Dual Coordinate Descent Method for Linear SVM

Roethel Albert, Brynda Marek

April 2021

1 Introduction

Support Vector Machines Classifier is a Machine Learning algorithm that aims to create such hyperplane that would separate observations that belong to different classes. More formaly, given a set of labeled observations $(\boldsymbol{x_i}, y_i)$, where $i \in \{1, 2, 3, ..., l\}$, $\boldsymbol{x_i} \in \mathbb{R}^n$, $y_i \in \{-1, +1\}$, we want to find such $\boldsymbol{w} \in \mathbb{R}^n$ and $b \in \mathbb{R}$ such that

$$\boldsymbol{x}_i^T \boldsymbol{w} + b \begin{cases} > 0 \text{ if } y_i = +1 \\ < 0 \text{ if } y_i = -1 \end{cases} . \tag{1}$$

Of course, finding of such a hyperplane is possible if and only if two classes are lineary separable. That is why in practice we allow for some misclassifications by using following unconstrained optimization problem with L2 regularization:

$$\min_{\boldsymbol{w}} \frac{1}{2} \boldsymbol{w}^T \boldsymbol{w} + C \sum_{i=1}^{l} \max(1 - y_i \boldsymbol{w}^T x_i, 0)^2. \tag{2}$$

In this formula, called primal problem, $C \geq 0$ is penalty parameter, and $x_i^T \leftarrow [x_i^T, 1]$, $w^T \leftarrow [w^T, b]$.

Starting from the L2 SVM optimization function:

$$\begin{gathered} f(\boldsymbol{w}) = \frac{1}{2} \|\boldsymbol{w}\|^2 + \frac{C}{2} \sum_{i=1}^{l} \xi_i^2 \\ \text{subject to } y_i(\boldsymbol{x_i} \boldsymbol{w} + b) \geq 1 - \xi_i \text{ i = 1,...,l} \end{gathered} \tag{3}$$

We can introduce primal Lagrangian with Lagrangian multipliers $\alpha_i \geq 0$:

$$L_p(\boldsymbol{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}) = \frac{1}{2} \|\boldsymbol{w}\|^2 + \frac{C}{2} \sum_{i=1}^{l} \xi_i^2 - \sum_{i=1}^{l} \alpha_i (y_i(\boldsymbol{x_i} \boldsymbol{w} + b) - 1 + \xi_i) \tag{4}$$

To get the optimal solution the following conditions must be satisfied:

$$\frac{\partial L_p}{\partial \boldsymbol{w}} = \boldsymbol{w} - \sum_{i=1}^{l} \alpha_i \beta_i \boldsymbol{x_i} = 0 \tag{5}$$

$$\frac{\partial L_p}{\partial b} = \sum_{i=1}^{l} y_i \alpha_i = 0 \tag{6}$$

$$\frac{\partial L_p}{\partial \xi_i} = C\xi_i - \alpha_i = 0 \tag{7}$$

$$\alpha_i(y_i(\boldsymbol{x_i}\boldsymbol{w} + b) - 1 + \xi_i) = 0 \text{ i = 1,...,l} \tag{8}$$

Substituting above into primal L2 SVM Lagrangian we obtain its dual form:

$$\text{minimize } L_D(\boldsymbol{\alpha}) = \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} \alpha_j \alpha_j y_i y_j (\boldsymbol{x_i}\boldsymbol{x_j} + \frac{\delta_{ij}}{C}) - \sum_{i=1}^{l} \alpha_i \tag{9}$$

$$\text{subject to } \sum_{i=1}^{l} y_i \alpha_i = 0, \tag{10}$$

$$\text{and } \alpha_i \geq 0 \text{ i = 1,...,l} \tag{11}$$

We can rewrite it using matrix notation:

$$L_D(\boldsymbol{\alpha}) = \frac{1}{2}\boldsymbol{\alpha}^T(\boldsymbol{y}\boldsymbol{y}^T \circ XX^T + \frac{1}{C}I)\boldsymbol{\alpha} - \mathbf{1}^T\boldsymbol{\alpha} \tag{12}$$

Where $\circ$ element-wise matrix multiplication.
Thus it can be transform this form:

$$\begin{aligned} \min_{\boldsymbol{\alpha}} f(\boldsymbol{\alpha}) &= \frac{1}{2}\boldsymbol{\alpha}^T\overline{Q}\boldsymbol{\alpha} - \boldsymbol{e}^T\boldsymbol{\alpha} \\ \text{subject to } 0 &\leq \boldsymbol{\alpha} \end{aligned} \tag{13}$$

where $\overline{Q} = Q + D$ (of size $l + 1 \times l + 1$), $Q_{ij} = y_i y_j \boldsymbol{x}_i^T \boldsymbol{x}_j$ and $D_{ii} = \frac{1}{2C}$.

