NYC DATA SCIENCE ACADEMY

# Python Machine Learning Class 1

Data Science Bootcamp

# Outline

- What is Machine Learning
- Introduction to Scikit-Learn
- Simple Linear Regression
  - Estimating Coefficients
  - Coefficient of Determination
- Multiple Linear Regression
- Statsmodels

# What is Machine Learning?

- **Task**: recognize a tree
- **Problem**:
  - It's hard to write a program to do this.
- **Solution**:
  - Learn from data (observations).
- ML-based tree recognition systems can be much more effective than hand-programmed systems.

# What is Machine Learning?

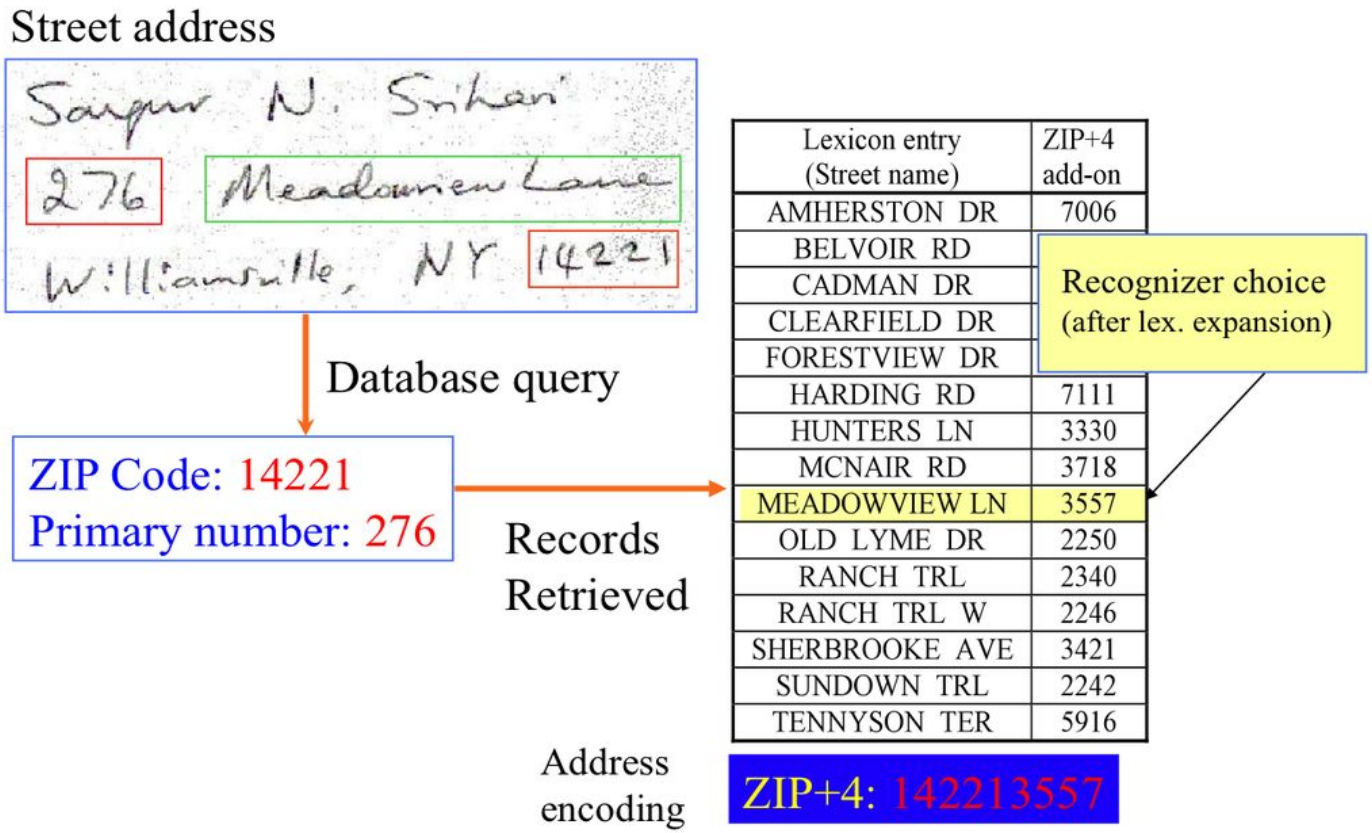
- Machine learning is a subfield of computer science that provides computers with the ability to learn without being explicitly programmed.
- The machine learning paradigm can be viewed as “programming by example”.
  - The learning is always based on some sort of observations or data.
  - The goal is to devise learning algorithms that do the learning automatically without human assistance.

# What can Machine Learning do?

- Applications that can't be programmed by hand:
  - Autonomous helicopters, handwriting recognition, most Natural Language Processing (NLP), Computer Vision.
- Self-customizing programs:
  - Amazon, Netflix product recommendations
- Database mining:
  - Large data sets from growth of automation/web. Web click data, medical records, biology, engineering...

