Machine Learning - Week 3 - Logistic Regression

T1 - Classification & Representation

L1 - Classification

Classification examples:

- Email: Spam or Not Spam
- Online Transactions: Fraudulent (Yes/No)?

$y \in \{0,1\}$, when 0: "Negative Class"
Binary Classification 1: "Positive Class"

Could also have $y \in \{0,1,2,3...\}$ (Multiclassification)

Threshold Classifier output $h_\theta(x)$ at 0.5: Uses Linear Regression

If $h_\theta(x) \geq 0.5$, predict $y=1$
If $h_\theta(x) < 0.5$, predict $y=0$

Note: Applying linear regression to a classification problem often isn't a great idea

Classification: $y = 0$ or 1
$h_\theta(x)$ can be >1 or <0
Logistic Regression: $0 \leq h_\theta(x) \leq 1$
└ A classification algorithm

L2 - Hypothesis Representation

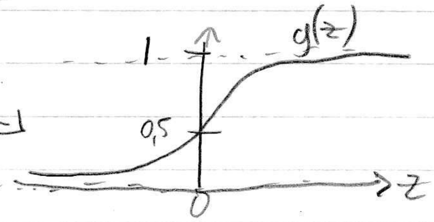
- What is the function we will use to represent our hypothesis when we have a classification problem.

We want $0 \leq h_\theta(x) \leq 1$

Sigmoid or Logistic function

$$h_\theta(x) = g(\theta^T x) \quad \text{where } g(z) = \frac{1}{1+e^{-z}}$$

$$h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}$$



→

Interpretation of Hypothesis Output

$h_\theta(x)$ = estimated probability that $y=1$ on input x

e.g. If $x = \begin{bmatrix} x_0 \\ x_1 \end{bmatrix} = \begin{bmatrix} 1 \\ \text{tumorSize} \end{bmatrix}$

$h_\theta(x) = 0.7$

∴ Tell patient that 70% chance of tumor being malignant

i.e. $h_\theta(x) = P(y=1 \mid x; \theta)$ ← "Probability that $y=1$ given x, parameterized by θ"

Also: $P(y=0 \mid x;\theta) + P(y=1 \mid x;\theta) = 1$

$P(y=0 \mid x;\theta) = 1 - P(y=1 \mid x;\theta)$

L3 - Decision Boundary

Recall $h_\theta(x) = g(\theta^T x) = P(y=1 \mid x;\theta)$

$g(z) = \frac{1}{1+e^{-z}}$

See graph on prev page

Suppose predict "$y=1$" if $h_\theta(x) \geq 0.5$

↳ i.e. if $\theta^T x \geq 0$

Predict "$y=0$" if $h_\theta(x) < 0.5$

↳ i.e. if $\theta^T x < 0$

$g(z) \geq 0.5$ when $z \geq 0$

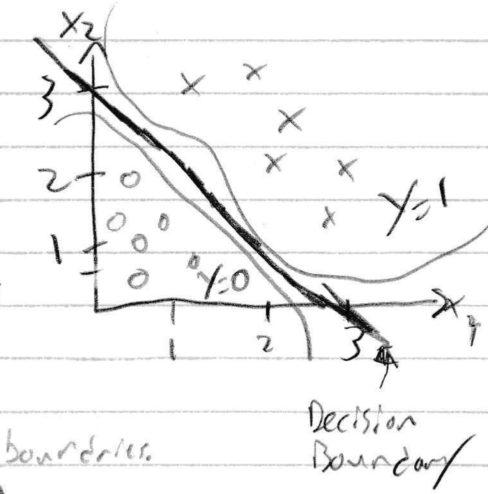
∴ $h_\theta(x) = g(\theta^T x) \geq 0.5$ wherever $\theta^T x \geq 0$

E.g. Suppose $h_\theta(x) = g(\underset{-3}{\theta_0} + \underset{1}{\theta_1} x_1 + \underset{1}{\theta_2} x_2)$

∴ $\theta = \begin{bmatrix} -3 \\ 1 \\ 1 \end{bmatrix}$

∴ Predict "$y=1$" if $\overbrace{-3 + x_1 + x_2}^{\theta^T x} \geq 0$

$x_1 + x_2 \geq 3$



Note that you can also have non-linear decision boundaries.

eg. $h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2)$

Machine Learning - Week 3 - Logistic Regression

T2 - Logistic Regression Model
L1 - Cost Function

Training set: $\{(x^{(1)},y^{(1)}),(x^{(2)},y^{(2)}),\ldots,(x^{(m)},y^{(m)})\}$
m examples

$$X \in \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{bmatrix} \quad \text{length of } n+1 \qquad x_0 = 1,\ y \in \{0,1\}$$

$$h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}$$

How to choose parameter θ?

