Machine Learning Lecture 2

Nathan Kullus

September 25, 2018

1 Supervised Learning

1.1 Definition

Def: Given examples $X_1, Y_1, \ldots, X_n, Y_n$, we want to learn a prediction rule f such that for any new unseen example X, Y we have Y $\approx$ *f(x)* (approximately loosely, can have different meaning).
i.e.: Given a new pair X, Y where Y is hidden and we need to guess its hidden value (but why hidden?)

- unknowable (we will only realize in the future)
- need such an expert to tell us the value

1.2 What is X and Y?

X represents the features/covariates while Y represents the outputs/responses/labels
We classify the cases as

- $G = \{a, b\}$ is a binary classification where a and b are any arbitrary.
  - e.g. $G = \{1, 0\}, \quad G = \{happy, sad\}$
- $G = \{a, b, c, d \ldots\}$ where $|G| < \infty$, is a multi class classification.
- $G = R$, where R is regression.

2 First Classification Algorithm: k-Nearest Neighbor

2.1 What is k-NN?

Suppose we have a query X,

1. Find the k "closest" x values among the examples $X_1, X_2 \ldots, X_n$
2. We then index them $i_1, \ldots, i_k \in [n] = \{1, 2 \ldots, n\}$
3. Classify as the majority of the corresponding labels:

$$\hat{Y} = mode(\{Y_{i1}, Y_{i2} \ldots, Y_{ik}\})$$

   where mode is just exactly its mathematical meaning .

Q: But what does "close" exactly mean?
A: It is a **distance measure**.
Suppose X is a vector of numerical features:

$$X_i = \left\{ \begin{array}{c} xi_1 \\ \ldots \\ xi_p \end{array} \right\}$$

$Xi \in \mathbb{R}^p$, it is also known as the Euclidean distance

$$\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

Or we could use other distance such as: $||x - x'||_1 = \sum_{j=1}^{p} |x_j - x'_j|$ or $||x - x'||_\infty = max_j |x_j - x'_j|$

2.2 Constructing feature vectors

1. If we want to use distance functions listed above, we may want to first standardize the data:

$$x_{ij} \leftarrow \frac{(x_{ij} - \widehat{\mu_j})}{\widehat{\sigma_j}}$$

where $\widehat{\mu_j}$ and $\widehat{\sigma_j}$ is the mean and standard deviation of x_{1j}, ..., x_{nj}.
We want to standardize the data because certain features can impact the distance disproportionately, as when measurements are done with wildly varying units such as ounces vs. tons. This may not be an issue when all the features are of a similar nature. For example, x represents pixel brightness in MNIST handwritten example.

2. If some features are categorical, we can encode those as one-hot encodings.
For example, if the question asks which of university did you attend, among { Yale, Columbia, Cornell}, we could use three binary variables and encode them into Yale as $\begin{pmatrix}1\\0\\0\end{pmatrix}$, Columbia as $\begin{pmatrix}0\\1\\0\end{pmatrix}$, Cornell as $\begin{pmatrix}0\\0\\1\end{pmatrix}$.

3. Later in class, we will discuss about more abstract distances that could be used in kNN. For example, x can be movie plot summary texts, and distance is some sort of similarity between texts.

2.3 Out-of-Sample Evaluation

We have learned that kNN tries to make $\widehat{Y}$ equal to Y. In order to evaluate kNN, we want to know how often kNN successfully make $\widehat{Y}$ equal to Y.

If X,Y are random variables representing new random examples drawn from the population of examples, then we want low risk

$$R(f) = E[l(Y, F(x))]$$

where $R(f)$ indicates how often f is wrong, and l is some loss function. For now, we define loss function as $l(y, \widehat{y}) = \mathbb{I}[y \neq \widehat{y}]$, which means that if prediction is wrong, penalty is 1. If prediction is right, there is no penalty (penalty = 0).

To estimate $R(f)$, we can take a test set of labeled examples, x_1^{test}, y_1^{test}, ..., x_n^{test}, y_n^{test}, which are drawn from the population of examples. Then, we want to compute

$$\widehat{R_{n_{test}}}(f) = \frac{1}{n_{test}} \Sigma_{i=1}^{n_{test}} l\left(Y_i, f(x_i^{test})\right)$$

which is a consistent and unbiased estimate of $R(f)$.

In most of the times, we get the test set by splitting randomly from the training data. We cannot just use the training data, because it will cause bias, and it will be biased downward(which means overly optimistic). f will look better than it really is, and this would encourage over-fitting. For example, when we test f_{1-NN} on training data, $\widehat{R_n}(\widehat{f_{1-NN}})$ would be equal to 0, which is wrong an it will make some mistakes.

3 Bayes Classifier (and Bayes Rate)

kNN is version of the Bayes classifier. The best classifier is one that has the smallest $\hat{R}_f$ value.

$$\begin{aligned} R[f] &= \mathbb{E}[l(y, f(x))] \\ &= \mathbb{E}[\mathbb{I}[y \neq f(x)]] \\ &= \mathbb{E}[1 - \mathbb{I}[y = f(x)]] \\ &= \mathbb{E}[\mathbb{E}[1 - \mathbb{I}[y = f(x)]|x]] \\ &= \mathbb{E}[1 - \mathbb{E}[\mathbb{I}[y = f(x)]|x]] \\ &= \mathbb{E}[1 - \sum_{y \in G} \mathbb{P}(y = \hat{y}|X)\mathbb{I}[y = f(x)]] \end{aligned}$$

We can choose one $y \in G$ for each x such that $f(x) = y$. As a result, we need to choose a suitable y to minimize the function $R[f]$.

$$R[f] = \mathbb{E}[\max_{\forall x} \mathbb{P}(y = \hat{y}|X)\mathbb{I}[y = f(x)]]$$

$$\boxed{f'(x) = \arg\max_{y \in G} \mathbb{P}(y = \hat{y}|x) = mode(y|x)}$$

The Bayes classifier is then defined by the above equation. Further, $f'(x)$ is defined as the Bayes Rate. It stands for the fundamental limit of the predictability of y from x.

3.1 kNN as a probabilistic classifier

Lets look at how we can modify our estimate (from kNN) as a probability estimate.

$$\hat{\mathbb{P}}(\hat{y} = y|\hat{x} = x) = \frac{1}{k} \sum_{j=1}^{k} \mathbb{I}[\hat{y}_{i_k} = y]$$

On the right hand side of the equation, we sum up over the fraction of k labels corresponding to the k nearest examples that are equal to $\hat{y}$.

Our kNN prediction can be written as $\hat{Y} = \arg\max_{y \in G} \hat{\mathbb{P}}(\hat{y} = y|\hat{x} = x)$ mimics the Bayes classifier with an estimate on the the conditional probability.

For $G = 0, 1$, this is going to translate to

$$\begin{aligned} \hat{Y} &= \mathbb{I}[\mathbb{P}(\hat{y} = 1|x) > \frac{1}{2} \\ &= \begin{cases} 0 & \mathbb{P} \leq \frac{1}{2} \\ 1 & \mathbb{P} > \frac{1}{2} \end{cases} \end{aligned}$$

Modifying the threshold changes classification rates which may be different for different classes of problems.