Report of MOOC

Machine Learning

Guangyue CHEN

2018/12/12

Contents

1 Introduction

Machine learning is the science of getting computers to act without being explicitly programmed. It involves many machine learning algorithms, which is a kind of automatic analysis from the data to obtain the law, and use the law of the algorithm to predict the unknown data. Because the learning algorithm involves a lot of statistical theory, machine learning and inferred statistics are particularly close, also known as statistical learning theory.

1.1 Machine Learning - Coursera

This Stanford online courses is taught by Andrew Ng who is one of the mostly influencing professors in Machine Learning. In this class, l learned about not only the theoretical underpinnings of learning, but also gain the practical experience to quickly and powerfully apply these techniques to new problems.

2 Course content

The course has a total of 11 weeks. Each week, it introduces us one or two common machine learning models or principle algorithms, which also involves some derivations of mathematical formula. It has the test every weak. What's more, it has also one project per week to practice the knowledge that we learned. In general, it is divided into the following sections.

2.1 Supervised learning

In supervised learning, we are given a data set and we already know what our correct output should look like, we should know that there is a relationship between the input and the output.

2.1.1 Linear Regression

In the linear regression model, through the feature extraction, we first select the characteristic variable x which affects the estimated variable y, and then through our training set and learning algorithm, we can get a hypothesis h. Here h can be expressed as:

$$h_\theta(x) = \theta_0 + \theta_1 * x_1 + \theta_2 * x_2 + ...$$

Here We measure the accuracy of our hypothesis function by using a cost function $J(\theta)$.

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

Then we choose the parameters who minimize the cost function. the main method optimized we used is **Gradient Descent** which is: repeat until convergence

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

where m is the number of the observations and α is the learning rate which we should choose by ourselves.

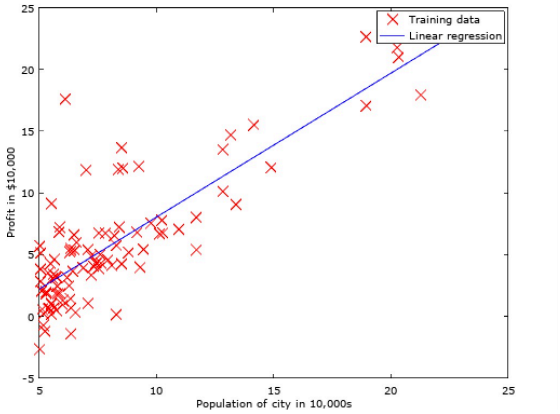


Figure1:The result of Linear Regression

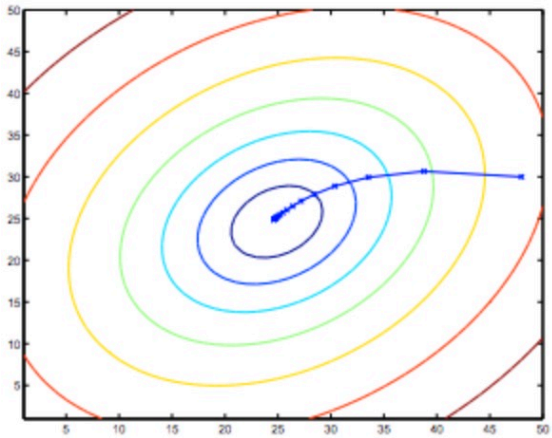


Figure2:The converge of Gradient Descent

Here we have one coding practice, after write the algorithm of Gradient Descent, we find that with the converge of Gradient Descent,whith is shown in figure 2, the model fit our data better.(Figure 1) **All code here.**

2.1.2 Logistic Regression

The classification problem is just like the regression problem, except that the values we now want to predict take on only a small number of discrete values. And here one inportant thing is to dicide the **decision boundary**.

$$h_\theta(x) \geq 0.5, y = 1.$$

$$h_\theta(x) < 0.5, y = 0.$$

Furthermore, with vectoration, we get:

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

$$when : \theta^T x \geq 0$$

We cannot use the same cost function that we use for linear regression because the Logistic Function will cause the output to be wavy, it might cause many local optima. In other words, it will not be a convex function.

