Business Analytics II

Lecture 1
Regression with Python

Pedro Hernandez Serrano
p.hernandezserrano@maastrichtuniversity.nl

April 10, 2018

BISS Institute

Today's session

- Data Science Applications on Business Cases
  - Data products and type of users
- Supervised Learning - Regression in Python
  - What is Regression?
  - Use cases: Bias, Spurious correlations, wrong use cases
  - Univariate Statistics
  - Linear Model
  - Multiple regression
  - Overfitting and Complexity
  - Regularization methods: Ridge, Lasso, Elastic-net
- Hands-on during the session on Jupyter Notebooks
  - Construct a linear regressor from scratch
  - Measuring Overfiting and Performance
  - Regression Models using Turicreate Machine Learning Library

Data Science Applications on Business Cases

Data Science Applications

Business Intelligence vs Business Analytics

Image: https://www.linkedin.com/pulse/business-intelligence-vs-analytics-what-needs-know-john-vuu/

Data Science Applications

Business Intelligence

- Recording to DWH
- Getting new insights from the business data
- Customized reporting
- Analyzing the history (past)

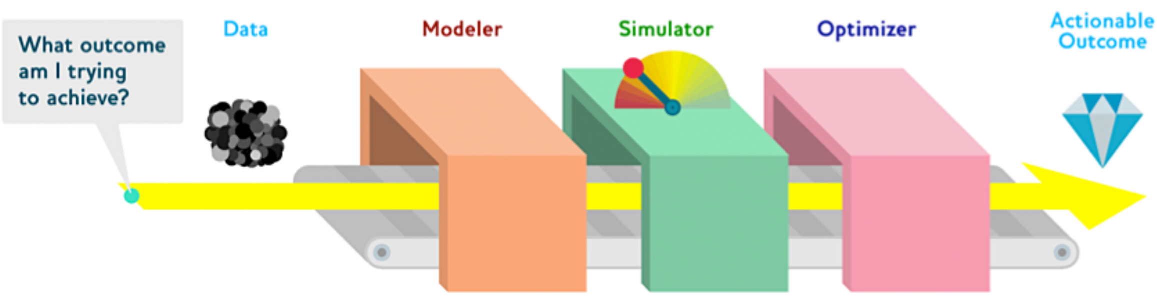
Business Analytics

- Exploiting from DWH
- Understanding the limitations of the business data
- Customized models
- Analyzing predictions (future)

Both conduct data-driven decision making.

Data Science Applications

Data Product



- Dynamic reports
- Static model for decision making
- Prototype for a new business product
- Production model for users (consumer, internal. executive, etc.)

Image: http://blog.kaggle.com/2012/03/28/drivetrain-approach-to-designing-great-data-products/

Data Science Applications

Pros

- Focuses on data driven decision over politics and gut feelings
- Automates decisions that might be financially and mentally taxing
- Improves consistency, accuracy and forces teams to draw out their decisions processes

Cons

- If an algorithm is incorrect the team might overly trust it
- Privacy regulations and data protection might be an issue
- GDPR and explainability of models

Data Science Applications

Example

- Fire-up Jupyter Notebook via **https://jupyter.org/try**
- Go to the drive folder of the class and make a personal copy
- Open intro.ipynb
- Discuss the problem

Supervised Learning
Regression with Python

Supervised Learning Regression

What is Regression?

A technique for determining the statistical relationship between two or more variables where a change in a dependent variable is associated with, and depends on, a change in one or more independent variables.

- A Supervised Learning Method of Machine Learning
- A Statistical Method in Bayesian or Frequentist Inference

regression
/rɪˈɡrɛʃ(ə)n/

noun

1. a return to a former or less developed state.
   "it is easy to blame unrest on economic regression"
2. STATISTICS
   a measure of the relation between the mean value of one variable (e.g. output) and corresponding values of other variables (e.g. time and cost).

