Assignment 1

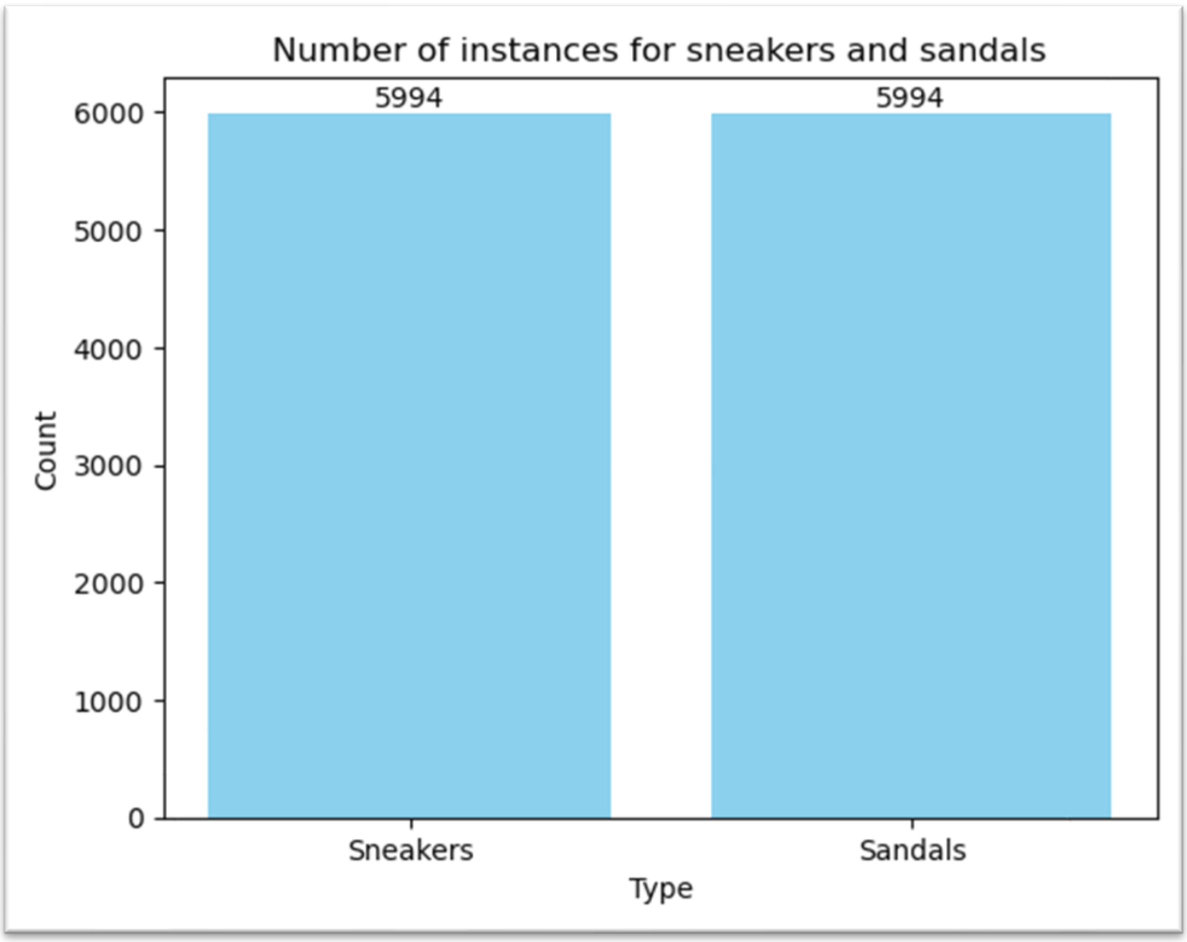
Adharsh Sundaram Soudakar (23796349)

D1:

Table. Dataset overview

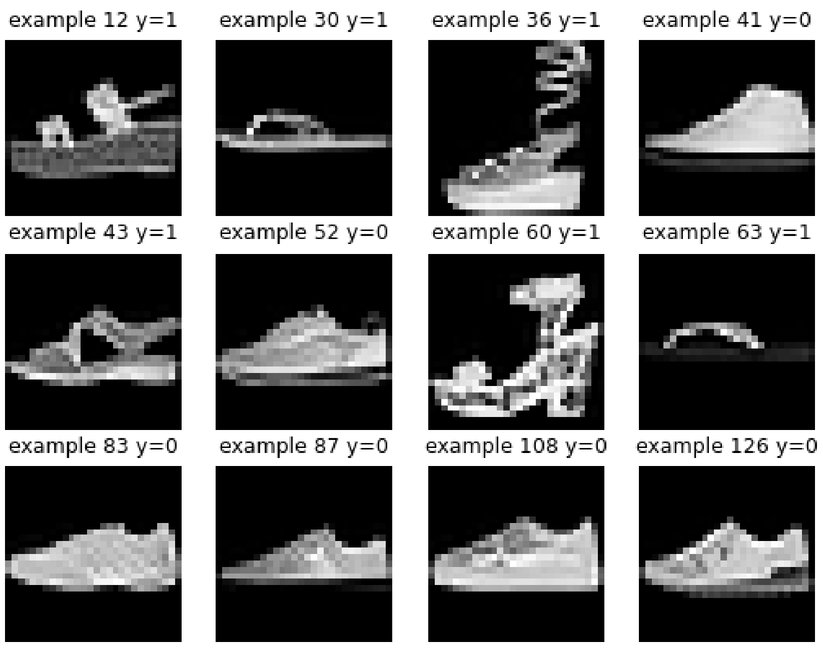
Training set Instances	Test set Instances	Total Instances
11988	2000	13988

D2:



- From the bar plot, we can infer that we don't have an imbalanced training set. There are 5994 instances for the class sneakers and 5994 instances for the class sandals.

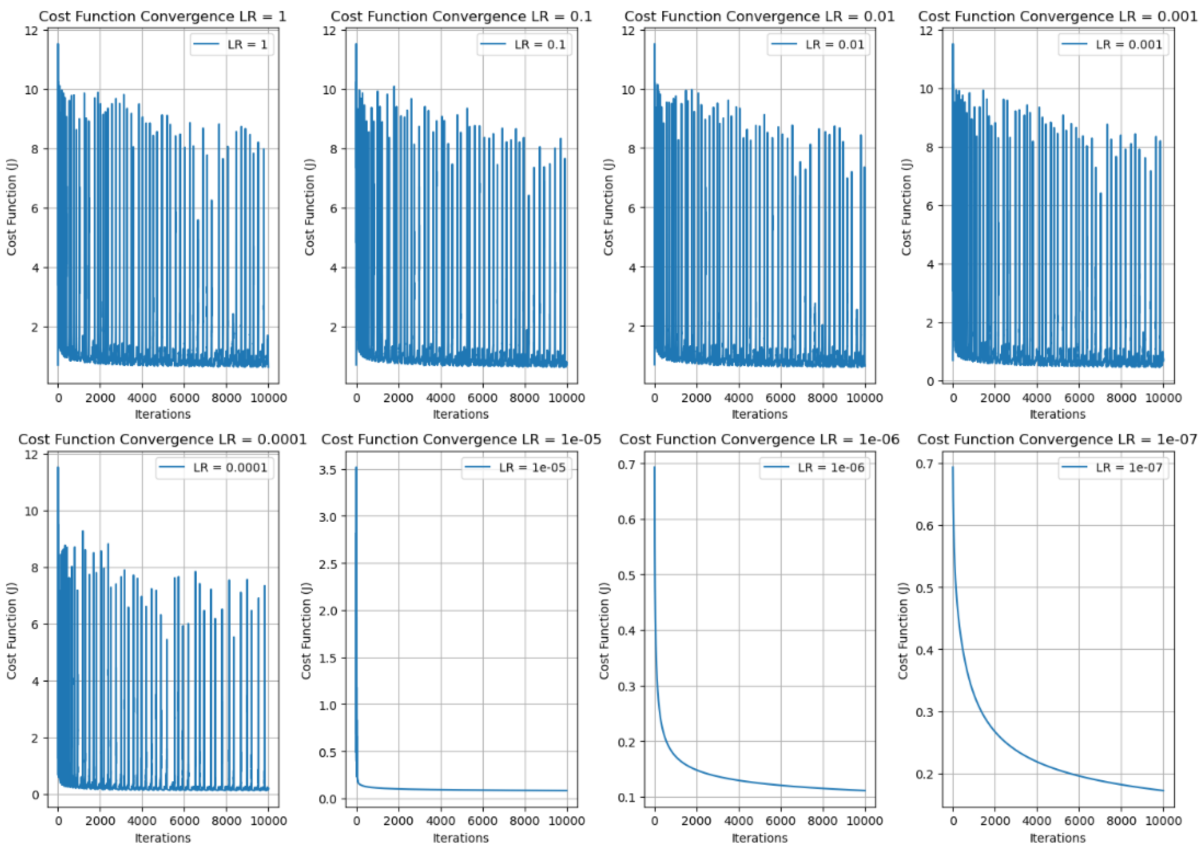
D3:



- The first 6 images/examples from each class with the corresponding example id and associated label.