Considering y equals 0 or 1, our cost function for logistic regression looks like:

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

Here we use the octave's "fminunc()" optimization algorithm along with the "optimset()" function that creates an object containing the options we want to send to "fminunc()".

```
% Set options for fminunc:
options = optimset('GradObj', 'on', 'MaxIter', 400);

% Run fminunc to obtain the optimal theta
% This function will return theta and the cost
[theta, cost] = ...
        fminunc(@(t)(costFunction(t, X, y)), initial_theta, options);
```

Le code completet: Logistic.

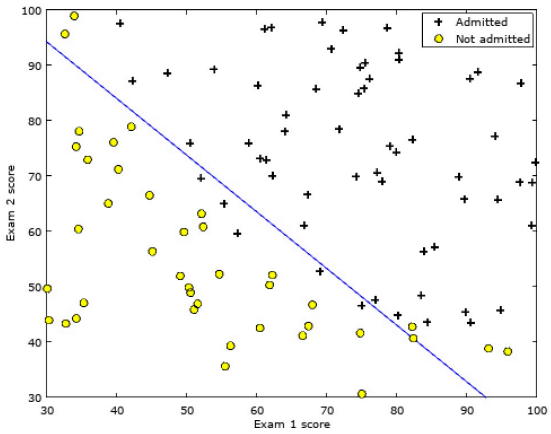


Figure3:The result of Logistic Regression

2.1.3 Regularization

Considering the problem of Overfitting, we add the regularization. So we could aregularize all of our theta parameters in a single summation as

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

The λ is the regularization parameter. It determines how much the costs of our theta parameters are inflated. Regularization is used in many projects of this course.

2.1.4 Multi-class Classification and Neural Networks

To classify data into multiple classes, we let our hypothesis function return a vector of values. Here we use the model **"Neural Networks"** the method **"Backpropagation"**.

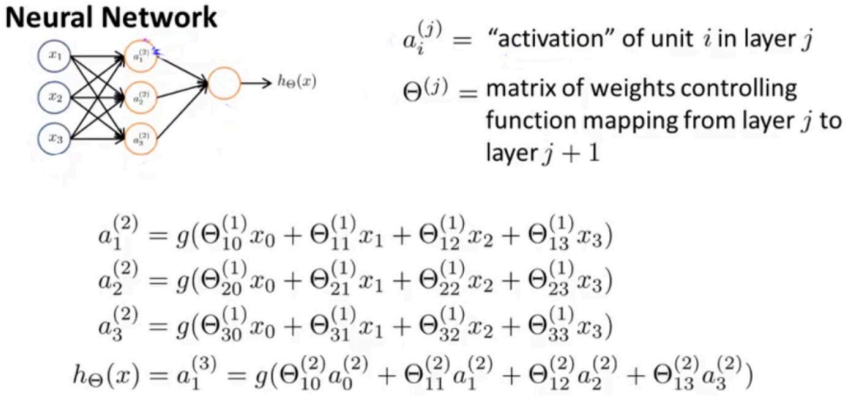


Figure4:Neural Networks Model

In the **Neural Networks** model, there are 3 types of layer: input layer, hidden layer, output layer. Each layer content n nodes. After the modelization, we can compute our activation nodes by using a matrix of parameters. We apply each row of the parameters to our inputs to obtain the value for one activation node.

