PH240C Supervised Learning (1): GLM/SVM

Jingshen Wang

September 8, 2021

Consider an input space $\mathcal{X}$ which is a subset of $\mathbb{R}^d$, and the output space $\mathcal{Y} = \{0, 1\}$, and let

$$f : \mathcal{X} \to \mathcal{Y},$$

be the target function. Given a set of functions $\mathcal{H}$ contains mapping from $\mathcal{X}$ to $\mathcal{Y}$, the binary classification task is formulated as follows. The learner receives a training sample $S = \{(X_i, Y_i)\}_{i=1}^n$ of size ni.i.d from $\mathcal{X}$ according to some unknown distribution $F(\cdot)$, with $Y_i = f(X_i)$ and $X_i \in \mathbb{R}^d$, $Y_i \in \{0, 1\}$. The supervised problem then aim to identify a function $h \in \mathcal{H}$, a binary classifier, with small generalization error (or risk):

$$R(h; F) = \mathbb{E}_F\big[\mathbf{1}(Y \neq h(X))\big] = \mathbb{P}_{X\sim F(\cdot)}\big(h(X) \neq f(X)\big).$$

Different functions $\mathcal{H}$ can be selected for this task. In this section, we shall disuses several methods that work with different class of functions $\mathcal{H}$.

Formalized by the Occam's razor principle[1], mappings with smaller complexity provide transparent interpretation, better learning guarantees. A natural choice for $\mathcal{H}$ of relatively low complexity is that of linear classifiers or hyperplanes, which can be defined as follows

$$\mathcal{H} = \{x \to \mathbf{1}(x'\beta > 0) : \beta \in \mathbb{R}^d\}.$$

A classifier of the form $x \to \mathbf{1}(x'\beta > 0)$ thus puts label "1" for all points falling on one side of the hyperplane $x'\beta = 0$ and "0" for all others. Such a problem is referred to as a linear classification problem. Without loss of generality, we include the intercept as the first component in each covariate X_i through out the lecture notes.

1 Logistic Regression

1.1 Review of regression

Regression is a method for studying the relationship between a response variable Y and a covariate X. One way to summarize the relationship between X and Y is through the regression function:

$$r(x) = \mathbb{E}[Y|X = x].$$

[1] According to Wikipedia: The Occam's razor principle is the principle of parsimony or law of parsimony. It is also a problem-solving principle that "entities should not be multiplied beyond necessity", sometimes inaccurately paraphrased as "the simplest explanation is usually the best one."

Our goal is to estimate the regression function $r(x)$ from the training sample $S = \{(X_i, Y_i)\}_{i=1}^n$. A classical parametric approach assumes $r(x)$ to be linear:

$$r(x) = x'\beta, \quad \overset{\text{hence}}{\Longrightarrow} \quad Y_i = X_i'\beta + \varepsilon_i, \quad \mathbb{E}[\varepsilon_i|X_i] = 0, \text{ and } \mathbb{V}\text{ar}[\varepsilon_i|X_i] = \sigma^2.$$

Occasionally, we add the assumption that $\varepsilon_i|X_i \sim N(0, \sigma^2)$. The maximum likelihood estimator of β coincides with the popular least squares estimator:

$$\widehat{\beta} = \arg\min \sum_{i=1}^n (Y_i - X_i'b)^2 = \big(\boldsymbol{X}'\boldsymbol{X}\big)^{-1}\boldsymbol{X}Y.$$

To this end, implicitly, we made three assumptions:

1. The pairs $(Y_1, X_1), \ldots, (Y_n, X_n)$ are independent and identically distributed (why we make this assumption?)
2. Conditional on X, the outcome Y follows a normal distribution: $Y|\boldsymbol{X} \sim N(\mu(\boldsymbol{X}), \sigma^2 \cdot I)$
3. The conditional mean of Y and X is linked through a linear function: $\mu(\boldsymbol{X}) = \boldsymbol{X}\beta$

In the presence of discrete outcome (e.g., $Y_i \in \{0, 1, 2, 3\}$), the last two assumptions are no longer appropriate. Generalized linear regression thus modify the last two assumptions to:

2. Conditional on X, the outcome Y follows certain discrete distribution
3. The conditional mean of Y and X is linked through a non-linear function $g(\cdot)$:

$$g\big(\mu(\boldsymbol{X})\big) = \boldsymbol{X}\beta, \quad \Longleftrightarrow \mu(\boldsymbol{X}) = g^{-1}(\boldsymbol{X}\beta)$$

Example 1 (Disease Occuring Rate)**.** *In the early stages of a disease epidemic, the rate at which new cases occur can often increase exponentially through time. Hence, suppose Y_i is the number of cases observed on day T_i and $\mu(T_i)$ is the expected number of new cases on day T_i, we assume that*

$$\mu(T_i) = \gamma \exp\left(\delta T_i\right),$$

where δ represents the exponential growth rate and γ represents the day-1 cases count. We then take log *on both side, which yields*

$$\log(\mu(T_i)) = \log(\gamma) + \delta T_i \triangleq \beta_0 + \beta_1 T_i.$$

Since Y_i is a count, assuming $Y_i|T_i \sim Poisson\big(\mu(T_i)\big)$ seems to be quite reasonable.

1.2 Logistic regression

In the presence of binary outcome $Y_i \in \{0, 1\}$, we assume

$$\mathbb{E}\big[Y_i|X_i\big] = \mu(X_i) \in [0, 1], \quad Y_i|X_i \sim \text{Bernoulli}\big(\mu(X_i)\big).$$

As $\mu(X_i)$ is a number between zero and one, we assume that

$$\log\left(\frac{\mu(X_i)}{1-\mu(X_i)}\right) \triangleq \operatorname{logit}\big(\mu(X_i)\big) = X_i'\beta.$$

Equivalently, we assume thta

$$\mathbb{P}\big[Y_i = 1|X_i\big] = \mu(X_i) = \frac{\exp\left(X_i'\beta\right)}{1+\exp\left(X_i'\beta\right)} \triangleq \operatorname{expit}(X_i'\beta).$$

The name "logistic regression" comes from the fact that $e^x/(1+e^x)$ is called "logistic function (or expit function)." A plot of the logistic function for a one-dimensional x is shown in Figure 1.

Thus, the likelihood function for the training sample $\{(Y_i, X_i)\}_{i=1}^n$ is

$$L_n\big(\beta; \{(Y_i, X_i)\}_{i=1}^n\big) = \prod_{i=1}^n \mu(X_i)^{Y_i}\big(1-\mu(X_i)\big)^{1-Y_i},$$

and the maximum likelihood estimator for β is obtained by

$$\begin{aligned}
\widehat{\beta} &= \operatorname*{arg\,max}_{b\in\mathbb{R}^d} \log L_n\big(b; \{(Y_i, X_i)\}_{i=1}^n\big) \\
&= \operatorname*{arg\,max}_{b\in\mathbb{R}^d} \sum_{i=1}^n \big[Y_i \log \mu(X_i) + (1-Y_i)\log(1-\mu(X_i))\big] \\
&= \operatorname*{arg\,max}_{b\in\mathbb{R}^d} \sum_{i=1}^n \big[Y_i \log \frac{\mu(X_i)}{1-\mu(X_i)} + \log(1-\mu(X_i))\big] \\
&= \operatorname*{arg\,max}_{b\in\mathbb{R}^d} \sum_{i=1}^n \big[Y_i \cdot X_i'b - \log\left(1+\exp(X_i'b)\right)\big].
\end{aligned}$$

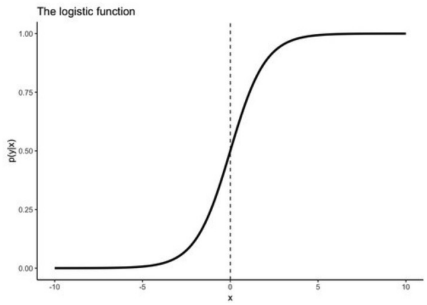

Figure 1: Logistic function $e^x/(1+e^x)$ illustration.