# What can Machine Learning do?

- Handwriting Recognition: address recognition



# What can Machine Learning do?

- Amazon Recommendation

# Machine Learning Terminology

- Variables that are considered to be measurable or present, and have some influence on other variables, are referred as **inputs**, **predictors**, **features**, or **independent variables**.
- Variables that are assumed to be influenced by others and that we want to predict, are called **outputs**, **responses**, or **dependent variables**.
- For example, to predict how smoking, together with other variables, would affect lung cancer rate, smoking is the *predictor* and lung cancer is the *response*.

# Machine Learning Problems

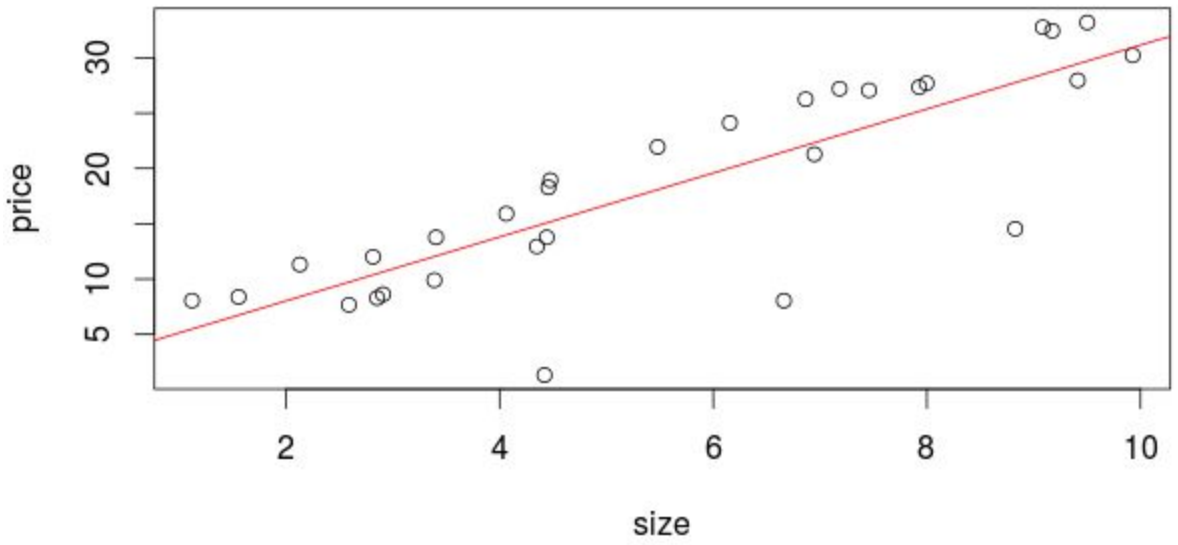
- There are two main kinds of machine learning problems:
  - *Supervised Learning*:
    - Predict outcome measurement *Y* using predictor measurements *X*.
    - In a *regression* problem, *Y* is quantitative (e.g., price).
    - In a *classification* problem, *Y* is categorical (digit 0-9).
  - *Unsupervised Learning*:
    - No outcome variables, inputs are often called *features*.
    - Objective is more fuzzy - e.g., find groups of people that behave similarly.

# Machine Learning Problems

- There are other kinds of learning problems, like reinforcement learning and recommendation systems.  We will not be covering those.
- The next few slides give examples of the main learning problems.

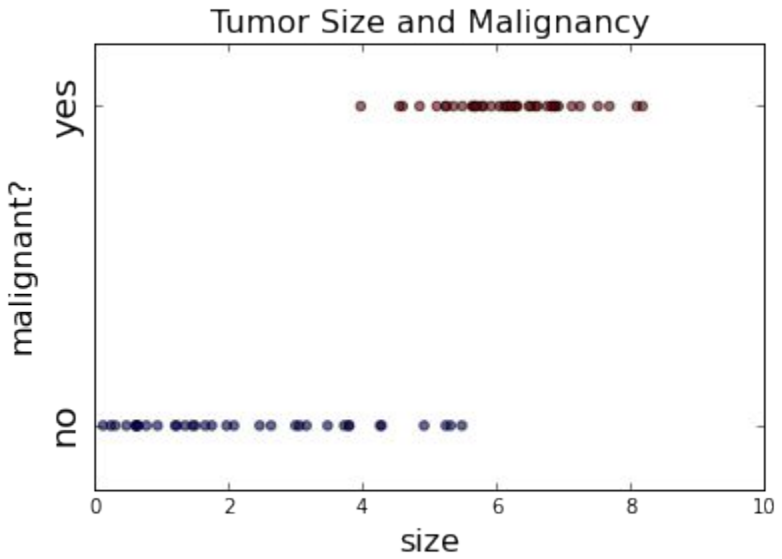
# Supervised Learning: regression

- Regression problem: predict continuous output.
- Example: You have a data set of house sizes and prices; given a new house's size, predict its price.