Recall Linear Regression Cost function:

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

Say $\mathrm{Cost}(h_\theta(x^{(i)}), y^{(i)}) = \frac{1}{2}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$

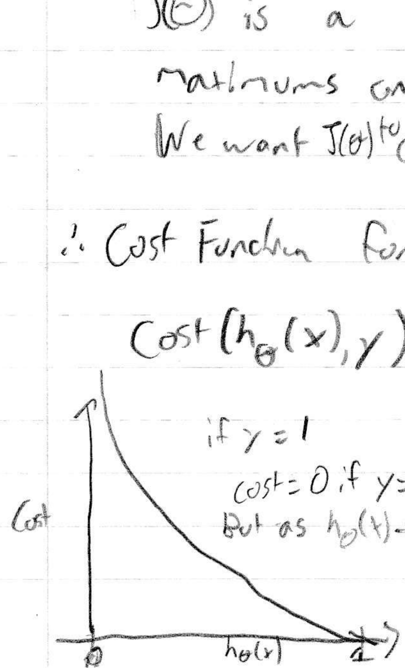
then $\mathrm{Cost}(h_\theta(x), y) = \frac{1}{2}\left(h_\theta(x) - y\right)^2$

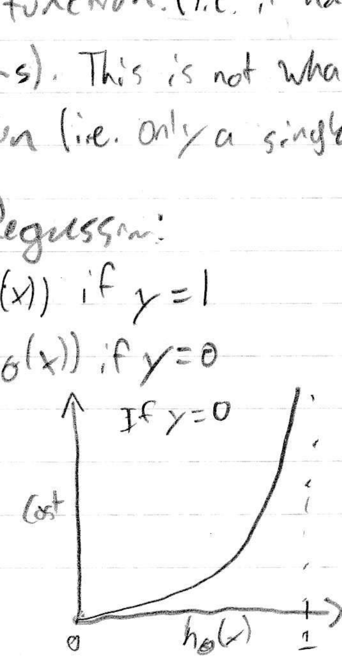
For Logistic Regression

If $h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}$ and we plug those values into $J(\theta)$, $J(\theta)$ is a non-convex function. (i.e. it has a lot of local maximums and minimums). This is not what we want. We want $J(\theta)$ to be a convex function (i.e. only a single minimum)

∴ Cost Function for Logistic Regression:

$$\mathrm{Cost}(h_\theta(x), y) \begin{cases} -\log(h_\theta(x)) & \text{if } y=1 \\ -\log(1-h_\theta(x)) & \text{if } y=0 \end{cases}$$





L2 - Simplified Cost Function and Gradient descent

Recall: $J(\Theta) = \frac{1}{m}\sum_{i=1}^{m} Cost(h_\Theta(x^{(i)}), y^{(i)})$

$$Cost(h_\Theta(x), y) = \begin{cases} -\log(h_\Theta(x)) & \text{if } y=1 \\ -\log(1-h_\Theta(x)) & \text{if } y=0 \end{cases}$$

Note: $y = 0$ or 1 always.

∴ We can say:

$$Cost(h_\Theta(x), y) = -y\log(h_\Theta(x)) - (1-y)\log(1-h_\Theta(x))$$

∴ Logistic Regression cost function

$$J(\Theta) = -\frac{1}{m}\left[\sum_{i=1}^{m} y^{(i)}\log h_\Theta(x^{(i)}) + (1-y^{(i)})\log(1-h_\Theta(x^{(i)}))\right]$$

To fit parameters Θ:

$\min_\Theta J(\Theta)$

To make a prediction given new x:

Output $h_\Theta(x) = \frac{1}{1+e^{-\Theta^T x}}$ Recall output is $p(y=1|x;\Theta)$

Want $\min_\Theta J(\Theta)$:

Repeat {

$\Theta_j := \Theta_j - \alpha \frac{d}{d\Theta_j} J(\Theta)$ [Simultaneously update all Θ_j]

}

$\hookrightarrow = \frac{1}{m}\sum_{i=1}^{m}(h_\Theta(x^{(i)}) - y^{(i)})x_j^{(i)}$

Note that algorithms for linear regression and logistic regression look exactly the same, but

For linear regression: $h_\Theta(x) = \Theta^T x$

For logistic regression: $h_\Theta(x) = \frac{1}{1+e^{-\Theta^T x}}$

L3 - Advanced Optimization

Say we have Cost Function $J(\theta)$. Want $\min_\theta J(\theta)$
Given θ, we have code that can compute
- $J(\theta)$
- $\frac{d}{d\theta_j} J(\theta)$ For $j = 0, 1, \ldots, n)$

Gradient descent:
Repeat {
$\theta_j := \theta_j - \alpha \frac{d}{d\theta_j} J(\theta)$
}

Other Optimization Algorithms
- Conjugate Gradient
- BFGS
- L-BFGS

Advantages
- No need to manually pick α
- often faster than gradient descent

Disadvantages:
- More complex

T3 - Multiclass Classification

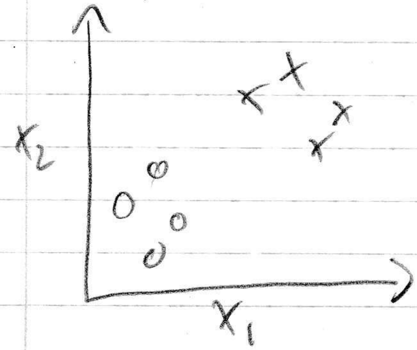
L1 - Multiclass Classification : One-vs-all

Multiclass Classification Example:

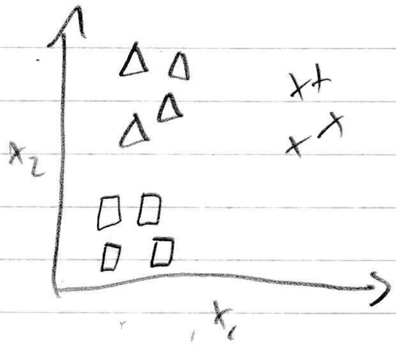
Email Foldering/tagging: Work, Friends, Family, Hobby

$y=1$ $\quad y=2$ $\quad y=3$ $\quad y=4$

Binary Classification



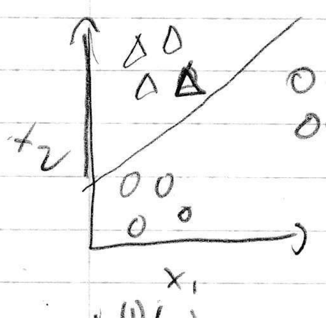
Multiclass Classification



One-vs-all (one-vs-Rest)

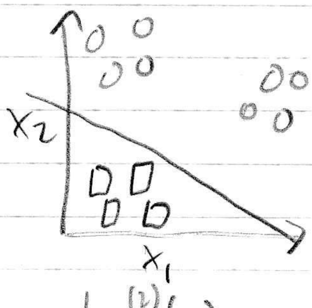
Divide into 3 classes

Class 1: $\triangle$



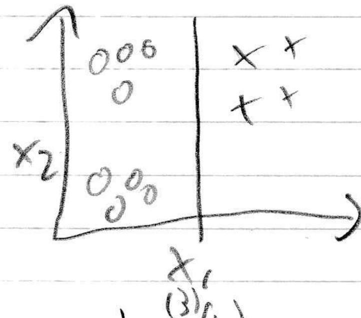
$h_\theta^{(1)}(x)$

Class 2: $\square$



$h_\theta^{(2)}(x)$

Class 3: $\times$



$h_\theta^{(3)}(x)$

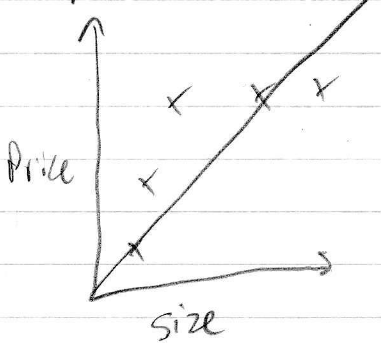
$$\therefore h_\theta^{(i)}(x) = P(y=i \mid x;\theta) \quad (i=1,2,3)$$

i.e. Train a logistic regression classifier $h_\theta^{(i)}(x)$ for each class i to predict the probability that $y=i$.