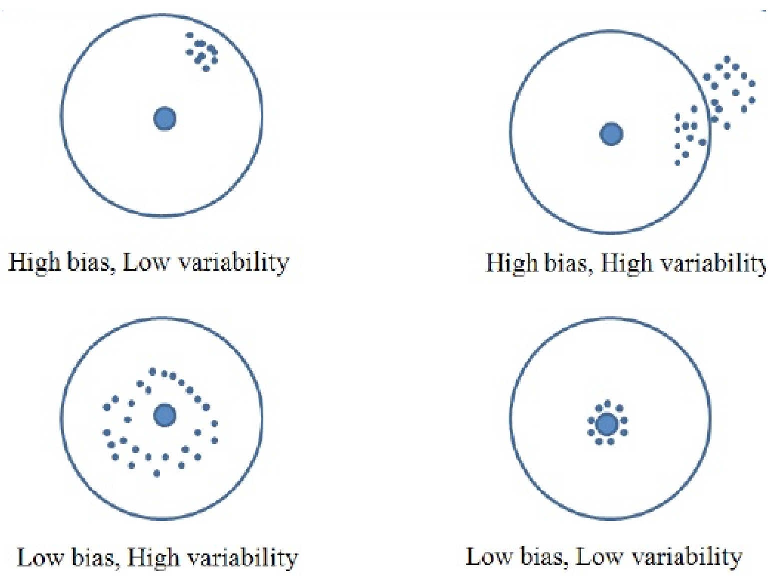
[1]http://www.businessdictionary.com/definition/regression.html

What is Bias?

Bias is derived from the ancient Greek word that describes an oblique line (i.e., a deviation from the horizontal). In Data Science, bias is a deviation from expectation in the data.

- Several definitions, common usage is decidedly negative.
- Systematic favoritism of a group
- Dataset Bias: Sampling bias in a given dataset, give rise to (often unintentional) wrong results when is not representative of the population
- Social Bias in a Dataset: Humans can introduce bias during data collection.

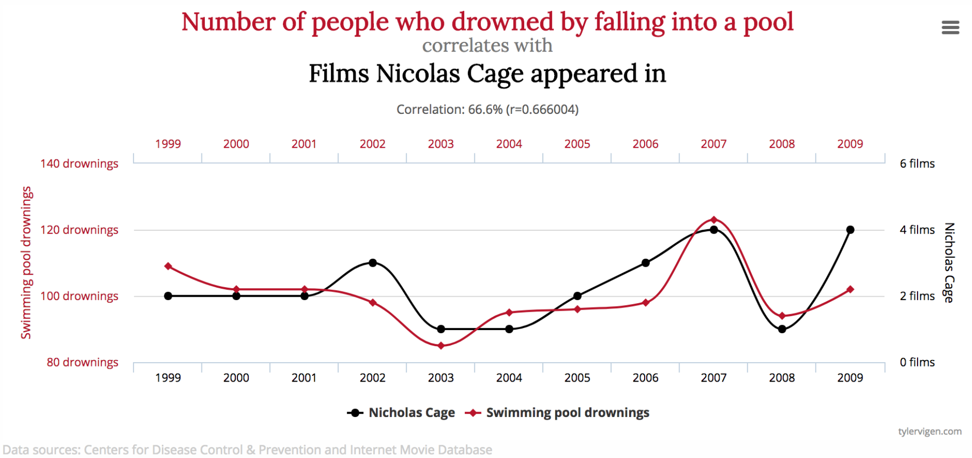
Bias vs Variability?



Supervised Learning Regression

Spurious Correlations

Is a relationship between two variables that appear to have interdependence or association with each other but actually do not.