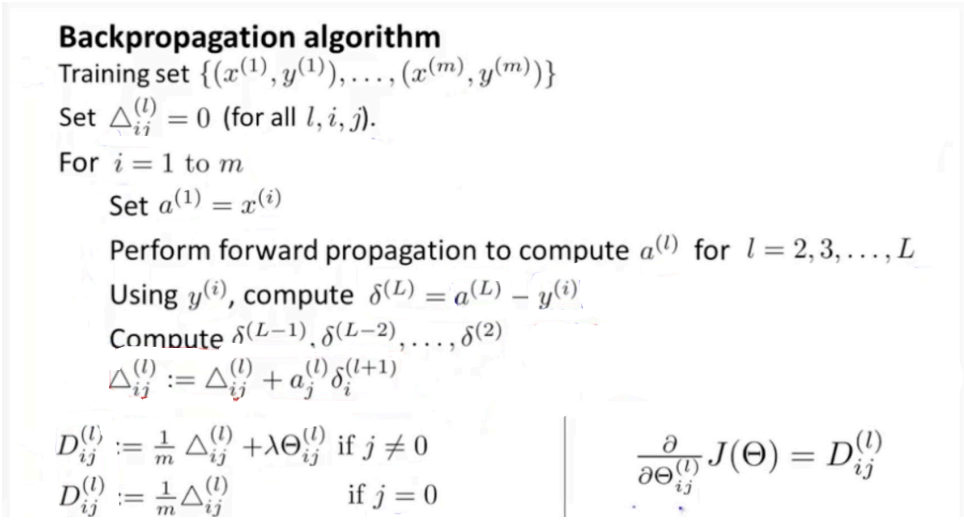


Figure5:Backpropagation Algorithm

"Backpropagation" is neural-network terminology for minimizing our cost function, just like what we were doing with gradient descent in logistic and linear regression.

Then we do the most important project of this course, we implement the backpropagation algorithm for neural networks and apply it to the task of hand-written digit recognition.

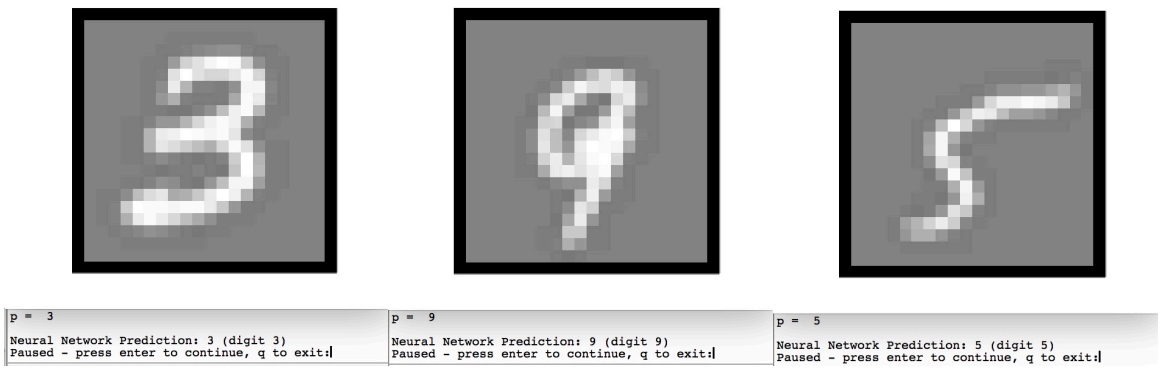


Figure6:The result

All code here.

2.1.5 Support Vector Machines

There's one more algorithm that is very powerful and is very widely used both within industry and academia, and that's called the support vector machine. SVM gives a cleaner, and sometimes more powerful way of learning complex non-linear functions.

$$min_\theta C \sum_{i=1}^{m} [y^{(i)} cost_1(\theta^T x^{(i)}) + (1 - y^{(i)}) cost_0(\theta^T x^{(i)})] + \frac{1}{2} \sum_{j=1}^{n} \theta_j^2$$

$$where : cost_1(\theta^T x^{(i)}) = -log h_\theta(x^{(i)})$$
$$cost_0(\theta^T x^{(i)}) = -log(1 - h_\theta(x^{(i)}))$$

Here we use a higher standard conditions:If y = 1, we want $\theta^T x \geq 1$, and if y = 0, we want $\theta^T x \leq -1$.

It will give us a large **Decision Boundry**. Here we have a project too. All code here.

