Machine Learning for Economics and Finance 1
Linear Regression & Logistic Regression

Kien C Nguyen

18 July, 2020

What is Machine Learning?

Definition

Machine Learning (ML) is a field of artificial intelligence that uses statistical techniques to give computer systems the ability to "learn" (e.g., progressively improve performance on a specific task) from data, without being explicitly programmed (Wikipedia).

`https://en.wikipedia.org/wiki/Machine_learning`

- Explores the study and construction of algorithms that can learn from and make predictions on data
- Such algorithms overcome following strictly static program instructions by making data-driven predictions or decisions, through building a model from sample inputs.
- Employed in a range of computing tasks where designing and programming explicit algorithms with good performance is difficult or infeasible
- Example applications include email filtering, detection of network intruders, NLP, and computer vision.

What is Machine Learning?

https://en.wikipedia.org/wiki/Machine_learning
Tom M. Mitchell (an American computer scientist) provided a widely quoted, more formal definition of the algorithms studied in the machine learning field:

Definition

"A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E."

Types of machine learning

- Machine learning is usually divided into two main types.
- In the predictive or supervised learning approach, the goal is to learn a mapping from inputs x to outputs y, given a labeled set of input-output pairs $D = \{(x_i, y_i)\}_{i=1}^N$. Here D is called the training set, and N is the number of training examples.
- The second main type of machine learning is the descriptive or unsupervised learning approach.
- Here we are only given inputs, $D = \{x_i\}_{i=1}^N$, and the goal is to find "interesting patterns" in the data. This is sometimes called knowledge discovery.
- We also have semi-supervised learning and reinforcement learning.

(from Murphy [1])

Supervised learning

For supervised learning,

- when y_i is categorical, the problem is known as classification or pattern recognition. For example, the problem of classifying emails into 'spam' and 'not spam'.
- y_i is real-valued, the problem is known as regression. For example, the problem of predicting the income level.
- Another variant, known as ordinal regression, occurs where label space Y has some natural ordering, such as grades A–F.

(from Murphy [1])

Contents

- How do we define a linear regression model?
- How do we learn the parameters of our model $f(\mathbf{x})$ from training examples?
- How do we extend linear models to nonlinear models?

Possible applications

- Predict tomorrow's stock market **prices** given current market conditions and other possible side information.
- Predict the **age** of a viewer watching a given video on YouTube.
- Predict the **temperature** at any location inside a building using weather data, time, door sensors, etc.
- Predict the **salaries** of graduate students given GPAs, number of social activities, gender, living location, etc.
- Predict the **number of users** sharing your post on Facebook based on your friend list, hashtag popularity, previous posts, etc.

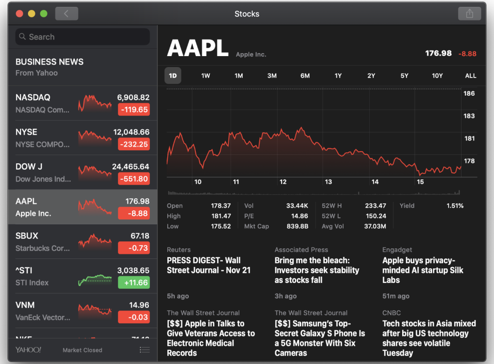


Figure: Apple stock prices (AAPL)

- Normally, we will use stock analysis techniques such as Fibonacci retracement, candlestick, bull/bear signal, etc.
- Can we use Machine Learning methods to help us **automate** the whole process with acceptable results?

How to solve these problems using Machine Learning?

1. Define the problem (e.g. **predicting** some outcome).
2. Collecting the appropriate **data set**.
3. Choose the right machine learning algorithm.
4. Define **evaluation metrics** of the model (e.g. Accuracy, AUC, Precision, Recall, etc.)