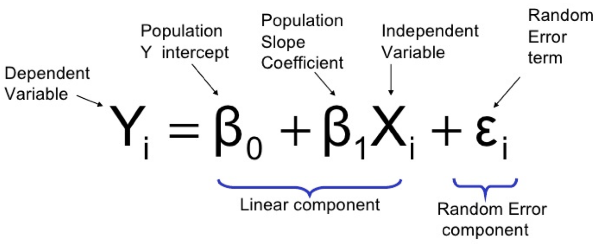


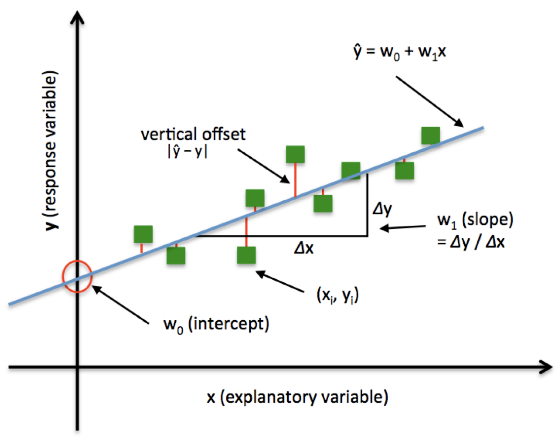
http://www.tylervigen.com/spurious-correlations

Supervised Learning Regression

Linear Model Representation: Statistics vs Machine Learning

$$\underbrace{Y_i = \beta_0 + \beta_1 X_i}_{\text{Linear component}} + \underbrace{\varepsilon_i}_{\text{Random Error component}}$$





Statistics

- β (Coefficient)
- Y = Dependent Variable
- X = Independent Variable

Machine Learning

- w (Weight)
- Y = Target/Response Feature
- X = Input Feature

Univariate Statistics

Univariate Statistics

Basics univariate statistics are required to explore a dataset:

- Discover associations between a variable of interest and potential predictors. It is strongly recommended to start with simple univariate methods before moving to complex multivariate predictors.
- Assess the prediction performances of machine learning predictors.
- Most of the univariate statistics are based on the linear model which is one of the main model in machine learning.
- Check every estimator of the main statistical measures

Univariate Statistics

Mean

Properties of the expected value operator $E()$ of a random variable X

$$E(X + c) = E(X) + c \tag{1}$$
$$E(X + Y) = E(X) + E(Y) \tag{2}$$
$$E(aX) = aE(X) \tag{3}$$

The estimator $\hat{X}$ on a sample of size $n : x = x_1, \ldots, x_n$ is given by

$$\hat{X} = \frac{1}{n} \sum_i x_i$$

$\hat{x}$ is itself a random variable with properties:

$E(\hat{X}) = \hat{X}$ $\qquad$ $Var(\hat{X}) \frac{Var(X)}{n}$

BISS Institute

Univariate Statistics

Variance

$$Var(X) = E((X - E(X))^2) = E(X^2) - (E(X))^2$$

The estimator is:

$$\sigma_x^2 = \frac{1}{n-1} \sum_i (x_i - \hat{x})^2$$

Statistical Bias

$$Bias(y, \hat{y}) = E((y - \hat{y})2) - Var(\hat{y})$$

Where MSE = $E((y - \hat{y})2)$

BISS Institute

Covariance

$$Cov(X,Y) = E((X - E(X))(Y - E(Y)) = E(XY) - E(X)E(Y)$$

Properties:

$$Cov(X,X) = Var(X) \tag{4}$$
$$Cov(X,Y) = Cov(Y,X) \tag{5}$$
$$Cov(cX,Y) = cCov(X,Y) \tag{6}$$
$$Cov(X+c,Y) = Cov(X,Y) \tag{7}$$

The estimator with $df = 1$ is

$$\sigma_{xy} = \frac{1}{n-1}\sum_i (x_i - \hat{x})(y_i - \hat{y})$$

Correlation

$$Cor(X,Y) = \frac{Cov(X,Y)}{Std(X)Std(Y)}$$

The estimator is:

$$\rho_{xy} = \frac{\sigma_{xy}}{\sigma_x \sigma_y}$$

Standard Error (SE)

As well as the standard deviation (of the sampling distribution) of a statistic:

$$SE(X) = \frac{Std(X)}{\sqrt{n}}$$

Commonly considered for the mean with the estimator: $SE(\hat{x}) = \sigma_x / \sqrt{n}$

Univariate Statistics

Testing pairwise associations

Exploring association between pairs of variables.

- **Categorical variable**: Is a variable that can take on one of a limited, and usually fixed, number of possible values, thus assigning each individual to a particular group or category. The levels are the possibles values of the variable (also known as a factor).
- **Ordinal variable**: Is a categorical variable with a clear ordering of the levels. (none, small, medium and high).
- **Continuous variable**: $x \in R$ is one that can take any value in a range of possible values, possibly infinite (salary, experience in years, weight) also known as quantitative.

