# Project A Report

# Problem 2

## Section 1: Methods for Sneaker-Sandal

Experiment Design:

The given data set contains 12000 examples, I divide them into training and validation set. I pick the first 8000 examples as training set , the rest 4000 examples as validation set. I do not use cross validation, instead I use fixed validation set because we are working on a very large data set, and the way I divide the data set makes the validation set large enough to neutralize randomness, and working on fixed validation set save a lot of time since we are training on large data set.

Method 0: Baseline: raw pixel features, fed into a Logistic Regression classifier

There is no feature transformation for baseline. The parameters input are raw 28*28 pixels with value between 0 and 1, and the output is 0 and 1 representing sneaker and sandal. They are fit using logistic regression. There is no concern about overfitting because we restrict the training with max iterations and penalty parameter C. There is no concern about convergence of the optimization because we restrict max iterations. The hyperparameters are C = 10, max_iter = 40, this is done via grid search. Candidate value for C is 30 values distributed evenly on $10^{-6}$ to $10^{6}$, candidate value for max_iter is integers from 20 to 60. I was trying to optimize log loss so I pick the complexity that gives the lowest log loss.

Method 1: A feature transform of your own design, fed into a Logistic Regression classifier

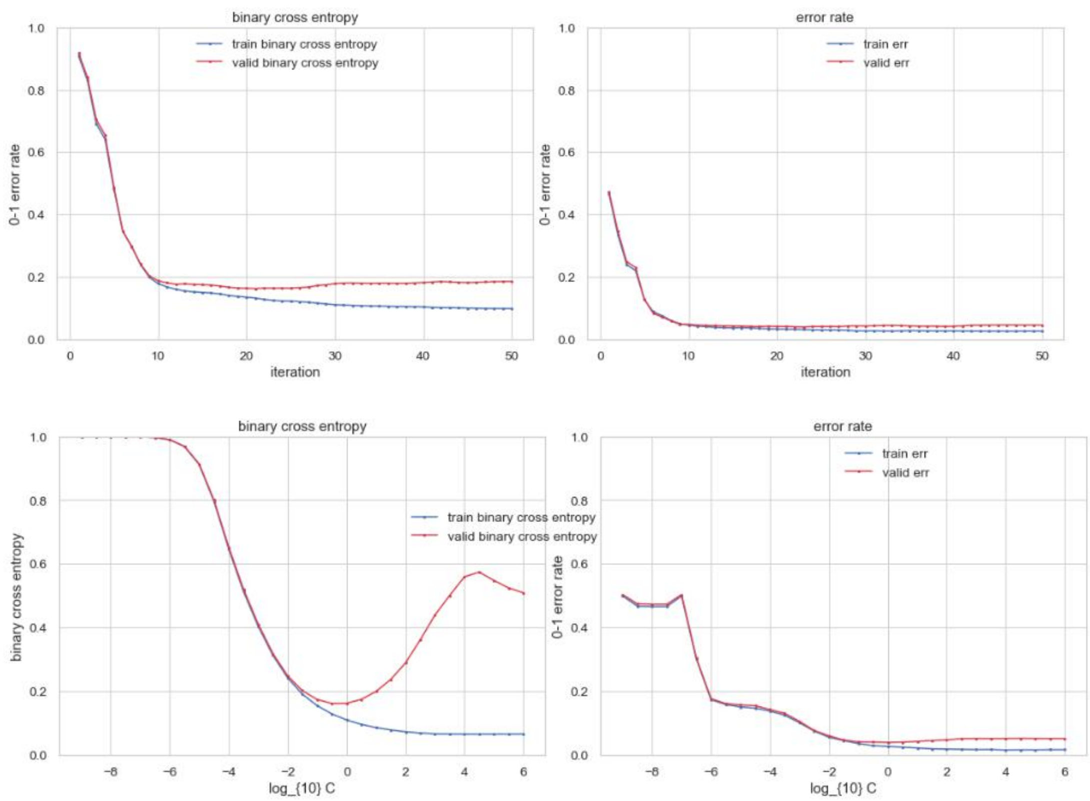
The feature transform is polynomial transform of degree 2 with only interaction between features. I think degree 2 polynomial establish relation between pixels so we can better capture spatial trends, and I do not want feature to interact with itself because that does not suggest any spatial information, so I set interaction_only to true, so we only get features like $X_1X_2$. They are fit with logistic regression, the first step is polynomial transformation, the second step is fed into logistic regression model. There is no concern of overfitting because the validation log loss will increase if iterations are too large, there is concern of convergence of the optimization because the log loss will not converge with iteration increase. The hyperparameters are C = 10, max_iter = 40, this is done via grid search. Candidate value for C is 30 values distributed evenly on $10^{-6}$ to $10^{6}$, candidate value for max_iter is integers from 20 to 60. I was trying to optimize log loss so I pick the complexity that gives the lowest log loss.

