ENSIIE

Rapport DE MOOC

Machine Learning Foundations : A Case Study Approach

Kai HUANG
09 Mai 2017

Mars 2017 — Avril 2017

Contents

1 Introduction

With the development of technology, more and more websites and applications are created, at the same time, tremendous data is being created by millions of people. For examples: the number of clicks on the specific website, clients' comments on the product, images, etc… There is a fact that we must realize which is that we are going to live in a data era. People who are able to collect data, and get valuable information from the data will probably be the next giant.

Machine learning is such a technology that helps us know what data can tell us. If we observe carefully in our life, it's easy to find the application of machine learning everywhere. If you have bought one product on Amazon, Amazon will recommend some products that you might want to buy; when you read some articles on the browser, it will show you some articles related to the article you have read the next time you come to the browser.

This course treats the machine learning method as a black box. In this course, we get hands-on experience with machine learning from a series of practical case-studies. Most of time, we focus on understanding tasks of interest, matching these tasks to machine learning tools, and assessing the quality of the output(model).

The following is the objective at the end of this course:

- Identify potential applications of machine learning in practice.
- Describe the core differences in analyses enabled by regression, classification, and clustering.
- Select the appropriate machine learning task for a potential application.
- Apply regression, classification, clustering, retrieval, recommender systems, and deep learning.
- Represent your data as features to serve as input to machine learning models.
- Assess the model quality in terms of relevant error metrics for each task.
- Utilize a dataset to fit a model to analyse new data.
- Implement these techniques in Python.

1.1 Syllabus

The first week is about the introduction of this course – "Machine learning foundations – A case study". Instructors told us the power of machine learning, and the multitude of intelligent applications. We learned the regression with a real case – predicting the house at the second week. In addition, the classification is introduced at the third week. With the help of classification, we learned how to analyse sentiment, we need to know the sentiment of the client is positive or negative from his or her comment. In the third case study (the fourth week), retrieving documents, we examine various document representations and an algorithm to retrieve the most similar document. We also consider structured representations of the documents that automatically group articles by similarity (for example: the topic of the document). The fifth week talked about the recommending products with the technology – collaborative filtering. At the last week, the neural network is introduced to us, and we used that to search images.

2 Development

2.1 Regression: Predicting house prices.

In real life, many people sell their houses. Then the question comes: how to estimate prices of houses? He may look at recent sales in his neighbourhood, that's a good solution. In our case, we decide to build an intelligent application that make predictions from data. Firstly, we would like to build a model which has only one feature (the square feet); later, we'll add some other features in order to better predicting house prices.

We collected some information about the house trade. So we have various variables in the dataset. For one sample, we have id, date, price, number of bedrooms, number of bathrooms, floors, waterfront, year built, etc…

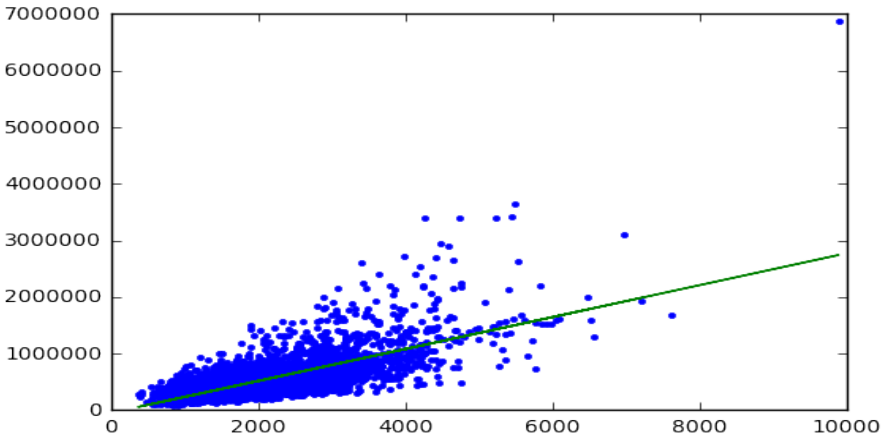
At first, we tried to visualize the data (x: square feet, y: house price). From the graph that we plotted, we guess that the model might be a linear regression model. So we fit a line through the data, so we have a function parameterized by w (w0, w1), and the function is f(x) = w0 + w1 * x. Given a line, how do we know if this is a good line? In other words, how can we find the best line? We created another function denoted RSS (Residual sum of squares):

$$RSS(w_0, w_1) = \sum_{i=1}^{n} (y_i - (w_0 + w_1 * x_1))^2$$

So the best line is the one that minimize the RSS function (called cost) over all possible w0, w1.