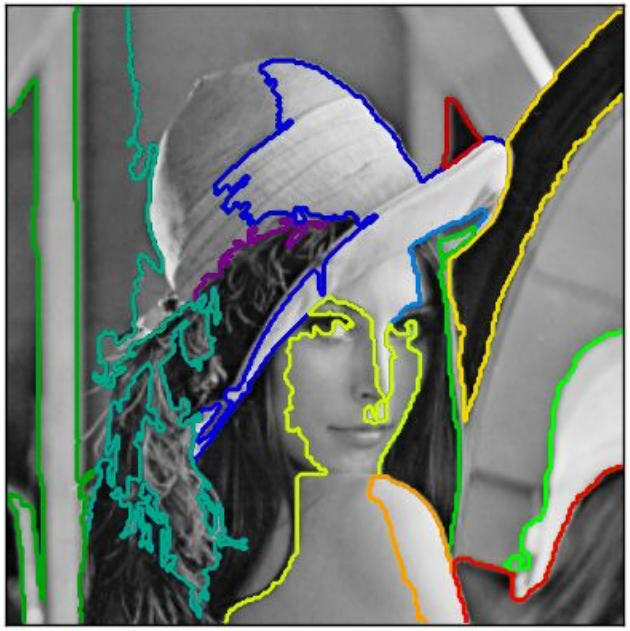
# Supervised Learning: classification

- Classification problem: predict discrete output.
- Example: classify whether a tumor is malignant or not by its size.



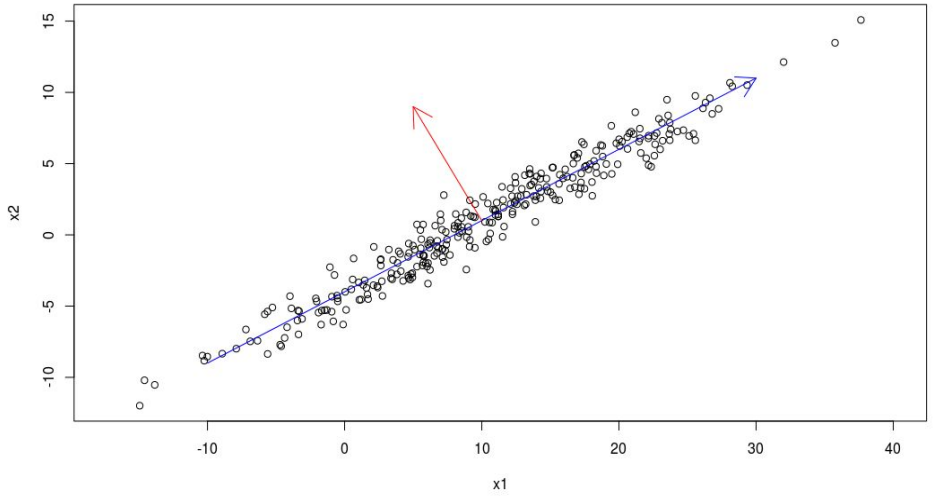
# Unsupervised Learning: clustering

- Given some features, cluster the observations into groups.
- Example: segmenting an image in regions based on voxel-to-voxel similarity using hierarchical clustering.



# Unsupervised Learning: dimension reduction

- Reduce the number of features without losing much information.
  - This is usually a required preprocessing step for supervised learning when the number of features is large or they are highly correlated.
- Example: These two variables are highly correlated. They can be converted into one combined variable without losing much information using Principal component analysis (PCA).



## Questions

- Which of the following problems are best suited to machine learning?
  - (a) Classifying numbers into primes and non-primes.
  - (b) Detecting potential fraud in credit card charges.
  - (c) Determining the time it would take a falling object to hit the ground.
  - (d) Determining the optimal cycle for traffic lights in a busy intersection.
  - (e) Determining the age at which a particular medical test is recommended.

## Questions

- Which of the following problems are best suited for machine learning?
  - (a) Classifying numbers into primes and non-primes
  - **(b) Detecting potential fraud in credit card charges**
  - (c) Determining the time it would take a falling object to hit the ground
  - **(d) Determining the optimal cycle for traffic lights in a busy intersection**
  - **(e) Determining the age at which a particular medical test is recommended**

## Questions

- Identify which type of learning can be used to solve each task below:
  - Categorize books into groups, knowing only what books were bought by each customer.
  - Decide the maximum allowed debt for a bank customer.

## Questions

- Identify which type of learning can be used to solve each task below:
  - Categorize books into groups, knowing only what books were bought by every customer.
  - Deciding the maximum allowed debt for each bank customer.
- The first is unsupervised learning. We have no "correct answers" to "supervise" our learning. More precisely, it is a clustering problem.
- The second is a supervised learning problem, more precisely, a regression problem.