Univariate Statistics

Pearson correlation test: Association between two quantitative variables

The test calculates a Pearson correlation coefficient and the *p*-value for testing non-correlation.
Let *x* and *y* two quantitative variables, where samples were obeserved. The linear correlation coeficient is defined as :

$$r = \frac{\sum_{i=1}^{n}(x_i - \hat{x})(y_i - \hat{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \hat{x})^2 \sum_{i=1}^{n}(y_i - \hat{y})^2}}$$

Under H_0 the test statistic $t = \sqrt{n-2}\frac{r}{\sqrt{1-r^2}}$ follow Student distribution with $n-2$ degrees of freedom

Correlation test with Python

```
import numpy as np
import scipy.stats as stats
n = 50
x = np.random.normal(size=n)
y = 2 * x + np.random.normal(size=n)

#Parametric Pearson correlation
cor, pval = stats.pearsonr(x, y)
print("Pearson r, cor: %.4f, pval: %.4f" %(cor, pval))

# Non-Parametric Spearman correlation
cor, pval = stats.spearmanr(x, y)
print("Spearman r, cor: %.4f, pval: %.4f" %(cor, pval))
```


Independency T-test with Python

```
import numpy as np
import scipy.stats as stats

#Simulate data
nx, ny = 50, 25
x = np.random.normal(loc=1.76, scale=0.1, size=nx)
y = np.random.normal(loc=1.70, scale=0.12, size=ny)

#t-Test Independency
tval, pval = stats.ttest_ind(x, y, equal_var=True)
print("t-Test, tval: %.4f, pval: %.4f" % (tval, pval))
```

Linear Model

Linear Model

Given n random samples $(y_i, x_i^1, ..., x_i^p), i = 1, ...n$, the linear regression models the relation between the observations and the independent variables x_i^p is formulated as

$$y_i = \beta_0 + \beta_1 x_i^1 + ... + \beta_p x_i^p + \epsilon_i \qquad i = 1, ..., n$$

- **An independent variable**. It is a variable that stands alone and is not changed by the other variables you are trying to measure. In Machine Learning, these variables are also called the predictors or input features.
- **A dependent variable**. It is something that depends on other factors. Usually when you are looking for a relationship between two things you are trying to find out what makes the dependent variable change the way it does. In Machine Learning this variable is called a target variable.

Linear Model

Case: House Price Problem

How much is my house worth?

Let's take a look at sales in my neighbourhood, How much did they sell for?

Linear Model

Case: House Price Problem

Data *input* *output*

(x_1 = sq.ft., y_1 = $)

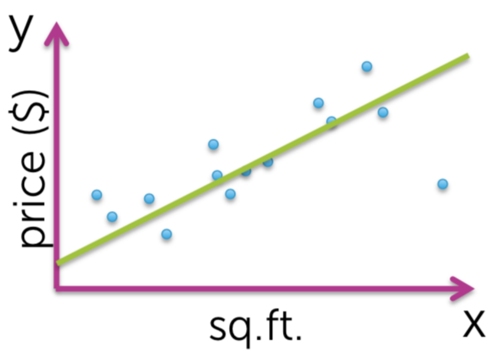
(x_2 = sq.ft., y_2 = $)

(x_3 = sq.ft., y_3 = $)

(x_4 = sq.ft., y_4 = $)

(x_5 = sq.ft., y_5 = $)

⋮



Linear Model

Case: House Price Problem

- Go to **simple regression** notebook
- Load the data of houses **housesdata.csv**
- Choose **price** as target variable
- Choose **sq.ft.** as predictor
- Answer the questions

Linear Model

Simple Linear regression

In simple linear regression, we attempt to model the relationship between two variables and it has the form:

$$y = \beta_0 + \beta_1 x + \epsilon$$

- β_1: the slope or coefficient or parameter of the model,
- β_0: the **intercept or bias** is the second parameter of the model,
- ϵ_i: the *i*th error, or residual with $\epsilon(0, \sigma^2)$.