2 Dual Coordinate Descent Algorithm

Algorithm proposed in [2] tries to solve dual problem (13) by splitting it into many smaller problems and iteratively converge to the solution. Algorithm starts with initial $\boldsymbol{\alpha}^0 = \mathbf{0}$ and $\boldsymbol{w} = \mathbf{0}$. Optimization process comprise of two loops/iterations: the outer one, and the inner one. In kth step of the outer loop we update the entire $\boldsymbol{\alpha}$ from $\boldsymbol{\alpha}^k$ to $\boldsymbol{\alpha}^{k+1}$. In the ith step of the inner loop we calculate the new value of ith component of $\boldsymbol{\alpha}^k$ (namely we update $\boldsymbol{\alpha}_i^k \to \boldsymbol{\alpha}_i^{k+1}$). So each outer loop consists of l inner loops. Each inner loop solves following one-variable problem: find such d which maximizes

$$f(\boldsymbol{\alpha}^{k+1} + d\boldsymbol{e}_i) \text{ subject to } 0 \leq \boldsymbol{\alpha}_i^k + d \tag{14}$$

where $\boldsymbol{e}_i$ is a zero vector in $\mathbb{R}^l$ with 1 in ith position. In this way we only modify the ith component of $\boldsymbol{\alpha}$ vector.

After substitution we have:

$$\begin{aligned} f(\boldsymbol{\alpha}^{k,i} + d\boldsymbol{e}_i) &= \frac{1}{2}(\boldsymbol{\alpha}^{k,i})^T Q \boldsymbol{\alpha}^{k,i} + dQ_i\boldsymbol{\alpha}^{k,i} + \frac{1}{2}d^2 Q_{ii} - \boldsymbol{\alpha}_i^{k,i} - d \\ &= \frac{1}{2}d^2 Q_{ii} + d(Q_i\boldsymbol{\alpha}^{k,i} - 1) + \frac{1}{2}(\boldsymbol{\alpha}^{k,i})^T Q \boldsymbol{\alpha}^{k,i} - \boldsymbol{\alpha}_i^{k,i} \\ &= \frac{1}{2}d^2 Q_{ii} + d(Q_i\boldsymbol{\alpha}^{k,i} - 1) + \mathrm{c} \end{aligned} \tag{15}$$

Note that $\boldsymbol{\alpha}^{k,i}$ is the vector $\boldsymbol{\alpha}$ from kth outer and ith inner loop. $\boldsymbol{\alpha}_j^{k,i}$ is the jth component of that vector.

One can easily notice that d that minimizes (15) is equal to

$$d = -\frac{Q_i\boldsymbol{\alpha}^{k,i} - 1}{Q_{ii}} \tag{16}$$

So the update of $\boldsymbol{\alpha}_i^{k,i}$ is as follows:

$$\boldsymbol{\alpha}^{k,i+1} = \max(\boldsymbol{\alpha}_i^{k,i} - \frac{Q_i\boldsymbol{\alpha}^{k,i} - 1}{Q_{ii}}, 0) \tag{17}$$

Lets now try to simplify it. We know that for SVM we have [1]:

$$\boldsymbol{w} = \sum_{j=1}^{l} y_j \boldsymbol{\alpha}_j^T \boldsymbol{x}_j \tag{18}$$

We can then do the following to obtain more handy equations to actually work with:

$$\begin{aligned} Q_i\boldsymbol{\alpha}^{k,i} - 1 =& \sum_{j=1}^{l} y_j y_i x_i^T x_j \boldsymbol{\alpha}_j - 1 + \boldsymbol{\alpha}_i/2C \\ =& y_i \boldsymbol{w}^T \boldsymbol{x}_i - 1 + \frac{\boldsymbol{\alpha}_i}{2C} \end{aligned} \tag{19}$$

So in the end algorithm looks as follows:

Algorithm 1: Dual Coordinate Descent Method

$\boldsymbol{\alpha}^0 \leftarrow 0$; $\boldsymbol{w}^0 \leftarrow 0$; $\epsilon \leftarrow 10^{-4}$; ;
while $\left\|\boldsymbol{w}^{k+1} - \boldsymbol{w}^k\right\|_2 \geq \epsilon$ **do**
  for $i = 0, i = l, i+ = 1$ **do**
    $\overline{\boldsymbol{\alpha}} \leftarrow \boldsymbol{\alpha}$
    $\mathbf{G} = \max(y_i \boldsymbol{w}^T \boldsymbol{x_i} - 1 + \boldsymbol{\alpha_i}/(2C), 0)$
    if $G \neq 0$ **then**
      $\boldsymbol{\alpha_i} \leftarrow \max(\boldsymbol{\alpha_i} - \frac{\mathbf{G}}{Q_{ii}+1/(2C)}, 0)$
      $\boldsymbol{w} \leftarrow \boldsymbol{w} + (\boldsymbol{\alpha_i} - \overline{\boldsymbol{\alpha}}_{\boldsymbol{i}}) y_i \boldsymbol{x_i}$
    end
  end
end

3 Improvements

3.1 Random permutation

In order to improve performance of presented algorithm one can permute the order of observations before entering another iteration of the outer loop.

3.2 Shrinking

During optimization many components of α vector might be zeroed out and stay this way for the rest of the training procedure. Thus, it is worth to do not make further caluclation based on them to speed up the computation. It can be accomplished by utilization of the shrinking method.

Shrinking algorithm works as follows:

The index $A = \{1, 2, 3, ..., l\}$ is created. It indicates the elements of $\boldsymbol{\alpha}$ that are being optimized.

Assume that we are on kth outer iteration of our method. We can define following bounds:

$$M^{k-1} = \max_j \nabla_j^P f(\boldsymbol{\alpha}^{k,j})$$

$$m^{k-1} = \min_j \nabla_j^P f(\boldsymbol{\alpha}^{k,j})$$

Both of them will converge to 0 as $k \to \infty$ [2].

At each inner step before performing the update of $\boldsymbol{\alpha}^{k,i}$ to $\boldsymbol{\alpha}^{k,i+1}$ we check whether the $\nabla_i^P f(\boldsymbol{\alpha}^{k,i}) > M^{k-1}$ and $\boldsymbol{\alpha}^{k,i} = 0$. If so, then ith component of $\boldsymbol{\alpha}^{k,i}$ is shrunken (removed from the A index).

The update of $\boldsymbol{\alpha}^k$ is calculated as in Algorithm 1 but based only on elements of $\boldsymbol{\alpha}^k$ that are in the index A.

This improvement can also provide us with another stopping criterion which can be defined as:

Figure 1: Toy dataset.

$$M^k - m^k < \epsilon$$

In our experiments we utilized both stopping criterions.

4 Experiments

4.1 Toy dataset

To validate the performance of our algorithm we created a toy dataset which comprises of 2000 observations on $\mathbb{R}^2$ plane which are organized into two partially overlapping blobs (see Fig. 1).

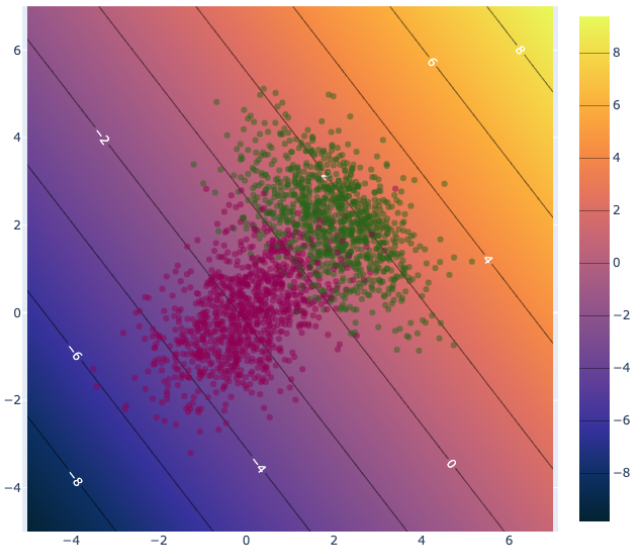

Figure 2: Decision function.

