INF283 Introduction to machine learning

Model solutions and grading criteria

18.2.2019

1 Basic concepts (10 p.)

1.1 Bias-variance tradeoff

a) Error can be decomposed into three parts: bias, variance, and irreducible error. The bias is an error due to modeling assumptions and the variance is an error due to variations in the training data. Typically, bias decreases and variance increases with model complexity. Thus, one typically trades between bias and variance.

b) Overfitting happens when a model has low bias and high variance.

Grading: 2p. for each item. Errors and omissions give deductions.

1.2 Cross-validation

c) Cross-validation can be used in model selection and evaluation to get an estimate of prediction error on unseen data points. Compared to hold-out validation, it reduces variation.

d) k-fold cross validation: Partition data randomly into k subsets. Repeat k times: train a model using $k-1$ subsets, evaluate it with the subset that was not used in training. In the end, take an average over all subsets.

Grading: 2p. for each item. Errors and omissions give deductions.

1.3 k-nearest neighbor classifier

When classifying a particular test point, we start by finding k training points that ar nearest to the test point. Then we predict the most common class among the k points.

Grading: 2p. Errors and omissions give deductions.

2 Probabilistic models (8 p.)

a)

$$\begin{aligned}
&P(C=0\,|\,x_1=1,x_2=0,x_3=1) \propto \\
&\quad P(C=0,x_1=1,x_2=0,x_3=1) \\
&= P(C=0)P(x_1=1\,|\,C=0)P(x_2=0\,|\,C=0)P(x_3=1\,|\,C=0) \\
&= \frac{4}{5}\frac{1}{3}\frac{1}{4}\frac{5}{8} \\
&= \frac{1}{24}
\end{aligned}$$

$$\begin{aligned}
&P(C=1\,|\,x_1=1,x_2=0,x_3=1) \propto \\
&\quad P(C=1,x_1=1,x_2=0,x_3=1) \\
&= P(C=1)P(x_1=1\,|\,C=1)P(x_2=0\,|\,C=1)P(x_3=1\,|\,C=1) \\
&= \frac{1}{5}\frac{2}{3}\frac{5}{6}\frac{1}{3} \\
&= \frac{1}{27}
\end{aligned}$$

Thus, $P(C=0 \mid x_1=1,x_2=0,x_3=1) > P(C=1 \mid x_1=1,x_2=0,x_3=1)$ and we predict class $C=0$

b)

$$\begin{aligned}
P(C=0\,|\,x_1=1,x_2=0) &\propto P(C=0,x_1=1,x_2=0) \\
&= P(C=0)P(x_1=1\,|\,C=0)P(x_2=0\,|\,C=0) \\
&= \frac{4}{5}\frac{1}{3}\frac{1}{4} \\
&= \frac{1}{15}
\end{aligned}$$

$$\begin{aligned}P(C=1\,|\,x_1=1,x_2=0) &\propto P(C=1,x_1=1,x_2=0)\\ &= P(C=1)P(x_1=1\,|\,C=1)P(x_2=0\,|\,C=1)\\ &= \frac{1}{5}\frac{2}{3}\frac{5}{6}\\ &= \frac{1}{9}\end{aligned}$$

Thus, $P(C=0\,|\,x_1=1,x_2=0) < P(C=1\,|\,x_1=1,x_2=0)$ and we predict class $C=1$

Grading:
2p. for knowing that you have to compute the probability of C given the observations, 4p. for computing the probabilities, and 2p. for predicting the class with the highest probability.

3 Ensemble methods (8 p.)

a) An ensemble model is a classifier (or a regressor) that is composed of several classifiers. Combining several models is useful because it can reduce error.

b) In bagging, one learns several independent classifiers for bootstrap samples. In boosting, one learns a sequence of classifiers such that the data points are weighted based on how well the previous classifiers classify them. Bagging can reduce only variance. Boosting, on the other hand, can reduce both bias and variance.

Grading: 4p. for each item. Errors and omissions give deductions.

4 Hierarchical clustering (8 p.)

Algorithm:

1. Put every point in a separate cluster
2. As long as there are more than one cluster:
   (a) Merge two clusters that are closest to each other

The distance between two clusters is measured by complete linkage, that is, it is the largest distance between a point belonging to the first cluster and a point belonging to the second cluster.