To minimize the mis-classification error rate (is mis-classification error always desirable? especially in health science), we predict the label for a new data point $x \in \mathcal{X}$ as

$$y = \begin{cases} 1 & \text{if } \operatorname{expit}(x'\widehat{\beta}) \geq 0.5 \quad \Leftrightarrow \quad x'\widehat{\beta} \geq 0 \\ 0 & \text{if } \operatorname{expit}(x'\widehat{\beta}) < 0.5 \quad \Leftrightarrow \quad x'\widehat{\beta} < 0. \end{cases}$$

Therefore, logistic regression gives us a linear classifier. The decision boundary separating two predicted class is the hyper-plane $x'\widehat{\beta} = 0$.

Logistic regression is one of the most commonly adopted tools for applied statistics. There are many reasons for this. First, logistic regression is easy-to-compute with Newton-Rapson and is being well-integrated into `R`. Second, the coefficient β has clear interpretation. When X_i contains the intercept and a univariate covariate (say gender) and the outcome indicate whether the individual has CAD, then $\beta_1 = 10$ represents the odds of having CAD for male is 10 times higher than females. In addition, the larger the coefficient β_1, the difference between female and male more strongly influences the disease status–because informally we can think of our prediction as being a very confidence one if $x'\widehat{\beta} \gg 0$ and vice versa.

2 Support Vector Machines

2.1 Hyperplane, margin and maximal margin classifiers

Motivated by the interpretation of logistic regression, given a training sample S, it seems that we would have found a good fit to the data if we can find β so that $x'\beta \gg 0$ whenever $y = 1$, and $x'\beta \ll 0$ whenever $y = 0$. Because this would reflect a very confidence (and maybe correct) set of classifications for the training sample. This seems to be a nice goal to aim for. To simplify notations, in this section, we change the label set $\mathcal{Y} \in \{0, 1\}$ to $\mathcal{Y} \in \{-1, 1\}$.

Consider a very simple example in the following Figure 2, in which X (green cross) represent positive outcomes, and red circles represent negative outcomes. The classifier $h_1$has larger margin than the classifier h_2, and the classifier h_2 is called the "maximal margin classifier." In plot C, h_3 is the maximal margin classifier. Nevertheless, h_3 may give false prediction given the yellow cross. This suggests the maximal margin classifier is sensitive to the presence of outliers. The decision boundary is specified by $\{x : x - a = 0\}$. We predict the outcome to be 1 (low risk) when $x > a$ and we predict the outcome to be -1 (high risk) when $x < a$.

When we are using a margin to determine the location of a threshold a, then we are using a maximal margin classifier–Support Vector Machine (SVM)– to classify observations. We will formally discuss SVM in the next section.

Now we have informally introduced margin and maximal margin classifiers, let's now define the margin of a given hyper-plane rigorously. A hyperplane is defined through $\beta = (a, \beta_1, \ldots, \beta_d)'$ as a set of points so that

$$\mathcal{A} = \left\{ x = (x_1, \ldots, x_d)' : a + \sum_{j=1}^{d} x_j \beta_j = 0 \right\},$$

and the margin γ is defined as the distance from the hyperplane to the closest point across both classes. Given a hyperplane, to decide the margin, we need to first calculate the distance of a point x to the hyperplane $\mathcal{A}$. For simplicity, we define $\mathrm{w} = (\beta_1, \ldots, \beta_d)'$.

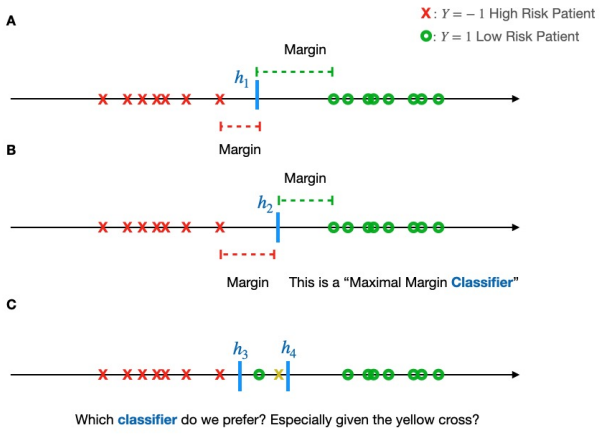

Figure 2: One-dimensional classification problem and the definition of margin.