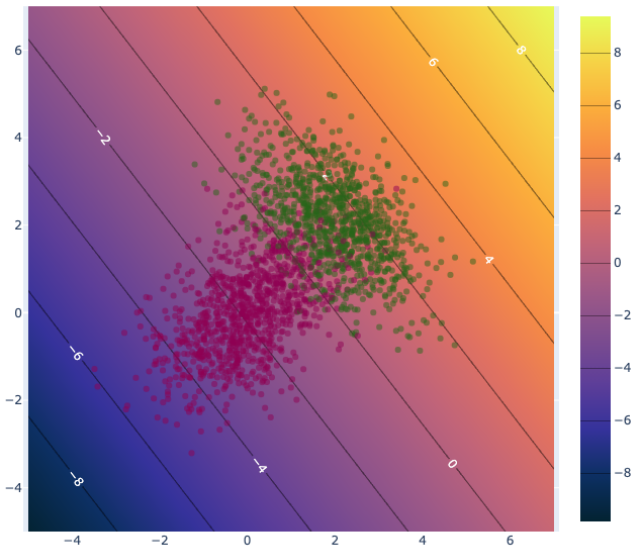

Figure 3: Decision function.

Then we trained our model on the dataset and we received the accuracy of 92.73%. In comparision, linear SVM implementation available in Python sklearn library, with identical parameters (tolerance= 10^{-4}, C= 10) also achieved 92.73% of accuracy. Next, we calculated the value of $\boldsymbol{x}^T\boldsymbol{w}$ on the $\mathbb{R}^2$ plane to see how the decision function looks like (see Fig 3).

4.1.1 Shrinking

As a next step we tested how shrinking algorithm performs on toy dataset. We monitored the state of A index through the learning process and visualized which observations are cancelled out during training (see Figure 4). It can be observed that the only points that are taken into consideration are those that lay near the intersection of two classes.

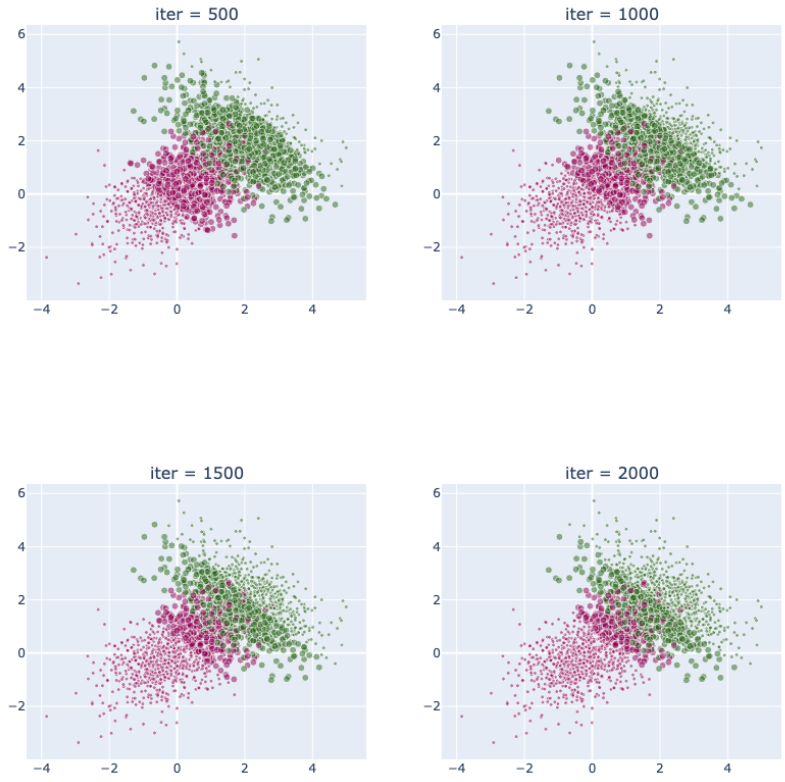

Figure 4: Shrinking on toy dataset. Smaller point corresponds to observations with $\alpha_i = 0$. For larger $\alpha_i \neq 0$.

4.2 Real world datasets

In next step we validated performance of our algorithm on real world datasets:

- **banknote dataset** - comprises of 1372 observations and 4 predictors: variance of image, skewness, kurtosis and its entropy. The aim is to predict whether banknote is forgery or not.
- **heart diseases dataset** - has 297 observations and 13 predictors, like age, sex, cholesterol, etc.. and the target is whether patient has a heart diseas or not.
- **MNIST dataset** - we selected 10853 images (784 pixels) of ones and sevens from the original MNIST dataset and we performed classification on them.

First two datasets comprise of relatively small number of observations and predictors. Third dataset is significantly larger and it has been chosen to check how our algorithm performs on bigger datasets. Our results are almost identical to the results obtained by sklearn.LinearSVC model, run with the same parameters (see Table 1). In some cases our algorithm converged in smaller number of steps than this provided in sklearn. However time of execution for our algorithm was significantly higher (see Table 2), but that is due to language selection - sklearn uses bindings to C under the hood, while our approach uses Python.

