

SoS Endterm report

Jayanth Dosapati

June 2022

1 Regression

1.1 Introduction

Regression analysis is a predictive modelling technique. It estimates relationship between dependent variable(target) and independent variable(predictor).

1.2 Logistic regression

Logistic regression estimates the probability of an event occurring, such as voted or didn't vote, based on a given dataset of independent variables. Since the outcome is a probability, the dependent variable is bounded between 0 and 1. Logistic regression is actually an approach to classification problems, not regression problems.

1.3 Binary classification

Instead of our output vector y being a continuous range of values, it will only be 0 or 1.

$$y \in 0, 1$$

Where 0 is usually taken as "negative class" and 1 is taken as "positive class". For two class problem known as "Binary classification Problem". One method is to use linear regression and map all predictions greater than 0.5 as a 1 and all less than 0.5 as a 0. This method doesn't work well because classification is not actually a linear function. Our hypothesis should satisfy:

$$0 \leq h_{\theta}(x) \leq 1$$

We use sigmoid function also called logistic function to do this

$$h_{\theta}(x) = g(\theta^T x) \tag{1}$$

$$z = \theta^T x \tag{2}$$

$$g(z) = \frac{1}{1 + e^{-z}} \tag{3}$$

1.4 Decision boundary

In order to get our discrete 0 or 1 classification, we can translate the output of the hypothesis function as follows:

$$h_{\theta}(x) \geq 0.5 \rightarrow y = 1 \quad (4)$$

$$h_{\theta}(x) \leq 0.5 \rightarrow y = 0 \quad (5)$$

2 Evaluating Hypothesis

2.1 Methods for solving errors

- Getting more training examples
- Trying smaller sets of features
- Trying additional features
- Trying polynomial features
- Increasing or decreasing λ

We don't just pick one of these avenues at random. There are some diagnostic techniques for choosing one of above solutions.

A hypothesis may have low error for the training examples but still be inaccurate (because of overfitting).

With a given dataset of training examples, we can split up the data into two sets: a training set and a test set.

The new procedure using these two sets is then:

- Learn Θ and minimize $J_{train}(\Theta)$ using training set
- Compute the test set error $J_{test}(\Theta)$

2.2 The test set error

- For linear regression:

$$J_{test}(\Theta) = \frac{1}{2m_{test}} \sum_{i=1}^{m_{test}} (h_{\Theta}(x_{test}^{(i)}) - y_{test}^{(i)})^2 \quad (6)$$

- For classification \sim Misclassification error (aka 0/1 misclassification error):

$$err(h_{\Theta(x),y}) = \begin{cases} 1 & \text{if } h_{\Theta}(x) \geq 0.5 \text{ and } y = 0 \text{ or } h_{\Theta}(x) < 0.5 \text{ and } y = 1 \\ 0 & \text{otherwise} \end{cases} \quad (7)$$

This gives us a binary 0 or 1 error result based on a misclassification.
The average test error for the test set is

$$TestError = \frac{1}{m_{test}} \sum_{i=1}^{m_{test}} err(h_{\Theta}(x_{test}^{(i)}), y_{test}^{(i)})$$

(8)

This gives us the proportion of the test data that was misclassified.

3 Model Selection and Train/Validation/Test Sets

- Model Selection and Train/Validation/Test Sets
- The error of your hypothesis as measured on the data set with which you trained the parameters will be lower than any other data set.

In order to choose the model of your hypothesis, you can test each degree of polynomial and look at the error result.

3.0.1 Without the Validation Set (note: this is a bad method - do not use it)

- Optimize the parameters in Θ using the training set for each polynomial degree.
- Find the polynomial degree d with the least error using the test set.
- Estimate the generalization error also using the test set with $J_{test}(\Theta^{(d)})$, (d = theta from polynomial with lower error);

In this case, we have trained one variable, d , or the degree of the polynomial, using the test set. This will cause our error value to be greater for any other set of data.

3.0.2 Use of the CV set

To solve this, we can introduce a third set, the Cross Validation Set, to serve as an intermediate set that we can train d with. Then our test set will give us an accurate, non-optimistic error.