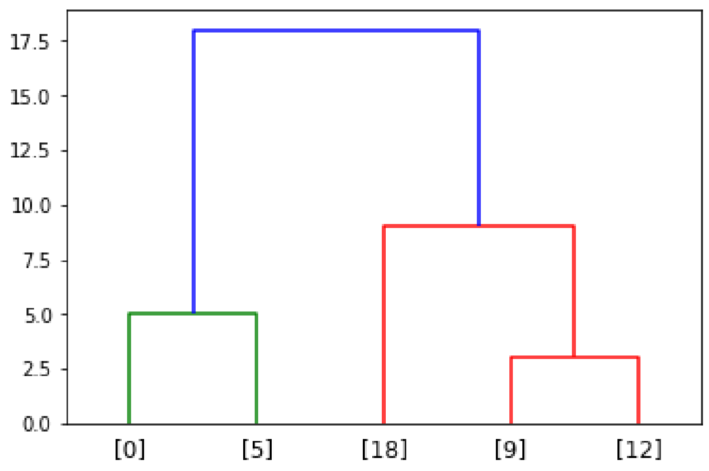
We start by putting all points in a separate cluster. The clusters are: $\{0\}$, $\{5\}$, $\{9\}$, $\{12\}$, $\{18\}$.

It is easy to see that the closest clusters are $\{9\}$ and $\{12\}$. Thus, we merge them. The new clusters are: $\{0\}$, $\{5\}$, $\{9, 12\}$, $\{18\}$. Distance from the new cluster $\{9, 12\}$ to $\{0\}$ is 12, distance to $\{5\}$ is 7 and distance to $\{18\}$ is 9.

Now, the closest clusters are $\{0\}$ and $\{5\}$ (distance 5) and we merge them. The new clusters are: $\{0, 5\}$, $\{9, 12\}$, $\{18\}$. Distance from the new cluster $\{0, 5\}$ to $\{9, 12\}$ is 12 and distance to $\{18\}$ is 18.

The closest clusters are $\{9, 12\}$, $\{18\}$ (distance 9). The new clusters are: $\{0, 5\}$, $\{9, 12, 18\}$. In the final step we merge the remaining two clusters.

The hierarchical clustering can be illustrated, for example, by the following tree:



Grading:

2p. if the output is a hierarchical clustering, 4p. for correct use of the algorithm, 2p. for correct use of complete linkage

5 Dimensionality reduction (8 p.)

5.1 PCA

a) The first principal component is the direction that maximizes the projected variance. Clearly, the direction is $[1, 1]$, that is, the line that goes through all data points.

b) The first principal component goes through all data points and thus it explains 100% of variance. That is, there is no need to use the second component.

Grading: 4p. Errors and omissions give deductions.

5.2 Neural networks

Autoencoders are neural networks that can be used in dimensionality reduction. They consist of two parts: encoder and decoder. The encoder map input $\mathbf{x}$ into an internal presentation $\mathbf{h}$, that is, $\mathbf{h} = f(\mathbf{x})$. Then the decoder produces a reconstruction of the input $\mathbf{r} = g(\mathbf{h})$. During training, we use loss $L(\mathbf{x}, g(f(\mathbf{x})))$.

When $\mathbf{h}$ has lower dimensionality than $\mathbf{x}$, we can use the encoder to produce a low dimensional representation of $\mathbf{x}$.

Grading:

Mentioning the term *autoencoder* 1p. A neural network that tries to predict its input 1p. A low dimensional middle layer 1p. After training, use the middle layer as a (low dimensional) representation of the input 1p.

6 Kernelized regression

a) A kernel if a function $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$. It is a dot product in some feature space: $K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$.

b) We can use kernels if data appears only as dot products between data vectors $\phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$. In that case, we replace all appearances of $\phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$ with $K(\mathbf{x}_i, \mathbf{x}_j)$. There are two such appearances in the formulation.

First, we note that $\mathbf{B}$ can be expressed as a kernel matrix. That is, $\mathbf{B}_{ij} = \phi(\mathbf{x}_i)^T\phi(\mathbf{x}_j) = K(\mathbf{x}_i, \mathbf{x}_j)$.

Second, the prediction can be written as

$$y^* = \boldsymbol{\beta}^T\phi(\mathbf{x}) = \sum_{i=1}^{n} \gamma_i \phi(\mathbf{x}_i)^T\phi(\mathbf{x}) = \sum_{i=1}^{n} \gamma_i K(\mathbf{x}_i, \mathbf{x}).$$

c) The data points do not have to be explicitly projected to d'-dimensional feature space but we can do our computations in the d-dimensional input space. This is computationally efficient.

Grading:
a) 1p. b) 6p. c) 1p.
In b), you get 3p. for correctly recognizing the kernel matrix and 3p. for correct prediction. Small errors deduct 1p. in both cases. A reasonable attempt to right direction gives 1p.