# Outline

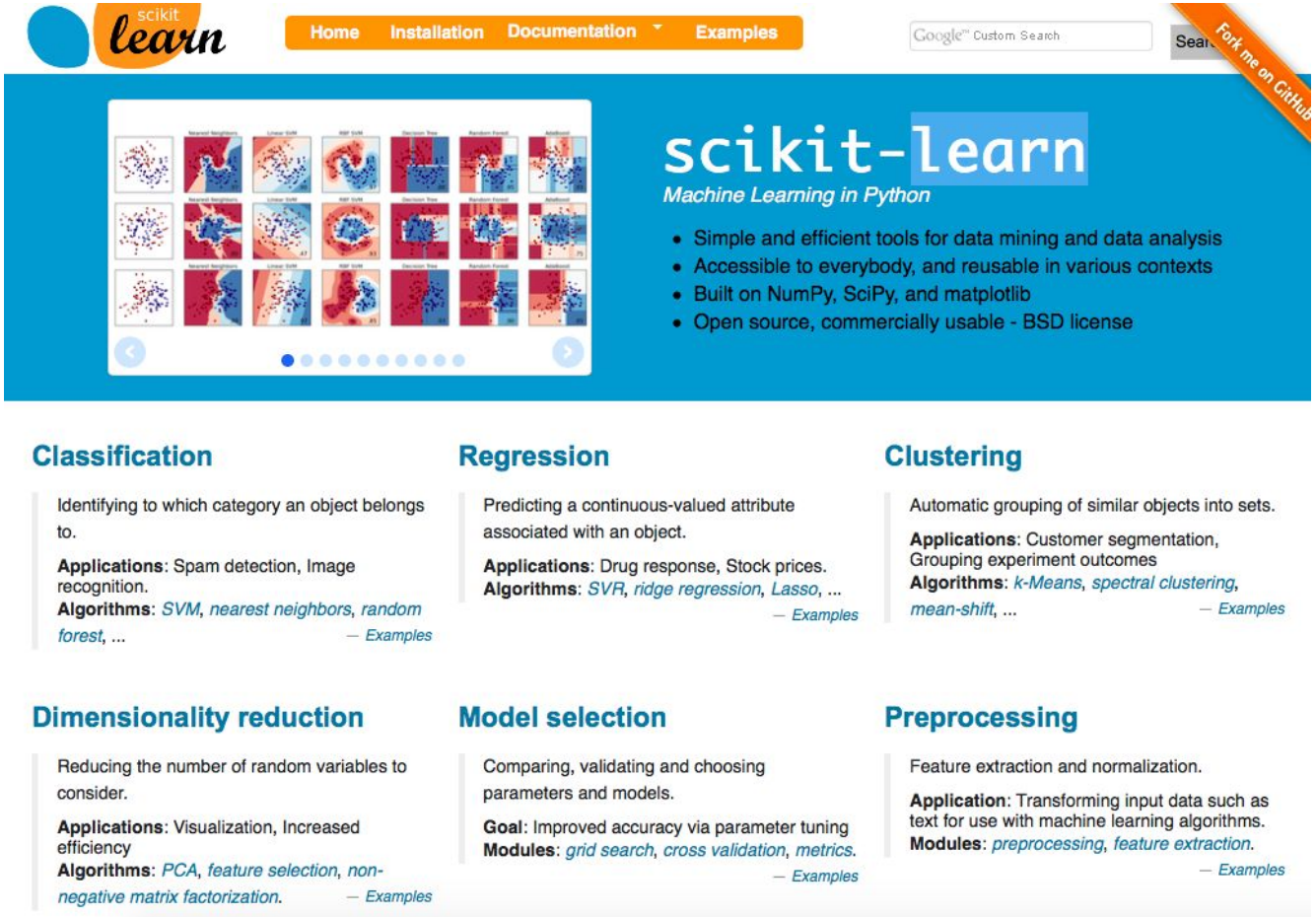
- What is Machine Learning
- **Introduction to Scikit-Learn**
- Simple Linear Regression
  - Estimating Coefficients
  - Coefficient of Determination
- Multiple Linear Regression
- Statsmodels

## Overview

- scikit-learn is an open source machine learning library for the Python programming language. It is built on Numpy, Scipy and matplotlib. It is designed to be an efficient tool box for machine learning and data mining. It provides user friendly functions to facilitate:
  - Supervised learning, including regression and classification.
  - Unsupervised learning.
  - Functions to help test your results, choose the correct algorithm and parameters, etc. *cross validation, feature and model selection.* We will cover those topic in later classes.

# Introduction to scikit-learn

- http://scikit-learn.org



# Introduction to scikit-learn

- The learning algorithms that scikit-learn provides:
  - Supervised learning
    - Regression
    - Classification
  - Unsupervised learning
    - Clustering
    - Dimension reduction
- We will focus on linear regression in this lecture.