3.0.3 With the Validation Set (note: this method presumes we do not also use the CV set for regularization)

- Optimize the parameters in Θ using the training set for each polynomial degree.
- Find the polynomial degree d with the least error using the cross validation set.
- Estimate the generalization error using the test set with $J_{test}(\Theta^{(d)})$, (d = theta from polynomial with lower error);
This way, the degree of the polynomial d has not been trained using the test set.

4 Diagnosing Bias vs. Variance

In this section we examine the relationship between the degree of the polynomial d and the underfitting or overfitting of our hypothesis.

- We need to distinguish whether bias or variance is the problem contributing to bad predictions. bias is underfitting and high variance is overfitting. We need to find a golden mean between these two.

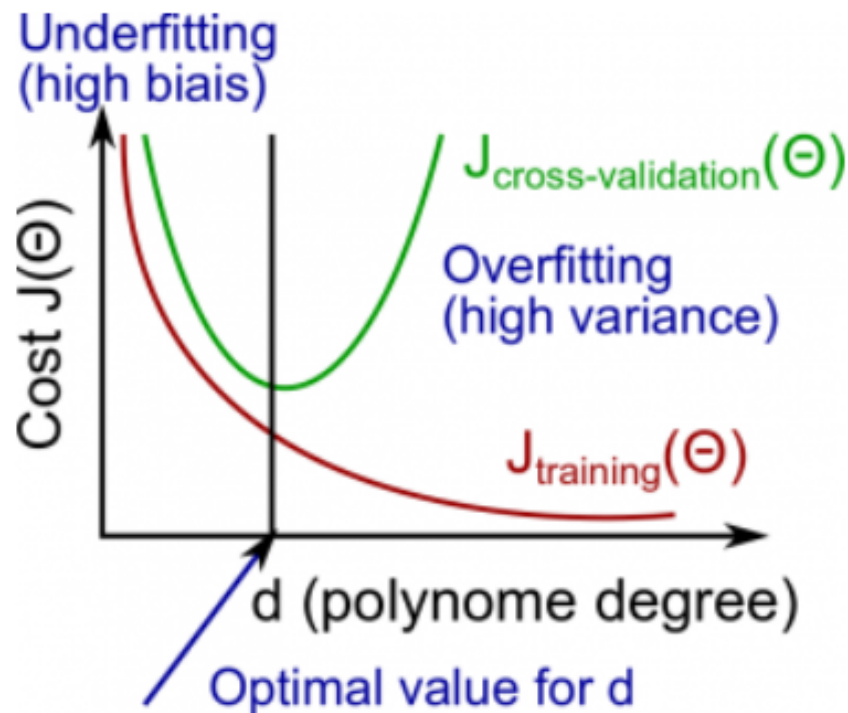
The training error will tend to decrease as we increase the degree d of the polynomial.

At the same time, the cross validation error will tend to decrease as we increased up to a point, and then it will increase as d is increased, forming a convex curve.

High bias (underfitting): both $J_{train}(\Theta)$ and $J_{CV}(\Theta)$ will be high. Also, $J_{CV}(\Theta) \approx J_{train}(\Theta)$

High variance (overfitting): $J_{train}(\Theta)$ will be low and $J_{CV}(\Theta)$ will be much larger than $J_{train}(\Theta)$

This is represented in following figure.