Choose the right machine learning algorithm

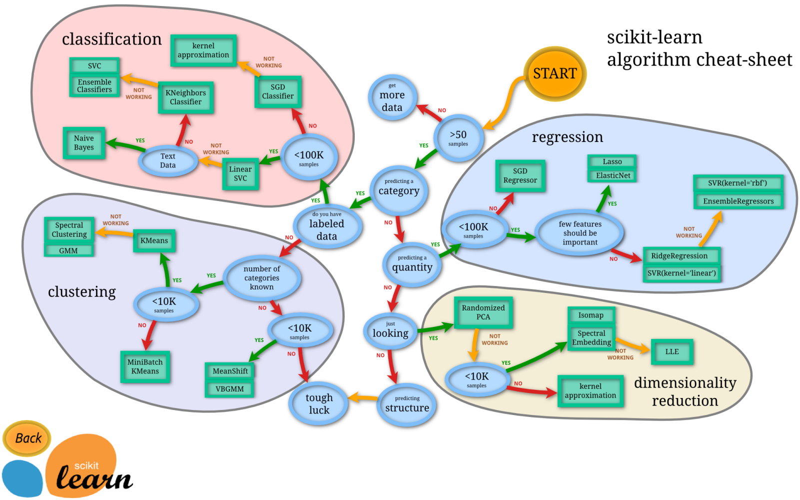


Figure: Machine learning map (sklearn)

Modeling process

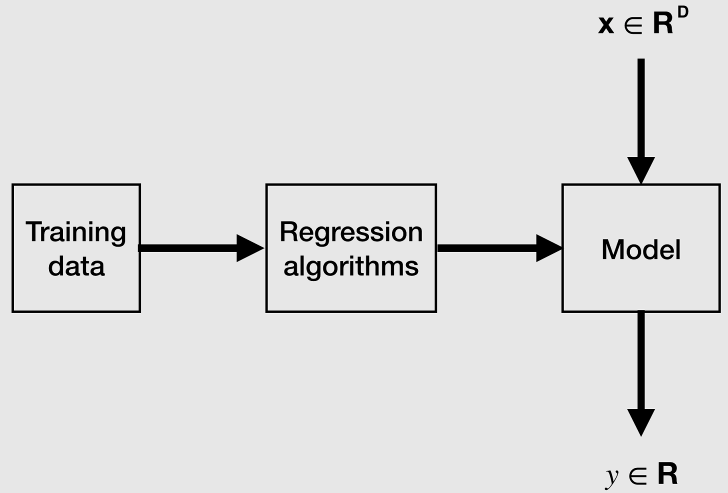


Figure: Regression process

Key points of model

- Need data to build prediction model (training process).
- Could predict *unseen data* in the future (**generalization**).
- **"No Free Lunch"** theorem states that there is no one model that works best for every problem.

Could you see the trend?

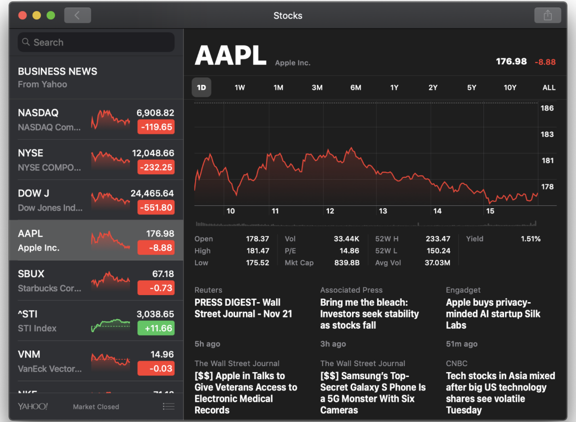


Figure: Apple stock prices (AAPL)

How to draw a straight line that express the trend of data?