Here is the result:

$$f(x) = -47114.02 + 281.96 * x$$



Linear Regression

Then, we tried to fit the data with a quadratic function:

$f(x) = w0 + w1 * x + w2 * x^2$.

By using this function, with an appropriate parameter w, we found out that the cost is smaller compared to the linear regression model. And we also higher order polynomial functions, the cost is even smaller than the one calculated in the quadratic function. So, this leads us to the notion of overfitting. It tells us that it's not that the smaller the cost is, the better the model is.

After building the model with only one feature, we found that the RMSE (Root Mean Squared Error) is still too large – $255170. So we tried to add some other features, e.g : number of bedroom, number of bathroom, floors, zipcode. We used the linear regression again with several features, and the RMSE goes down from $255170 to $179508, which means we had a great improvement of the model.

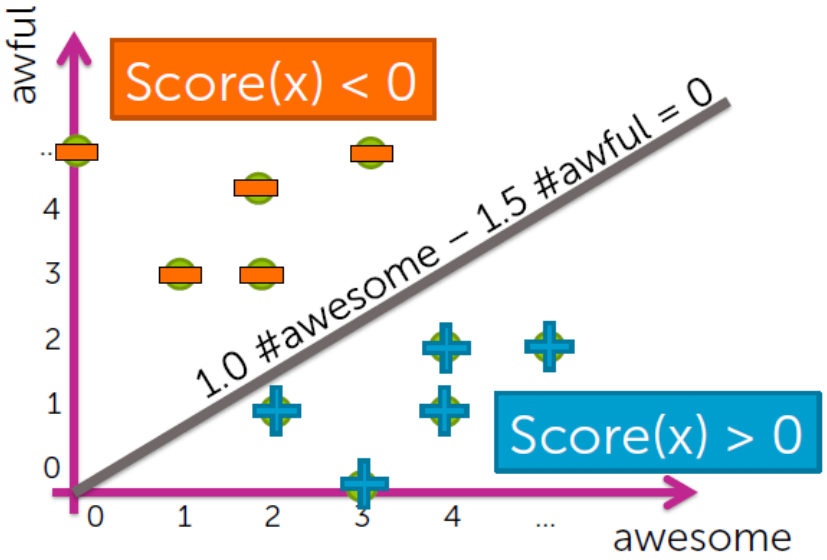
2.2 Classification: Analysing sentiment

How do we guess whether a person felt positively or negatively about an experience, just from a show comment they wrote? In our second case study, analysing sentiment, we focus on predicting a class (positive/negative) from input features (text of view). This task is an example of classification, one of the most widely used areas of machine learning, with a broad array of applications, including advertisement targeting, spam detection, medical diagnosis and image classification.

We started from building a linear classifier. Given a dataset which contains clients' views about products on Amazon, we, firstly, utilize the word count function to count words in every view. We defined that views with rating greater than 3 are positive and

views with rating smaller than 3 are negative. Then, we created a linear classifier with logistic regression.

Theoretically, after counting words in every view, we need to give every word a weight (eg: awesome is more positive than good, awful is more negative than bad). Then, we are supposed to find a decision boundary which helps us decide that views on the one side of the decision boundary are positive, yet views on the other side of the decision of boundary are negative (Like the following graph).



Decision Boundary

For linear classifiers, when there are 2 weights, the decision boundary is a line; when there are 3 weights, the decision boundary is a plane; when there are many weights, the decision boundary is a hyperplane.