Distance between a point $x \in \mathbb{R}^d$ to the hyperplane $\mathcal{A}$ Consider some point x, and let d be the vector from $\mathcal{A}$ to x of the minimum length. Our goal is to calculate the length of d (l_2 norm of d). Let x^{p} be the projection of x onto $\mathcal{A}$. Since d is parallel to w, we can write

$$x^{\mathrm{p}} = x - d, \quad d = \alpha \mathrm{w}, \ \alpha \in \mathbb{R}.$$

Since x^{p}, it satisfies $a + x^{\mathrm{p}'}\mathrm{w} = 0$. Therefore,

$$0 = a + x^{\mathrm{p}'}\mathrm{w} = a + (x - d)'\mathrm{w} = a + (x - \alpha \mathrm{w})'\mathrm{w},$$
$$\implies \alpha = \frac{\mathrm{w}'x + a}{\mathrm{w}'\mathrm{w}}.$$

The length of d:

$$||d||_2 = \sqrt{d'd} = \alpha \cdot \sqrt{\mathrm{w}'\mathrm{w}} = \frac{|a + \mathrm{w}'x|}{||\mathrm{w}||_2}.$$

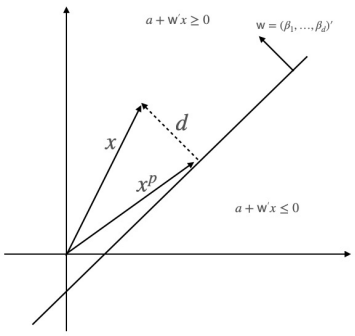

Figure 3: Distance between a point $x \in \mathbb{R}^2$ to the hyperplane $\mathcal{A}$.

Now, given training covariates $\{X_i\}_{i=1}^n$ and a hyperplane $\mathcal{A}$, the *margin* of $\mathcal{A}$ with respect to S is defined as:

$$\gamma(\mathrm{w}, a) = \min_i \frac{|a + \mathrm{w}'X_i|}{||\mathrm{w}||_2}.$$

By definition, the margin and hyperplane are scale invariant: $\gamma(c{\cdot}\mathrm{w}, c{\cdot}a) = \gamma(\mathrm{w}, a)$, for any $c \neq 0$.

2.2 Maximal Margin Classifier-SVM

The name SVM comes from the fact that the observations on the edge and within the margin are called *Support Vectors*. We will circle back for a more precise definition of support vectors at the end of the section. See a two-dimensional illustration in Figure 4. We can formulate our search for the maximum margin separating hyperplane as a constrained optimization problem. The objective is to maximize the margin under the constraint that all data points must lie on the correct side of the hyperplane:

$$\begin{aligned} \max_{a,\mathrm{W}} \quad & \gamma(\mathrm{w}, a) \\ \text{s.t.} \quad & Y_i(a + \mathrm{w}'X_i) \geq 0, \quad i = 1, \ldots, n. \end{aligned}$$

Or equivalently:

$$\begin{aligned} \max_{a,\mathrm{W}} \quad & \min_i \frac{|a + \mathrm{w}'X_i|}{||\mathrm{w}||_2} \\ \text{s.t.} \quad & Y_i(a + \mathrm{w}'X_i) \geq 0, \quad i = 1, \ldots, n. \end{aligned}$$

Because the hyperplane is scale invariant, we can fix the scale of b and w such that

$$\min_i |a + \mathrm{w}'X_i| = 1.$$

This suggests we can further relax our objective function to

$$\begin{aligned} \min_{a,\mathrm{W}} \quad & \mathrm{w}'\mathrm{w} \\ \text{s.t.} \quad & Y_i(a + \mathrm{w}'X_i) \geq 0, \quad i = 1, \ldots, n \\ & \min_i |a + \mathrm{w}'X_i| = 1. \end{aligned}$$

We can further show that the optimal solution of the above problem is equivalent to (How?)

$$\min_{a,\mathrm{W}} \quad \mathrm{w}'\mathrm{w}$$

$$\text{s.t.} \quad Y_i(a + \mathrm{w}'X_i) \geq 1, \quad i = 1, \ldots, n.$$

We've now transformed the problem into a form that can be efficiently solved. The above is an optimization problem with a convex quadratic objective and only linear constraints. Its solution gives us the optimal margin classifier. This optimization problem can be solved using commercial quadratic programming (QP) code.