Figure: There are many possible straight lines for predicting trends

Contents

Linear function

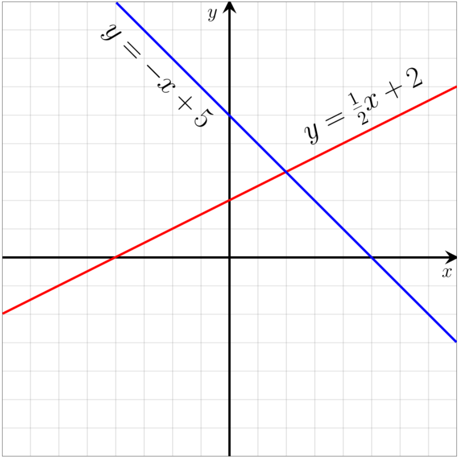


Figure: Linear function (https://en.wikipedia.org/)

Linear model for regression

Linear model for regression is a linear combination of the input variables. It assumes the dependency of the response variable y on the explanatory variables $\mathbf{x}$ is linear.

Formula

$$y(\mathbf{x}, \mathbf{w}) = w_0 + w_1 x_1 + ... + w_D x_D = w_0 + \sum_{j=1}^{D} w_j x_j$$

where

- $y \in \mathbf{R}$: response variable, dependent variable, outcome.
- D: number of dimensions of the input vector $\mathbf{x}$.
- $\mathbf{x} = (x_1, ..., x_D)^T$: input vector (explanatory variable, independent variable, features).
- $\mathbf{w} = (w_0, ..., w_D)$: parameters.
- $D + 1$: total number of parameters.

Hyperplane

Linear is a **straight line** in 2 dimensions space, a **plane** in 3 dimensions space, and a hyperplane in D-dimensional space.

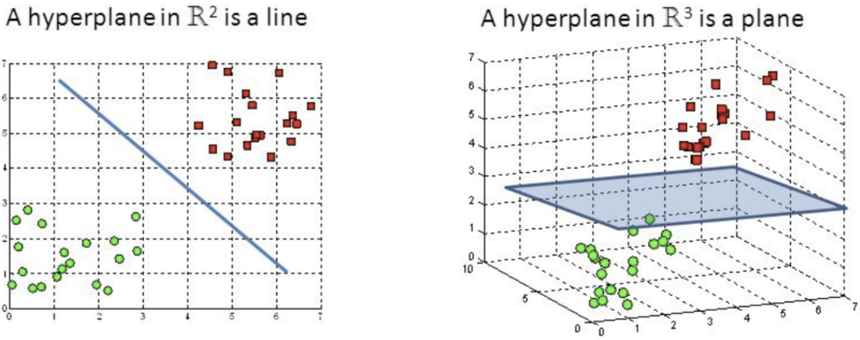


Figure: https://towardsdatascience.com/

Linear regression in one picture

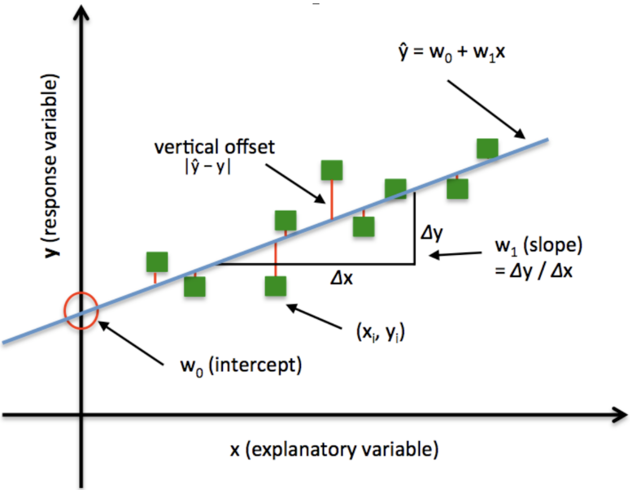


Figure: http://rasbt.github.io/

Loss function

Given the features $\mathbf{x}$, the predicted value of y, $\hat{y}$, is given by
$\hat{y} = f(\mathbf{x}) = w_0 + \sum_{j=1}^{D} w_j x_j$

Loss function

A loss function is a measure of how good a prediction model does in terms of being able to predict the expected outcome.

$$L(\mathbf{w}) = \frac{1}{2} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2$$

where N is the number of training examples.

