Advanced Topics in Machine Learning - Assignment 3

Christoffer Thrysøe - dfv107

September 26, 2017

1. SVM and regularization

We consider the primal optimization of the 1-norm soft margin SVM:

$$\text{minimize}_{\xi,w,b} \quad \frac{1}{2}\langle w, w\rangle + C\sum_{i=1}^{l}\xi_i \tag{1}$$

$$\text{subject to} \quad y_1(\langle w, \phi(x_1)\rangle + b) \geq 1 - \xi_i \quad , \quad i = 1, ..., l \tag{2}$$

$$\xi_i \geq 0 \quad , \quad i = 1, ..., l \tag{3}$$

and we wish to prove that, when w, b is fixed, the optimal slack variable is given by:

$$\xi_i = \max(0, 1 - y_i(\langle w, \phi(x_i)\rangle + b)) = L_{hinge}(y_i, f(x_i)) \tag{4}$$

where the hinge loss is given by the following:

$$L_{hinge}(y, f(x)) = \max(0, 1 - yf(x)) \tag{5}$$

Given that the hyperplane is fixed in the optimization problem (1), the only variable we are minimizing with respect to is the slack variables. Clearly the problem is minimized, when we have the smallest feasible slack variable, i.e the one which still satisfies it's constraints, listed in (2) and (3). We note that we can re-write the constraint (2) as followed:

$$y_1(\langle w, \phi(x_1)\rangle + b) \geq 1 - \xi_i \Rightarrow \tag{6}$$

$$\xi_i \geq 1 - y_1(\langle w, \phi(x_1)\rangle + b) \tag{7}$$

It is clear that given the constraint in (2) and that $\xi_i \geq 0$ for all i, to minimize (1), the inequality in (7) must be an equality, because we want the smallest feasible slack variable. Thus we have:

$$\xi_i = 1 - y_1(\langle w, \phi(x_1)\rangle + b) \tag{8}$$

by constraint (3), ξ_i must be greater than or equal to zero. (8) will get negative for points correctly classified and above/below the margin, in this case we do not need a slack variable, thus we take the following:

$$\xi_i = \max(0, 1 - y_i(\langle w, \phi(x_i)\rangle + b)) \tag{9}$$

which is equal to the hinge-loss defined in (5), where y is the target value and $(\langle w, \phi(x_1)\rangle + b)$ is the predicted value.

2. Karush-Kuhn-Tucker (KKT) theorem

Given an optimization problem, we formulate the problem as followed:

$$\text{minimize } f(w) \tag{10}$$
$$\text{subject to } g_i(w) \leq 0 \tag{11}$$
$$h_j(w) = 0 \tag{12}$$

Thus an optimization problem can have multiple inequality and equality constraints. For a primal optimization problem, we define the Lagrangian function as:

$$L(w, \lambda, \mu) = f(w) + \sum_{i=1}^{m} \lambda_i h_i(w) + \sum_{i=1}^{k} \mu_i g_i(w) \tag{13}$$

Then if the following Karush-Kuhn-Tucker conditions are satisfied, we have that w^* is a a global minimum, if we can find values for λ^* and μ^* such that the following conditions are satisfied:

$$\frac{\partial L(w^*, \lambda^*, \mu^*)}{\partial w} = 0 \tag{14}$$
$$\frac{\partial L(w^*, \lambda^*, \mu^*)}{\partial \lambda} = 0 \tag{15}$$
$$\mu_i^* g_i(w^*) = 0 \tag{16}$$
$$g_i(w^*) \leq 0 \tag{17}$$
$$\mu_i^* \geq 0 \tag{18}$$

for all $i = 1, .., k$
Note that for the following problems, instead of minimizing $||w||$, I will be minimizing $||w||^2$ as written on the discussion forum.

