Machine Learning I (DATS 6202)

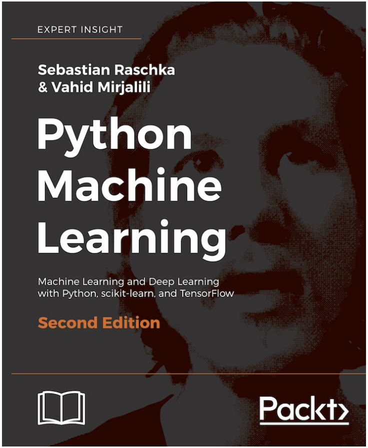
Linear Regression

Yuxiao Huang

Data Science, Columbian College of Arts & Sciences
George Washington University
yuxiaohuang@gwu.edu

September 10, 2018

Reference



Picture courtesy of the website of the book code repository and info resource

Reference

- This set of slices is an excerpt of the book by Raschka and Mirjalili, with some trivial changes by the creator of the slides
- Please find the reference to and website of the book below:
  - *Raschka S. and Mirjalili V. (2017). Python Machine Learning. 2nd Edition.*
  - `https://sebastianraschka.com/books.html`
- Please find the website of the book code repository and info resource below:
  - `https://github.com/rasbt/python-machine-learning-book-2nd-edition`

Overview

1. The linear regression model
2. Exploring and visualizing datasets
3. Evaluating regression models and diagnosing common problems
4. Implementing linear regression models
5. Training regression models that are robust to outliers

Simple (univariate) linear regression

- Simple linear regression expresses the relationship between two continuous-valued variables, x and y:

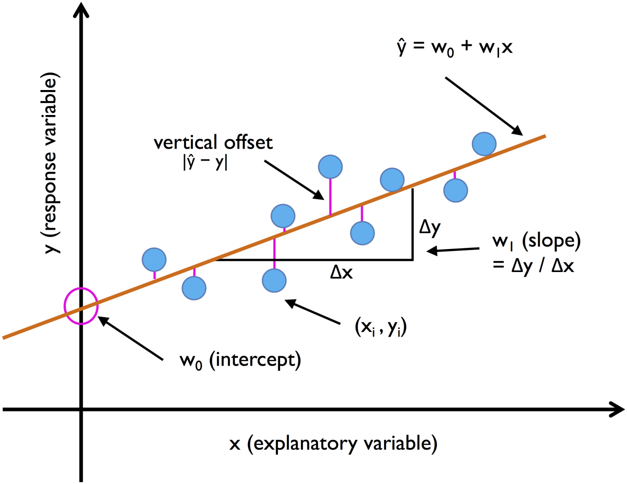
$$y = w_0 + w_1 x$$

- Here:
    - x: explanatory variable (or independent variable, regressor)
    - y: response variable (or dependent variable, regressand)
    - w_0: intercept
    - w_1: slope
- The goal is to:
    1. learn w_0 and w_1
    2. predict the value of y based on the value of x

The best fitting line (regression line)

- Linear regression can be understood as finding the best-fitting straight line through the sample points, as shown in Figure 1 (see next page)
- The best-fitting line is also called the regression line
- The vertical lines from the regression line to the sample points are the errors of our prediction (offsets, or residuals)

Figure 1


ŷ = w0 + w1x
vertical offset
|ŷ − y|
Δy
Δx
w1 (slope)
= Δy / Δx
(xi , yi)
w0 (intercept)
y (response variable)
x (explanatory variable)

Multiple linear regression

- We can generalize simple linear regression with only one explanatory variable to a model with multiple explanatory variables:

$$y = w_0 x_0 + w_1 x_1 + \cdots + w_m x_m = \sum_{i=0}^{m} w_i x_i = \mathbf{w}^T \mathbf{x}$$

- Such model is called multiple linear regression

Visualizing the important characteristics of a dataset

- Exploratory Data Analysis (EDA) is an important and recommended first step prior to the training of a machine learning model
- The graphical EDA toolbox may help
  - visually detect the presence of outliers
  - the distribution of the data
  - the relationship between features