Method 2: Another feature transform of your own design, fed into a Logistic Regression classifier or some other classifier

There is feature transform is using sklearn.MinMaxScaler, use scaler to make each feature is scaled to unit size, so when training the model with MLP the weight can be more distinctly distributed. The parameters are scaled pixels value, and they are fit into a single layer MLP using stochastic gradient decent. There is no concern of overfitting as we add penalty alpha. There might be concern of convergence of the optimization, we need to select appropriate batch size and learning rate to make it converge. They hyperparameters are batch size = 100, learning rate = 0.2, they are selected via grid search. Considered values for batch size are [100, 500 , 2000], considered values for learning rate are [ 0.2, 0.5, 1.0], the performance metrics I optimize is the log loss and heldout error on validation set

## Section 2: Results for Sneaker-Sandal

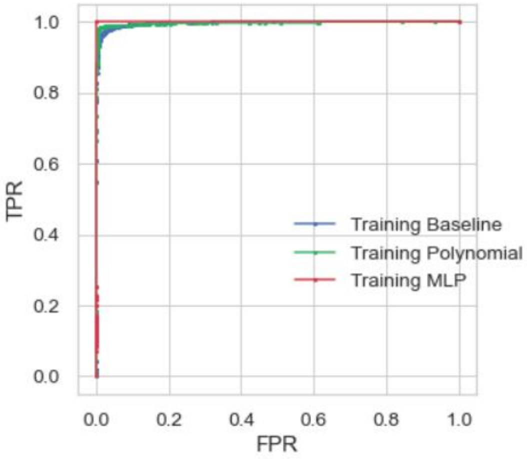
**Figure: evidence of hyperparameter selection for Baseline**

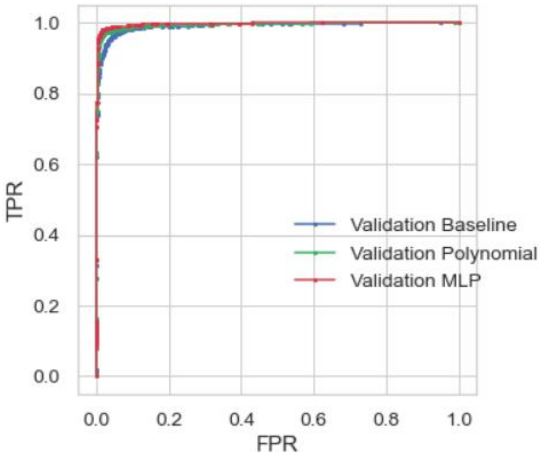



For max iterations, the value should be greater than 10, and between 10 and 50 according to the graph. With the iterations increase, the log loss and error rate increase, if max iterations is too large there will be overfitting as we can see the training log loss and error approach 0. So I chose max iteration 40.

For penalty term C, I did grid search on log space from $10^{-6}$ to $10^{6}$, to achieve lowest log loss I chose C = 0, and if C is too large there will be overfitting.