Our goal: find parameters $\mathbf{w}$ that minimize the loss function. How?

Contents

We have

$$L(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{N}(y^i - \mathbf{w}\mathbf{x}^{(i)})^2$$

where we let $x_0^{(i)} = 1$ to simplify the notation.

Our goal is to find $\hat{\mathbf{w}}$:

$$\hat{\mathbf{w}} = argmin_{\mathbf{w}} L(\mathbf{w}) = argmin_{\mathbf{w}} \left(\frac{1}{2}||\mathbf{y} - \mathbf{X}\mathbf{w}||_2^2\right)$$

Using Ordinary Least Squares

$$
\begin{aligned}
L(\mathbf{w}) &= \frac{1}{2}(y - \mathbf{X}\mathbf{w})^T(y - \mathbf{X}\mathbf{w}) \\
&= \frac{1}{2}\left(y^Ty - 2\mathbf{w}^T\mathbf{X}^Ty + \mathbf{w}^T\mathbf{X}^T\mathbf{X}\mathbf{w}\right).
\end{aligned}
$$

Setting the gradient to 0:

$$
\begin{aligned}
\frac{\partial L(\mathbf{w})}{\partial \mathbf{w}} &= -\mathbf{X}^Ty + \mathbf{X}^T\mathbf{X}\mathbf{w} = 0 \\
&\iff \mathbf{X}^T\mathbf{X}\mathbf{w} = \mathbf{X}^Ty \\
&\iff \mathbf{w} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^Ty.
\end{aligned}
$$

Notes:

- The Hessian in this case is $\mathbf{X}^T\mathbf{X}$, which is a positive semidefinite matrix.
- The matrix $\mathbf{X}^T\mathbf{X}$ must be invertible and difficult to scale with high dimension input vector.
- The case in which $\mathbf{X}^T\mathbf{X}$ is non-invertible will be addressed later.

Solve with Gradient descent

Gradient descent algorithm

Initialize $\mathbf{w} = [0, ..., 0]$;
for $t = 1, ..., T$ **do**
  $\mathbf{w} \leftarrow \mathbf{w} - \eta \nabla L(\mathbf{w})$
end

- η: step size
- $\nabla L(\mathbf{w})$: gradient

Cons: requires the entire set of data samples to be loaded in memory, since it operates on all of them at the same time

Stochastic Gradient descent algorithm

Initialize $\mathbf{w} = [0, ..., 0]$;
for $t = 1, ..., T$ **do**
  for $(x, y) \in D_{train}$ **do**
    $\mathbf{w} \leftarrow \mathbf{w} - \eta \nabla L(x, y, \mathbf{w})$
  end
end

- Pros: during learning, compute $L(x, y, \mathbf{w})$ before updating $\mathbf{w}$, so require less memory.
- Cons: requires a number of hyperparameters such as the regularization parameter and the number of iterations.

How about complex data set?

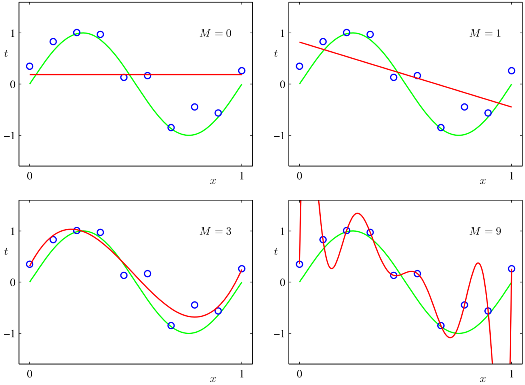


Figure 1.4 Plots of polynomials having various orders M, shown as red curves, fitted to the data set shown in Figure 1.2.

