Machine Learning Notes

beck

2021 年 1 月 7 日

Week1

1. Classification problem: discrete valued output. $(0, 1, 2 \ldots)$
2. Regression problem: to predict a continuous valued output.
3. Supervised learning: w/ correct answer.
4. Unsupervised learning: w/o correct answer, to find structure in data.
5. Linear regression: fit a straight line. (hypothesis function is linear, e.g.$H(x) = Ax + B$)
6. Hypothesis function: maps input x to output y.

$$H(x) = y$$

7. Cost function: choose model parameters θ_i.

$$J(\theta_0, \theta_1, \ldots, \theta_i)\ (i = 0, 1, \ldots)$$

8. Linear regression cost function: mean squared error(MSE).

$$J(\theta_i) = \frac{1}{2m} \sum_{i=1}^{m} (h(x_i) - y_i)^2$$

9. Contour plots: 等高线图
10. Converge/Diverge: 收敛/发散
11. Gradient descent algorithm: minimize cost function.

$$\begin{aligned} & \textit{repeat until convergence } \{ \\ & \theta_j = \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1, \ldots, \theta_j) \\ \} \end{aligned}$$

12. Gradient descent simultaneous update:

$$
\begin{aligned}
&temp0 = \theta_0 - \alpha \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1, \ldots, \theta_j) \\
&temp1 = \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1, \ldots, \theta_j) \\
&\ldots \\
&tempj = \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1, \ldots, \theta_j) \\
&\theta_0 = temp0 \\
&\theta_1 = temp1 \\
&\ldots \\
&\theta_j = tempj
\end{aligned}
$$

13. No need to decrease α over time, because gradient descent will automatically take smaller steps. ($|\frac{\partial}{\partial \theta} J(\theta)|$ decreases as approaching local minimum, and finally turn 0)

14. Batch gradient descent: each step of gradient descent uses all the traning examples.

15. For the specific choice of cost function $J(\theta)$ used in linear regression, there are no local optima (other than the global optimum) (optimum, plural noun: optima)

16. Multivariate linear regression: n features

$$
\begin{aligned}
H_\theta(X) &= \theta_0 + \theta_1 x_1 + \ldots + \theta_n x_n \\
&= \theta^T X \\
&= \begin{bmatrix} \theta_0 & \theta_1 & \ldots & \theta_n \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{bmatrix} (x_0 = 1)
\end{aligned}
$$

17. Gradient descent with multiple features:

$$
\theta_j = \theta_j - \alpha \frac{1}{m} \sum i = 1^m (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)}
$$

18. Feature scaling: mean normalization(均值归一化), replace x_i with $\frac{x_i-\mu}{\sigma}$ or $\frac{x_i-\mu}{S}$ where S is the scale of data.($S = Max - Min$)

19. Gradient descent should decrease after every iteration, if not, consider using smaller α

20. Polynomial regression

21. Normal equation: no need to choose α, no need to iterate, slow if n is very large.(e.g.$n = 100000$);
When X^TX is non-invertable/singular/degenerate, use pseudo invertion.

$$\begin{cases} redundant\ features: & linearly\ dependent \\ too\ many\ features(m \leq n): & use\ regularization\ or \\ & delete\ some\ features \end{cases}$$

$$\theta = (X^TX)^{-1}X^Ty$$