# Outline

- What is Machine Learning
- Introduction to Scikit-Learn
- **Simple Linear Regression**
  - Estimating Coefficients
  - Coefficient of Determination
- Multiple Linear Regression
- Statsmodels

# Linear Regression

- Linear regression is a supervised machine learning method that aims to uncover the relationship between continuous variables:
  - One or more **explanatory/independent/input** variables: $X_1, X_2, \ldots X_p$
  - The **response/dependent/output** variable $Y$.

# Simple Linear Regression

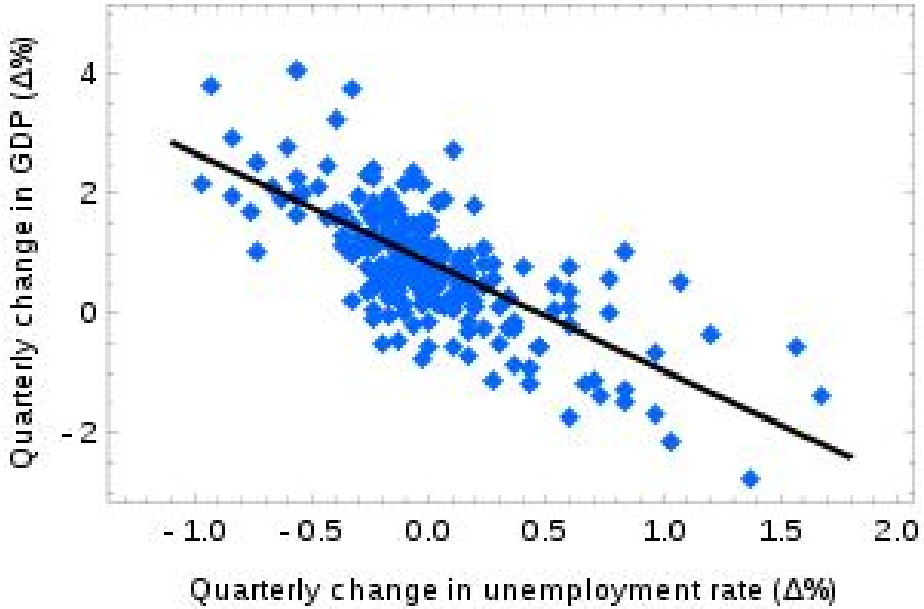
- **Simple linear regression** in a special case when there is only one explanatory variable *X* . Then the relation can be represented quantitatively by:

$$Y = \beta_0 + \beta_1 X + \epsilon$$

  - $\beta_0$ and $\beta_1$ are two unknown constants that represent the intercept and slope.
  - $\epsilon$ is called the **error term**. This represents the deviation of the observed value from the true value.

# Simple Linear Regression

- For example, Okun's law in macroeconomics can be modeled by simple linear regression. Here the GDP growth is presumed to be in a linear relationship with the changes in the unemployment rate.




Source: https://en.wikipedia.org/wiki/Simple_linear_regression

# Simple Linear Regression

- The Okun's law from previous slide can then be modeled as

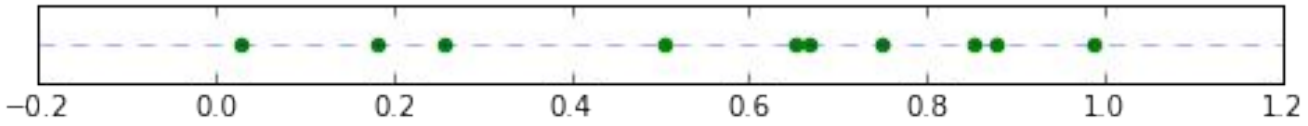
$$GDP = \beta_0 + \beta_1 unemployment\_rate + \epsilon$$

# The Coefficients

- Below we visualize our simple linear model with an example:

$$Y = 1 + 0.5X + \epsilon. \ (\beta_0 = 1 \text{ and } \beta_1 = 0.5)$$

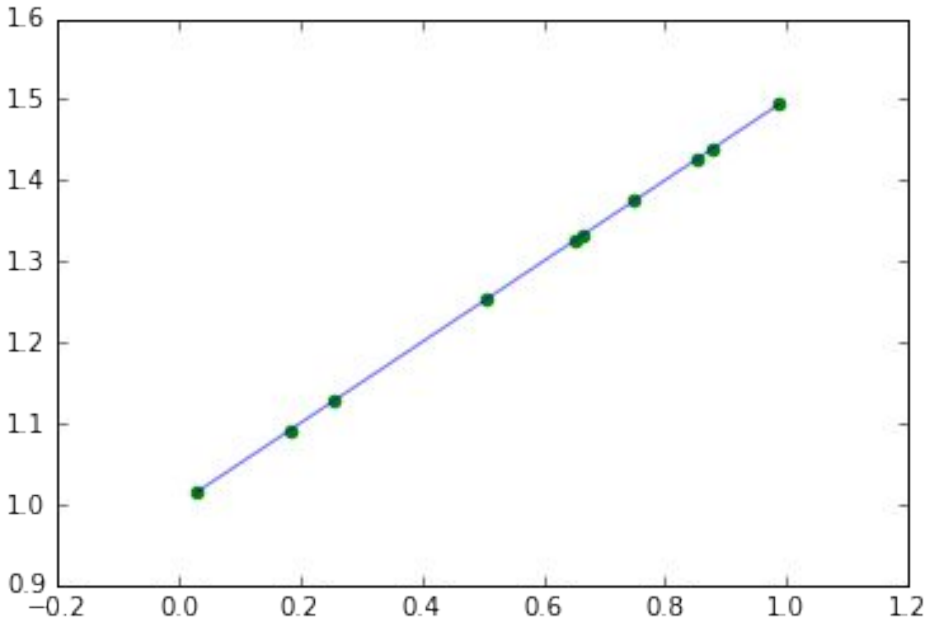
  - $\beta_0$ and $\beta_1$ defines the linear relation. That says, if we observe a set of n independent variables $X = (x_1, x_2, \ldots, x_n)$ :



# The Coefficients

- The linear relation indicates that the outcome $Y = (y_1, y_2, \ldots, y_n)$

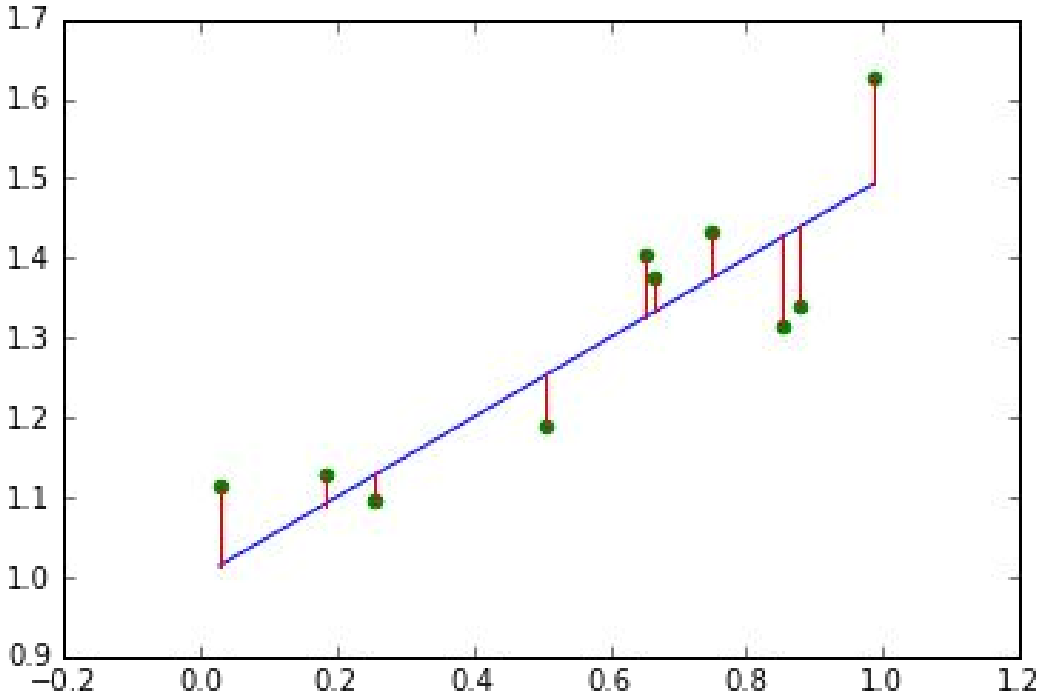
  Should be:




Note that there is **NO** randomness involved.

# The Errors

- All the randomness are attributed to $\epsilon$.
  - The relation $Y = 1 + 0.5X + \epsilon$ becomes



# The Basic Assumptions on Linear Regression

- The basic assumptions of a simple linear model are:
  - Linearity
  - Normality
  - Constant Variance
  - Independent Errors