D4:

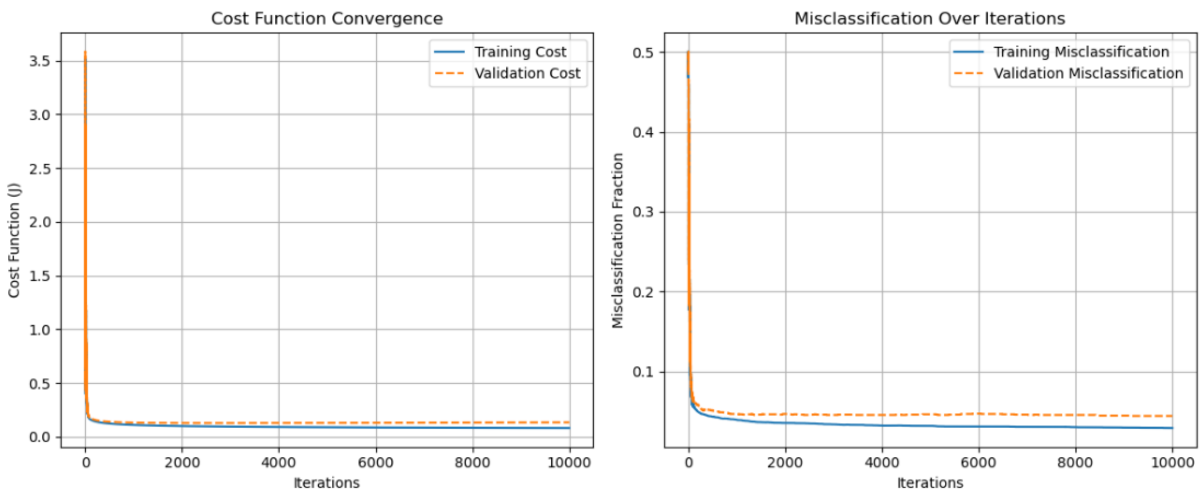
Fig. Plots for different learning rates



- From the plots, the final value of η should be 1e-5. The cost function curve is smooth enough and converges to minimum within 10000 iterations.

- Although the cost function curve is smoother for η = [1e-6,1e-7], it does not converge to the minimum within the 10000 iterations.
- For η = [1 to 1e-4], the cost function oscillates significantly and does not converge to the minimum within the 10000 iterations.

D5: **Fig. Plots for LogisticRegression without Regularization with η = 1e-5**



D6: Interpretations and Conclusions

- From the proximity of the training and validation curves in both the plots, we can infer the model's generalization capability. It should perform well on unseen data.
- The model distinguishes sneakers from sandals well as the misclassifications are very low.
- The curves don't raise after reaching the minimum. This shows that overfitting does not occur.
- To conclude, η = 1e-5 is a very effective learning rate.

D7:

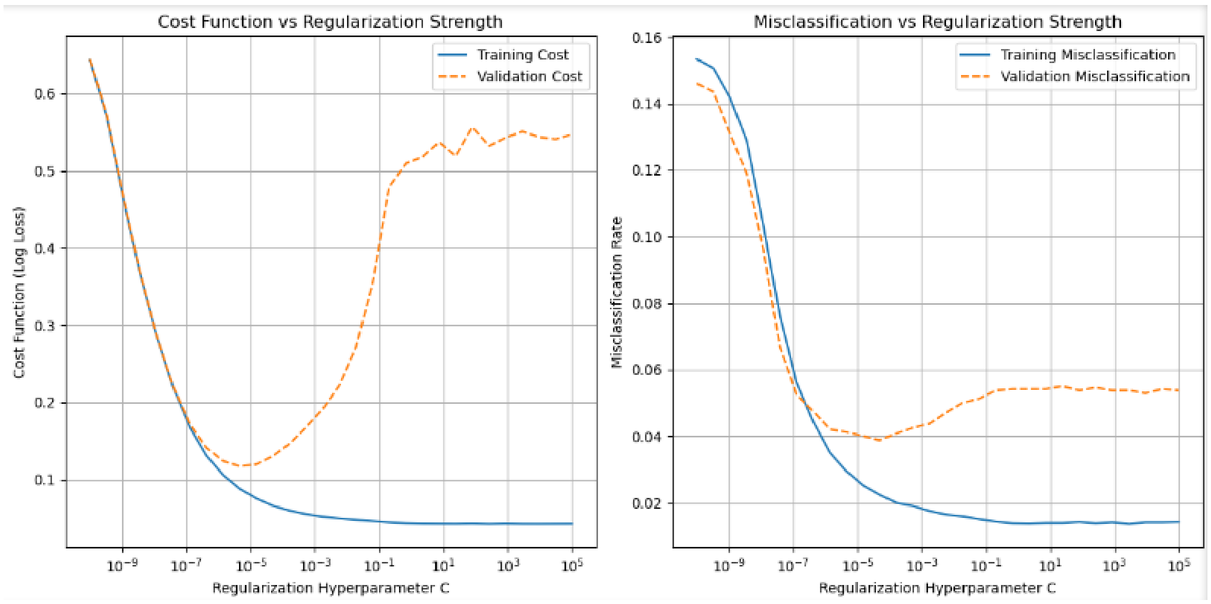


Fig. Plots for LogisticRegression with Regularization

D8:

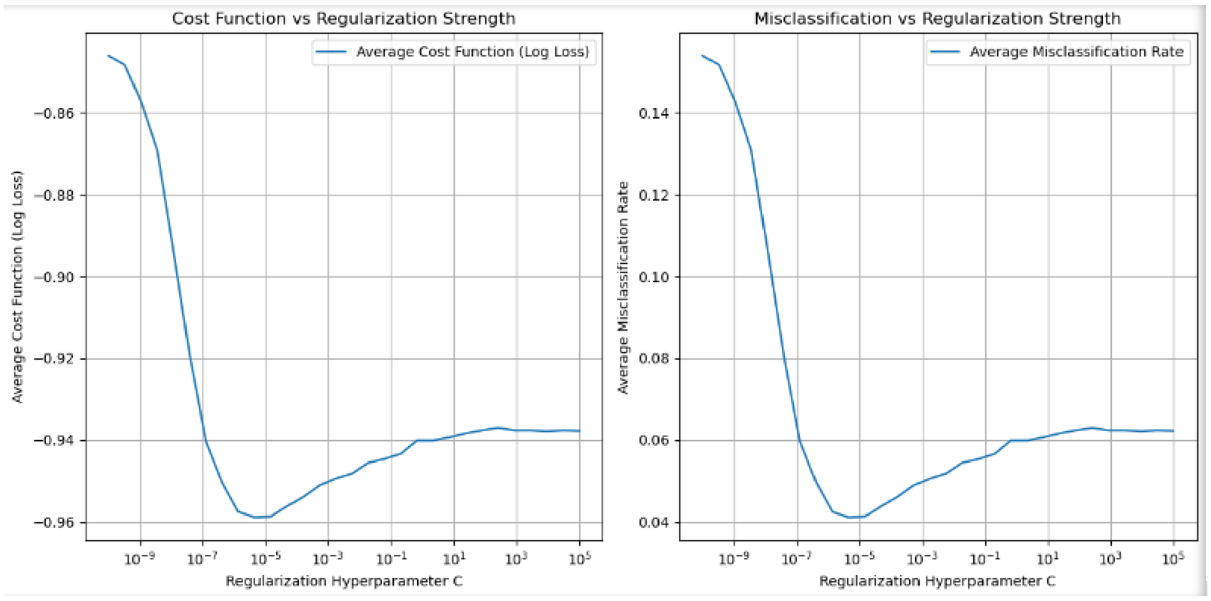


Fig. Plots for LogisticRegressionCV with 10 folds Cross-Validation

D9:

Interpretation of D7:

- The Training and validation cost are close until a certain C, after which the validation cost starts to raise, the model could be overfitting for values of C beyond this point.

Interpretation of D8:

- The cost function and misclassification curve are the opposites here. But they follow the same trend, beyond a certain C value,
- The best C value is the point where the mean log loss (average cost function) and the misclassification are lowest.

Based on these interpretations, the optimal C value would be the 11th point in the curve which in the np.logspace(-10,5,30) is 1.487532e-5.

Imapct:

The impact of using 10-fold cross-validation instead of a fixed validation set is that it led to less bias and reduced overfitting as all data points were used for validation. But the difference in computation time between these are very big. D7 approx. took 25 minutes whereas D8 approx. took 1 hour and 45 minutes to finish execution.