Two kinds of useful graphical summaries

- Scatterplot matrix: allows us to visualize the pair-wise correlations between different features
- Correlation matrix:
  - a square matrix that contains the Pearson product-moment correlation coefficients (or Pearson's r), which measures the linear dependence between pairs of features
  - can be calculated as the covariance between two features x and y, divided by the product of their standard deviations

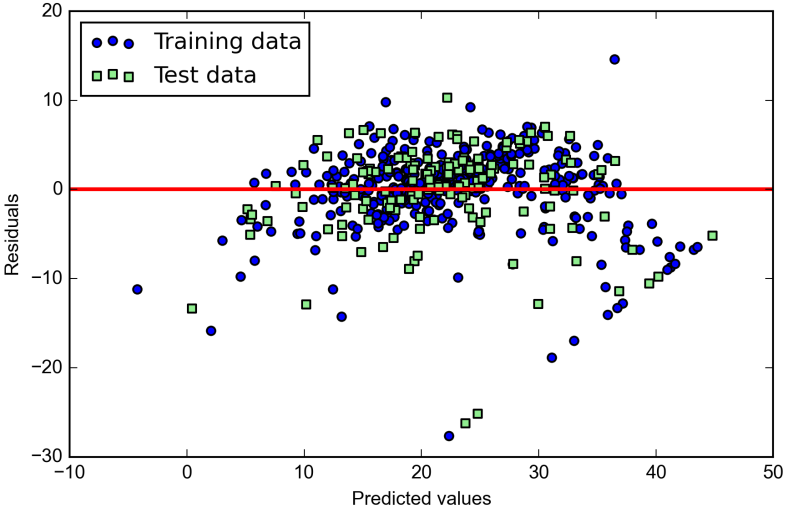
$$r = \frac{\sum_{i=1}^{n}\left[\left(x^{(i)} - \mu_x\right)\left(y^{(i)} - \mu_y\right)\right]}{\sqrt{\sum_{i=1}^{n}\left(x^{(i)} - \mu_x\right)^2}\sqrt{\sum_{i=1}^{n}\left(y^{(i)} - \mu_y\right)^2}} = \frac{\sigma_{xy}}{\sigma_x \sigma_y}$$

  - identical to a covariance matrix computed from standardized data
- Combine the two summaries for choosing explanatory variables

Evaluating the performance using residual plot

- When our model uses multiple explanatory variables, we cannot visualize the model in a two-dimensional plot
- Instead we can plot the residuals (the difference or vertical distances between the actual and predicted values) versus the predicted values to diagnose our regression model
- One residual plot is shown in Figure 2 (see next page)

Figure 2



Training data
Test data
Residuals
Predicted values
20
10
0
−10
−20
−30
−10
0
10
20
30
40
50

Evaluating the performance using Mean Squared Error

- Another useful quantitative measure of a model's performance is the so-called Mean Squared Error (MSE)

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(y^{(i)} - \hat{y}^{(i)}\right)^2$$

- The MSE is useful for comparing different regression models or for tuning their parameters via a grid search and cross-validation
- See details in `ch10.ipynb`

Estimating the parameters with gradient descent

- The parameters of the regression can be approximated by minimizing the cost function
- The cost function here is the Ordinary Least Squares (OLS) function

$$J(w) = \frac{1}{2} \sum_{i=1}^{n} \left(y^{(i)} - \hat{y}^{(i)} \right)^2$$

- We can minimize the cost function to learn the weights via optimization algorithms such as Gradient Descent (GD)
- Using GD, the rule for updating the weights can be written as

$$\mathbf{w} = \mathbf{w} + \Delta\mathbf{w} \quad \text{where} \quad \Delta\mathbf{w} = -\eta \nabla J(\mathbf{w}).$$

  Here:
  - η is the learning rate
  - $\nabla J(\mathbf{w})$ the gradient of $J(\mathbf{w})$
- **Q:** Why the negative sign?