5 Regularization and Bias/Variance

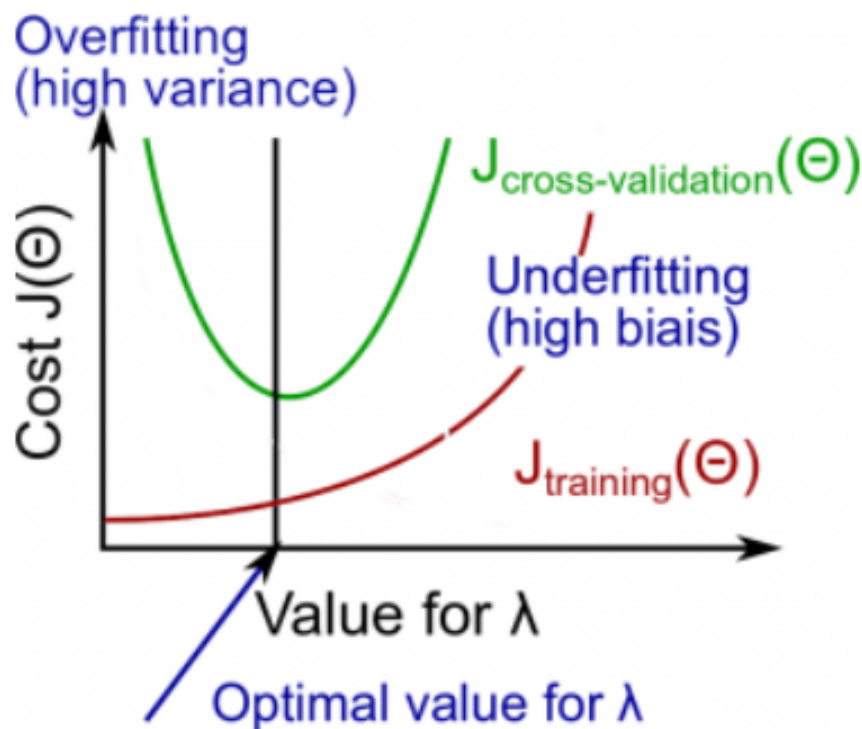
Instead of looking at the degree d contributing to bias/variance, now we will look at regularization parameter λ .

- Large λ : High bias (underfitting)
- Intermediate λ : just right
- Small λ : High variance (overfitting)

A large lambda heavily penalizes all the parameters, which greatly simplifies the line of our resulting function, so causes underfitting.

The relationship of λ to the training set and the variance set is as follows: **Low λ** : $J_{train}(\Theta)$ is low and $J_{cv}(\Theta)$ is high (high variance/overfitting). **Intermediate λ** : $J_{train}(\Theta)$ and $J_{cv}(\Theta)$ are somewhat low and are $J_{train}(\Theta) \approx J_{cv}(\Theta)$ **large λ** : both $J_{cv}(\Theta)$ and $J_{train}(\Theta)$ will be high (underfitting /high bias)

The figure below illustrates the relationship between lambda and the hypothesis:



In order to choose the model and the regularization λ , we need:

- Create a list of lambdas (i.e. $\lambda \in 0, 0.01, 0.02, 0.04, 0.08, 0.16, 0.32, 0.64, 1.28, 2.56, 5.12, 10.24$);

- Create a set of models with different degrees or any other variants.
- Iterate through the λ s and for each λ go through all the models to learn some Θ .
- Compute the cross validation error using the learned Θ (computed with λ) on the $J_{cv}(\Theta)$ without regularization or $\lambda=0$.
- Select the best combo that produces the lowest error on the cross validation set.
- Using the best combo of Θ and λ apply it on $J_{test}(\Theta)$ to see if it has good generalization problem.

6 Optimization Objective

The Support Vector Machine (SVM) is yet another type of supervised machine learning algorithm. It is sometimes cleaner and more powerful. Recall that in logistic regression, we use the following rules:

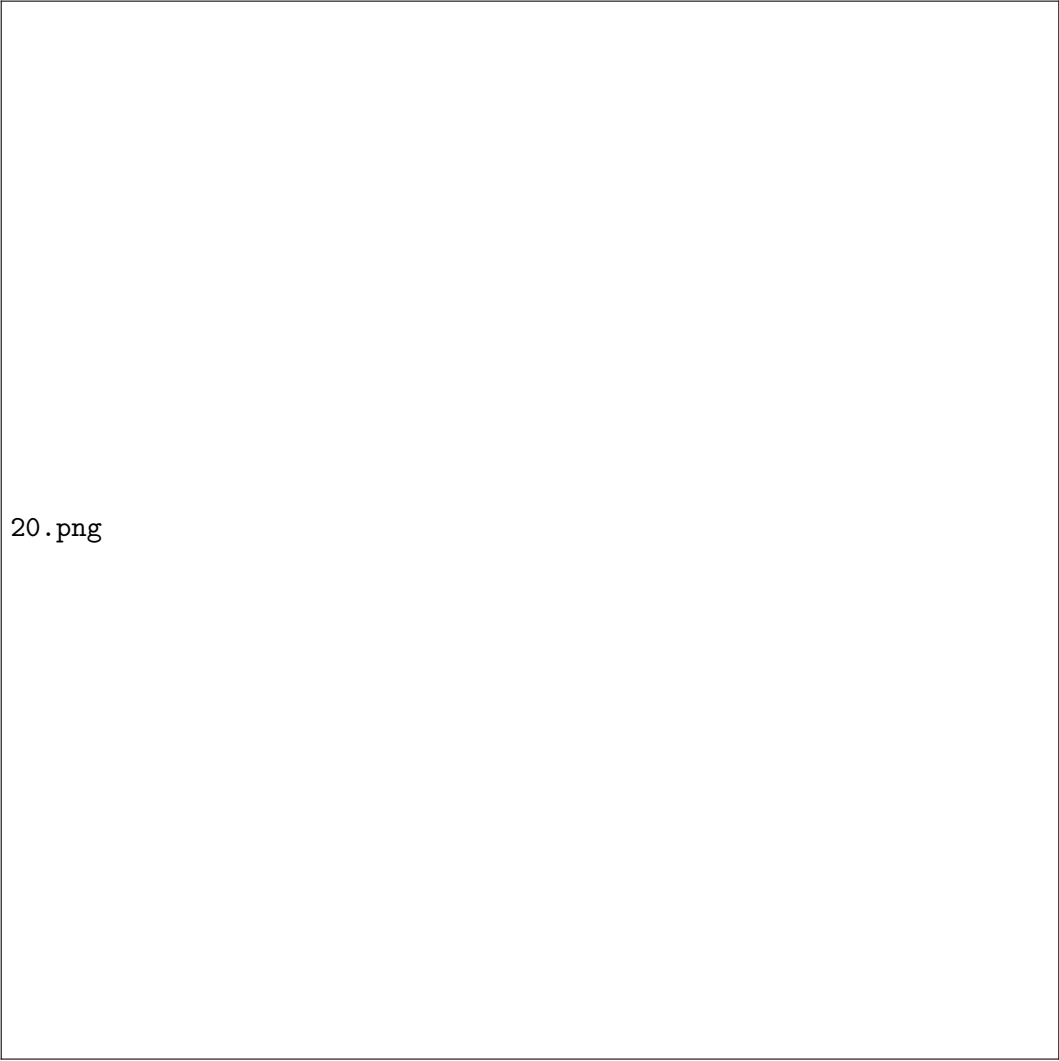
if $y=1$, then $h_{\theta}(x) \approx 1$ and $\Theta^T x \gg 0$ if $y=0$, then $h_{\theta}(x) \approx 0$ and $\Theta^T x \ll 0$ Recall the cost function for (unregularized) logistic regression:

$$J(\Theta) = \frac{1}{m} \sum_{i=1}^m -y^i \log(h_{\theta}(x^{(i)})) - (1 - y^i) \log(1 - h_{\theta}(x^{(i)}))$$

(9)