After building a classifier, we need to evaluate the model. So we have

error = (# of mistakes) / (total # of sentences)

accuracy = (# of corrects) / (total # of sentences)

And we need to decide what accuracy our application needs, we also need to know the impact of mistakes we make (false positives, false negatives).

2.3 Clustering and similarity: Retrieving documents

A reader is interested in a specific news article and we want to find a similar article to recommend. What is the right notion of similarity? How do I automatically search

over documents to find the one that is most similar? This third case study aims at answering these questions.

In this case study, we study on the introduction about famous people from wikipedia. As we did in the last case, we, firstly, utilized word count function to count words. Then we used TF-IDF (Term frequency – inverse document frequency) to do document representations.

- Term frequency (count of the word in the current article)
- Inverse document frequency (log ((#docs) / (1 + #docs using word)))

(Note: docs is the number of article in the corpus, docs using word is the number of articles that contain this word.)

After doing the transformation of TF-IDF, we get a vector from each article, then the dot product of two vectors from two articles is the similarity of these two articles. This is how we measure similarity. When we want to retrieve similar documents, we apply 1-nearest neighbour algorithm:

- Search over each article in corpus
- Compute s = similarity (doc1, Query_article)
- If s > Best_s, record Most_similar_article = doc1 and set Best_s = s
- Return Most_similar_article

In this algorithm, we gave an input (query article), and it returns the most similar article in the corpus. This is the algorithm that we apply when samples are labelled, when samples have no labels, we need to structure it by clustering.

And here we use the k-means algorithm to do the clustering of articles.

K-means algorithm:

- 1. Initialize cluster centers.
- 2. Assign observations to closest cluster center.
- 3. Revise cluster centers as mean of assigned observations.
- 4. Repeat 2 + 3 until convergence

Clustering is widely used in real life, for example: clustering images, discovering users on Amazon, grouping patients by medical condition.

2.4 Recommending products

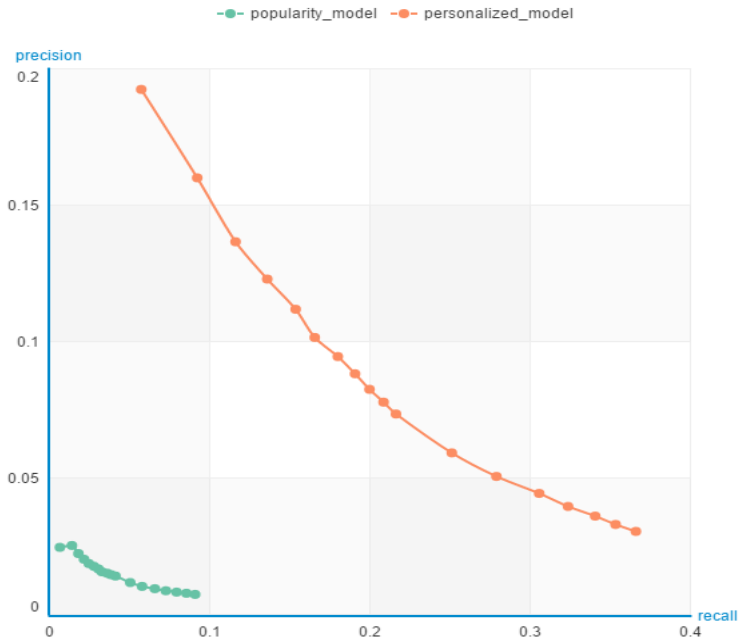
Recommending products is an important issue in e-commerce, like Amazon, Taobao. And there are several ways to recommend products for clients. We may recommend songs for listeners according to the popularity of songs, and this is what we have done at first. Yet this model has a disadvantage, because it recommends same songs for everyone, in other words, this model is not personalized for clients. As everyone's taste is probably different, so we want to create a model which is personalized for our clients. So we used collaborative filtering to recommend songs for users. Collaborative filtering is a method of making automatic predictions(filtering) about the interests of a user by collecting preferences or taste information from many users (collaborating). The underlying assumption of the collaborative filtering approach is that if a person A has the same opinion as a person B on an issue, A is more likely to have B's opinion on a different issue than that of a randomly chosen person.