Estimating the parameters with gradient descent

- The parameters of the regression can be approximated by minimizing the cost function
- The cost function here is the Ordinary Least Squares (OLS) function

$$J(w) = \frac{1}{2} \sum_{i=1}^{n} \left(y^{(i)} - \hat{y}^{(i)} \right)^2$$

- We can minimize the cost function to learn the weights via optimization algorithms such as Gradient Descent (GD)
- Using GD, the rule for updating the weights can be written as

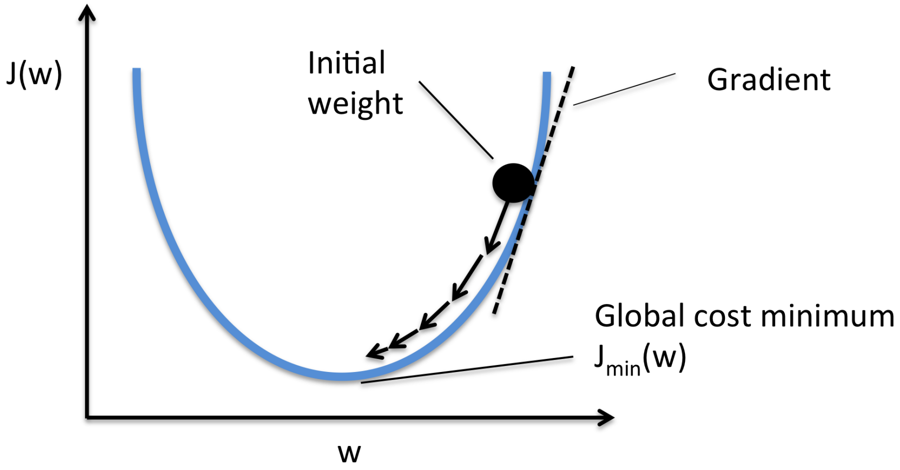
$$\mathbf{w} = \mathbf{w} + \Delta \mathbf{w} \quad \text{where} \quad \Delta \mathbf{w} = -\eta \nabla J(\mathbf{w}).$$

  Here:
  - η is the learning rate
  - $\nabla J(\mathbf{w})$ the gradient of $J(\mathbf{w})$
- **Q:** Why the negative sign?
- **A:** Two kinds of explanations

The visual explanation

- The negative sign updates the weights by taking a step in the opposite direction of the gradient
- Thus we can minimize, rather than maximize, the cost function
- This is also the reason why it is called gradient descent, rather than gradient assent
- Figure 3 (see next page) shows the idea

Figure 3



The mathematical explanation

- The updating rule for the weight of x_j, w_j, is

$$w_j = w_j + \Delta w_j \quad \text{where} \quad \Delta w_j = -\eta \frac{\partial J}{\partial w_j}.$$

Here, $\frac{\partial J}{\partial w_j}$ is the partial derivative of J with respect to w_j:

$$\frac{\partial J}{\partial w_j} = -\sum_i \left(y^{(i)} - \hat{y}^{(i)}\right) x_j^i.$$

- Thus, the updating rule for w_j can be written as

$$w_j = w_j + \eta \sum_i \left(y^{(i)} - \hat{y}^{(i)}\right) x_j^i.$$

The mathematical explanation

- **Q:** Why does the updating rule work:

$$w_j = w_j + \Delta w_j \quad \text{where} \quad \Delta w_j = \eta \sum_i \left(y^{(i)} - \hat{y}^{(i)}\right) x_j^i$$

- **A:** Because it pulls the predicted value ($\hat{y}$) closer to the actual one (y)
- Assume $y^{(i)} = 1$, $\hat{y}^{(i)} = -1$, and $x_j^{(i)} > 0$, then:
  1. $\Delta w_j > 0$
  2. $w_j \uparrow$
  3. $\hat{y} = \mathbf{w}^T \mathbf{x} \uparrow$

Fitting a robust regression model using RANSAC

- Linear regression models can be heavily impacted by the outliers
- In certain situations, a very small subset of our data can have a big effect on the estimated model coefficients (the parameters)
- RANdom SAmple Consensus (RANSAC) algorithm is an alternative to throwing out outliers, by fitting a regression model to a subset of the data, the so-called inliers

The RANSAC algorithm

- Basic steps in RANSAC
  1. Select a random number of samples to be inliers and fit the model
  2. Test all other data points against the fitted model and add those points that fall within a user-given tolerance to the inliers
  3. Refit the model using all inliers
  4. Estimate the error of the fitted model versus the inliers
  5. Terminate the algorithm if the performance meets a certain user-defined threshold or if a fixed number of iterations has been reached; go back to step 1 otherwise
- See details in `ch10.ipynb`