Table 1: Accuracy on real datasets (test set).

	our			our shrinking			sklearn		
C	1	10	100	1	10	100	1	10	100
dataset									
banknote	**98.90**	**98.90**	98.68	**98.90**	**98.90**	**98.90**	**98.90**	**98.90**	**98.90**
heart	80.81	80.81	**82.83**	80.81	80.81	80.81	80.81	**82.83**	74.75
mnist	99.55	**99.58**	**99.58**	99.55	**99.58**	**99.58**	99.55	**99.58**	**99.58**
toy	93.48	93.48	**93.64**	93.48	93.48	**93.64**	93.48	93.48	93.48

Table 2: Working time of algorithms for real datasets (test set). Time in seconds

	our			our shrinking			sklearn		
C	1	10	100	1	10	100	1	10	100
dataset									
banknote	3.3	12.0	11.9	4.2	10.2	7.2	0.0029	0.0099	0.0068
heart	1.8	4.3	4.3	1.9	4.5	4.3	0.0063	0.0072	0.0066
mnist	45.9	80.9	81.5	65.7	109.8	107.6	0.1399	0.1133	0.1095
toy	4.1	23.7	25.1	5.3	22.9	23.9	0.0078	0.0196	0.0315

For C equal to 1 we observed the fulfillment of the condition $||w^{k+1} - w^k||_2$. The changes of the parameter through the epoch have been compared for algorithm with shrinking and without this mechanism.