D10: **Table. Performance of best model from GridSearchCV's best_estimator_**

Optimal regularisation hyperparameter (C) value	0.062102
Cost function value in training set for best C	0.092701
Cost function value in validation set for best C	0.013112
Fraction of the misclassification rate in training set	0.033994
Fraction of the misclassification rate in validation set	0.047123

D11:

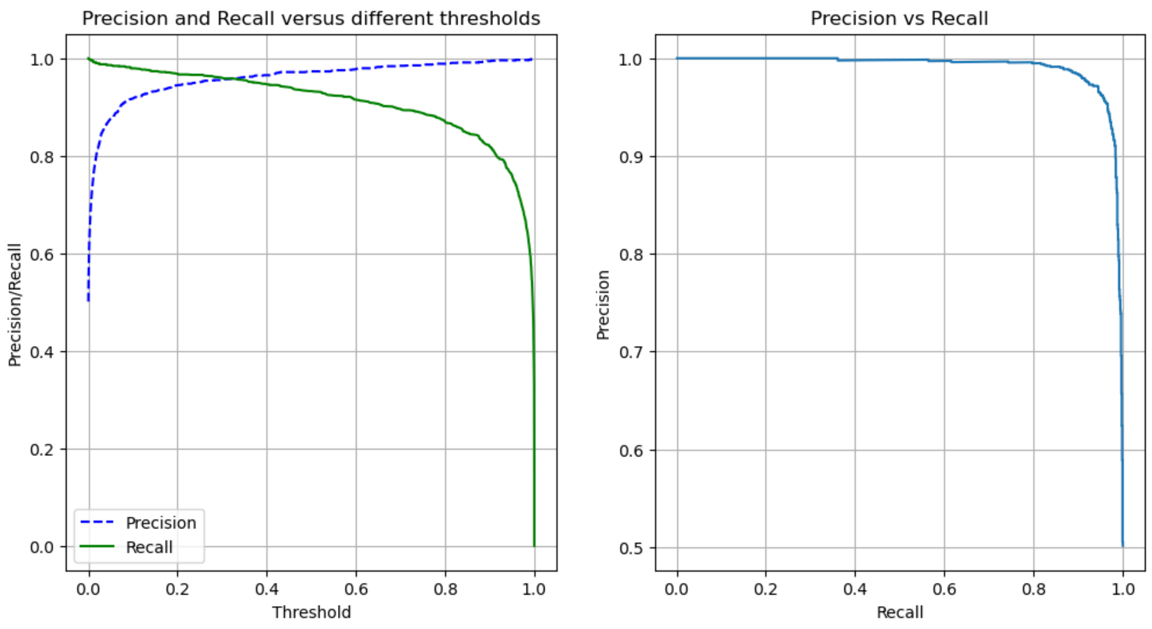
Interpretation:

- The low log loss and misclassification values on both the training and validation set indicates the prediction capabilities of the model, better generalization on unseen data.

Comparison:

- The C value is very different from D8, this could possibly be due to the different optimization algorithms used by the SGDClassifier with the GridSearch and LogisticRegressionCV.
- For D8, the log loss and misclassification rates are slightly higher than the values in D10.

D12: **Fig. Plots of precision and recall of best model.**



Comments and behaviour:

- The Precision and recall curve are closest around 0.25 to 0.3 and looking at the precision versus recall plot, precision falls fast around 85-90% recall. It behaves as intended, as threshold increases, precision increases (being strict and hence classification becomes aggressive) but recall decreases (on the cost of being stricter, we miss some positive cases).

Choice of threshold:

- I would choose a threshold value that's close to [0.25...0.3], as that's where precision and recall are balanced (the two lines are closest to each other). This would make the model be not so aggressive with classification and classify more positive cases. And from the precision vs recall plot, you would want to select a trade-off point right before where the curve starts to fall sharply.

D13:

Comments:

The value that grid search suggested is 0.2727..., this is the point where the precision and recall curve are closest to each other. This is in line with my reflection as 0.2727 ϵ [0.25...0.3]. To cross verify this result we can also calculate the F1 score which is the harmonic mean of precision and recall (weighs both equally). The F1 score is 0.959 for a corresponding threshold of 0.2727 (high score = balanced performance).

