Homework 6

Jakob Horvath - u1092049

December 3, 2020

1 Logistic Regression

1.

$$log(1+e^{-y_iw^Tx_i})$$
$$\frac{d}{dw}log(1+e^{-y_iw^Tx_i}) = \frac{d}{dw}log(1+e^{-y_iw^Tx_i}) * \frac{d}{dw}(1+e^{-y_iw^Tx_i})$$
$$= \frac{1}{1+e^{-y_iw^Tx_i}} * (0 + -y_ix_ie^{-y_iw^Tx_i}) = \frac{-y_ix_ie^{-y_iw^Tx_i}}{1+e^{-y_iw^Tx_i}}$$

2.

$$min_w(log(1+e^{-y_iw^Tx_i}) + \frac{1}{\sigma^2}w^Tw)$$

3.

$$\frac{d}{dw}(log(1+e^{-y_iw^Tx_i}) + \frac{1}{\sigma^2}w^Tw)$$
$$= \frac{d}{dw}log(1+e^{-y_iw^Tx_i}) + \frac{d}{dw}(\frac{1}{\sigma^2}w^Tw)$$
$$= \frac{-y_ix_ie^{-y_iw^Tx_i}}{1+e^{-y_iw^Tx_i}} + \frac{d}{dw}(\frac{1}{\sigma^2}w^Tw)$$
$$= \frac{-y_ix_ie^{-y_iw^Tx_i}}{1+e^{-y_iw^Tx_i}} + \frac{2w}{\sigma^2}$$
$$\nabla J^t(w) = \frac{-y_ix_ie^{-y_iw^Tx_i}}{1+e^{-y_iw^Tx_i}} + \frac{2w}{\sigma^2}$$

4. Given a training set $S = \{(x_0, y_0), ..., (x_n, y_n)\}$, $x \in \mathbb{R}^n$, $y \in \{-1, 1\}$

```
1. Initialize $w^0 = \epsilon \in \mathbb{R}^n$, for some small $\epsilon$
2. For epoch = 1...T:
    1. Randomly shuffle the set $S$
    2. For each example in $S$
        1. Treat $(x_i, y_i)$ as a full dataset and take the derivative of the objective at
           the current $w^{t-1}$ to be $\nabla J^t(w^{t-1})$
               $$\nabla J^t(w) = \frac{-y_i x_i e^{-y_i w^T x_i}}{1 + e^{-y_i w^T x_i}} + \frac{2w}{\sigma^2}$$
        2. If $y_i * x_i * w_i \leq 0$:    $w^t \leftarrow w^{t-1} - \gamma_t \nabla J^t(w)$
3. Return final $w$
```

2 Experiments

1. All algorithms were written in Python 3. Every algorithm used the same transformed data. This transformation involved recreating the original dense feature vectors to be sparse (inserting 0's for an attribute if it wasn't present beforehand), as well as folding in the bias term.

   (a) **Support Vector Machine**
   SVM used stochastic sub-gradient descent and updated its learning rate every epoch using the method described in the instructions ($\frac{\gamma}{1+t}$). The unique aspect of the algorithm is in its stopping criteria. A threshold variable was adjusted until the value of 0.0001 was determined to offer a good balance between accuracy and runtime. The algorithm only stops running when the difference between successive predictions ($y_i * x_i * w_i - y_{i-1} * x_{i-1} * w_{i-1}$) is smaller than the threshold unit. This lead to many epochs and most likely a longer runtime than sub-gradient descent would typically allot, but was also the only way I found to produce reasonably accurate results.

   (b) **Logistic regression**
   In order to combat numerical over/underflow, the logistic regression algorithm I implemented ensures that the product within the exponential term ($-y_i * x_i * w_i$) is always within the range [-10, 10]. Since regular stochastic gradient descent is implemented, it was important to shuffle the data with each epoch. This also allowed for a hard-coded number of epochs, and 5 was found to be the optimal number for this value. Finally, the weight vector is only updated when the predicted label is incorrect.

   (c) **SVM over trees**
   The SVM over trees algorithm uses the same previously described SVM function. Using decision trees to transform the feature vectors for SVM to work with turned

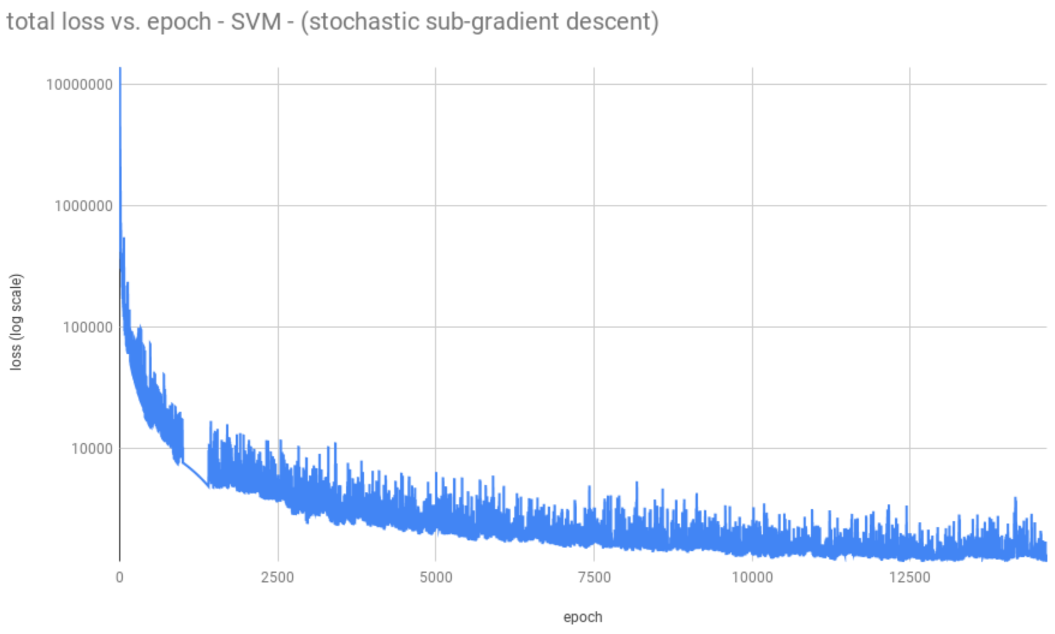
out to reduce the overall prediction accuracy by a significant margin. Further, the terrible test set accuracy suggests that there was major over-fitting going on. This could be due to the small stopping threshold of the SVM function.