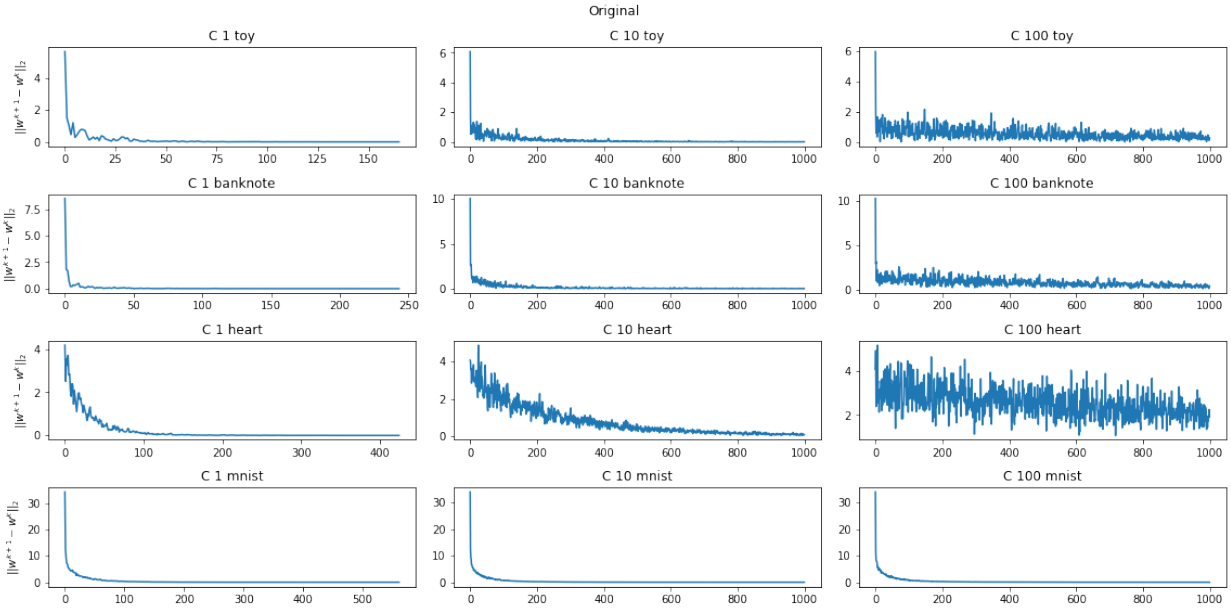

Figure 5: Changes of values $||w^{k+1} - w^k||_2$ for original algorithm

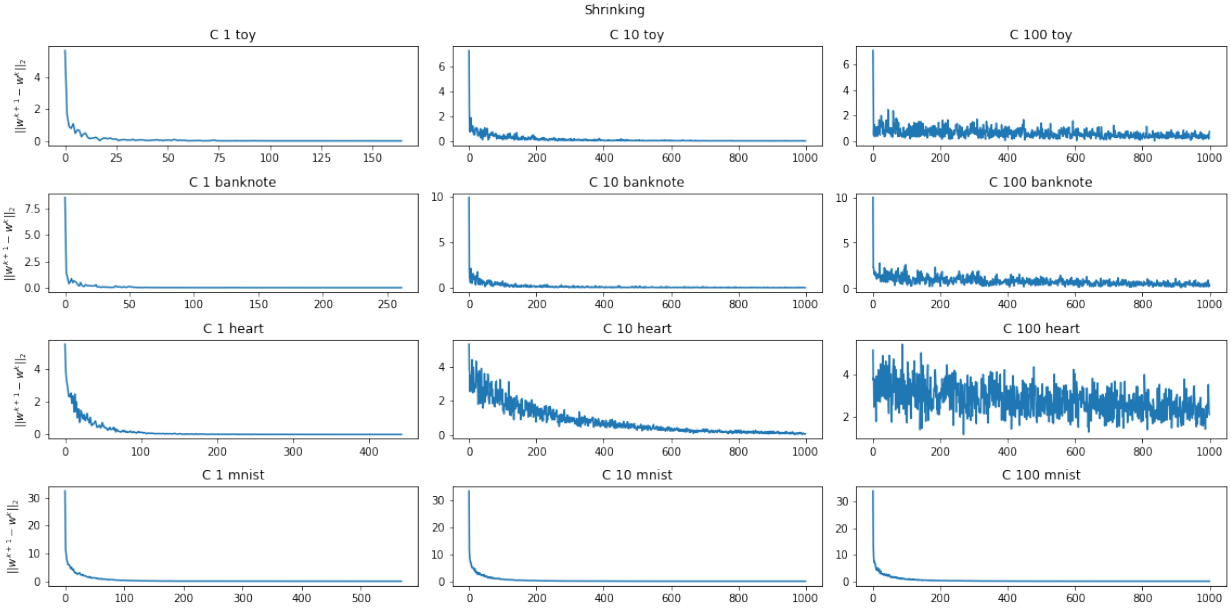

Figure 6: Changes of values $||w^{k+1} - w^k||_2$ for algorithm with shrinking

In case of of our algorithm it made few mistakes during classification seven and one from mnist dataset (the case with parameter C equal to 100). Most of them are easy for the human to classify properly but some of them could be really hard - like 2nd row 2nd column or 3rd row 1t and 2nd columns, those are not so obvious.

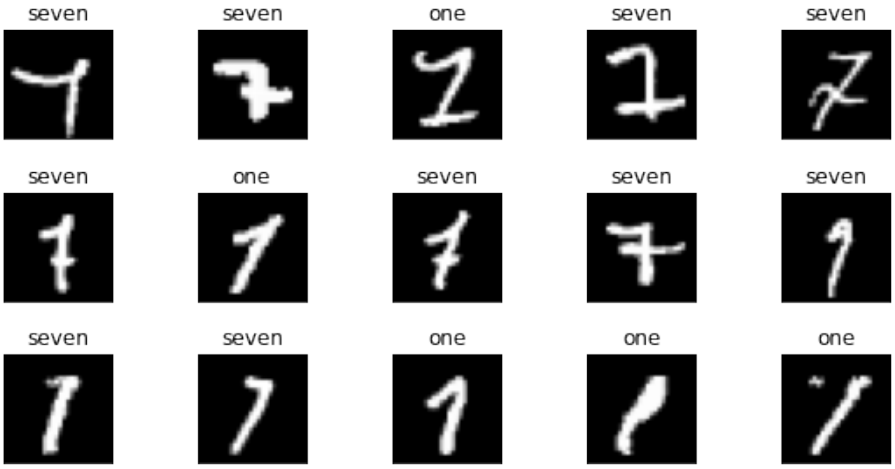

Figure 7: Mnist images which have been classified to wrong class

References

[1] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. New York, NY, USA: Springer New York Inc., 2001.

[2] Cho-Jui Hsieh et al. "A Dual Coordinate Descent Method for Large-Scale Linear SVM". In: *Proceedings of the 25th International Conference on Machine Learning*. ICML '08. Helsinki, Finland: Association for Computing Machinery, 2008, pp. 408–415. ISBN: 9781605582054. DOI: 10.1145/1390156.1390208. URL: https://doi.org/10.1145/1390156.1390208.