1

$$\text{minimize } ||w||^2 \tag{19}$$
$$\text{subject to } w_1 + w_2 + 1 \geq 0 \tag{20}$$

First the inequality is rephrased to the same form of (11), thus the constraint is as followed:

$$g(w) = -w_1 - w_2 - 1 \leq 0 \tag{21}$$

writing up the Lagrangian function we get:

$$L(w, \mu) = ||w||^2 + \mu(-w_1 - w_2 - 1) \tag{22}$$
$$= \sqrt{w_1^2 + w_2^2}^2 + \mu(-w_1 - w_2 - 1) \tag{23}$$

note that we don't include the λ term, because we only have an equality constraint. Taking the derivative of (23) with respect to w_1 and w_2, and setting them to zero we get:

$$\frac{\partial L(w, \mu)}{\partial w_1} = 2w_1 - \mu = 0 \tag{24}$$
$$\frac{\partial L(w, \mu)}{\partial w_2} = 2w_2 - \mu = 0 \tag{25}$$

for the two derivatives, shown in (24) and (25), we note that w_1 and w_2 must be equal, otherwise the two equations don't hold. We also note that they must not be negative as otherwise we would break the condition of $\mu \geq 0$ If we have that $w_1 = w_2 = 0$ then the above would hold for $\mu = 0$. If we check the KKT conditions we have:

$$\frac{\partial L(w^*, \mu^*)}{\partial w_1} = 2w_1 - \mu^* = 2 \times 0 - 0 = 0 \tag{26}$$

$$\frac{\partial L(w^*, \mu^*)}{\partial w_2} = 2w_2 - \mu^* = 2 \times 0 - 0 = 0 \tag{27}$$

$$\mu^* g(w^*) = \mu^*(-w_1 - w_2 - 1) = 0(0 - 0 - 1) = 0 \tag{28}$$

$$g(w^*) = (-w_1 - w_2 - 1) = (0 - 0 - 1) = -1 \leq 0 \tag{29}$$

$$\mu^* = 0 \geq 0 \tag{30}$$

Thus all conditions are satisfied and we have that $w^* = (0, 0)$ is a solution to the minimization.

2

$$\text{minimize } ||w||^2 \tag{31}$$

$$\text{subject to } w_1 + w_2 + 1 \leq 0 \tag{32}$$

We have the following Lagrangian function:

$$L(w, \mu) = \sqrt{w_1^2 + w_2^2}^2 + \mu(w_1 + w_2 + 1) \tag{33}$$

The derivative, with respect to w, is given by:

$$\frac{\partial L(w, \mu)}{\partial w_1} = 2w_1 + \mu = 0 \tag{34}$$

$$\frac{\partial L(w, \mu)}{\partial w_2} = 2w_2 + \mu = 0 \tag{35}$$

We can easily identify that the constraint is satisfied when $w_1 = w_2 = -0.5$. If we use this value of w in (34) and (35) we get that $\mu = 1$ and therefore:

$$2 \times -0.5 + 1 = 0 \tag{36}$$

We have shown that:

$$\frac{\partial L(w^*, \mu^*)}{\partial w} = 0 \tag{37}$$

now we wish to show that the rest of the KKT conditions are satisfied as well:

$$\mu^* g(w^*) = \mu^*(w_1 + w_2 + 1) = 1(-0.5 + (-0.5) + 1) = 0 \tag{38}$$

$$g(w^*) = (w_1 + w_2 + 1) = (-0.5 + (-0.5) + 1) = 0 \leq 0 \tag{39}$$

$$\mu^* = 1 \geq 0 \tag{40}$$

Thus all conditions are satisfied and we have that $w^* = (-0.5, -0.5)$ is a solution to the minimization problem.

3

$$\text{minimize } ||w||^2 \tag{41}$$

$$\text{subject to } 2w_1 + 2w_2 + 2 = 0 \tag{42}$$

We have the following Lagrangian function:

$$L(w,\mu) = \sqrt{w_1^2 + w_2^2}^2 + \lambda(2w_1 + 2w_2 + 2) \tag{43}$$

Taking the derivative of (43) with respect to w_1 and w_2, and setting them to zero we get:

$$\frac{\partial L(w,\mu)}{\partial w_1} = 2w_1 + 2\lambda = 0 \tag{44}$$

$$\frac{\partial L(w,\mu)}{\partial w_2} = 2w_1 + 2\lambda = 0 \tag{45}$$

we note again that the above is only satisfied when $w_1 = w_2$. The constraint in (42) is satisfied when $w_1 = w_2 = -0.5$. From (44) we set $w_1 = w_2 = -0.5$ and solve for λ:

$$2 \times -0.5 + \lambda = 0 \Rightarrow \lambda = 0.5 \tag{46}$$

To prove that the KKT conditions are satisfied we also need to prove that:

$$\frac{\partial L(w^*,\lambda^*)}{\partial \lambda} = 0 \tag{47}$$

Taking the derivative of (43) with respect to λ and inputting the value for w we get:

$$\frac{\partial L(w^*,\lambda^*)}{\partial \lambda} = 2w_1 + 2w_2 + 2 = 2 \times -0.5 + 2 \times -0.5 + 2 = 0 \tag{48}$$

Thus all conditions are satisfied and we have that $w^* = (-0.5, -0.5)$ is a solution to the minimization problem.

3. PAC-Bayesian Aggregation

For this assignment, I have used the Support Vector Machine library LibSVM [1]. First I will describe how the baseline was implemented, and then how the experiment from [2] was performed. The implementation of the baseline is given in the file `crossValidate.py`, the experiment is implemented in the file `experiment.py` and the plotting is given in the file `plotgraph.py`.
For the baseline, the Ionosphere data [3] was first loaded and the target values was converted such that:

$$y_i = \begin{cases} 1 & \text{if } y_i = b \\ -1 & \text{if } y_i = g \end{cases} \tag{49}$$

The data was then randomly partitioned into a training and test set, where the training set contains 225 observations, and the test set contains 126 observations. This division follows the one used in [2]. For both parts of the assignment, the RBF-kernel SVM is used. To find appropriate values for the RBF-kernel bandwidth parameter γ, the Jaakkola heuristic was implemented. This was done by taking the distance for each training example to the rest of the training examples. These distances was sorted in ascending order. The closest training example, with a different target label, was then located, and it's distance to the training point was added to a list G. Thus index i in G is defined as:

$$G(x_i) = \min_{(X_j,Y_j)\in S \wedge Y_i \neq Y_j} ||X_i - X_j|| \tag{50}$$

the median of G was then taken and the Jaakkola seed was defined by:

$$\gamma_J = \frac{1}{2 \cdot \text{ median}(G)^2} \tag{51}$$

Getting γ_J on the entire data (that is training and test) the following Jaakkola seed was achieved:

$$\gamma_J = 0.14349 \tag{52}$$

(The γ_J used for the baseline is determined solely on the training data, i.e. no access to the test data). The γ and C values, used for training the Support Vector Machine was then defined as:

$$\gamma \in \{y_J \cdot 10^i | i \in \{-4, -3, -2, -1, 0, 1, 2, 3, 4\}\} \tag{53}$$

$$C \in \{10^{-3}, 10^{-2}, 10^{-1}, 10^0, 10^1, 10^2, 10^3\} \tag{54}$$

Grid search was then performed using 5-fold cross validation, to find optimal hyper parameters. For each value of C and γ, the training set was randomly split into 5 equal sized sets. For each round of the cross validation, the SVM was trained on the four partitions combined, where the fifth partition was used as a validation set. The average error over these five validation sets was then taken. The choosing of C and γ, which resulted in the lowest average error over the five validation sets, was used as final hyper parameters. The found hyper parameters was then used for training of the SVM on the entire training set. This trained model was then used to get the test error on the test set. Running the baseline 30 times, the following hyper parameters was found:

$$C = 10^1 = 10$$
$$\gamma = \gamma_J \cdot 10^0 = \gamma_J$$

The following table shows the average training ($\hat{L}(h, S)$) and test ($L(h)$) errors and their standard deviation over the 30 runs:

Table 1: Average training and test errors over 10 runs and their standard deviation.

$\hat{L}(h, s)$	$L(h)$	$\sigma_{\hat{L}(h,s)}$	$\sigma_{L(h)}$
0.0192	0.0577	0.0140	0.0171

The average running time of the baseline was:

$$t_{baseline} = 2,87 \text{ sec} \tag{55}$$

The running time included the cross validation, hyper parameter selection, the training of the final model and the applying of the model on the unseen test data to get the test error.