Figure: C. Bishop, Pattern Recognition and Machine Learning

Basis function

Extend the class of models by considering linear combinations of fixed **nonlinear functions** of the form

$$y(\mathbf{x}, \mathbf{w}) = w_0 + \sum_{j=1}^{M-1} w_j \phi_j(\mathbf{x})$$

where $\phi_j(\mathbf{x})$ are known as basis functions. Identity "basis function" is $\phi(\mathbf{x}) = \mathbf{x}$.

Some basis function

Polynomial basis function

$$\phi_j(x) = x^j$$

Gaussian basis function

$$\phi_j(x) = exp\{-\frac{(x-\mu_j)^2}{2s^2}\}$$

Sigmoidal basis function

$$\phi_j(x) = \sigma(\frac{x-\mu_j}{s})$$

where $\sigma(a)$ is the logistic sigmoid function defined by

$$\sigma(a) = \frac{1}{1+exp(-a)}$$

Some basis function

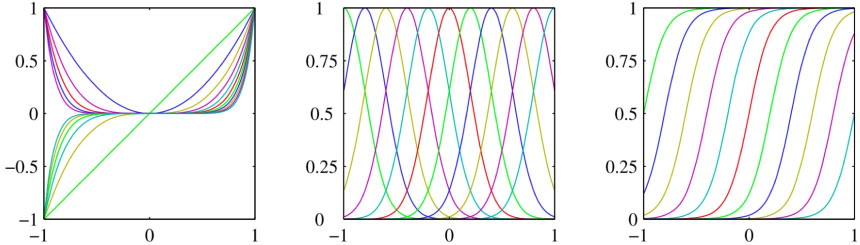


Figure 3.1 Examples of basis functions, showing polynomials on the left, Gaussians of the form (3.4) in the centre, and sigmoidal of the form (3.5) on the right.

Figure: C. Bishop, Pattern Recognition and Machine Learning

Assumptions of the (Multiple) Linear Regression Model

Formula

$$y^{(i)}(\mathbf{x}^{(i)}, \mathbf{w}) = w_0 + \sum_{j=1}^{D} w_j x_j^{(i)} + \epsilon^{(i)}$$

- The relationship between the dependent variable (y) and the independent variables ($x_j,\ j = 1, \dots, D$) is linear.
- The independent variables ($x_j,\ j = 1, \dots, D$) are not random. There is no exact linear relation between two or more of the independent variables (multi-collinearity).
- The expected value of the error term, conditioned on the independent variables, is 0. $E\left[\epsilon^{(i)}|x^{(i)}\right] = 0$
- The variance of the error term is the same for all observations $E\left[\left(\epsilon^{(i)}\right)^2\right] = \sigma_\epsilon^2$ (homoscedasiticity).
- The error term is uncorrelated across observations $E\left[\epsilon^{(i)}\epsilon^{(j)}\right] = 0,\ i \neq j$
- The error term is normally distributed.

Contents

Binary Classification

- Recall that in supervised learning, if the targets (labels) are categorical, the problem is called classification.
- Classify an email as Not Spam / Spam
- In credit scoring, classify a customer as Good / Bad
- In network intrusion detection, classify a connection as Normal / Attack
- Detect the gender (Male / Female) using profile pictures

Logistic Regression

- Recall that in linear regression, $\hat{y} = \mathbf{w}^T\mathbf{x}$.
- This model can only be used if y is not upper-bounded and not lower-bounded.
- In logistic regression, we predict the probability of a Positive Class (vs a Negative Class).
- E.g. Probability that an email is Spam, probability that a customer is a Bad customer.

