Machine Learning Project
Digit classification with the Kernel Perceptron

Natalia Kartasheva (943350)

I declare that this material, which I now submit for assessment, is entirely my own work and has not been taken from the work of others, save and to the extent that such work has been cited and acknowledged within the text of my work. I understand that plagiarism, collusion, and copying are grave and serious offences in the university and accept the penalties that would be imposed should I engage in plagiarism, collusion or copying. This assignment, or any part of it, has not been previously submitted by me or any other person for assessment on this or any other course of study.

Introduction

Our task is as follows:
Download the MNIST dataset for multiclass classification of handwritten digits. Implement the kernel Perceptron from scratch and run it to train 10 binary classifiers, one for each of the 10 digits (one-vs-all encoding), using the polynomial kernel. In order to extract a binary classifier, do the following: run the algorithm for a given number of epochs (e.g., cycles over the entire training data) on a random permutation of the training set and collect the ensemble of predictors ω_t used by the Perceptron when predicting each training datapoint (x_t, y_t). Then use:

1. the average of the predictors in the ensemble;
2. the predictor achieving the smallest training error among those in the ensemble.

Based on the given split of the data in training and test set, evaluate the multiclass classification performance (zero-one loss) for different values of the number of epochs (go up to at least 10 epochs) and the degree of the polynomial (go up to at least degree 6). In order to use the 10 binary classifiers to make multiclass predictions, use the fact that binary classifiers trained by the Perceptron have the form $h(x) = sgn(g(x))$ and predict using $argmax_i(g_i(x))$ where g_i corresponds to the binary classifier for class i.

Dataset Overview

MNIST Handwritten Digits dataset is used for this task. It contains images of digits taken from a variety of scanned documents, normalized in size and centered. This makes it an excellent dataset for evaluating models, allowing the developer to focus on the machine learning with very little data cleaning or preparation required. Each image is a 28 by 28 pixel square (784 pixels total). The dataset contains 60,000 images for model training and 10,000 images for the evaluation of the model. The dataset consists of two files:

- mnist_train.csv - contains 60,000 training examples and labels
- mnist_test.csv - contains 10,000 test examples and labels

The first value in the dataset is the label (a number from 0 to 9) and 784 values are the pixel values (a number from 0 to 255).

MyPerceptron

The Percentron as a class is created using the $fit()$ and $avg\_predict()$ functions.

Binary Classification

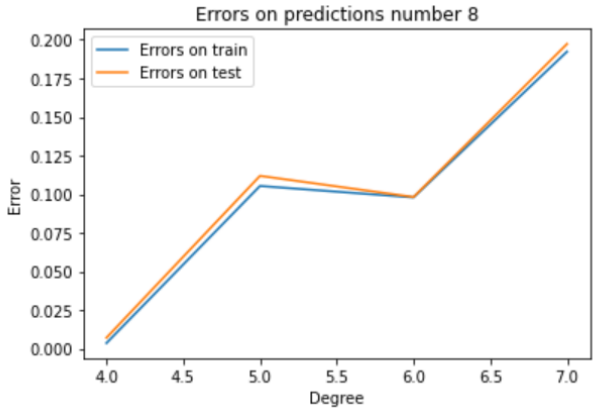
Fitting the binary model with the Average Predictor

Defining S_t as the active set of indices of examples where was made a mistake at time t, S_0 by definition is the empty set. Also by definition of the Kernel Perceptron Alghoritm $S_1 = 1$. At time $t = 2$ is possible

that $S_2 = 1, 2$ or $S_2 = 1$ if the second training example was correctly classified. At time $t = n$ the active set is S_n and defined as $\omega_{s=}\frac{n+1-s}{n}$ the average importance of the misclassified training example (x_s, y_s) the classifier becomes

$$h_\omega = sgn(\sum_{s \in S} y_s \omega_s K(x_s \omega_s, x))$$

By definition $\omega_1 = 1$ and $\omega_t < 1 \forall t \neq 1$. This give a bigger importance to the set of early misclassified examples. Heuristically the advantages of this predictor is backed by the fact that the major reduction in the training error in the binary classification was observable in the early time stamp. Also an advantages could be a reduction in the probability of overfitting for a large enough number of epochs. In other words, at this step, we create empty sets of indices for the train data as well as for the test data, and fill them with the indices of points at which the prediction made with a polynomial is wrong. This is then validated with polynomials of different degrees to figure out which of them is a better predictor for any given digit. The number of epochs is fixed.



