[redacted] **Modern Statistical Prediction and Machine Learning** [redacted]

Homework 3 (due March 12): Linear Regression, Classification

[redacted] [redacted]

Theoretical Exercise

Problem 1: Margin

We say that a set of labeled vectors S_N (in $\mathbb{R}^p$) is linearly separable with a margin γ if there is a vector $v \in \mathbb{R}^p \backslash \{0\}$ such that for any $(x, y) \in S_N$, where $x \in \mathbb{R}^p$ and $y \in \{1, -1\}$:

$$\frac{y\langle v, x\rangle}{||v||} \geq \gamma$$

- Verify that this condition indeed corresponds to linear separability with distance γ from points to the hyperplane.

  Define the hyperplane by the equation $\langle v, x\rangle = 0$, where v is the normal vector to the hyperplane. The distance from a point x to the hyperplane is given by $\frac{|\langle v,x\rangle|}{||v||}$.

  Let $(x, y) \in S_N$ be any point. We can rewrite the condition given as:

  $$y\langle v, x\rangle \geq \gamma ||v||$$

  Substituting $\langle v, x\rangle = \langle v, x_0\rangle$ and $||v|| = 1$, we get:

  $$y(\langle v, x\rangle - \langle v, x_0\rangle) \geq \gamma$$

  This means that the distance of any point $(x, y) \in S_N$ to the hyperplane is at least γ, since y is either 1 or -1. Therefore, the hyperplane separates the points in S_N with a margin of at least γ, which implies that S_N is linearly separable.

- Generalize this formula for the case where the separating hyperplane does not necessarily pass through the origin.

  If the separating hyperplane does not pass through the origin, but we can to add an extra term to the equation of the hyperplane to account for it.

  Let v be the vector given in the condition, and let $w = \frac{v}{||v||}$, where w is the normal vector to a hyperplane in $\mathbb{R}^p$. Let b be the signed distance of the hyperplane from the origin, so that the hyperplane is defined by the equation $\langle w, x\rangle = b$ for any point x on the hyperplane.

  Choosing any point $(x_0, y_0) \in S_N$ and solve for b in the equation $\langle w, x_0\rangle = b + \delta$, where δ is the distance of the hyperplane from the origin along the direction of w. Then the hyperplane is given by the equation $\langle w, x\rangle = b + \delta$.

  Let $(x, y) \in S_N$ be any point. We can rewrite the condition given as:

  $$y\langle w, x\rangle \geq \gamma ||w||$$

  Substituting $\langle w, x\rangle = \langle w, x_0\rangle - \delta + \langle w, x - x_0\rangle$ and $||w|| = 1$, we get:

  $$y(\langle w, x_0\rangle - \delta + \langle w, x - x_0\rangle) \geq \gamma$$

  This means that S_N is linearly separable with distance γ from points to the hyperplane and with distance δ from the origin along the direction of w.

Problem 2: Lasso interpretation Suppose we are minimizing

$$\sum_{i=1}^{n}(y_i - \beta^T x_i - \beta_0)^2 \text{ subject to } ||\beta||_1 \leq s, \beta \in \mathbb{R}^p$$

.

- How will the training RSS change as s increases from 0 to infinity?

  As s increases from 0 to infinity, RSS will decrease as s increases, because the model will fit the data better. This will lead to overfitting.

- How will the test RSS change in the same setup?

  When the constraint s is close to 0, many of the β_i's are forced to be 0. This results in a very sparse model. The model is simple and the RSS is high. It cannot capture the complexity of the data. Both the bias and variance are high.

  As s increase from 0, constraint on many of the β_i's are relaxed, the test RSS will decreases. At some optimal value of s, the test RSS will be the smallest, achieving a balance between bias and variance.

  As s gets close to ∞, many of the β_i's are nonzero. The model here is complex and has a low bias but high variance. the test RSS will also be high, because the model overfits the training data and it does not generalize well to the test data.

- Assuming the Gaussian linear model, how will the bias of the estimator $\hat{\beta}_s^T x$ change as s increases from 0 to infinity?

