EECS 445: Machine Learning **Fall 2016**

Discussion 5: Naive Bayes and SVM

Written by: Chansoo Lee

Edited by:

Last Updated: October 13, 2016 10:47am

5.1 Naive Bayes

Notations: Let C be number of classes, D be the number of features, and M be the number of values each feature can take. The Naive Bayes prior parameters are $\boldsymbol{\pi}$ and posterior parameters are $\{\boldsymbol{\theta}_{cd} : c = 1, \ldots, C, d = 1, \ldots, D\}$. For simplicity we simply use π, θ when discussing them as general parameters and thus dimensions are not important in the context.

5.1.1 Naive Bayes: MLE

Excercise 5.1 (Review Question)**.** What are the semantics of these parameters? What are the dimensions of $\boldsymbol{\pi}$ and $\boldsymbol{\theta}_{cd}$? They are called *probability vectors*. What is the key property?

The individual elements of a probability vector sums up to 1.

Furthermore, suppose $M = 2$. When dealing with sparse features such that x_d is rarely 1 (`Spambase` data in HW2, after our preprocessing step, has this property), we only see a small number of examples to accurately estimate θ_{cd1}. This causes overfitting. (*Understand exactly why this is the case.*)

Excercise 5.2. Discuss what happens if $x_d = 1$ for every training example x, regardless of its class label? What happens if we recieve a test data with $x_d = 0$?

5.1.2 Naive Bayes: Mean estimate

The overfitting problem can be fixed by putting Dirichlet priors on $\boldsymbol{\pi}$. Dirichlet distribution over M-dimensional probability vector, parameterized by a vector $\boldsymbol{\alpha} \in \mathbb{R}^M$ of positive values, has the following probability density function:

$$p(\boldsymbol{u}; \boldsymbol{\alpha}) = Z(\boldsymbol{\alpha}) u_1^{\alpha_1 - 1} \cdots u_M^{\alpha_M - 1}$$

where $Z(\boldsymbol{\alpha})$ is the normalizing constant, which makes the above function integrates to 1.

Excercise 5.3. Let's inspect the Dirichlet pdf.

- Fix every coordinates but α_1, and increase α_1. What happens to the distribution of u_1? What would happen to the mean of u_1?.

  We are more likely to get a high value, so the mean would increase.

- Set all coordinates of $\boldsymbol{\alpha}$ to λ. As you increase λ, what would happen to the expected value of $\boldsymbol{u}$?

  The expected value is always $\boldsymbol{u} = (1/M, \ldots, 1/M)^\top$ *because the sum* $\sum_{i=1}^M u_i$ *is constrained to be 1 and pdf is symmetric.*

With Dirichlet prior, we can do the MAP estimate as usual

$$(\hat{\pi}, \hat{\theta}) = \arg\max_{\pi,\theta} P(\theta, \pi | \mathcal{D}_{\text{train}}).$$

But unfortunately the expression for the maximizer isn't very pretty (functionally they are fine). So, we use the mean estimate instead:

$$\hat{\pi} = E[\pi | \mathcal{D}_{\text{train}}], \quad \hat{\theta} = E[\theta | \mathcal{D}_{\text{train}}]$$

where $\boldsymbol{\pi} \sim \text{Dirichlet}(\boldsymbol{\alpha})$ and $\theta_{cd} \sim \text{Dirichlet}(\boldsymbol{\beta}_{cd})$. If we compute the means, we get

$$\hat{\pi}_c = \frac{N_c + \alpha_c}{N + \sum_c \alpha_c}, \quad \hat{\theta}_{cdm} = \frac{N_{cdm} + \beta_{cdm}}{N_c + \sum_{m'} \beta_{cdm'}}$$

Excercise 5.4. Redo Exercise 5.2. What happens at the prediction time?

5.2 SVM

Let $\{(\mathbf{x}_i, t_i) \in \mathbb{R}^D \times \{-1, 1\}\}_{i=1}^N$ be our training set.