Here we can see that the four optional degrees of the polynomial kernel are compared using the plots depicting the test and training errors. Binary classification predictor can be used iteratively to predict all 10 digits (classes). The plots show that the lower degree of the polynomial always provides better predictions with binary classification.

Finding a Best Predictor

At time $t = T$ we have S_1, S_2,...,S_T active sets of indices and we know that $S_1 \subseteq S_2$ ... $\subseteq S_T$. Defining the set of classifier as

$$h_1(x) = sgn(\sum_{s \in S_1} y_s K(x_s,x))$$

$$h_2(x) = sgn(\sum_{s \in S_2} y_s K(x_s,x))$$

...

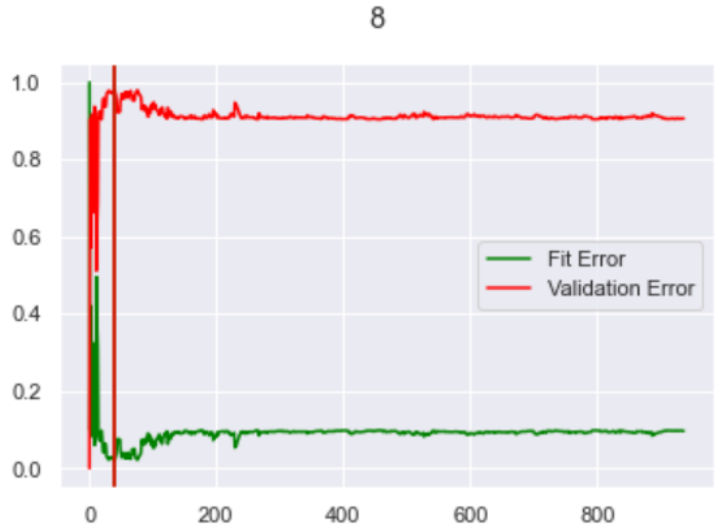
$$h_n(x) = sgn(\sum_{s \in S_n} y_s K(x_s,x))$$

Given $\Omega = h_1, h_2$,...,h_T the best predictor is the one minimizing the training error formally

$$h_b(x) = argmin_{h \in \Omega} \frac{1}{T} \sum_{t=1}^{T} L(h(x_t), y_t)$$

```
Min loss on ind 885 train: 0.0006
Min loss on ind 739 test: 0.0032
```

We take one digit at a time and find the index where the total error is minimal. We can see, that in our example with digit 6, the minimum achieved error of the training set is 0.0006, and on the test set - 0.0032.

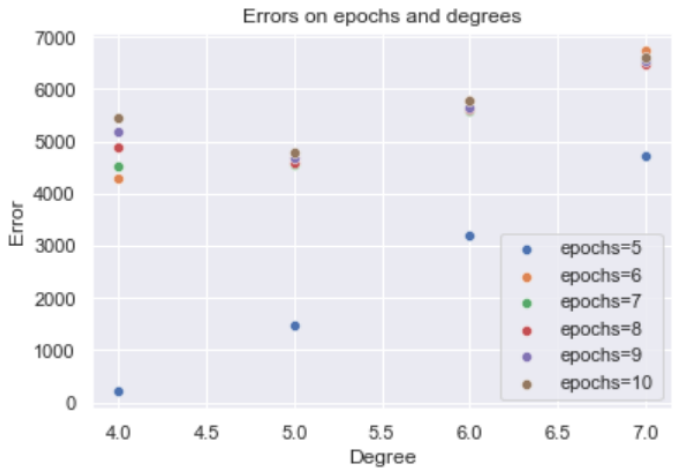


The entire range of error frequencies for all predictors on the train and test arrays can be plotted against each other for comparison. (Note, the combination of the polynomial kernel degree and the number of epochs is pre-defined at the previous step). The best predictors are marked with vertical lines.

Multiclass Classification

Fitting and finding an Average Prediction

The multiclass model is built on the basis of the average binary predictor and validated with zero-one loss. We compare a range of different polynomial kernel degrees (up to 7) and a range of numbers of epochs (up to 10).



Then the errors are compared with the help of a plot. It's obvious that the number of epochs 5 produces significantly poorer results than the rest, and the difference between other options is compensated by the change in the degree. The lower degrees again win, degree = 4 being the best option.

Conclusion

In this work, we have implemented the Kernel Perceptron from scratch and we have built a binary classification model and a multiclass model to predict a handwritten digit and saw that a low polynomial kernel degree = 4 produces the best predictions. On the other hand, a big number of epochs leads to overfitting, so this number should also be kept low.