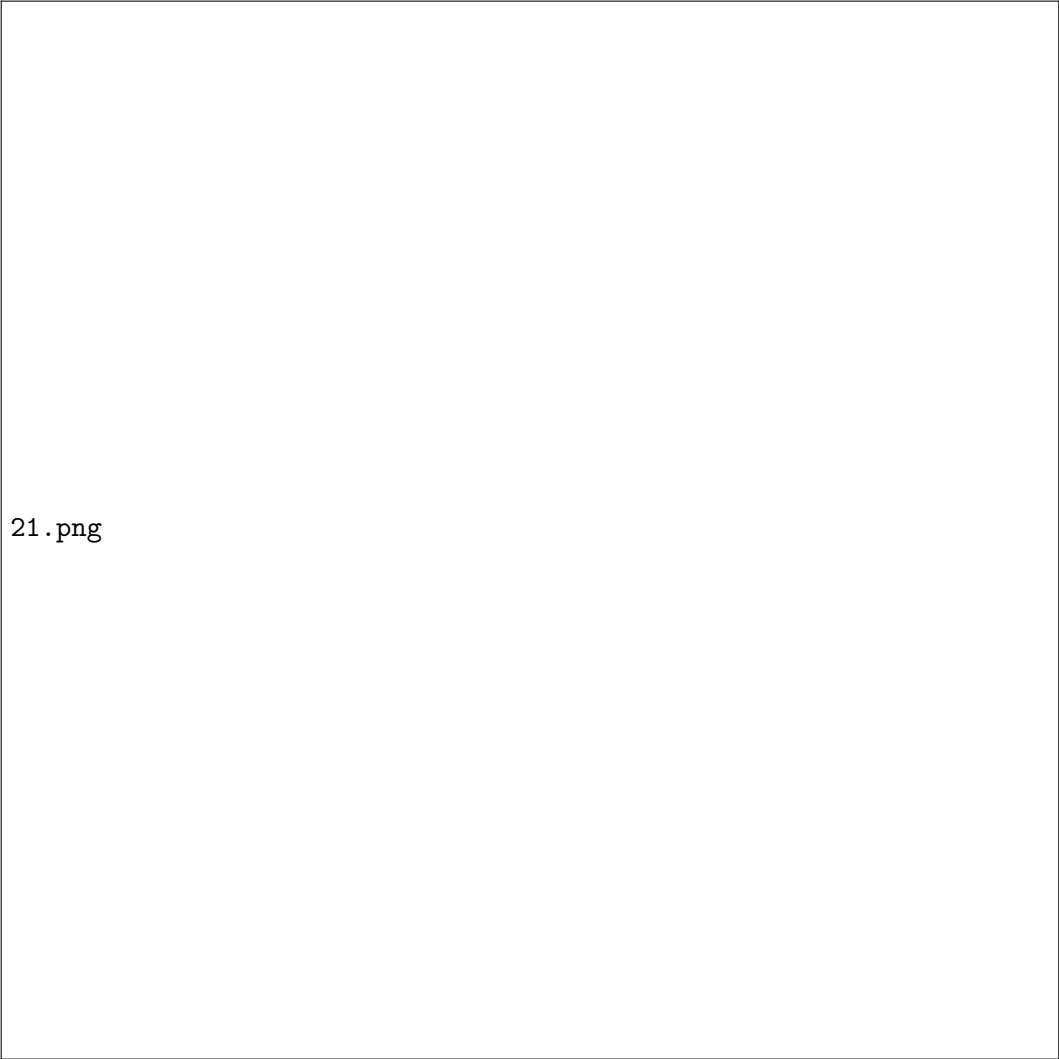
$$= \frac{1}{m} \sum_{i=1}^m -y^{(i)} \log\left(\frac{1}{1 + e^{-\Theta^T x^{(i)}}}\right) - (1 - y^{(i)}) \log\left(1 - \frac{1}{1 + e^{-\Theta^T x^{(i)}}}\right) \quad (10)$$

To make a support vector machine, we will modify the first term of the cost function $-\log(h_{\theta}(x)) = -\log\left(\frac{1}{1 + e^{-\Theta^T x^{(i)}}}\right)$ so that when $\Theta^T x$ (from now on, we shall refer to this as z) is greater than 1, it outputs 0. Furthermore, for values of z less than 1, we shall use a straight decreasing line instead of the sigmoid curve. (In the literature, this is called a hinge loss function.



20.png

Similarly, we modify the second term of the cost function $-\log(1 - h_{\theta(x)}) = -\log\left(1 - \frac{1}{1 + e^{-\theta^T x}}\right)$ so that when z is less than -1 , it outputs 0 . We also modify it so that for values of z greater than -1 , we use a straight increasing line instead of the sigmoid curve.



21.png

7 Unsupervised Learning: Introduction

Unsupervised learning is contrasted from supervised learning because it uses an unlabeled training set rather than a labeled one.

In other words, we don't have the vector y of expected results, we only have a dataset of features where we can find structure.

Clustering is good for:

- Market segmentation
- Social network analysis
- Organizing computer clusters
- Astronomical data analysis

8 K-Means Algorithm

The K-Means Algorithm is the most popular and widely used algorithm for automatically grouping data into coherent subsets.