Hyperplane is a *set* of D-dimensional vectors that "constitute" an $(D-1)$ dimensional object. The key here is *set*, as a hyperplane is usually described in the form of:

$$\{\boldsymbol{v} : \mathbf{w}^\top \boldsymbol{v} + b = 0\}.$$

The distance between the above hyperplane and a point $\mathbf{x}$ is

$$\frac{|\mathbf{w}^\top \mathbf{x} + b|}{\|\mathbf{w}\|}.$$

5.2.1 Hard SVM

The Hard SVM maximizes the *margin distance*, which is the minimum distance between a point and the hyperplane $\{\boldsymbol{v} : \mathbf{w}^\top \boldsymbol{v} + b = 0\}$:

$$\arg\max_{\mathbf{w},b} \min_{i=1,\dots,N} \frac{|\mathbf{w}^\top \mathbf{x}_i + b|}{\|\mathbf{w}\|} \text{ such that } t_i(\mathbf{w}^\top \mathbf{x}_i) + b > 0 \text{ for all } i. \tag{5.1}$$

Excercise 5.5. (Review)Does the solution always exist? What is the condition under which the solution exists?

When the solution exists, (5.1) is equivalent to

$$\arg\max_{\mathbf{w},b} \min_{i=1,\dots,N} \frac{t_i(\mathbf{w}^\top \mathbf{x}_i) + b}{\|\mathbf{w}\|} \text{ such that } t_i(\mathbf{w}^\top \mathbf{x}_i) + b > 0 \text{ for all } i. \tag{5.2}$$

Excercise 5.6. Show the equivalence of (5.1) and (5.2).

Excercise 5.7. (Review) Given a solution $(\mathbf{w}^*, b^*)$ to the above, can we generate another solution? *Hint: As you scale* $(\mathbf{w}^*, b^*)$ *with* γ *and set* $\mathbf{w} = \gamma \mathbf{w}^*$ *and* $b = \gamma b^*$*, what happens to* $t_i(\mathbf{w}^\top \mathbf{x}_i) + b$*?*

By fixing $t_{i^*}(\mathbf{w}^\top \mathbf{x}_{i^*}) + b = 1$ for the minizer i^*, we get an optimization problem whose unique solution is the maximizer of (to be precise, *one of* the maximizers of) (5.2):

$$\arg\max_{\mathbf{w},b} \min_{i=1,\dots,N} \frac{t_i(\mathbf{w}^\top \mathbf{x}_i) + b}{\|\mathbf{w}\|} \text{ such that } t_i(\mathbf{w}^\top \mathbf{x}_i) + b = 1 \text{ for some } i \text{ and } t_i(\mathbf{w}^\top \mathbf{x}_i) + b \geq 1 \text{ for all other } i. \tag{5.3}$$

The above is equivalent to

$$\arg\max_{\mathbf{w},b} \frac{1}{\|\mathbf{w}\|} \text{ such that } t_i(\mathbf{w}^\top \mathbf{x}_i) + b = 1 \text{ for some } i \text{ and } t_i(\mathbf{w}^\top \mathbf{x}_i) + b \geq 1 \text{ for all other } i. \tag{5.4}$$

which is again equivalent to

$$\arg\min_{\mathbf{w},b} \|\mathbf{w}\| \text{ such that } t_i(\mathbf{w}^\top \mathbf{x}_i) + b = 1 \text{ for some } i \text{ and } t_i(\mathbf{w}^\top \mathbf{x}_i) + b \geq 1 \text{ for all other } i. \tag{5.5}$$

Excercise 5.8. Show that the above is equivalent to

$$\arg\min_{\mathbf{w},b} \|\mathbf{w}\| \text{ such that } t_i(\mathbf{w}^\top \mathbf{x}_i) + b \geq 1 \text{ for all } i. \tag{5.6}$$