On a new input x, to make a prediction, pick the class i that maximizes $\max_i h_\theta^{(i)}(x)$

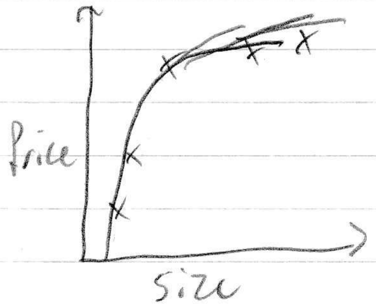
T4 - Solving the Problem of Overfitting (Regularization)
L1 - The problem of Overfitting

Example: Linear Regression

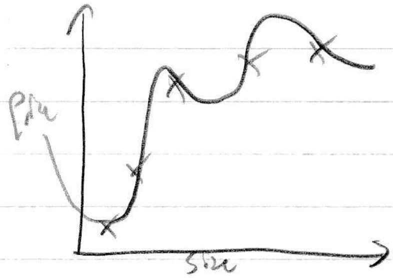


- Underfitting
- High bias

$\theta_0 + \theta_1 x$



$\theta_0 + \theta_1 x + \theta_2 x^2$



- Overfitting
- High variance

$\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$

Overfitting: If we have too many features, the learned hypothesis may fit the the training set very well ($J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 \approx 0$), but fail to generalize to new examples (predict prices on new examples).

Similar thing can happen with logistic regression.

Addressing Overfitting:

Options:

1) Reduce number of features
   - → Manually select which features to keep
   - → Model selection algorithm

2) Regularization
   - → Keep all the features, but reduce the magnitude/values of parameters θ_j.
   - → Works well when we have a lot of features, each of which contributes a bit to predicting y.

L2 - Cost Function

Regularization.

Small values for parameters $\theta_0, \theta_1, \ldots, \theta_n$
- Simpler hypothesis
- less prone to overfitting

Ex. Housing:
- Features: $x_1, x_2, \ldots, x_{100}$
- Parameters: $\theta_0, \theta_1, \theta_2, \ldots, \theta_{100}$

$$J(\theta) = \frac{1}{2m}\left[\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 + \underbrace{\lambda}_{\text{Regularization Parameter}} \underbrace{\sum_{j=1}^{n}\theta_j^2}_{\text{Regularization term}}\right]$$

L3- Regularized Linear Regression

Gradient Descent

Repeat {

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

$$\theta_j := \theta_j - \alpha \left[\frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)} + \underline{\frac{\lambda}{m} \theta_j} \right] \quad (j = \cancel{0}, 1, 2, \ldots, n)$$

}

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

$1 - \alpha \frac{\lambda}{m} < 1$, usually around 0.99 or something.

$\therefore \theta_j \cdot 0.99$

Normal Equation

$$X = \begin{bmatrix} (x^{(1)})^T \\ \vdots \\ (x^{(m)})^T \end{bmatrix} \qquad y = \begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(m)} \end{bmatrix} \mathbb{R}^m$$

$m \times (n+1)$

$\min_\theta J(\theta)$

$$\theta = \left(X^T X + \lambda \underbrace{\begin{bmatrix} 0 & & \text{zeros} \\ & 1 & \\ & & 1 \\ \text{zeros} & & & \ddots & 1 \end{bmatrix}}_{(n+1) \times (n+1)} \right)^{-1} X^T y$$

L4 - Regularized Logistic Regression

Cost Function:

$$J(\theta) = -\left[\frac{1}{m}\sum_{i=1}^{m} y^{(i)}\log h_\theta(x^{(i)}) + (1-y^{(i)})\log(1-h_\theta(x^{(i)}))\right] + \frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2$$

Gradient Descent

Repeat {

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

$$\theta_j := \theta_j - \alpha\underbrace{\left[\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)} + \frac{\lambda}{m}\theta_j\right]}_{\frac{d}{d\theta_j}J(\theta)}, \quad (j = \cancel{0},1,2,\dots,n)$$

}

Recall $h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}$