D14:

Confusion matrix:

LR1	Sneakers_class [0]	Sandals_class[1]
Sneakers_class[0]	968	32
Sandals_class[1]	87	913

LR2	Sneakers_class [0]	Sandals_class[1]
Sneakers_class[0]	950	50
Sandals_class[1]	62	938

LR3	Sneakers_class [0]	Sandals_class[1]
Sneakers_class[0]	949	51
Sandals_class[1]	63	937

LR4	Sneakers_class [0]	Sandals_class[1]
Sneakers_class[0]	954	46
Sandals_class[1]	40	960

Precision, recall and False Positive Rate:

Model	Precision	Recall	False Positive Rate
LR1	0.966	0.913	0.032
LR2	0.949	0.938	0.05
LR3	0.948	0.937	0.051
LR4	0.954	0.96	0.046

D15:

Comments:

LR1:

It has the highest precision of all models at the cost of losing a little recall. The false positive rate is also the lowest, hence best specificity. Based on these metrics, LR1 has the second-best generalization capacity amongst other models.

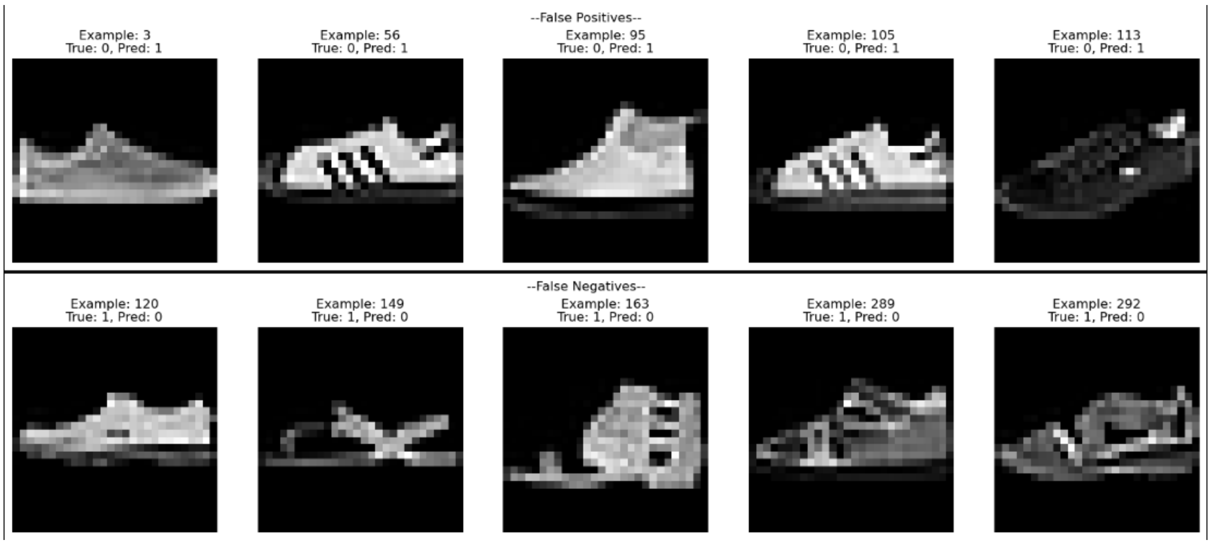
LR2 and LR3:

All their metrics are almost the same and not better than LR1. The possible reasons could be the regularization intensity and the dataset could be optimal without much of overfitting issues. The generalization capacities of LR2 and LR3 are not as good as LR1.

LR4:

This model has the most balanced metrics. The difference between precision and recall is the lowest amongst other models. The false positive rate is better than LR3 and LR4. Hence, it has the best generalization capacity of the four models as it has the best precision to recall ratio (missing very few positive case). Hence on unseen data this model should perform the best.

D16:


--False Positives--
Example: 3 True: 0, Pred: 1
Example: 56 True: 0, Pred: 1
Example: 95 True: 0, Pred: 1
Example: 105 True: 0, Pred: 1
Example: 113 True: 0, Pred: 1
--False Negatives--
Example: 120 True: 1, Pred: 0
Example: 149 True: 1, Pred: 0
Example: 163 True: 1, Pred: 0
Example: 289 True: 1, Pred: 0
Example: 292 True: 1, Pred: 0

- The first 5 images are the False positive results where the true class was 0 (sneaker) but was predicted as 1 (sandal).
- The next 5 images are the False negative results where the true class was 1 (sandal) but was predicted as 0 (sneaker).