Probability of passing an exam versus hours of study

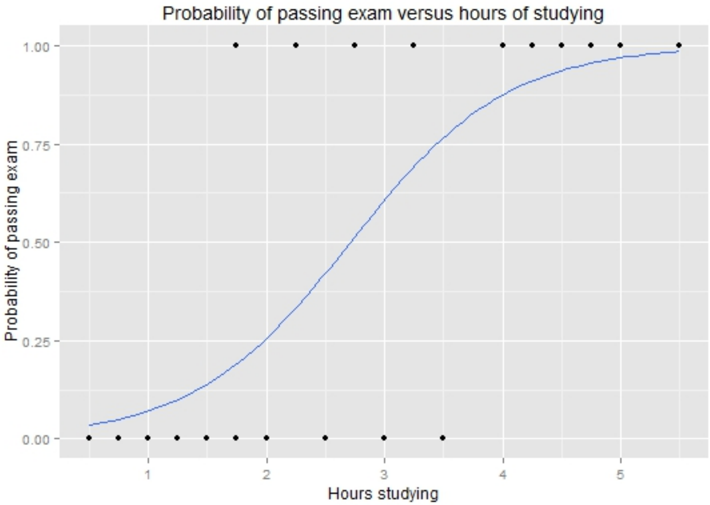
- A group of 20 students spend between 0 and 6 hours studying for an exam. How does the number of hours spent studying affect the probability that the student will pass the exam?
- We predict the probability that a student passes the exam ($y = 1$) using the number of hours that student spent.

Source: `https://en.wikipedia.org/wiki/Logistic_regression`

Hours	Pass	Hours	Pass
.5	0	2.75	1
.75	0	3	0
1	0	3.25	1
1.25	0	3.5	0
1.5	0	4	1
1.75	0	4.25	1
1.75	1	4.5	1
2	0	4.75	1
2.25	1	5	1
2.5	0	5.5	1

Source: `https://machinelearningcoban.com/2017/01/27/logisticregression/`

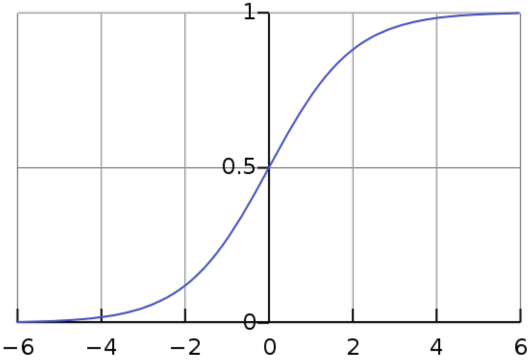
Probability of passing an exam versus hours of study



Source: `https://machinelearningcoban.com/2017/01/27/logisticregression/`

- Use a function $\Phi(\mathbf{w}^T\mathbf{x})$
- As this is a probability, we want $0 \leq \Phi(\mathbf{w}^T\mathbf{x}) \leq 1$
- Sigmoid function (Logistic function)

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

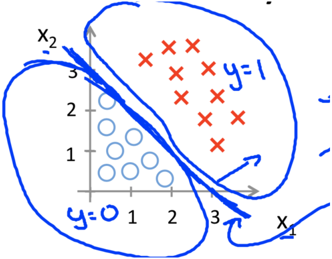


Source: `https://en.wikipedia.org/wiki/Logistic_regression`

Suppose we predict $y = 1$ if $P\{y = 1\} \geq 0.5$.

$$\sigma(z) \geq 0 \iff \mathbf{w}^T\mathbf{x} \geq 0$$

Predict $y = 0$ if $P\{y = 1\} < 0.5 \iff \mathbf{w}^T\mathbf{x} < 0$



Source: Andrew Ng – Machine Learning (Coursera)

Contents

Recall that the training set is $\left(\left(\mathbf{x}^{(1)}, y^{(1)}\right), \left(\mathbf{x}^{(2)}, y^{(2)}\right), \ldots \left(\mathbf{x}^{(N)}, y^{(N)}\right)\right)$.