The experient was done as followed. First a training and test set was created, following the same partition size as for the baseline. Given a number of m subsets, each subset was assigned $d+1 = r$ random points from the training data. The remaining $n - r$ points was used as a validation set, where n is the length of the training data. Each m subsets, containing 35 points each, was then used for training an SVM, where the validation set, containing $n - r$ points each, was used for validation of the weak classifier. The bandwidth parameter γ of the RBF-kernel was randomly selected from the same list of γ as used in the baseline. For the value of C [2] notes that setting C was not necessary as the hyperplane was able to perfectly separate the 35 training points. I did however experience that setting C was necessary in order to correctly classify the 35 training points, this setting of C however did not influence the final result of the classifier. Once the validation error was achieved for each m subsets, the distribution ρ was found using alternating minimization of the following bound, which holds with probability greater than $1 - \delta$:

$$\mathbb{E}_\rho[L(h)] \leq \frac{\mathbb{E}_\rho[\hat{L}^{val}(h, S)]}{1 - \frac{\lambda}{2}} + \frac{KL(\rho||\pi) + \ln \frac{(n-r)+1}{\delta}}{\lambda(1 - \frac{\lambda}{2})(n - r)} \tag{56}$$

where $\lambda \in (0, 2)$. The alternating minimizations consist of alternating minimizing ρ and λ. The update rule for ρ is defined as:

$$\rho(h) = \frac{\pi(h)e^{-\lambda(n-r)(\hat{L}^{val}(h,S)-\hat{L}^{val}_{\min})}}{\sum_{h'} \pi(h')e^{-\lambda(n-r)(\hat{L}^{val}(h,S)-\hat{L}^{val}_{\min})}} \tag{57}$$

The update rule for λ is defined as followed in Yevgeny's lecture notes (equation 3.15):

$$\lambda = \frac{2}{\sqrt{\frac{2n\mathbb{E}\rho\left[\hat{L}(h,S)\right]}{\mathrm{KL}(\rho||\pi) + \ \ln\ \frac{n+1}{\delta}} + 1} + 1} \tag{58}$$

The update term can be re-written to suit our validation error, and use the definition of $KL(\rho||\pi)$ from the lecture notes:

$$\lambda = \frac{2}{\sqrt{\frac{2(n-r)\mathbb{E}\rho\left[\hat{L}^{val}(h,S)\right]}{\mathrm{KL}(\rho||\pi) + \ \ln\ \frac{(n-r)+1}{\delta}} + 1} + 1} \tag{59}$$

$$= \frac{2}{\sqrt{\frac{2(n-r)\mathbb{E}\rho\left[\hat{L}^{val}(h,S)\right]}{\mathbb{E}_\rho\left[\ln\frac{1}{\pi}\right] - H(\rho) + \ \ln\ \frac{(n-r)+1}{\delta}} + 1} + 1} \tag{60}$$

$$= \frac{2}{\sqrt{\frac{2(n-r)\left(\sum\limits_{h\in\mathcal{H}} \rho(h)\hat{L}^{val}(h,S)\right)}{\left(\sum\limits_{h\in\mathcal{H}} \rho(h)\ln\frac{1}{\pi}\right) - \left(-\sum\limits_{h\in\mathcal{H}} \rho(h)\ \ln\ \rho(h)\right) + \ \ln\ \frac{(n-r)+1}{\delta}} + 1} + 1} \tag{61}$$

Where in both terms, the prior distribution π was chosen to be uniform, that is $\pi(h) = 1/m$. The alternating minimization was performed, by initially setting $\lambda = 2.0$ and then finding the distribution ρ from (57), using this obtained value for ρ to find λ from (61). This alternating minimization was performed until the change in the norm of ρ (treated as a list for each $\rho(h)$) was below a threshold ϵ and the change in the parameter λ was below a threshold ϵ, where for both stopping criteria $\epsilon = 2.224 \cdot 10^{-16}$. For the minimization we had $\delta = 0.05$. The obtained distribution ρ minimizing the bound in (56) was then used to perform ρ-weighted majority vote, which is defined as followed:

$$MV_\rho(X) = \mathrm{sign} \sum_{h\in\mathcal{H}} \rho(h)h(X) \tag{62}$$

where each h is the SVM model, trained on the random r points, used for getting the validation errors. The majority vote was used on the yet unseen test data to get the test error. The bound from theorem 3.13 from Yevginev's lecure notes, which we would like to plot is defined as:

$$\mathrm{kl}\left(\mathbb{E}_\rho\left[\hat{L}^{val}(h,S)\right] ||\mathbb{E}_\rho\left[L(h)\right]\right) \geq \frac{\mathrm{KL}(\rho||\pi) + \ln\frac{(n-r)+1}{\delta}}{n-r} \tag{63}$$