D17:

Comments:

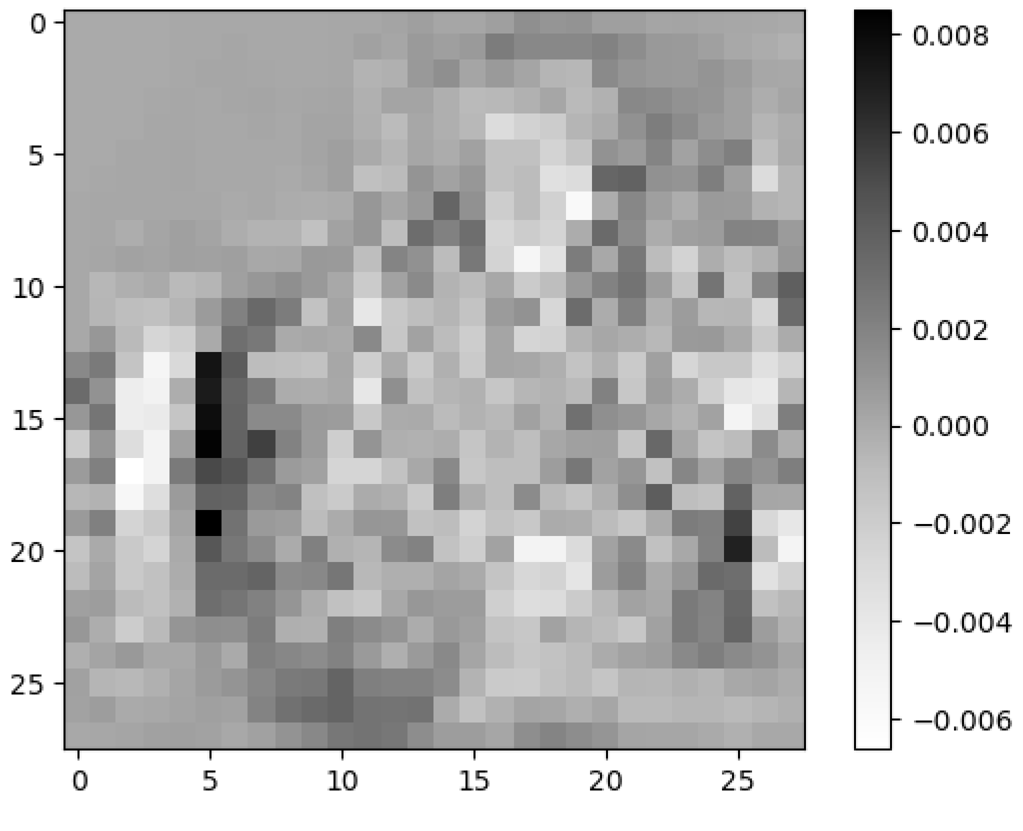
For false positives, the mistakes/wrong classifications could be due to:

- Horizontal inversion of images causing the sneaker to look differently.
- Patterns, colour, shape, or design on the shoe that merge with the black background, causing the image to maybe look like a sandal.

For false negatives, the mistakes/wrong classifications could be due to:

- Some sandals having shapes like shoes, e.g. Woman's heels that are covered on the sides, sandals that are covered (crocs like sandals).
- Sandals that have different design from typical sandals, i.e. some sandals have extra straps that may possibly be mistaken as a shoelace.