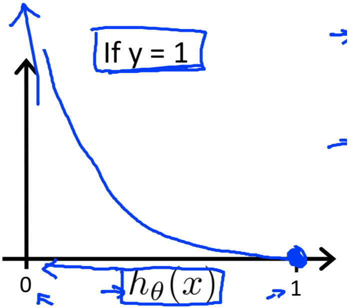
where $\mathbf{x}^{(i)}$ is given by $\mathbf{x}^{(i)} = \begin{bmatrix} x_0^{(i)} \\ x_1^{(i)} \\ \ldots \\ x_D^{(i)} \end{bmatrix}$

$x_0^{(i)} = 1,\ y^{(i)} \in \{0, 1\}$

Loss for each training example

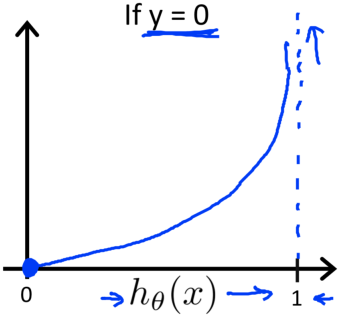
$$
\begin{aligned}
L(\hat{y}, y) &= \begin{cases} -log(\Phi(\mathbf{w}^T\mathbf{x})) & if \quad y = 1 \\ -log(1 - \Phi(\mathbf{w}^T\mathbf{x})) & if \quad y = 0 \end{cases} \\
&= -ylog(\Phi(\mathbf{w}^T\mathbf{x})) - (1 - y)log(1 - \Phi(\mathbf{w}^T\mathbf{x}))
\end{aligned}
$$

Loss for each training example: $y = 1$



$L = 0$ when $\Phi(\mathbf{w}^T\mathbf{x}) = 1$
$L \to \infty$ as $\Phi(\mathbf{w}^T\mathbf{x}) \to 0$
Source: Andrew Ng – Machine Learning (Coursera)

Loss for each training example: $y = 0$



$L = 0$ when $\Phi(\mathbf{w}^T\mathbf{x}) = 0$
$L \to \infty$ as $\Phi(\mathbf{w}^T\mathbf{x}) \to 1$
Source: Andrew Ng – Machine Learning (Coursera)

Loss function

$$
\begin{aligned}
L(\mathbf{w}) &= \frac{1}{N}\sum_{i=1}^{N} -y^{(i)} log(\Phi(\mathbf{w}^T\mathbf{x}^{(i)})) - (1-y^{(i)}) log(1-\Phi(\mathbf{w}^T\mathbf{x}^{(i)})) \\
&= -\frac{1}{N}\sum_{i=1}^{N} y^{(i)} log(\Phi(\mathbf{w}^T\mathbf{x}^{(i)})) + (1-y^{(i)}) log(1-\Phi(\mathbf{w}^T\mathbf{x}^{(i)}))
\end{aligned}
$$

We find the $\hat{\mathbf{w}}$ such that

$$\hat{\mathbf{w}} = argmin_{\mathbf{w}} L(\mathbf{w}) \tag{1}$$

Once we have $\hat{\mathbf{w}}$, the prediction for a new $\mathbf{x}$ is

$$P\{\hat{y} = 1\} = \Phi(\mathbf{w}^T\mathbf{x}) \tag{2}$$

Contents

Initialize $\mathbf{w} = [0, ..., 0]$;
Repeat $\mathbf{w} \leftarrow \mathbf{w} - \eta \nabla_{\mathbf{w}} L(\mathbf{w})$

- η: step size
- $\nabla_{\mathbf{w}} L(\mathbf{w})$: gradient

The update for each w_j:

$$w_j = w_j - \eta \sum_{i=1}^{N} \left(\Phi(\mathbf{w}^T \mathbf{x}^{(i)}) - y^{(i)} \right) x_j^{(i)} \tag{3}$$

References

[1] Bishop, C. M. (2013). Pattern Recognition and Machine Learning. Journal of Chemical Information and Modeling (Vol. 53).
[2] Wikipedia. Gradient descent, Ordinary least squares, Stochastic gradient descent.
[3] Wikipedia – Logistic Regression – https://en.wikipedia.org/wiki/Logistic_regression
[4] Vu Huu Tiep – Machine Learning Co Ban – https://machinelearningcoban.com/2017/01/27/logisticregression/
[5] Andrew Ng – Machine Learning (Coursera) – https://www.coursera.org/learn/machine-learning