In order to know if we have improved our model by using collaborative filtering, we plot the Precision-recall curve.

Precision = #(liked and shown) / #shown

Recall = #(liked and shown) / #liked

For a given precision, we want recall as large as possible; for a given recall, we want precision as large as possible. In general, we select the model with the largest AUC (area under the curve).



Precision and recall curve

From the graph, we can tell that our personalized model has improved significantly.

2.5 Deep learning: Searching for images

Deep Learning is making news across the world as one of the most promising techniques in machine learning. Industries are dedicating resources to unlock the deep learning potential, including for tasks such as image tagging, object recognition, speech recognition, and text analysis.

In our final case study, searching for images, we learn how layers of neural networks provide very descriptive (non-linear) features that provide impressive performance in image classification and retrieval tasks.

Features are keys to machine learning, as some features are "very" non-linear, neural network is such a technology that solve this problem.

Artificial neural networks (ANNs) systems are a computational model used in computer science and other research disciplines, which is based on a large collection of simple neural units (artificial neurons), loosely analogous to the observed behavior of a biological brain's axons. Each neural unit is connected with many others, and links can enhance or inhibit the activation state of adjoining neural unit. Each individual neural unit computes using summation function. There may be a threshold function or

limiting function on each connection and on the unit itself, such that the signal must surpass the limit before propagating to other neurons. These systems are self-learning and trained, rather than explicitly programmed, and excel in areas where the solution or feature detection is difficult to express in a traditional computer program.

In our case, deep learning is used to retrieve similar images. Firstly, we used deep learning model's extract features to extract deep features from images, then we apply the K-nearest neighbours algorithm on building a knn model. With this model, we are able to retrieve similar images with one input (one image).

Deep learning is now applied in many domains: visual product recommender, image classification, autonomous driving.

3 Project

3.1 Introductions

As there is not project in this course, I selected one task in another Machine Learning course. In this project, we are supposed to implement hand-written digit recognition by constructing a neural network. There are 5000 training examples in data file, where each training example is a 20 pixels by 20 pixels grayscale image of the digit. Each pixel is represented by a floating point number indicating the grayscale intensity at that location. The 20 by 20 grid of pixels is unrolled into a 400-dimensional vector. Each of these training examples becomes a single row in our data matrix. This gives us a 5000 by 400 matrix where every row is a training example for a handwritten digit image. The second part of the training set is a 5000-dimensional vector y which contains labels for the training set.

3.2 Solution

In this project, we used a neural network with 3 layers to recognize hand-written digits. The input layer is 400 dimensions, the hidden layer is 25 dimensions and the output layer is 10 dimensions. Before constructing a neural network, we need to implement a few algorithms.

3.2.1 1.Implement the forward propagation algorithm to get the cost function:

Firstly, we follow the instructions below to compute $\mathrm{h}_\theta(x)$.

$z^{(2)} = \Theta^{(1)} * a^{(1)}$

$a^{(2)} = g(z^{(2)})$

$z^{(3)} = \Theta^{(2)} * a^{(2)}$

$h_\Theta(x) = a^{(3)} = g(z^{(3)})$

$a_i^{(j)}$: "Activation" of unit i in layer j;

$\Theta^{(j)}$: Matrix of weights controlling function mapping from layer j to layer j+1;

g is the sigmoid function:

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

And then we compute the cost function. The cost is: regularized cost function (J(Θ))

3.2.2 2.Implement the back propagation algorithm to get the gradient:

The instruction is as follows:

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

Set $\Delta_{ij}^{(l)} = 0$

for i = 1 to m:

- Set $a^{(1)} = x^{(i)}$;
- Perform forward propagation to compute $a^{(l)}$(for l = 2, 3, …, L);
- Using $y^{(i)}$, computer $\delta^{(l)} = a^{(l)} - y^{(i)}$;
- Compute $\delta^{(L-1)}, \delta^{(L-2)}, \delta^{(2)}$
- $\Delta_{ij}^{(l)} := \Delta_{ij}^{(l)} + a_j^{(l)} \delta_i^{(l+1)}$;

$D_{ij}^{(l)} := \frac{1}{m}\Delta_{ij}^{(l)} + \lambda\Theta_{ij}^{(l)}$ if y != 0

$D_{ij}^{(l)} := \frac{1}{m}\Delta_{ij}^{(l)}$ if y = 0

The derivate of J(Θ) is : $D_{ij}^{(l)}$.