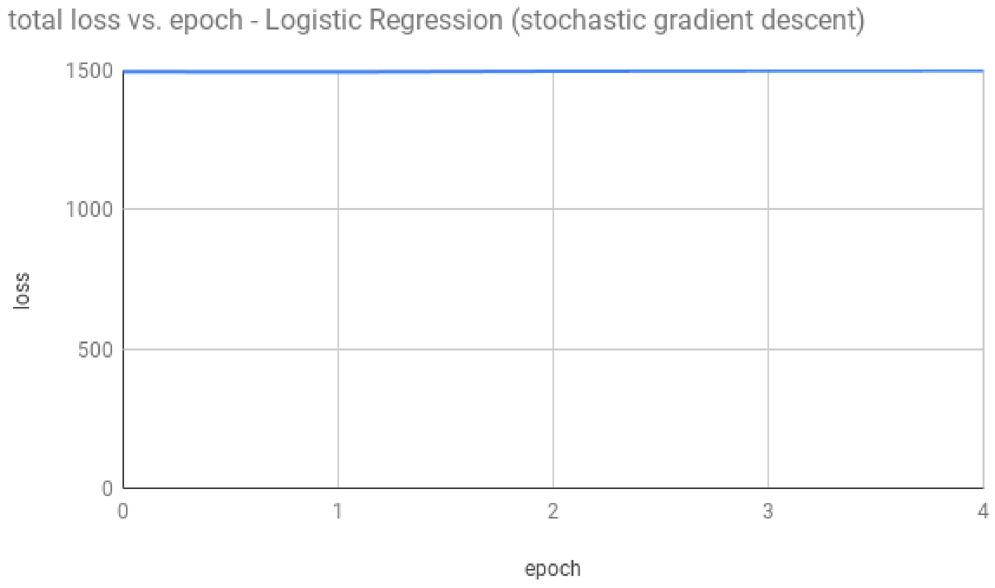
2. -

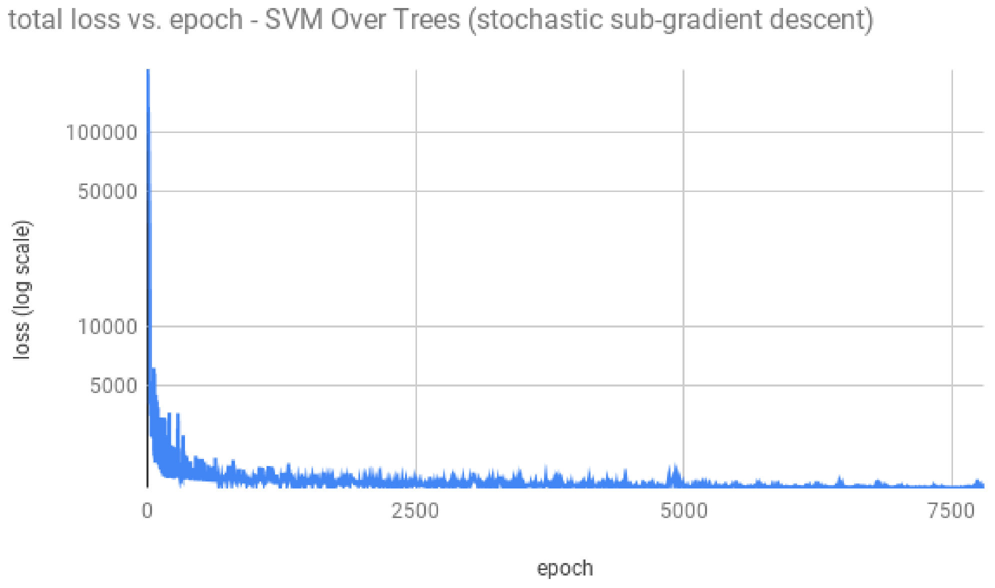
	Best hyper-parameters	Average Cross-validation accuracy	Training accuracy	Test accuracy
SVM	$\gamma_0 = 1$, $C = 1000$	0.75561	0.79552	0.78136
Logistic regression	$\gamma_0 = 0.01$, $\sigma^2 = 10000$	0.74305	0.80000	0.76344
SVM over trees	$\gamma_0 = 0.001$, $C = 1000$, $d = 8$	0.68027	0.69013	0.54301

Table 1: Results table

3. -




total loss vs. epoch - Logistic Regression (stochastic gradient descent)
1500
1000
500
0
loss
0
1
2
3
4
epoch


total loss vs. epoch - SVM Over Trees (stochastic sub-gradient descent)
100000
50000
10000
5000
loss (log scale)
0
2500
5000
7500
epoch