Regression

In this tutorial, we will have a look at three basic regression algorithms in Machine Learning: linear regression, logistic regression, and softmax regression.

I. Linear Regression

Linear regression is the most basic yet powerful tool of statistics and (supervised) machine learning. Given a dataset $D = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ where $x_i \in \mathbb{R}^d$ are the features and $y_i \in \mathbb{R}$ is the real-valued label of the i-th data point, linear regression aims to find a function $f_\theta : \mathbb{R}^d \to \mathbb{R}$ *parameterized*[1] by $\theta \in \mathbb{R}^m$ that predicts a label y given a new x.

1. Common misconceptions

These are some common mistakes about linear regression that should be avoided:

1. Linear regression is just a line/plane/hyperplane/linear function with respect to x, i.e., $f_\theta(x) = \theta^T x,\ \theta \in \mathbb{R}^d$ (1) (see Figure 1).
2. Linear regression cannot model non-linear relationships.

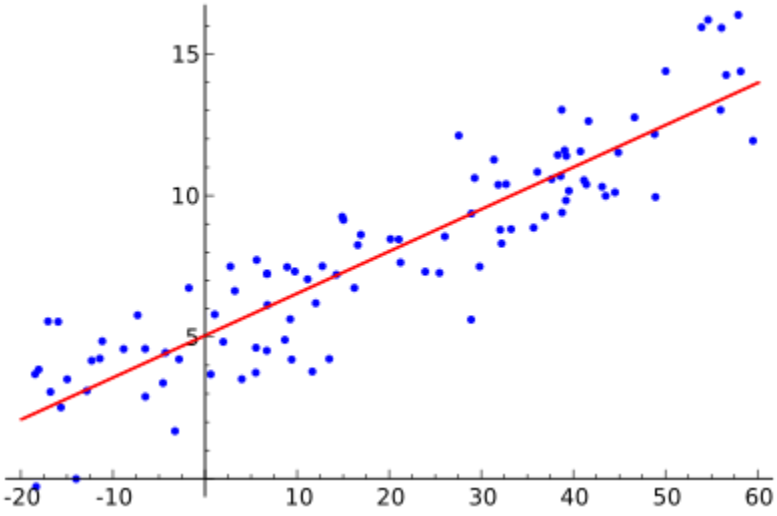

Figure 1: A typical illustration of linear regression where it fits a straight line (colored in red) onto the data points (colored in blue).

Equation (1) is the well-known basic form of linear regression (there is actually an additional bias term, but I will omit it here for simplicity). However, there is a more general form of linear regression that is less popular.

[1] Describe or represent in terms of a parameter or parameters.

The general form of linear regression[2]**:**

$$f_\theta(x) = \theta^T \phi(x) = \sum_{i=1}^{m} \theta_i \phi_i(x)$$

Where $\phi : \mathbb{R}^d \to \mathbb{R}^m$ is a **basis function**.

Remark #1: When augmented with an appropriate basis function, linear regression can model any (linear or non-linear) relationships.

For example: Let $x, y \in \mathbb{R}$; $\theta \in \mathbb{R}^m$; $\phi(x) = \left[1, x, x^2, \ldots, x^{m-1}\right]$ we have $f_\theta(x) = \theta_1 + \theta_2 x + \cdots + \theta_m x^{m-1}$ (this is also known as the polynomial regression). Now our hypothesis function can be fitted onto more complex data (see Figure 2).

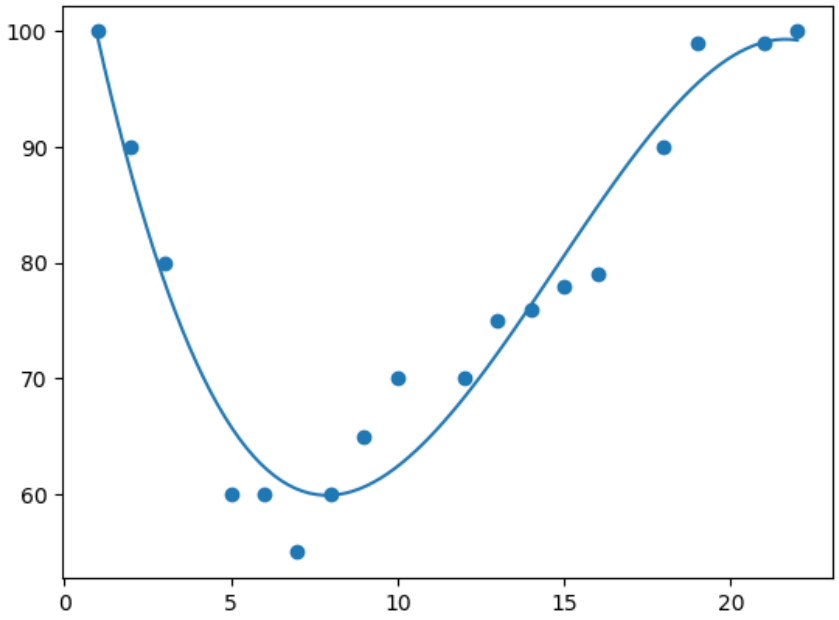

Figure 2: An example of linear regression with polynomial basis.
It can model more than just linear relationships.