D18: **Fig. LR4's estimated weights associated with images.**



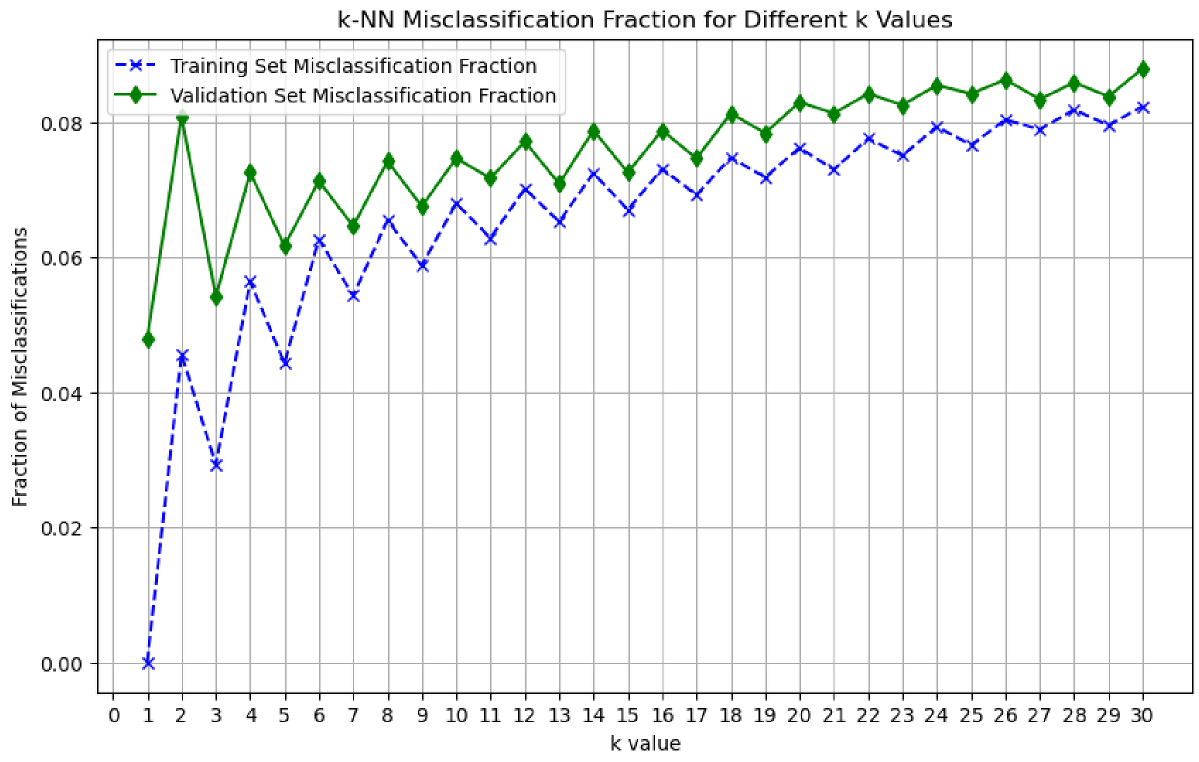
D19:

Comment:

- The area over the toe box and heel are darkest, meaning these areas could have been the main distinguishing feature for the model to classify an image as a sneaker (positive class).
- The area around the ankle is lighter, this could be the model trying to learn sandals (negative class) having designs/straps (e.g. High heels).
- The weights are lightest over the toe box and heel again, meaning that the model could have learnt that all sandals (negative class) have distinctive features at these areas.

D20:

Fig. Misclassification fraction for k-nn model



D21:

- For low k values, the model is underfitting and for higher k values, the model is overfitting. This could be because of high bias when k is small and as for high k values, the model is turning too complex, introducing noise into training data...poor generalization.
- As k value increases, misclassification gradually increases, which indicates degradation in generalization capacity.
- k = 9 is my preferred choice as the difference between misclassification fractions of training and validation sets are not too large as this k value, meaning that no overfitting is occurring.
- One major difference between LR1 without regularization and k-nn models is that in the former, even after convergence, there's a significant gap between the training and validation misclassification rate (potential overfitting) comparatively. Hence k-nn should have better generalization capacity after optimization and with an optimal k value.

D22:

Confusion matrix and performance metrics for k = 9:

k-nn	Sneakers_class [0]	Sandals_class[1]
Sneakers_class[0]	998	2
Sandals_class[1]	147	853

Precision	0.997
Recall	0.853
False Positive Rate	0.002

D23:

Comments:

- Compared to the logistic regression models, the precision of k-nn is clearly high but at the cost of recall, it has the highest difference between precision and recall of all the models. Although the false positive rate is lower, the number of false negatives is very high compared to all these models. This means that a lot of sandals were misclassified.

- Based on this, it is clear the k-nn model requires optimisation and LR4 would still be the best model when it comes to generalization capacity.

D24:

- Dataset: This dataset is not the right dataset to train a model to classify between sandals and sneakers, as this dataset contains other items as well. We could probably find better datasets that is apt for this scenario with more instances.
- Experimenting with parameters: Provided a powerful machine, we could experiment with different values for parameters that affect the model and select the best one. (e.g. Learning rates)
- Image manipulation: Instead of this using just the images present in the dataset, we could try to manipulate the present images in different ways, this could be rotating/flipping the image. This would create variations in the dataset which could help while training the model.
- Collaborate models: We could use a pre-trained model and extract its features/parameters to train a new model, iterating this process for several times would lead to better model each iteration.
- Learning techniques: We could try different learning techniques other than supervised learning.