Proof technique: Suppose our minimizer $(\mathbf{w}^*, b^*)$ *gives* $\min_i t_i(\mathbf{w}^{*\top} \mathbf{x}_i) > 1$. *Can you find a better* $(\mathbf{w}, b)$ *and get a contradictiction?*

Now we can change $\|\mathbf{w}\|$ into any increasing function of $\|\mathbf{w}\|$ (like we changed likelihood to log likelihood). So why not something that looks exactly like our regularizer for linear and logistic regression?

$$\arg\min_{\mathbf{w},b} \lambda\|\mathbf{w}\|^2 \text{ such that } t_i(\mathbf{w}^\top \mathbf{x}_i) + b \geq 1 \text{ for all } i. \tag{5.7}$$

where $\lambda > 0$ is the regularization coefficient.

5.2.2 Soft-SVM

We relax the problem so that we allow c_i to be less than 1 (**margin error**), but we add a penalty term.

$$\arg\min_{\mathbf{w},b,\boldsymbol{\xi}} \lambda\|\mathbf{w}\|^2 + \sum_{i=1}^{N} \xi_i \text{ such that } t_i(\mathbf{w}^\top \mathbf{x}_i) + b \geq 1 - \xi_i \text{ and } \xi_{\geq} 0 \text{ for all } i. \tag{5.8}$$

In the HW, you will show that the above is equivalent to

$$\arg\min_{\mathbf{w},b} \lambda\|\mathbf{w}\|^2 + \sum_i \max(0, 1 - (t_i(\mathbf{w}^\top \mathbf{x}_i) + b)). \tag{5.9}$$

Excercise 5.9. Plot the **hinge loss** function $f(z) = \max(0, 1 - z)$. What does it look like? What does the SVM penalty function do intuitively? How is it different from RMSE?

5.2.3 Duality and Kernel

The dual formulation optimizes over a vector in $\mathbb{R}^N$, instead of $\mathbb{R}^D$.

$$\begin{aligned} \underset{\boldsymbol{\alpha}}{\text{maximize}} \quad & -0.5 \sum_{i,j=1}^{N} \alpha_i \alpha_j t_i t_j \mathbf{x}_i^\top \mathbf{x}_j + \sum_{i=1}^{N} \alpha_i \\ \text{subject to} \quad & 0 \leq \alpha_i \leq C/n \quad \forall i \\ & \sum_{i=1}^{n} \alpha_i t_i = 0 \end{aligned}$$

Given a solution to the dual problem, we can get the primal solution, and vice versa. The main (arguably the only) reason we study the SVM dual formulation is that it helps us observe that the optimal $\mathbf{w}^*$ is a linear combination of examples:

$$\boxed{\mathbf{w}^* = \sum_{i=1}^{N} \alpha_i^* t_i \mathbf{x}_i}$$

The name "support vector" comes from this observation.

Because we know that the optimal $\mathbf{w}^*$ is a linear combination of $\mathbf{x}_i$, we can rewrite (5.9) as:

$$\arg\min_{\boldsymbol{\alpha},b} \lambda\|\mathbf{w}\|^2 + \sum_{i=1}^{N} \max\left(0, 1 - t_i\left(\left(\sum_{j=1}^{N} \alpha_j \mathbf{x}_j\right)^\top \mathbf{x}_i + b\right)\right). \tag{5.10}$$

Using kernelized features, we get

$$\arg\min_{\boldsymbol{\alpha}} \lambda\|\mathbf{w}\|^2 + \sum_{i=1}^{N} \max\left(0, 1 - t_i\left(\sum_{j=1}^{N} \alpha_j \phi(\mathbf{x}_j)\right)^\top \phi(\mathbf{x}_i)\right). \tag{5.11}$$

The bias term is removed because we can always add a bias coordinate in our feature mapping $\phi(\mathbf{x}_j)$.

Excercise 5.10. Take the gradient of the objective function. In particular, take the derivative with respect to α_k.

Observe that the derivative depends only on the dot product of the features. So we can implement the gradient descent as long as the dot products are well-defined, even if we are in infinite-dimensional space.