next, different values of the C parameter were explored using a regularly-spaced grid of values ranging from small decimals to a million. The purpose of this C value is to reduce overfitting, as a higher regularization will give a greater penalty on overfitting. The C value measures the inverse of regularization, so a lower C value increases regularization and vice versa.



It is expected to have a value of C that is not too large as it will result in small regularization and an overfit model; Vice-versa, a really small C value results in significant heavy regularization and an underfit model. Based on the logistic loss figure,

C = 0.7543120063354607

minimizes the logistic loss on testing data.

Penalty: L1 vs L2 Regularization

L1 and L2 penalties were explored to observe changes in model performance. The L1 Lasso regression gave a higher accuracy and lower logistic loss compared to the L2 Ridge regression.

	L1	L2
Accuracy	0.95375	0.9533333333333334
Logistic Loss	0.13738203489907558	0.14345207699405943

This makes sense because the L1 regression technique shrinks the less important feature's coefficient to zero and by doing so, the specific feature becomes irrelevant. In this case, since the pixels around the corners/sides of the images are similar in scales due to noise reduction, they may have weights of zero as the model observes that it is not as relevant in predicting whether or not the image is a trouser or dress.

SUMMARY

Below is the error rate and AUROC for the predictions of the input transformed testing set, split into 3 rounds:

- Round 1: Best model after cross-validation (model 2)
- Round 2: Best model after cross-validation + C regularization
- Round 3: Best model after cross-validation + C regularization + L1 vs. L2

	Round 1	Round 2	Round 3
Parameters	Max-iter = 7 C = default 1.0 Penalty = default L2	Max-iter = 7 C = 0.7543120063354607 Penalty = default L2	Max-iter = 7 C = 0.7543120063354607 Penalty = L1
Error Rate	0.05449999999999999	0.05449999999999999	0.05400000000000005
AUROC	0.9806829999999999	0.980774	0.981258

Both the error rate and AUROC values are similar respectively, though as the round goes on, there was a slight difference; The error rate went down, and the AUROC increased. The error rate is equal to 1 minus accuracy, so the closer the error rate to 0, the better the model performance. Whereas high AUROC values indicate that the area below the ROC curve is high and the closer it is to 1, the better the model performance. Even though the change was not much, it still indicated that the regularizations did help in improving the model performance and an error rate of 0.054 and AUROC of 0.981 is a good model.

Though, there could always be better models, once there are more experimentations of data transformations and parameter regularizations. This project focuses on noise reduction as its data transformation, and averaging pixel values according to its 3x3 surroundings was a relatively easy implementation that reduces noise by blurring and blending pixels. However, it is only a 28 by 28-pixel image so the trouser/dress may blend to its background, which is something to consider whether it will improve model performance or not; There are other complex noise reduction techniques that may result in better model performance. Another to consider is if the training input-output data was 100% accurate at all. Given 12000 images, it cannot be guaranteed that the output labels are all correct unless someone confirmed that it is by manually looking at each image.

Overall, given that no testing output data was given, this project was successful in applying cross-validations and regularization techniques to reduce the possibility of overfitting the training data and find the optimal model for the testing data.