1. Introduction

In this work we study batch learning and how this batch algorithm behaves as we use different data sets (permutations of the same data set), also analyzing how much training is enough and how to optimize the parameters for learning model. The objective of this work is to get familiar with logistic regression, plotting performance curves and understanding early stopping and parameters/model optimization.

In the following we describe the main results of this work in which different scenarios of interest were defined and performance curves were plotted. The scenarios, the plots and the respective comments and conclusions are presented in the following sections, but first we briefly present the algorithm itself and the datasets used as input during this experiments.

2. The problem

We implemented logistic regression for predicting whether or not emails are to be considered spam. The data set consisted of a matrix 2001x2000, where the first value in each row is the class label, in \{0,1\}, where 0 is "not spam" and 1 is "spam". Remaining 2000 values in each row are also \{0,1\} values that indicate presence 1 or non-presence 0 of words in the message.

Figure 1 shows the presence (dark points) or absence (blank spaces) of the features in each row of the matrix. Figure 1 is the same dataset but now with its rows permuted.
Figure 1: Data set plotted in MATLAB using the command "imagesc(1-data); colormap gray"

Figure 2: Same data set plotted in MATLAB, but now with the examples/email (rows of the matrix) permuted using the command "p=randperm(2000); permData=data(p(:,:)); imagesc(1-permData); colormap gray;"
3. Experiment 1 – Studding early stopping:

In our first experiment, we implemented the logistic regression for training our model, using the first $\frac{3}{4}$ of the data set as training set and the remaining $\frac{1}{4}$ as testing set. Our prediction function was a sigmoid function called here “$y_{\text{hat}}_t = \frac{1}{1 + \exp(-a_{\text{hat}})}$”, where $a_{\text{hat}}$ is the linear activation “$a_{\text{hat}} = (W_i'X'i)'$”. The loss function here is defined as the logistic loss “$\text{Logistic Loss} = \log(1+\exp(a_{\text{hat}}))-y.*a_{\text{hat}}$”, where “$y$” is the class label of the batch examples.

The loss gradient “$\text{Loss Grad} = (y_{\text{hat}}_t-y)'*X'i$” was used for updating the weights, using gradient descent, “$W_i = W_i - (1/\text{Numb of examples Training})*\eta* ((y_{\text{hat}}_t-y)'*X'i)$”, where $\eta$ is the learning rate of the algorithm and $\text{Numb of examples Training}=1500$, in our case.

The main question to be answered here was when to stop training. In this experiment our stopping criteria was the absolute value of the logistic loss gradient. We evaluated the performance of the algorithm for 5 different values of the gradient, where we stopped the training phase when reached these thresholds of $[10^0, 10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}]$.

![Early Stopping for Logistic Loss](image)

**Figure 3:** Early stopping using the absolute value of the gradient loss as the stopping criteria to avoid overtraining.
It is easy to see in Figure 3 how much the loss during the training phase drops the longer you train, achieving losses very close to zero when running the algorithm until the gradient reaches $10^{-4}$. After reaching the stopping criteria, we stopped training and used the new set of weights acquired in the training phase to predict in the testing set. We got the best result at $10^{-2}$ and observed a poorer performance at the testing set when using the model trained until $10^{-3}$ and $10^{-4}$, configuring overtraining at these last 2 points. In this results, no cross validation or regularization were used and we set $\text{eta}$ to be equal to 0.3.

4. Experiment 2 – Changing stopping criteria:

In this experiment we used to different stopping criteria: (a) the 1-norm of the gradient and (b) the 2-norm of the gradient loss. For each pass on the batch we updated the weight, incurred in loss and recalculated the gradient loss. When the stopping criteria (a) or (b) reached a threshold, we stopped the training phase and applied the given model to the data set and averaged the loss. We varied the threshold $[10^0, 10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}]$. Figure 4 shows the results.

Figure 4: Early stopping using the 1-norm and 2-norm of the gradient loss as the stopping criteria to avoid overtraining.
Using the 1-norm and 2-norm of the gradient loss as the stopping criteria to avoid overtraining, it is possible to see that for the 2-norm stopping criteria, the loss is smaller for bigger trainees. For these experiments, no cross validation was used.

5. Experiment 3 – Implementing cross-validation and optimizing the model:

In the third experiment, we implemented the 5-fold cross validation with 3-way split, using a piece of the data set for training, a piece for validating and a third for testing. We divided the training set by 5 (5x300 examples) plus the data set with 500 examples. From the 5 divisions of the training set, 4 were used for training and 1 for validation. We ran 5 experiments where every fold was used 4 times as training and once as validation. This was done for every value of the learning rate $\eta$ tested. For every value of $\eta$ we reported the average logistic loss over the 5 losses measured during the 5 validation phases.

![Choosing the best eta (small eta)](image)

Figure 5: Very small difference between the tested etas, led to very small difference in the average final loss.
Figure 5 shows the values of \( \eta \) evaluated and its respective average losses. It is possible to see a difference, but for the suggested \( \eta \) update (\( \eta_0 \alpha^{\text{Pass}^{-1}} \)), the values of \( \eta \) are very close to each other, making the weights to be updated very slowly and leading to a very similar final result and very similar average losses. We can see by this figure that the difference in the losses is only visible at the order of \( 10^{-4} \). For all experiments in this section, we ran the training phase to the precision of gradient loss equal to \( 10^{-2} \).

We then tried another experiment for 20 different values of \( \eta \), but for a much larger range of the learning rate, from 15 to 0.0163. For each value, we trained in 4 folds and validated in the 1 fold left for 5 permutation of the folds, as we did before for the first set of small \( \eta \)s. Figure 6 reports the average loss over the 5 validations. Larger difference between tested learning rates allowed a better tune of this parameter, since its impact now is larger in the weights update.

Figure 6: Average logistic loss over 5 folds as a function of the Learning rate. Larger difference between tested learning rates allowed a better tune of this parameter, since its impact now is larger.
It was interested also to notice during the experiments that larger learning rates leaded to the gradient to converge faster, since the weights were updated much faster. But, as we can see at Figure 6, there is a tradeoff between the learning rate and accuracy, which means that we cannot choose a big $\eta$ arbitrarily.

After plotting the curve at Figure 6, we choose the best and the worse learning rate (and it’s respective set of weights) measured in our cross validation step. For each one of these models (the best and the worse) we ran 10 times over the testing set. Table I reports the average and standard deviation (STD) over the 10 runs for both models.

<table>
<thead>
<tr>
<th>Model</th>
<th>Best</th>
<th>Worst</th>
</tr>
</thead>
<tbody>
<tr>
<td>Learning rate (eta)</td>
<td>5.1084</td>
<td>0.4138</td>
</tr>
<tr>
<td>Average Loss</td>
<td>0.0331</td>
<td>0.0668</td>
</tr>
<tr>
<td>STD of Loss</td>
<td>0.0066</td>
<td>0.0399</td>
</tr>
</tbody>
</table>

6. Future Work:

Given the limited time and the huge computational effort to perform this work, the following experiments were studied implemented but no result was generated until the deadline for this work.

- Analysis of the impact of shrinking;
- Regularization implementation;

7. Conclusions:

In this work we studied logistic regression for classification problems and applied to a case of study where it was used to estimate if an email is or not a spam. We analyzed the case for overtraining and how to avoid it, using early stopping. Different early stopping criteria were also used as 1-norm and 2-norm of the gradient loss. We also learned how to tune our model parameters in order to optimize our algorithm by using cross validation. More specifically, the parameter tuned was the learning rate. Final results using the optimized model were also presented.