To get a bound on the expected loss $\mathbb{E}_\rho[L(h)]$, the kl divergence from (63) was inverted numerically, taking the upper inverse. To take the upper inverse, we must solve the following:

$$kl^{-1^+}(\hat{p}, z) = \max\{p : kl(\hat{p}||p) \leq z\} \tag{64}$$

giving the following bound, which holds with probability greater than $1-\delta$:

$$p \leq \max\left\{p : p \ln\frac{\hat{p}}{p} + (1-\hat{p}) \ln \frac{1-\hat{p}}{p} \leq z\right\} \tag{65}$$

where $\hat{p} = \sum_{h\in\mathcal{H}} \rho(h)\hat{L^{val}}(h,S)$, $p = \mathbb{E}_\rho[L(h)]$ and z is defined as:

$$z = \frac{KL(\rho||\pi) + \ln\frac{(n-r)+1}{\delta}}{n-r} \tag{66}$$

$$= \frac{\left(\sum_{h\in\mathcal{H}} \rho(h)\ln\frac{1}{\pi}\right) - \left(-\sum_{h\in\mathcal{H}} \rho(h) \ln \rho(h)\right) + \ln\frac{(n-r)+1}{\delta}}{n-r} \tag{67}$$

The kl inequality was inverted numerically using a binary search.
The above was performed for 20 different values of m from [1,200]. As with the baseline, for each choosing of m, the experiment was performed 30 times, where it's average error, bound, running time and standard deviation was taken. The running time of the Pac-Baysian aggregation included training the m subsets, getting the validation error, applying alternating minimization and performing ρ-weighted majority vote to get the test error. Below, the error and standard deviation is listed for some of the values for m (the error for all m is shown in the figure):

Table 2: The test error and standard deviation shown for some values of m using Pac-Baysian Aggregation

m	10	30	50	70	90	110	130	150	170	190
$L(h)$	0.115	0.089	0.080	0.078	0.079	0.071	0.075	0.072	0.074	0.073
$\sigma_{\text{Pac-bayes}}$	0.038	0.035	0.025	0.022	0.023	0.025	0.022	0.021	0.026	0.023

As evident from table 2 the error is decreasing and almost as low as the baseline for high values of m. It is also clear that the standard deviation in the results are higher than for the baseline. Figure 1 shows the baseline average error and time, it also shows the average error obtained from the Pac-Bayesian Aggregation, it's bound and the running as a function of the number of subsets m. The reported error is the zero-one loss, and the time is plotted as $seconds/10$. It is clear that when the number of subsets increases the error goes down, however the running time goes up, so there is a trade-off in achieving a low error and a reasonable running time. Interestingly for the Pac-Baysian aggregation, we are able to achieve a somewhat low error, while maintaining a very low running time. The baseline has the lowest error, but also the largest running time. As m increases, the error goes towards the baseline and the running time also approaches the baseline running time.