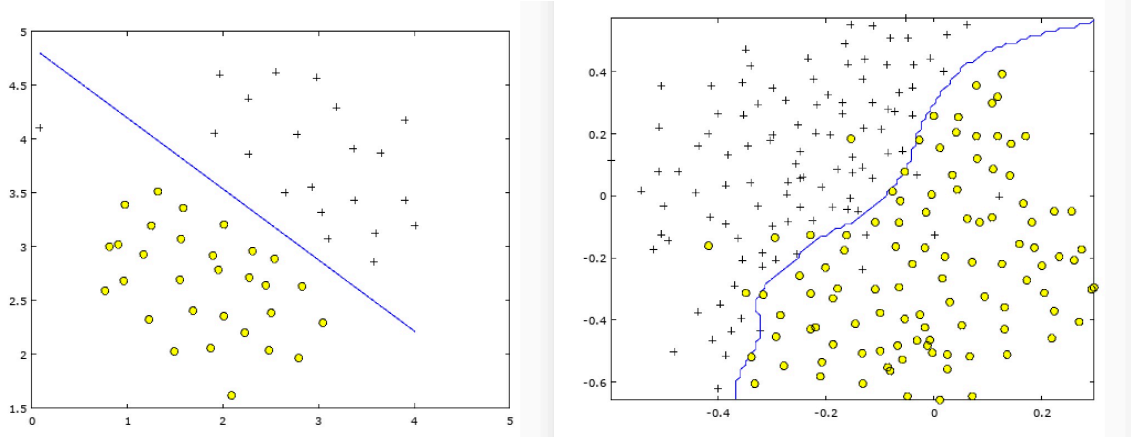


Figure7:Decision Boundry made by SVM

So we can see the higher standard conditions make the SVM get the largest Margin's decision boundary.

2.2 Unsupervised learning

Unsupervised learning allows us to approach problems with little or no idea what our results should look like. We can derive structure from data where we don' t necessarily know the e ect of the variables.

2.2.1 Principal Component Analysis

PCA is an unsupervised linear dimensionality reduction algorithm , which is used to find a more meaningful basis or coordinate system

for our data and works. It based on covariance matrix to find the strongest features in the samples. So what PCA formally does is that it tries to find a lower dimensional surface , onto which we can project the data with the minimize sum of squares of variance.

We have learnt many knowledge about PCA, PCoA and KPCA from the course 'MAD', which teach us more things. But this course spend more at practice. So I gain that PCA can help us to Reduce memory/disk needed to store data. And speed up learning algorithm.

2.2.2 K-Means Clustering

K-means clustering is a type of unsupervised learning, which is used when we have unlabeled data. The goal of this algorithm is to find groups in the data, with the number of groups represented by the variable K. The algorithm works iteratively to assign each data point to one of K groups based on the distances between the data points and the group centers.

K-means algorithm

Randomly initialize K cluster centroids $\mu_1, \mu_2, ..., \mu_k$
Repeat{
for i = 1 to m
$c^{(i)}$:= index(from 1 to K) of cluster centroid closest to $x^{(i)}$
for k = 1 to m
μ_k:= average(mean) of points assigned to cluster k
}
Project:

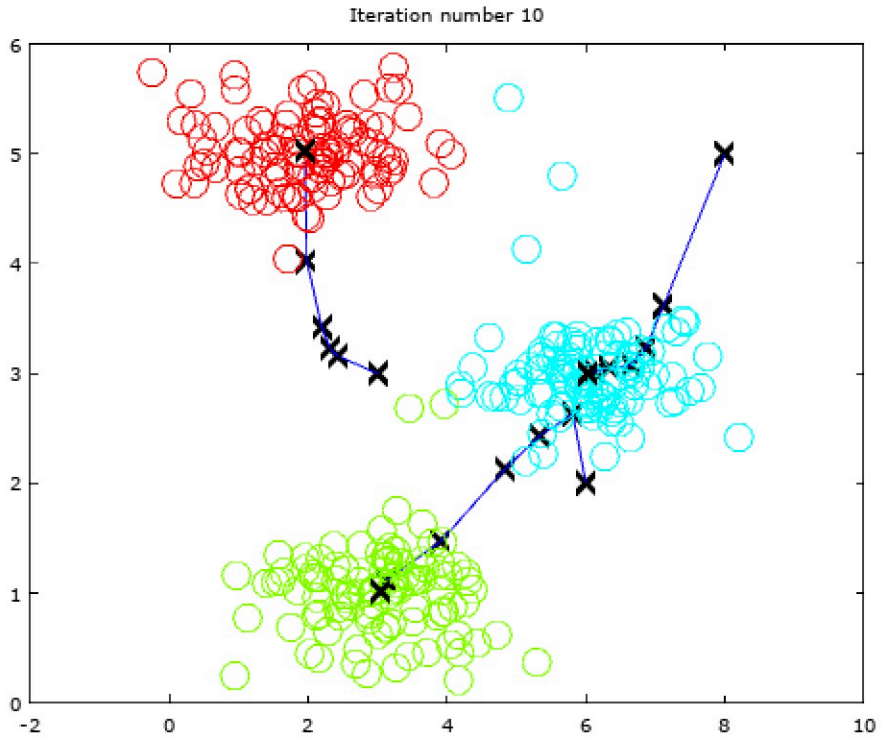


Figure8:Iteration of K-means

So we can see with the ieration, the centroids is closer to the center of real class. All code here.

