Machine Learning Review

Molin Liu

December 16, 2019

Introduction

This is a review work for the Machine Learning course in *Univeristy of Glasgow*.

1 Regression

1.1 Linear Regression

For $x = (x_1, x_2, ..., x_m)$, where x_i is the i_{th} attribute of x, a lenear model tries to train a linear combination function from these attributes, i.e:

$$f(x) = w_1x_1 + w_2x_2 + ... + w_mx_m + b$$

which can be written in vector as:

$$f(x) = w^Tx + b$$

where $w = (w_1, w_2, ..., w_m)$.

The linear regression model is trained by *loss function*. Generally, we define our *loss function* as:

$$min \sum_{i=1}^{m} (y_i - f(x_i))^2$$

1.1.1 Loss Function

There are several kinds of *loss functions* we can choose from.

- **L1-norm**: L1-norm can be represented as follow:

$$S = \sum_{i=1}^{m} |y_i - f(x_i)|$$

- **L2-norm**: L2-norm is what we used above:

$$S = \sum_{i=1}^{m} (y_i - f(x_i))^2$$

1.1.2 Least Square Solution

The solution of **Least Square** is:

$$\vec{w} = \left(X^\top X\right)^{-1} X^\top Y$$

Prove:
The error vector $(Y - X\vec{w})$ should be orthogonal to every column of X:

$$(Y - X\vec{w}) \cdot X_j = 0$$

which can be written as a matrix equation:

$$(Y - X\vec{w})^\top X = \overrightarrow{0}$$

Then we can easily derive the equation from the following process:

$$\begin{aligned} X^\top(Y - X\vec{w}) = X^\top Y - X^\top X\vec{w} = 0 \\ \Longrightarrow \left(X^\top X\right)\vec{w} = X^\top Y \\ \Longrightarrow \quad \vec{w} = \left(X^\top X\right)^{-1} X^\top Y \end{aligned}$$

1.1.3 Conclusion

1.2 Polynomial Regression

The **Polynomial Regression** can be written as:

$$t = w_0 + w_1 x + w_2 x^2 + w_3 x^2 + \ldots + w_K x^K = \sum_{k=0}^{K} w_k x^k$$

Define the loss funcion:

$$\mathcal{L} = \frac{1}{N}(\mathbf{t} - \mathbf{X}\mathbf{w})^\top(\mathbf{t} - \mathbf{X}\mathbf{w})$$

1.2.1 Generalization & Overfitting

We can find out that the **loss** will always decrease as the model is made more complex.
How to choose the right model complexity? **Cross-validation**

1.2.2 Cross-Validation

2 Classification

The **Classification** task is to classify a set of N objects x_i with attributes. Each object has an associated label t_i

Probabilistic classifier produce a probability of class membership:
$P(t_{\text{new}} = k|\mathbf{x}_{\text{new}}, \mathbf{X}, \mathbf{t})$

non-Probabilistic classifier produce a hard assignment:
$t_{new} = 1$ or $t_{new} = 0$

2.1 KNN

K-Nearest Neighbours(KNN)
- Non-probabilistic classifier;
- Supervised trainning;
- Fast;
- We can use CV to find the right K;

2.1.1 Problem

- As K increases, the small classes will disppear.

2.2 Logistic Regression

2.3 SVM

2.3.1 Hard Margin

If the training data is linearly seperable, we can select two parallel hyperplanes that separate the two classes of data, so that the distance between them is as large as possible.

These hyperplanes can be described by the equations:

$$\vec{w} \cdot \vec{x}_i - b \geq 1, \text{ if } y_i = 1$$

or

$$\vec{w} \cdot \vec{x}_i - b \leq -1, \text{ if } y_i = -1$$

We can easily infer that

$$y_i (\vec{w} \cdot \vec{x}_i - b) \geq 1, \quad \text{for all } 1 \leq i \leq n$$

We want to maximise $\gamma = \frac{1}{||\mathbf{w}||}$, equivalent to minimising $||\mathbf{w}||$

Note: y_i is the label in the data, which is in $\{-1, 1\}$, rather than vertical axis value of the data.

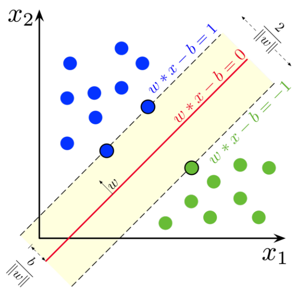

Figure 1: Hard margin

2.3.2 Soft Margin

Soft-margin function for the data are not linearly seperable.

2.3.3 Inner Product

3 ROC

Sensitivity/Recall

$$S_e = \frac{TP}{TP + FN}$$

Specificity

$$S_p = \frac{TN}{TN + FP}$$

4 Unsupervised Learning

4.1 K-Means

4.2 Gaussion Mixture Model