INF283 Introduction to machine learning

Model solutions and grading criteria

3.12.2018

1 Basic concepts

1.1 Overfitting

a) Overfitting means that our model fits too tightly to the particular training data and does not generalize well.

b) The goal of learning is to generalize (perform well on unseen data) and overfitting prevents generalization

c) Typically, test (or validation) error is much larger than training error

d) For example, use more data, use simpler models, regularization, pruning, early stopping, dropout, ensemble models, dimensionality reduction, Bayesian models with priors

Grading:
a) 1p. b) 1p. c) 1p. d) 3p. (1p. for each method, if method listed does not help to avoid overfitting then 0p.)
Small errors or unclarities give 0.5p. deductions

1.2 Model selection and evaluation

e) Degree 5 polynomial has the smallest validation error and therefore we choose it.

f) We cannot get an unbiased estimate with the current information. We have used our validation set for model selection and thus validation

error is not an unbiased estimate of error on the unseen data. To get an unbiased estimate, we should evaluate the degree 5 polynomial on a separate test set.

Grading:
e) 2p. f) 2p.
Small errors or unclarities give 1p. deductions

2 Neural networks

Forward pass:

$z = w_1 x = 2 \cdot 1 = 2$
$h = f(z) = \max(0, 2) = 2$
$\hat{y} = w_2 h = 3 \cdot 2 = 6$

Backward pass:

We are going to need the following derivatives:
$\frac{\partial L}{\partial \hat{y}} = -(y - \hat{y}) = \hat{y} - y$
$\frac{\partial \hat{y}}{\partial w_2} = h$
$\frac{\partial \hat{y}}{\partial h} = w_2$

$$\frac{\partial h}{\partial z} = \begin{cases} 1 & \text{if } z > 0 \\ 0 & \text{if } z \leq 0 \end{cases}$$

$\frac{\partial z}{\partial w_1} = x$

The compute the gradient, we use backpropagation. Using chain rule, we get the following partial derivatives:

$$\begin{aligned} \frac{\partial L}{\partial w_1} &= \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial h} \frac{\partial h}{\partial z} \frac{\partial z}{\partial w_1} \\ &= (\hat{y} - y) \cdot w_2 \cdot 1 \cdot x \\ &= (6 - 5) \cdot 3 \cdot 1 \cdot 1 \\ &= 3 \end{aligned}$$

and

$$\begin{aligned}\frac{\partial L}{\partial w_2} &= \frac{\partial L}{\partial \hat{y}}\frac{\partial \hat{y}}{\partial w_2} \\ &= (y-\hat{y})\cdot h \\ &= (6-5)\cdot 2 \\ &= 2\end{aligned}$$

Updates with gradient descent:

$$\begin{aligned}w_1 &\leftarrow w_1 - \gamma\frac{L}{\partial w_1} \\ &= 2 - 0.1\cdot 3 \\ &= 1.7\end{aligned}$$

$$\begin{aligned}w_2 &\leftarrow w_2 - \gamma\frac{L}{\partial w_2} \\ &= 3 - 0.1\cdot 2 \\ &= 2.8\end{aligned}$$

Grading:
Forward pass 2p. Backpropagation 4p. Parameter update 2p.
Deductions:
Errors in derivatives 0.5-1p. Numerical errors 0.5-1p.

3 K-means clustering

Initial cluster centers: $\mu_1 = 3$ and $\mu_2 = 7$

Assign each data points to the closest cluster center. We get partition $D_1 = \{0, 1, 3\}$ and $D_2 = \{7, 9\}$

Update cluster means: $\mu_1 = \frac{0+1+3}{3} = 1\frac{1}{3}$ and $\mu_2 = \frac{7+9}{2} = 8$

Assign data points to the closest cluster center. We get partition $D_1 = \{0, 1, 3\}$ and $D_2 = \{7, 9\}$. We observe that the partition did not change. Thus, we can stop.

Grading:

The idea of k-means clustering 4p. Correct cluster assignment step 2p. Correct cluster mean update step 2p.

Deductions:

Numerical errors 0.5-1p.

Note that the task was to simulate Lloyd's algorithm. If you get the correct clustering using some other method then you will be penalized 4-7 points depending on how closely your clustering algorithm resembles Lloyd's algorithm.

4 PCA

PCA is a dimensionality reduction technique. The goal is to represent the data with less dimensions while preserving as much variance as possible.

PCA finds principal components, that is, directions that have high variance. For example, the first principal component is the direction with maximum variance,the second PC the direction with second most variance etc. The new features are projections of the original data points to the principal components. In other words, new features are linear combinations of the original ones.

Grading:

Goal 4p. Output 4p.

Unclarities or missing key points results in deductions. Factually incorrect statements give deductions.

5 Independencies in Bayesian networks

There are 8 d-separated pairs (recall that if there is no arc between two variables then they are d-separated given some set of variables). In all cases there are several correct separating sets. The column "Example sep. set" gives an example of a valid separating set. The column "Conditions" in the table below tells conditions that all valid separating set have to satisfy.

Pair	Example sep. set	Conditions
A, C	{B}	All sets must contain B. If a set contains F then it must contain D
A, E	{B}	All sets must contain B. If a set contains F then it must contain either C or D
A, F	{B, D}	All sets must contain D. All sets must contain either B or C
B, D	{A}	All sets must contain A. If a set contains F then it must contain also C
B, F	{A, C}	All sets must contain C. All sets must contain either A or D
C, D	{A}	All sets must contain either A or B. Set cannot contain F
D, E	{A}	All sets must contain A or B. If a set contains F then it must contain C
E, F	{A, C}	All sets must contain C. All sets must contain either A, B, or D

Grading:

Each correct pair with correct conditioning set +1p. Each correct pair with incorrect conditioning set +0.25p. Each incorrect pair -1p.

Minimum 0p.

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

Second, the prediction can be written as

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

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.