- Randomly initialize two points in the dataset called the cluster centroids.
- Cluster assignment: assign all examples into one of two groups based on which cluster centroid the example is closest to.
- Move centroid: compute the averages for all the points inside each of the two cluster centroid groups, then move the cluster centroid points to those averages.
- Re-run (2) and (3) until we have found our clusters.

Our main variables are:

- K (number of clusters)
- Training set $x^{(1)}, x^{(2)}, \dots, x^{(m)}$
- Where $x^{(i)} \in \mathbb{R}^n$

8.1 The algorithm:

Randomly initialize K cluster centroids $\mu(1), \mu(2), \dots, \mu(K)$ Repeat: for $i = 1$ to m : $c(i) := \text{index (from 1 to K) of cluster centroid closest to } x(i)$ for $k = 1$ to K : $\mu(k) := \text{average (mean) of points assigned to cluster } k$

9 ML:Anomaly Detection

9.1 Problem Motivation

Just like in other learning problems, we are given a dataset $x^{(1)}, x^{(2)}, \dots, x^{(m)}$ We are then given a new example, x_{test} , and we want to know whether this new example is abnormal/anomalous.

We define a "model" $p(x)$ that tells us the probability the example is not anomalous. We also use a threshold ϵ (epsilon) as a dividing line so we can say which examples are anomalous and which are not.

A very common application of anomaly detection is detecting fraud:

- $x^{(i)}$ = features of user i 's activities.
- Model $p(x)$ from the data.
- Identify unusual users by checking which have $p(x) < \epsilon$

If our anomaly detector is flagging too many anomalous examples, then we need to decrease our threshold ϵ

10 Gaussian Distribution

The Gaussian Distribution is a familiar bell-shaped curve that can be described by a function $\mathcal{N}(\mu, \sigma^2)$. Let $x \in \mathbb{R}$. If the probability distribution of x is Gaussian with mean μ , variance σ^2 then :

$x \sim \mathcal{N}(\mu, \sigma^2)$ The little 'or tilde' can be read as "distributed as."

The Gaussian Distribution is parameterized by a mean and a variance.

μ , or μ , describes the center of the curve, called the mean. The width of the curve is described by sigma, or σ , called the standard deviation.

The full function is as follows:

$$p(x; \mu, \sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} \quad (11)$$

We can estimate the parameter μ from a given dataset by simply taking the average of all the examples:

$$\mu = \frac{1}{m} \sum_{i=1}^m x^{(i)}$$

We can estimate the other parameter, σ^2 with our familiar squared error formula:

$$\sigma^2 = \frac{1}{m} \sum_{i=1}^m (x^{(i)} - \mu)^2$$

11 Learning with Large Datasets

We mainly benefit from a very large dataset when our algorithm has high variance when m is small. Recall that if our algorithm has high bias, more data will not have any benefit.

Datasets can often approach such sizes as $m = 100,000,000$. In this case, our gradient descent step will have to make a summation over all one hundred million examples. We will want to try to avoid this – the approaches for doing so are described below.

12 Stochastic Gradient Descent

Stochastic gradient descent is an alternative to classic (or batch) gradient descent and is more efficient and scalable to large data sets.

Stochastic gradient descent is written out in a different but similar way:

$cost(\theta, (x^{(i)}, y^{(i)})) = \frac{1}{2}(h_{\theta}(x^{(i)}) - y^{(i)})^2$ The only difference in the above cost function is the elimination of the m constant within $\frac{1}{2}$

$$J_{train}(\theta) = \frac{1}{m} \sum_{i=1}^m cost(\theta, (x^{(i)}, y^{(i)}))$$

$J_{train}(\theta)$ is now just the average of cost applied to all of our training examples.

This algorithm is as follows

1. Randomly 'shuffle' the dataset
2. For $i = 1 \dots m$

$\Theta_j := \Theta_j - \alpha(h_{\theta}(x^{(i)}) - y^{(i)})x_j^{(i)}$ This algorithm will only try to fit one training example at a time. This way we can make progress in gradient descent without having to scan

all m training examples first. Stochastic gradient descent will be unlikely to converge at the global minimum and will instead wander around it randomly, but usually yields a result that is close enough. Stochastic gradient descent will usually take 1-10 passes through your data set to get near the global minimum.