Here are some other interesting basis functions that you can study further: Radial basis function[3], Fourier basis function[4], Wavelets basis function[5].

---

[2] Murphy, K. P. (2012). Linear regression. In Machine learning: a probabilistic perspective. MIT Press.
[3] Orr, M. J. (n.d.). Introduction to Radial Basis Function Networks. https://www.cc.gatech.edu/~isbell/tutorials/rbf-intro.pdf.
[4] A. Silvescu, "Fourier neural networks," IJCNN'99. International Joint Conference on Neural Networks. Proceedings (Cat. No.99CH36339), 1999, pp. 488-491 vol.1, doi: 10.1109/IJCNN.1999.831544.
[5] Jun Zhang, G. G. Walter, Y. Miao and Wan Ngai Wayne Lee, "Wavelet neural networks for function learning," in IEEE Transactions on Signal Processing, vol. 43, no. 6, pp. 1485-1497, June 1995, doi: 10.1109/78.388860.

Remark #2: The “linear” in “linear regression” refers to the linear relationship between the output variable y and the optimizing variables (the parameters θ), not necessarily between y and x.

2. If linear regression can model any relationship, why not use it for every ML problem?

Although I mentioned earlier that linear regression can theoretically fit on any arbitrary data given an appropriate basis function, in reality, we do not see linear regression used in complex problems because choosing the right basis ϕ can be a bit tricky. For example, in the polynomial regression above, how do we choose the right degree m that is not too simple to underfit but also not too complex to overfit the data? Or how do we make sure all generated features from the basis function are not redundant? Not mentioning other limitations such as being prone to overfitting, outliers, etc.

3. Linear regression models are neural networks, too[6]

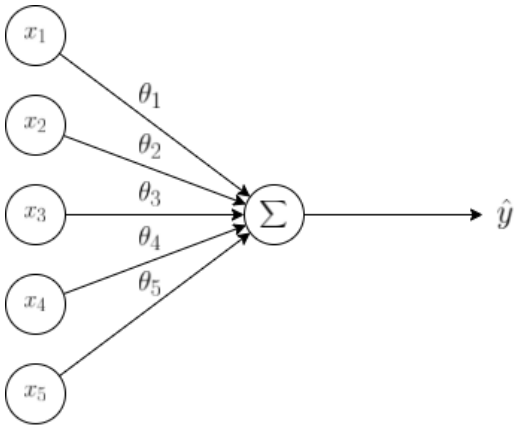

Figure 3: A single layer neural network

Look at Figure 3, it’s a single layer neural network, but isn’t it just $\hat{y} = f(x) = \theta^T x$? Which is the same as Equation (1), which is a linear regression.

4. Bayesian linear regression

If you know about the frequentist vs bayesian debate in statistics[7], I will tell you one interesting fact: the linear regression we have discussed so far is actually the frequentist version of linear regression. There is another version of linear regression within the context of bayesian statistical inference that is very interesting and worth mentioning here.

[6] https://d2l.ai/chapter_linear-networks/linear-regression.html#from-linear-regression-to-deep-networks
[7] Are you Bayesian or Frequentist?

Quick recap: Bayes' theorem

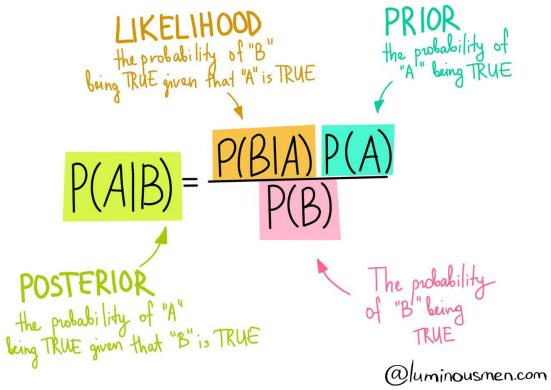

Figure 4: Bayes' theorem

- **Prior distribution**: represent one's initial belief/knowledge on event A.
- **Posterior distribution**: represent one's updated belief/knowledge on event A after observing some evidence/event B.
- Bayes' theorem equation should be thought of as an iterative process of updates, where the posterior of the previous iteration becomes the prior of the next iteration. We may start with a naive (and probably incorrect) prior, after a series of observing new evidence and updating our belief, our final posterior (updated knowledge about the event A) will be more and more accurate.

In the Bayesian viewpoint, we formulate linear regression using probability distributions rather than point estimates. The response, y, is not estimated as a single value, but is assumed to be drawn from a probability distribution. The model for Bayesian Linear Regression with the response sampled from a normal distribution is:

$$y \sim \mathcal{N}\left(\theta^T X, \sigma^2 I\right)$$

The output, y is generated from a normal (Gaussian) Distribution characterized by a mean and variance. The mean for linear regression is the transpose of the weight matrix multiplied by the predictor matrix. The variance is the square of the standard deviation σ (multiplied by the Identity matrix because this is a multi-dimensional formulation of the model).