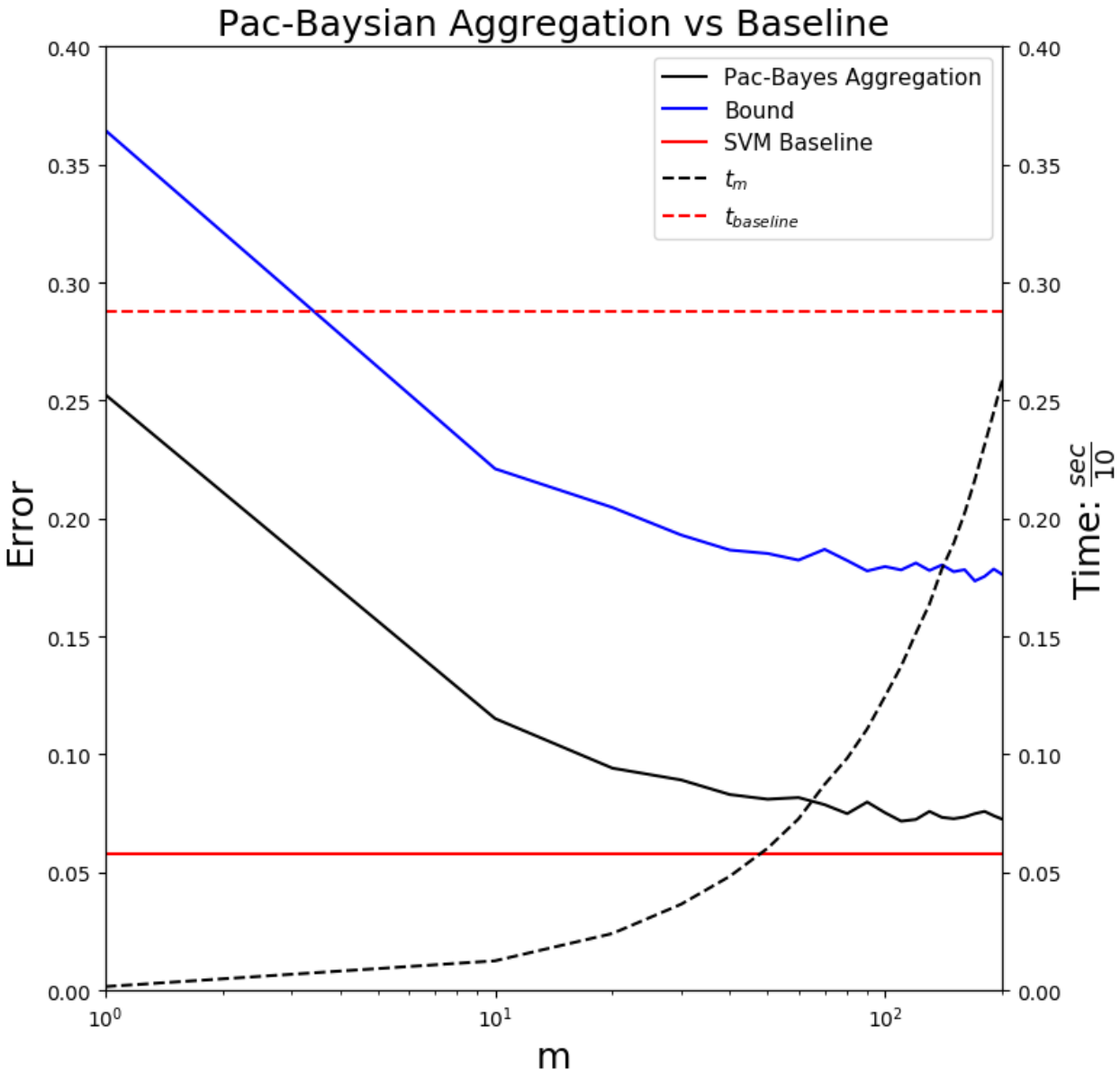

Figure 1: The Pac-Baysian aggregation compared to an RBF kernel SVM tuned by cross-validation. The left y axis shows the zero-one loss, where the right axis shows the time in $sec/10$. The x axis shows the number of subsets used in the Pac-Baysian aggregation.

References

[1] Chih-Chung Chang and Chih-Jen Lin, *LIBSVM – A Library for Support Vector Machines* https://www.csie.ntu.edu.tw/ cjlin/libsvm/

[2] Niklas Thiemann, Christian Igel, Olivier Wintenberger, and Yevgeny Seldin. *A strongly quasiconvex PAC-Bayesian bound*. In Proceedings of the International Conference on Algorithmic Learning Theory (ALT), 2017. http://arxiv.org/abs/1608.05610.

[3] Arthur Asuncion and David J. Newman. UCI machine learning repository, 2007. http://archive.ics.uci.edu/ml/index.php.