2.3 Best practices in machine learning

2.3.1 Anomaly Detection

For this part, the teacher modelize the real problem into the mixture model. With eht estimation of the distribution, we can evidently know which object or which user is anomal.

Anomaly Detection Method

1. Choose features x_i which might be indicative of anomalous examples.
2. Fit parameters $\mu_1, \mu_2, ..., \mu_n, \sigma_1, \sigma 2, ..., \sigma_n$

$$\mu_j = \frac{1}{m}\sum_{i=1}^{m} x_j^{(i)}$$

$$\sigma_j^2 = \frac{1}{m}\sum_{i=1}^{m}(x_j^{(i)} - \mu_j)^2$$

3. Given new example x, compute $p(x)$:

$$p(x) = \Pi_{j=1}^{n} p(x_j; \mu_j; \sigma_j^2) = \Pi_{j=1}^{n} \frac{1}{\sqrt{2\pi}\sigma_j} exp(1\frac{(x_j - \mu_j)^2}{2\sigma_j^2})$$

4. Anomaly if p(x) $< \varepsilon$.

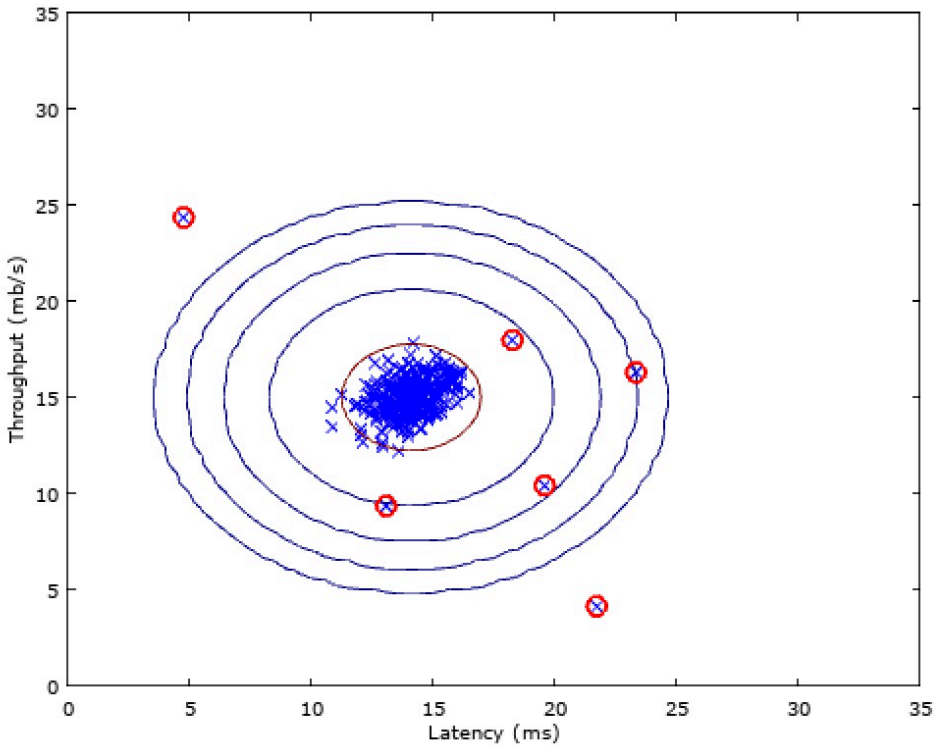


Figure8:Anomaly Detection

For this project, at first we visualize the dataset, and then we estimate a Gaussian model . Then we implement an algorithm to select the threshold using the F1 score on a cross validation set. So we can find the anomal data in the end. L'algorithm for select the threshold . All code here.

2.3.2 Recommender Systems

The recommender systems is an important application of supervised machine learning. So we can recommender users the movies or the novels after they rate the same kinds movies or novels. Here the main conception is **Collaborative Filtering**. The system learn better features and then these features can be used by the system to make better movie predictions for everyone else. And also every user is helping the system learn better features for the common good when they rate new movies.