**ROC Curve: Left is training set, right is validation set**

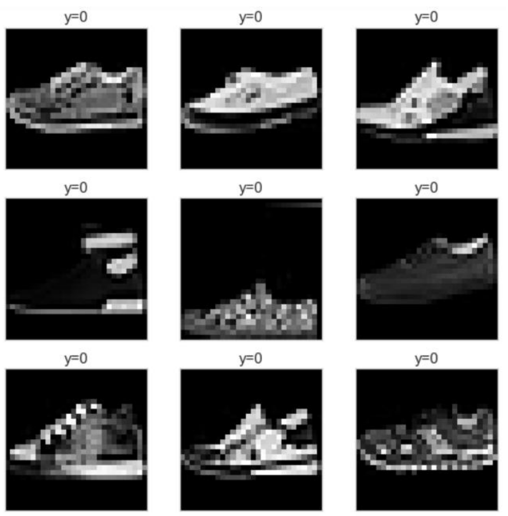


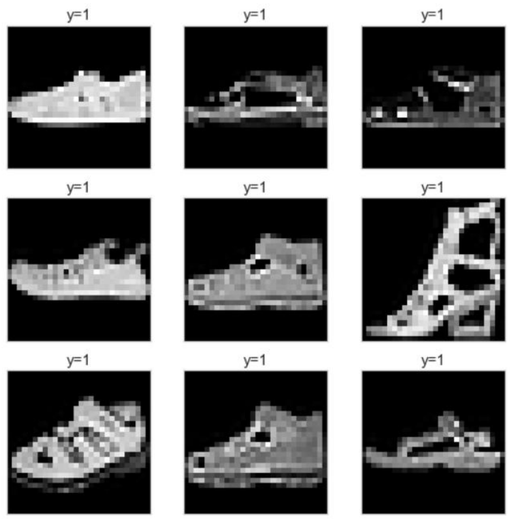




Blue curve is method 0 baseline. Green curve is method 1 polynomial transform of degree 2. Red curve is training with MLP.

The MLP perform best among the 3 methods, its ROC curve is above the other 2 methods, and the area above the curve is very small.

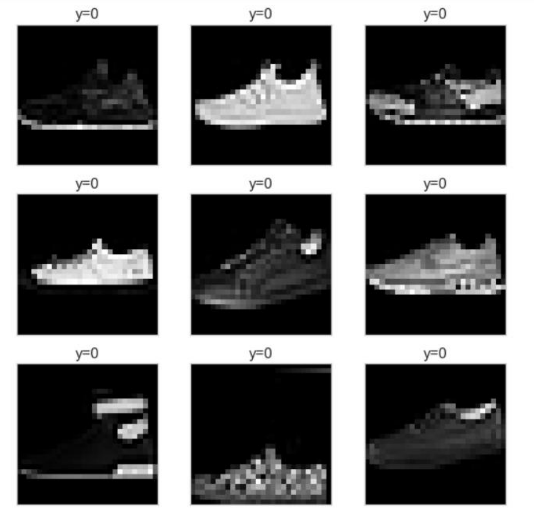
**Mistakes of MLP model, Left is false positive, Right is false negative**

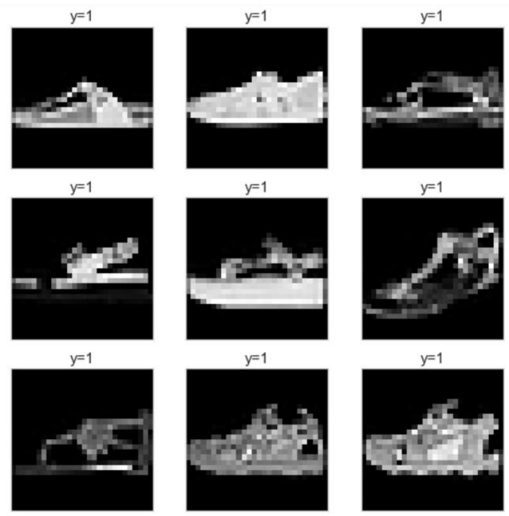



Some of the mistakes are because the image is ambiguous or too dark, like image 4,5 on the left and image 2 3 on the right graph. And for shoes with complicated textures and patterns like image 7 8 9 on the left and 6 7 on the right, the MLP may make mistakes. The angel of the shoes may also cause mistakes, like the image 7 on the right graph is taken from 45 degree above the sandal rather than side view of the shoes.

**Mistakes of Polynomial model, Left is false positive, Right is false negative**




Some of the mistakes are because the image is ambiguous or too dark, like the image 1,3,7,8 on the left graph. And for image that is too bright there are also mistake like image 2,4 on the left and 1,2,5 on the right. Images with complicated texture and patterns are also likely to cause mistakes, like image 5 on the left, image 8, 9 on the right.