Contents

Chapter 1

Session 1

1.1 Matrix notes

Multiplying Matrices: Number of columns in the first matrix must equal the number of rows in the second matrix

$$n \times m \ * \ m \times z \ = \ n \times z$$

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \times \begin{bmatrix} a & b \\ c & d \end{bmatrix} = \begin{bmatrix} 1 * a + 2 * c & 1 * b + 2 * d \\ 3 * a + 4 * c & 3 * b + 4 * d \end{bmatrix}$$

Chapter 2

test

2.1 Linear Regression with One Variable/Univariate Linear Regression

Link: Model Representation

$$h_\theta(x) = \theta_0 + \theta_1 x$$

To break it apart, it is $\frac{1}{2}\bar{x}$ where $\bar{x}$ is the mean of the squares of $h_\theta(x^{(i)}) - y^{(i)}$, or the difference between the predicted value and the actual value.

This function is otherwise called the "Squared error function", or "Mean squared error". The mean is halved $(\frac{1}{2})$ as a convenience for the computation of the gradient descent, as the derivative term of the square function will cancel out the $\frac{1}{2}$ term.

2.2 Model and Cost Function

Link: Model and Cost Function

We can measure the accuracy of our hypothesis function by using a cost function. This takes an average difference (actually a fancier version of an average) of all the results of the hypothesis with inputs from x's and the actual output y's.

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$$

2.3 Batch Gradient Descent for Linear Regression One Variable

Link: Gradient Descent for Linear Regression

When **specifically applied to the case of linear regression**, a new form of the gradient descent equation can be derived. We can substitute our actual cost function and our actual hypothesis function and modify the equation to:

$$\textit{Repeat until convergence}:$$

$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_0^{(i)}$$

$$\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_1^{(i)}$$

where m is the size of the training set, θ_0 a constant that will be changing **simultaneously** with θ_1 and x_i, y_i are values of the given training set (data). h_θ is the hypothesis, in this case

Univariate Linear Regression - section 2.1. It is possible to omit $x_0^{(i)}$ as it is 1 in case of univariate linear regression. Note also that

$$\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_0^{(i)} = \frac{\partial}{\partial\theta_0}J(\theta_0, \theta_1)$$

$$\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_1^{(i)} = \frac{\partial}{\partial\theta_1}J(\theta_0, \theta_1)$$

Where $J(\theta_0, \theta_1)$ is the Model and Cost Function in section 2.2.

Generalised algorithm

$$Repeat\ until\ convergence:$$
$$\theta_j := \theta_j - \alpha\frac{\partial}{\partial\theta_0}J(\theta_0, \theta_1)$$
$$(for\ j = 1\ and\ j = 0)$$

2.4 Unclassified

Supervised Learning `https://www.coursera.org/learn/machine-learning/lecture/1VkCb/supervised-learning`.
We give the algorithm data sets with right answers (Training set). Algorithm must produce more right answers.
Training set -¿ Learning Algorithm -¿ outputs function h (hypothesis). Job of hypothesis is get the size of house as input and output estimated value. h maps from x's to y's

- **Regression** House price prediction is regression problem - predict continuous valued output (price).
- **Classification** Classification: Predict a discrete valued output (0 or 1). It can be more than two values.
  Also contains a different way of plotting classification problems - different symbols on one axis instead of two axes.

Unsupervised Learning `https://www.coursera.org/learn/machine-learning/lecture/olRZo/unsupervised-learning`
Given data with the same label. "Here is a data set, find some structure in it".

- **Clustering algorithm** E.g. breast cancer thing. Or google news e.g. clusters news related to specific news story. Genes issue - cluster people.
- **Examples** : Organise computing clusters, social network analysis, Market segmentation, Astronomical data analysis. Cocktail party problem: Overlapping voices

2.4.1 Model Representation

Training set: Data set with right answers

2.4.2 Notation

m	Number of training examples
$x's$	"input" variable / features
$y's$	"output" variable / "target" variable
(x, y)	one training example
$(x^{(i)}, y^{(i)}$	i^{th} training example
$h(hypothesis)$	maps from $x's$ to $y's$
Linear Regression with one variable or Univariate linear regression	$h_\theta(x) = \theta_0 + \theta_1 x$
θ_0,θ_1	Model Parameters
Linear Regression Cost Function	
Linear Regression Minimize Square Difference Minimise over $\theta_0\theta_1$	$$\frac{1}{2m}\sum_{i=1}^{m}((h_\theta(x^(i)) - y^{(i)})^2$$
Cost function (Squared error cost function) Minimise cost function over $\theta_0\theta_1$	$$J(\theta_0, \theta_1) = \frac{1}{2m}\sum_{i=1}^{m}((h_\theta(x^(i)) - y^{(i)})^2$$
Batch Gradient Descent	For minimising. It goes to a local minimum. Batch because it looks at the entire training set.
α	Learning rate (Always positive)
Linear Regression	It is always a Convex function (Bowl-shaped)

Table 2.1: Symbols

Chapter 3

Error analysis

3.1 Error Metrics for Skewed Classes

An example of skewed classes is when the ratio of positive to negative examples is very close to one of two extremes. In this case a simple fixed predictor of $y = 0$ or $y = 1$ might give a high accuracy/low error. In such cases we can use precision and recall, which are more robust error metrics.

Predicted class	Actual class 1	0
1	true positive	false positive
0	false negative	true negative

Table 3.1: Comparing predicted classes with actual classes.

$$\text{Accuracy} = \frac{\text{true positives} + \text{true negatives}}{\#\ \text{samples}} \tag{3.1}$$

$$\text{Precision} = \frac{\text{true positives}}{\#\ \text{predicted positives}} = \frac{\text{true positives}}{\text{true positives} + \text{false positives}} \tag{3.2}$$

$$\text{Recall} = \frac{\text{true positives}}{\#\ \text{actual positives}} = \frac{\text{true positives}}{\text{true positives} + \text{false negatives}} \tag{3.3}$$

The recall is zero for a constant predictor $y = 0$. In cases with skewed classes it is not possible for simple constant classifiers like $y = 0$ or $y = 1$ to have both a high precision and recall. The usual convention is to set $y = 1$ for the rare class.

The F_1 score can be used to compare precision and recall in order to evaluate the trade off between high precision and high recall:

$$F_1 = \frac{2PR}{P + R}, \tag{3.4}$$

where P is the precision and R is the recall. If either P or R is zero the F_1 score is zero. Both P and R have to be one for the F_1 score to be one.