2.3.3 Application: Photo OCR

This part is just the introduction of one popular machine learning technology: Photo Optical Character Recognition, or OCR. It can convert the photo into editable data. We can use the **Sliding windows** to do the **Text detection** or **Pedestrian detection**.

2.4 Tool and Language

Octave

The Octave syntax is largely compatible with Matlab. The Octave interpreter can be run in GUI mode, as a console, or invoked as part of a shell script. Here all the projects are written in Octave.

3 Conclusion

After a period of study, I learned about the most effective machine learning techniques, and gain practice implementing them and getting them to work for myself. It helps me to gain deeper and more various machine learning models and algorithms which is useful to be a data scientist. To conclusion, I benefit a lot from this course.

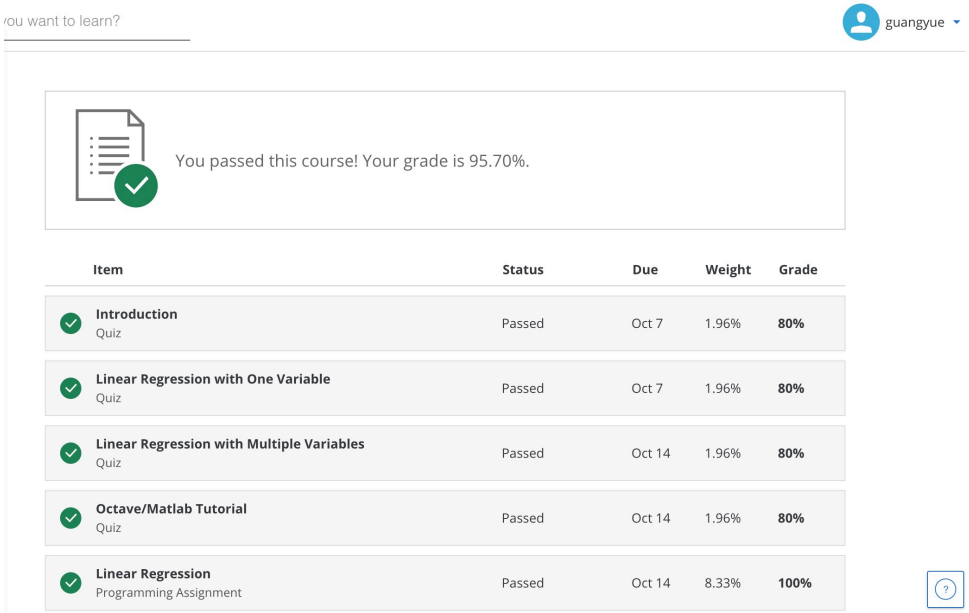
4 Appendix

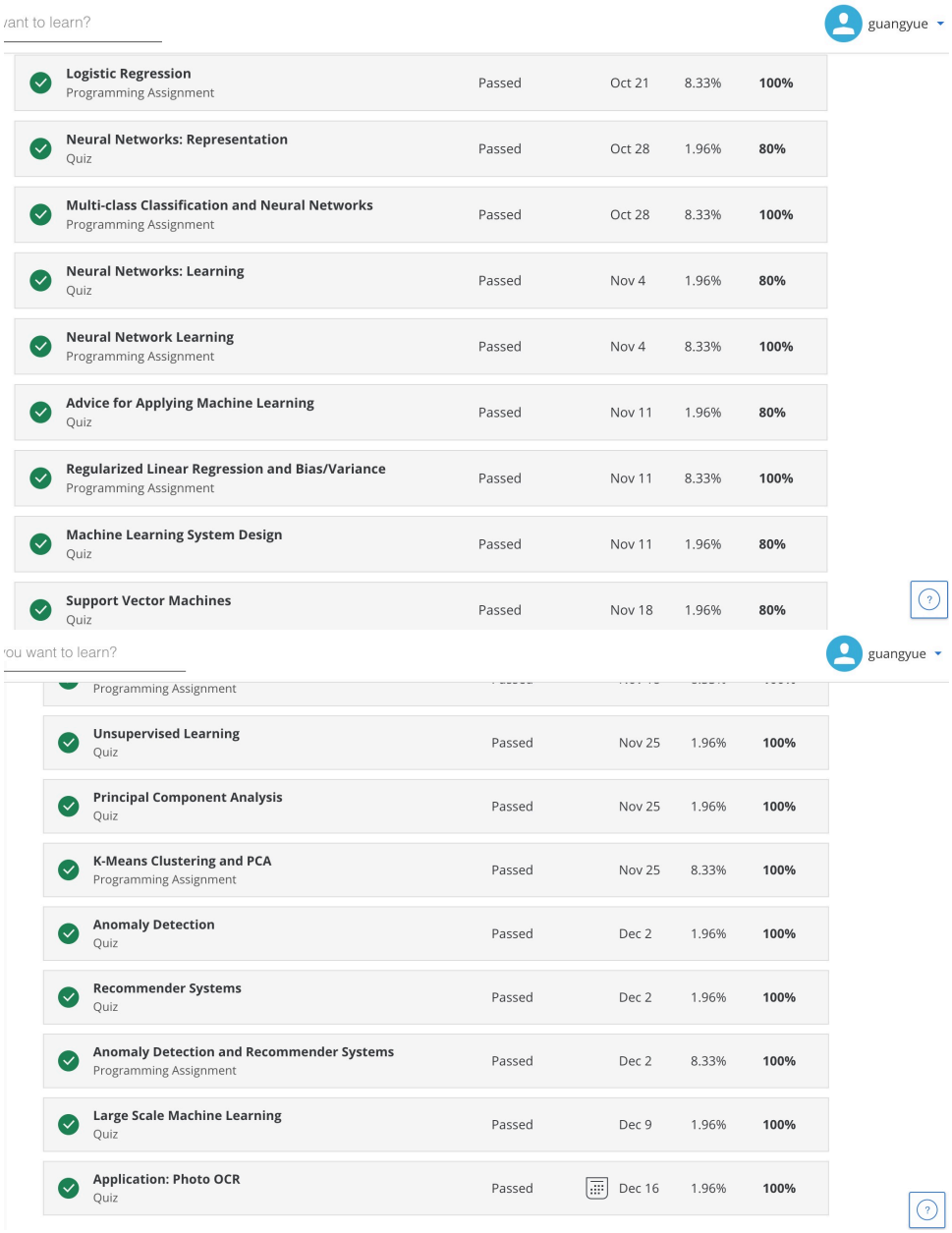
4.1 Course and Github

Course:https://www.coursera.org/learn/machine-learning/
Github(all 7 projects):https://github.com/GuangYueCHEN/ENSIIE/MachineLearn

4.2 progress

After 11 weeks studying, I passed this course. My final resault is 96/100.