  The LASSO estimate $\hat{\beta}_s$ is biased because the L-1 norm shrinks some of the coefficients towards zero with a penalty proportional to their absolute values.

  When we increase s, we are relaxing the L1 constraint on $\hat{\beta}_i$ coefficients. This means there coefficients will increase and the model is less bias. As s increases the model is less and less bias. When s approaches infinity, the coefficient becomes the least squares estimate.

- The same question for the variance of this estimator.

  As s increases from 0 to infinity, we are relaxing the L1 constraint on the coefficients. For the variance of training data, the variance will decrease as the model is getting less bias. For the test data, variance will be high when s is close to zero, and decrease when s increases to its optimal value, then increase again as s approaches infinity, because the model is overfitting the training data, thus giving high variance on the test data.

Problem 3: Cross validation error of Perceptron and SVM

- Upper bound the Leave-one-out cross-validation error of SVM in terms of the number of essential support vectors.

  The leave-one-out cross validation in terms of essential support vectors (ESV):

  $$\text{LOOCV error} = \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}[y_i \neq f_i(x_i)] \leq \frac{|ESV|}{n}$$

  where $|ESV|$ is the number of ESVs and n is the number of observations.

  We claim that we can upper bound the LOOCV error by counting how many times an ESV is misclassified. This is because, if we remove an ESV from the training set, it will change the optimal decision boundary. However, if we remove a non-ESV from the training set, it will not change the optimal decision boundary and have no effect on other observations.

- Assume that the Perceptron algorithm is run through the sample multiple times until it makes the first pass through the data without a single mistake. Upper bound the leave-one-our error of the classifier that corresponds to the weights that made this mistakes-free pass in terms of the margin.

  Let w be the weight vector that the Perceptron algorithm found after making a pass through the data without making any mistakes. Let γ be the margin of the classifier, i.e., the minimum distance from any point in the training set to the decision boundary.

  We will use Novikoff Theorem to show upper bounded of the margin. The theorem states that if $S_n = \{(x_1, y_1), ..., (x_n, y_n)\}$ in $\mathbb{R}^p$ is linearly separable with a margin γ, then the number of mistakes made by the Perceptron algorithm before it finds a linear separator is upper bounded by $\left(\frac{R}{\gamma}\right)^2$, where R is the maximum norm of any vector in S_N.

  Since the Perceptron algorithm has made a mistake-free pass through the data, we know that it has found a linear separator. Let k be the number of mistakes made by the Perceptron algorithm before it found this separator. Then, by the Novikoff Theorem, we have $k \leq \left(\frac{R}{\gamma}\right)^2$.

  In leave-one-out error of the classifier that makes a mistake-free pass, we remove the i^{th} vector $(x_i, y_i) \in S_n$, and run the algorithm on the remaining vectors. Since (x_i, y_i) is linearly separable, by the Novikoff Theorem, the number of mistakes made by the Perceptron algorithm on this reduced set is upper bounded by $\left(\frac{R}{\gamma}\right)^2$. Now, if we sum over all vectors in S_n, we get an upper bound on the leave-one-out error of $n\left(\frac{R}{\gamma}\right)^2$.

  Therefore, the leave-one-out error of the classifier that corresponds to the weights that made a mistake-free pass through the data is upper bounded by $n\left(\frac{R}{\gamma}\right)^2$.

- Explain how your findings imply that both algorithms have small prediction risk when the sample size goes to infinite. Specify which statistical model (data-generating model) you are using.

  We will use the central limit theorem. For a fixed dimensional space, the number of essential support does not change while the data points increases. In this inequality, $\frac{1}{n}\sum_{i=1}^{n} \mathbb{1}[y_i \neq f_i(x_i)] \leq \frac{|ESV|}{n}$ the risk decreases as n, the sample size increases. For the data generating process, we can use uniform distribution.

Problem 4: Optional

Problem 5: Classification on MNIST

see code

Problem 6: Ridge Regression

see code