And now we're able to constructing the neural network. Steps are as follows:

1. Randomly initialize weights.

2. Implement forward propagation to get $h_\Theta(x^{(i)})$ for any $x^{(i)}$.

3. Compute cost function J(Θ).

4. Implement back propagation to compute partial derivatives of J(Θ).

5. Use gradient checking to compare partial derivatives of J(Θ)

with the one using numerical estimate of gradient of J(Θ).

And then, disable gradient checking (this is very important).

6. Use gradient descent to try to minimize J(Θ)as a function of parameter Θ.

And we get an accuracy of 95%.

4 Conclusion

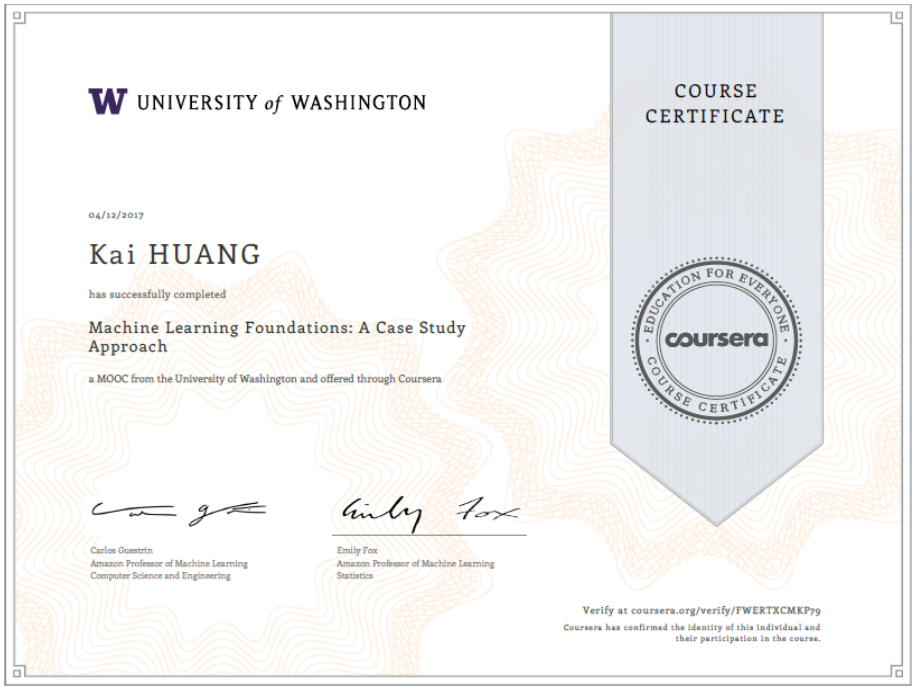
This course helps me identify applications of machine learning, core differences in regression, classification and clustering. And it helps me select the appropriate machine learning task for a potential application. The work flow of machine learning is following:

- 1. Data pre-processing
- 2. Separate data into training set and test set.
- 3. Extract features from training set
- 4. Build a model with training set
- 5. Evaluate the model with test set and repeat 2, 3, 4, 5 if the model doesn't satisfy our needs

As this course treats the machine learning method as a black box, so it doesn't tell a lot details about those technology: how do we do logistic regression manually, how do we build a neural network, what's structure of neural network, what algorithms are used in neural network, etc… So I will continue to study more about machine learning, and I think it's a very promising domain.

5 Reference

Source code of project: https://github.com/axelhuang24/NeuralNetwork



Certificate