ant to learn?
guangyue
Logistic Regression
Programming Assignment
Passed
Oct 21
8.33%
100%
Neural Networks: Representation
Quiz
Passed
Oct 28
1.96%
80%
Multi-class Classification and Neural Networks
Programming Assignment
Passed
Oct 28
8.33%
100%
Neural Networks: Learning
Quiz
Passed
Nov 4
1.96%
80%
Neural Network Learning
Programming Assignment
Passed
Nov 4
8.33%
100%
Advice for Applying Machine Learning
Quiz
Passed
Nov 11
1.96%
80%
Regularized Linear Regression and Bias/Variance
Programming Assignment
Passed
Nov 11
8.33%
100%
Machine Learning System Design
Quiz
Passed
Nov 11
1.96%
80%
Support Vector Machines
Quiz
Passed
Nov 18
1.96%
80%
ou want to learn?
guangyue
Programming Assignment
Unsupervised Learning
Quiz
Passed
Nov 25
1.96%
100%
Principal Component Analysis
Quiz
Passed
Nov 25
1.96%
100%
K-Means Clustering and PCA
Programming Assignment
Passed
Nov 25
8.33%
100%
Anomaly Detection
Quiz
Passed
Dec 2
1.96%
100%
Recommender Systems
Quiz
Passed
Dec 2
1.96%
100%
Anomaly Detection and Recommender Systems
Programming Assignment
Passed
Dec 2
8.33%
100%
Large Scale Machine Learning
Quiz
Passed
Dec 9
1.96%
100%
Application: Photo OCR
Quiz
Passed
Dec 16
1.96%
100%