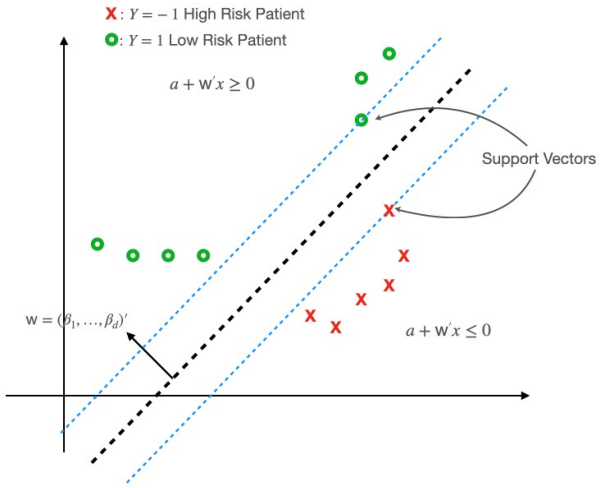

Figure 4: Support Vector Machine in 2-dimensions.

Support vectors For the optimal solution of the above problem, some training points will have tight constraints (why?), i.e.,

$$Y_i(a + \mathrm{w}'X_i) = 1.$$

We refer to these training points as support vectors. Support vectors are special because they are the training points that define the maximum margin of the hyperplane to the data set S. Thus, they determine the shape of the hyperplane. If you were to move one of them and retrain the SVM, the resulting hyperplane would change.

2.3 SVM with soft constraints

In the previous section, we discussed SVM under the constraint that no mis-classification is allowed. Let's circle back to the example given in Figure 2.C and compare the classifier h_3 and h_4:

1. Classifier h_3: no mis-classification error, but predict the high risk patient (yellow cross) as low risk
2. Classifier h_4: miss classify one low risk patient as high risk, but correctly predict the high risk patient profile

Which classifier would you prefer if you were a doctor?

In some cases, we would deliberately make mistakes so that our algorithm can detect high risk patients with a higher accuracy. Motivated by this consideration, we can revise the SVM with so-called "soft constraints." In addition, in practice,when there exists no separating hyperplane between the two classes, soft constraints can be helpful as well.

The soft constraints are done by the introduction of slack variables:

$$\begin{aligned} \min_{a,\mathrm{W}} \quad & \mathrm{w}'\mathrm{w} + C\sum_{i=1}^{n} \xi_i \\ \text{s.t.} \quad & Y_i(a + \mathrm{w}'X_i) \geq 1 - \xi_i, \\ & \xi_i \geq 0, \quad i = 1, \ldots, n. \end{aligned}$$

The slack variables ξ_i's allow the input x_i to move closer to the hyperplane, but there is a penalty in the objective function for such slackness. C is a tuning parameter: If C is very large, the SVM becomes very

strict and tries to get all points on one side of the hyperplane. If C is very small, the SVM becomes very loose and may "sacrifice" some points to obtain a simpler solution.

Two questions before ending this lecture:

1. If we do not want to misclassify any high-risk patient, what options do we have?
2. How can we revise the current algorithm so that SVM can work with the problem presented in Figure 5?

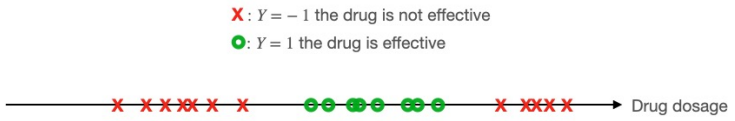

Figure 5: Can SVM be used to predict the drug-effective outcome?