The simple regression is equivalent to the Pearson correlation.

Linear Model

Simple regression

Regression Estimators The goal is to estimate, β, and β_0. We have to minimize the mean squared error (MSE) or the Sum squared error (SSE). The so-called Ordinary Least Squares (OLS)

$$\hat{\beta} = \frac{\frac{1}{n}\sum_i x_i y_i - \bar{y}\bar{x}}{\frac{1}{n}\sum_i (x_i \bar{x})} = \frac{Cov(x, y)}{Var(x)} = \frac{(\bar{xy}) - (\bar{x})(\bar{y})}{(\bar{x^2}) - (\bar{x})^2}$$

$$\hat{\beta}_0 = E(y) - \beta E(x) = \bar{y} - \beta\bar{x}$$

Fitting

$$\hat{Y} = \hat{\beta}_0 + \hat{\beta}x$$

Linear Model


```python
def simple_linear_regression(x, y):
    xy = x * y #create xy variable
    mean_xy = xy.mean()
    mean_x = x.mean()
    mean_y = y.mean()
    xx = x * x # create xx variable
    mean_xx =  xx.mean()
    x_sqr = mean_x * mean_x
    #compute slope
    cov_xy = mean_xy - mean_x * mean_y
    var_x = mean_xx - x_sqr
    slope = cov_xy / var_x
    #use the slope for the intercept
    intercept = mean_y - slope * mean_x
    return intercept, slope
```

Linear Model

Goodness of fit

The goodness of fit of a statistical model describes how well it fits a set of observations. Measures of goodness of fit typically summarize the discrepancy between observed values and the values expected under the model in question.

- Residual Sum of Squares

$$RSS = \sum_{i=1}^{n}(y_i - \hat{y})^2$$

- Root Mean Squared Error

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y})^2}$$

Multiple Regression

Multiple Regression

Muliple Linear Regression is the most basic supervised learning algorithm. Given: a set of training data $x_1, \ldots, x_n$ with corresponding targets $y_1, \ldots, y_n$. In linear regression, we assume that the model that generates the data involves only a linear combination of the input variables, i.e

$$y(x_i, \beta) = \beta^0 + \beta^1 x_i^1 + \ldots + \beta^P x_i^P,$$

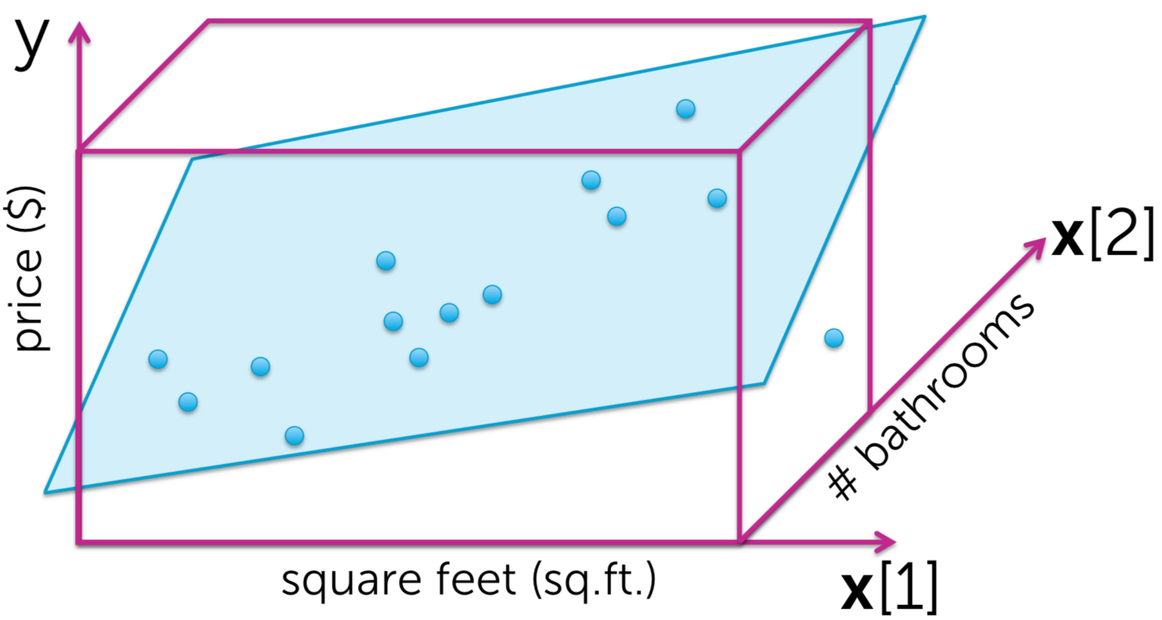
Extending each sample with an intercept $x_i := [1, x_i] \in R^{P+1}$ allows us to use a more general notation based on linear algebra and write it as a simple dot product:

$$y(x_i, \beta) = x_i^T \beta,$$

Where $\beta \in R^{P+1}$ is a vector of weights that define the $P+1$ parameters of the model. From now we have P regressors + the intercept.

Multiple regression

- Linear regression with multiple features
- More complex functions of a single input



Multiple regression

Case: House Price Problem

- Go to **multiple regression** notebook
- Load the data of houses **housesdata.csv**
- Check the turicreate API
- Run the multiple regression example
- Create the new features
- Learn multiple models proposed

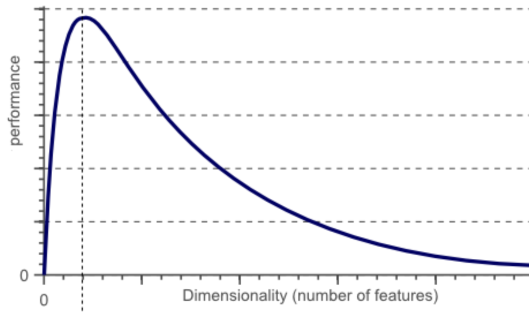
Overfitting and Complexity

Overfitting and Complexity

In statistics and machine learning, overfitting occurs when a statistical model describes random errors or noise instead of the underlying relationships. A model that has been overfitted will generally have poor predictive performance, as it can exaggerate minor fluctuations in the data.

The overfitting phenomenon has three main explanations:

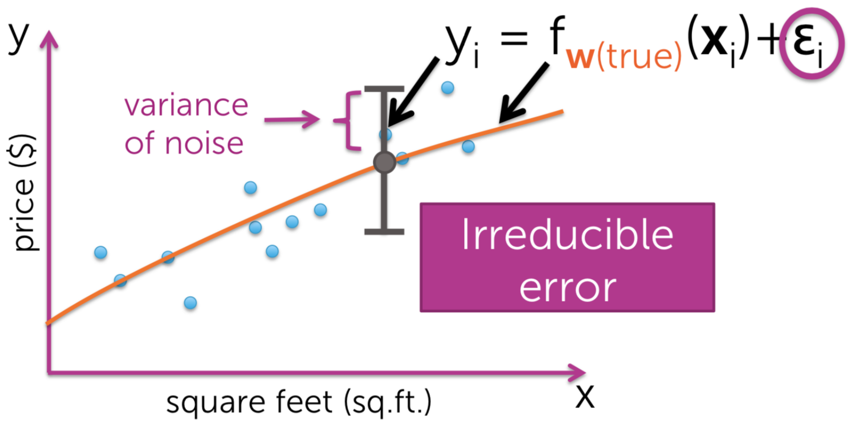
- Excessively complex models
- Multicollinearity
- High dimensionality



Overfitting and Complexity

3 sources of error

- Noise
- Bias
- Variance



MSE in terms of Bias and Variance

$$MSE = Var(\hat{y}) + E((y - \hat{y})2) = \frac{1}{n-1}\sum_i (y_i - \hat{y})^2 + \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y})^2$$