The aim of Bayesian Linear Regression is not to find the single "best" value of the model parameters, but rather to determine the posterior distribution for the model parameters. Not only is the response generated from a probability distribution, but the model parameters are assumed to come from a distribution as well.

Frequentist Linear Regression vs Bayesian Linear Regression

1. **Frequentist Linear Regression cannot represent uncertainty while Bayesian Linear Regression can**
   One limitation of Frequentist linear regression is that it does not provide us a certainty measure of its prediction. For example, in Figure 2, the regression line fits nicely onto the training data points, but it cannot tell us anything about the likelihood of predictions on new data points. In contrast, we can achieve this in Bayesian linear regression. The result of performing Bayesian Linear Regression is a distribution of possible model parameters based on the data and the prior. This allows us to quantify our uncertainty about the model: if we have fewer data points, the posterior distribution will be more spread out (see Figure 5).
2. **In Bayesian Linear Regression, we can inject external domain knowledge into our model**
   If we have domain knowledge, or a guess for what the model parameters should be, we can include them in our model through the prior distribution, unlike in the frequentist approach which assumes everything there is to know about the parameters comes from the data. If we don't have any estimates ahead of time, we can use non-informative priors[8] for the parameters such as a normal distribution.
3. **Bayesian can be computationally intractable**
   In practice, evaluating the posterior distribution for the model parameters is intractable for continuous variables, so we use sampling methods to draw samples from the posterior in order to approximate the posterior. The technique of drawing random samples from a distribution to approximate the distribution is one application of Monte Carlo methods.

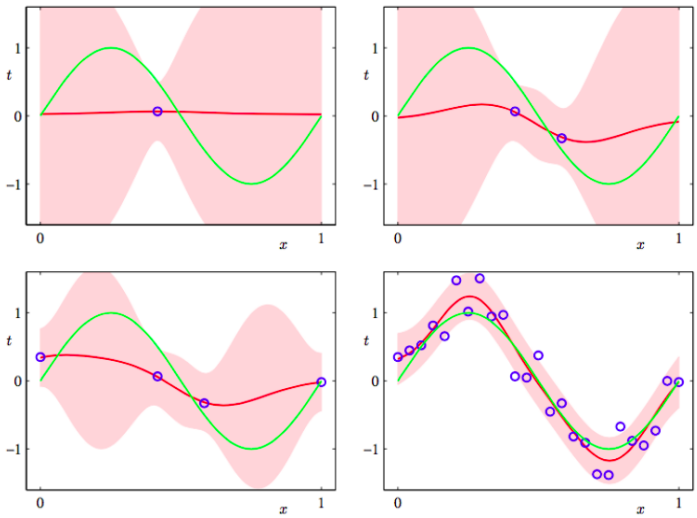

Figure 5: Visualization of the confidence intervals based on the posterior predictive mean and variance at each point. Regions that do not have many observations will be more spread out (i.e., uncertain).

[8] https://www.ime.unicamp.br/~veronica/MI402/Randi21998.pdf

II. Logistic and Softmax ~~Regression~~ Classification

Although logistic regression has “regression” in its name, it is actually an algorithm to solve classification tasks. The same applies to softmax regression.

Once you understand the gist of linear regression, you will see that logistic regression is nothing but an extension of linear regression to handle the classification task and softmax regression is just a generalization of logistic regression to perform multi-class classification.
Given a data $D = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ where $x_i \in \mathbb{R}^d$ are the features and $y_i \in \{0, 1\}$ is the discrete label of the i-th data point, logistic regression aims to find a function $f_\theta : \mathbb{R}^d \to \{0, 1\}$ parameterized by $\theta \in \mathbb{R}^m$ that predicts a label y given a new x. f_θ has the following simple form:

$$f_\theta(x) = \sigma\big(\theta^T x\big),\ \theta \in \mathbb{R}^d$$

where $\sigma(x) = \dfrac{1}{1 + e^{-x}}$ is the sigmoid function, also known as the logistic function (hence the name logistic regression).

Similar to linear regression, the simple form of logistic regression can only learn a linear decision boundary between two classes. In order to learn non-linear decision boundaries, we augment it with basis function expansion:

The general form of logistic regression:

$$f_\theta(x) = \sigma\big(\theta^T \phi(x)\big) = \sigma\left(\sum_{i=1}^{m} \theta_i \phi_i(x)\right)$$

Where $\phi : \mathbb{R}^d \to \mathbb{R}^m$ is a **basis function**.

Regarding softmax regression for the n-class classification problem, now instead of having a single output head, we have n outputs with softmax activation function:

$$f_\theta(x) = \mathrm{softmax}(\Theta \phi(x)),\ \Theta \in \mathbb{R}^{n \times m}$$

where $\mathrm{softmax}(x)_i = \dfrac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}} \quad \text{for } i = 1, \ldots, n$ is the generalization of the logistic function to multiple dimensions[9].

[9] https://en.wikipedia.org/wiki/Softmax_function