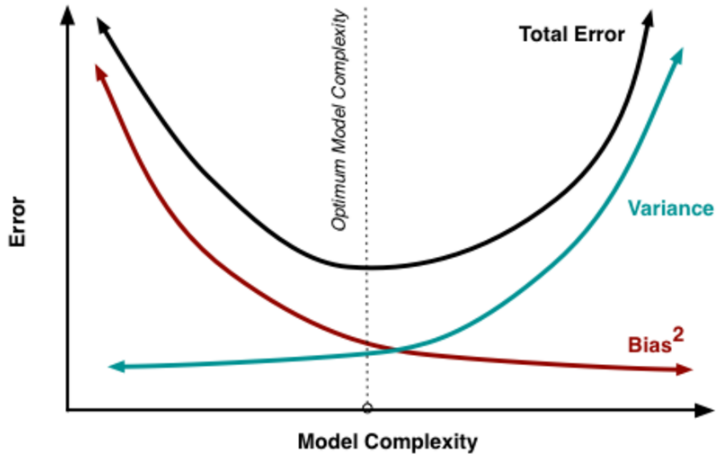


Image: http://scott.fortmann-roe.com/docs/MeasuringError.html

Regularization Methods

Regularization Methods

Ridge regression(L_2-regularization)

Overfitting generally leads to excessively complex weight vectors, accounting for noise or spurious correlations within predictors. To avoid this phenomenon the learning should constrain the solution in order to fit a global pattern. This constraint will reduce (bias). Adding such a penalty will force the coefficients to be small, i.e. to shrink them toward zeros.

$$Ridge(\beta) = ||y - \mathbf{X}\beta||_2^2 + \lambda||\beta||_2^2$$

- Increasing shrinks the coefficients toward 0.
- Penalizes the objective function by the Euclidian norm of the coefficients such that solutions with large coefficients become unattractive.

Regularization Methods

Lasso regression (L_1-regularization)

Lasso regression penalizes the coefficients by the âĎŞ1 norm. This constraint will reduce (bias) the capacity of the learning algorithm. To add such a penalty forces the coefficients to be small, i.e. it shrinks them toward zero. The objective function to minimize becomes:

$$Lasso(\beta) = ||y - \mathbf{X}\beta||_2^2 + \lambda||\beta||_1$$

- Choose the model with the smallest coefficient vector, i.e. smallest $L2$ or $L1$ norm
- Choose the model that uses the smallest number of predictors. In other words, choose the model that has many predictors with zero weights.

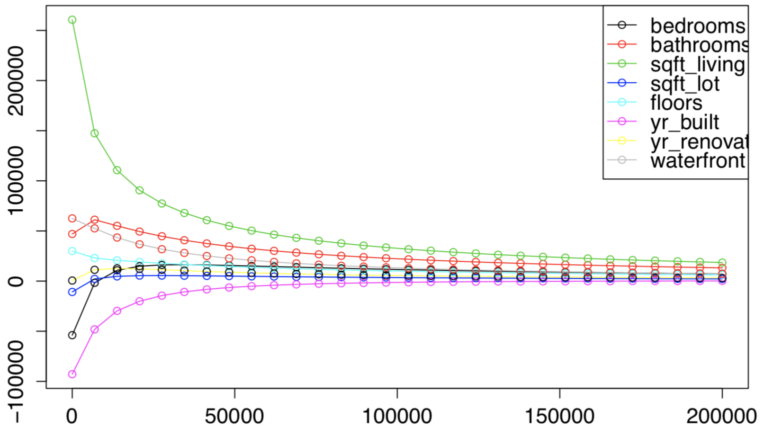
Regularization Methods

Elastic-net regression ($L_2 - L_1$-regularization)

The Elastic-net estimator combines the $L1$ and $L2$ penalties, and results in the problem to:

$$Enet(\beta)||y - \mathbf{X}^T\beta||_2^2 + \alpha(\rho||\beta||_1 + (1-\rho)||\beta||_2^2, \tag{8}$$

where α acts as a global penalty and ρ as an $L1/L2$ ratio.



Regularization Methods

Summary Regularization

- Powerful techniques generally used for creating parsimonious models in presence of a large number of features.
- Large enough to enhance the tendency of a model to overfit (as low as 10 variables might cause overfitting).
- Large enough to cause computational challenges. With modern systems,(millions or billions of features).
- If a feature is too large, means that we are putting a lot of emphasis on that feature.