# The Basic Assumptions on Linear Regression

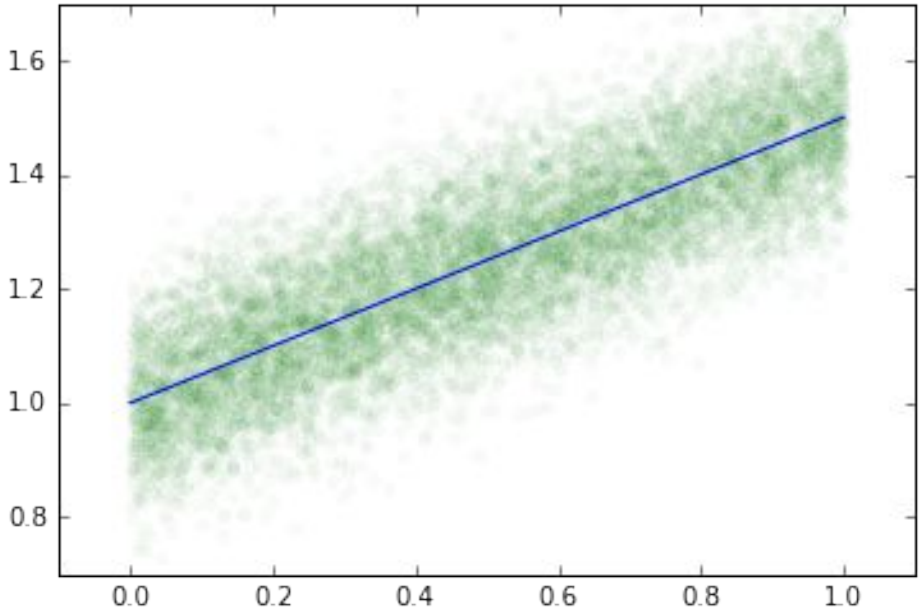
- Linearity:
  - Linearity defines the relation between *X* and *Y*. As we saw in the previous plot, it is represented by $\beta_0$ and $\beta_1$.
- We will discuss how these two constants are estimated.

# The Basic Assumptions on Linear Regression

- Assumptions on the Errors:
  - We **cannot** estimate $\epsilon$ mainly because it is random. However, we can still study some properties of the randomness. The last three (except the linearity) assumptions on linear model describe what kind of randomness $\epsilon$ should be.

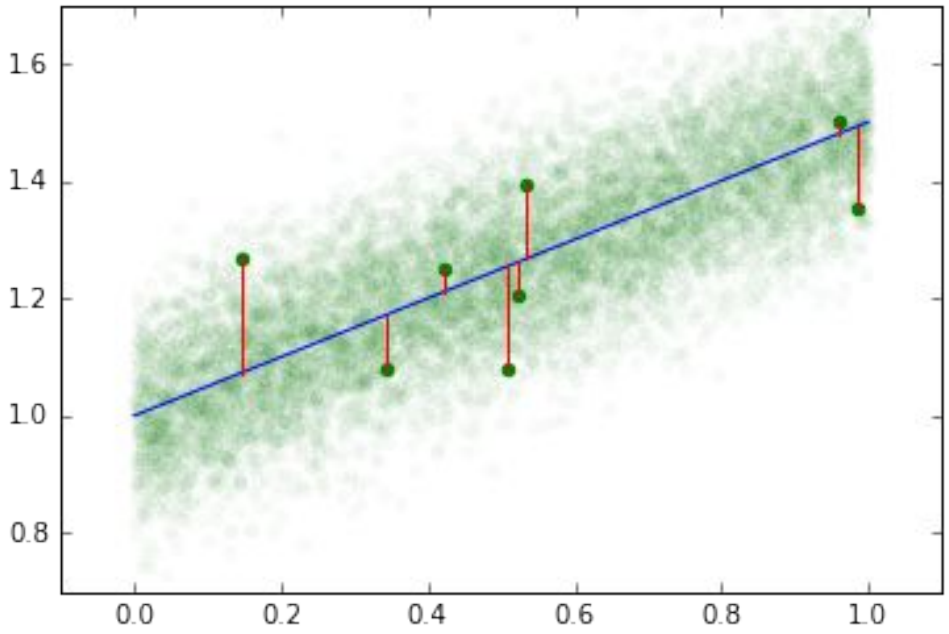
# The Basic Assumptions on Linear Regression

- Normality:
  - Randomness are often described by **distribution**, which can be seen only when we have a lot of samples. So let's create a much larger sample set:



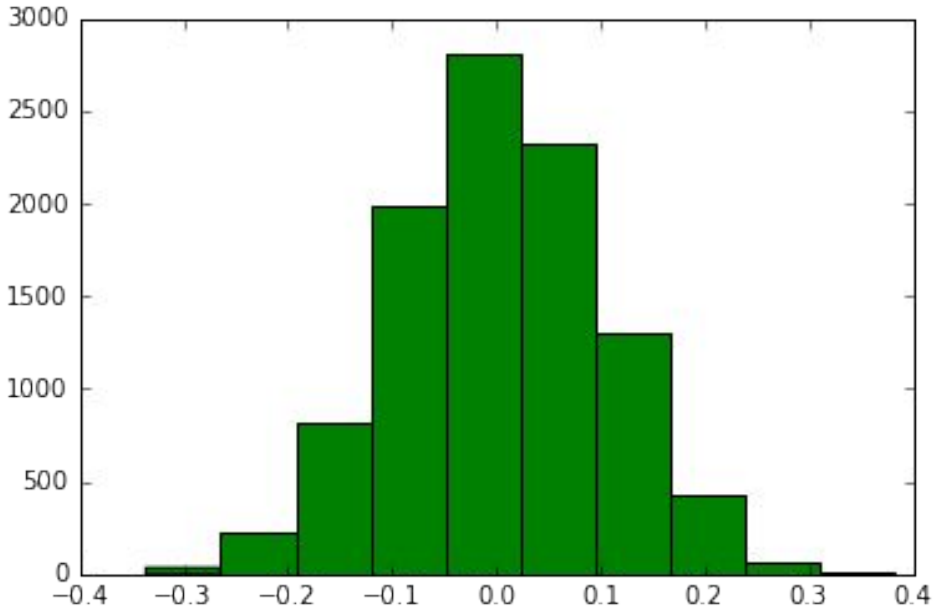
# The Basic Assumptions on Linear Regression

- Normality:
  - And again we visualize the error with some of the *X*:



# The Basic Assumptions on Linear Regression

- Normality:
  - The normality assumption means if we sketch the histogram of the errors, it looks like:

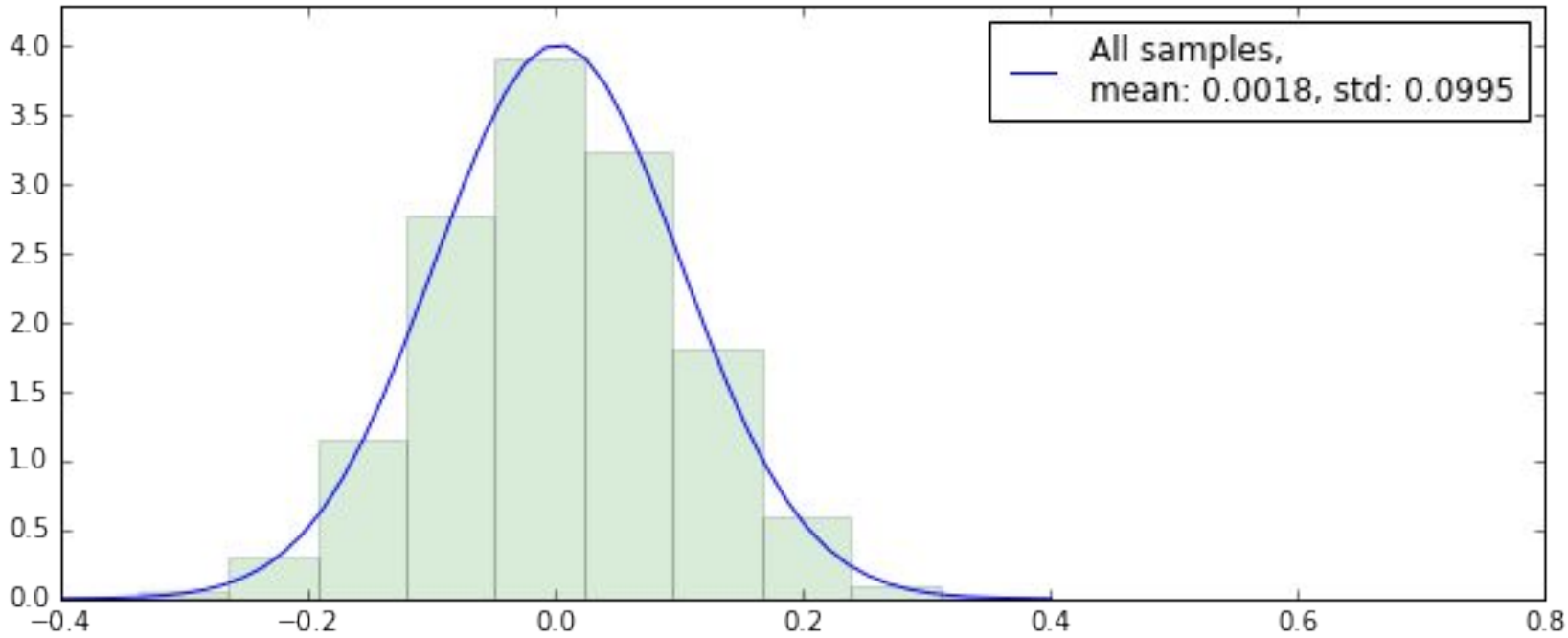


# The Basic Assumptions on Linear Regression

- Normality:
  - Note that our errors have:
    - mean equals to 0.0018
    - standard deviation equals to 0.0995
  - Then we can compare the **normalized** histogram and the pdf curve of a normal distribution.
    - Note the difference between the ***y* axes** of the plot below and of the previous one.

# The Basic Assumptions on Linear Regression

- Normality:



# The Basic Assumptions on Linear Regression

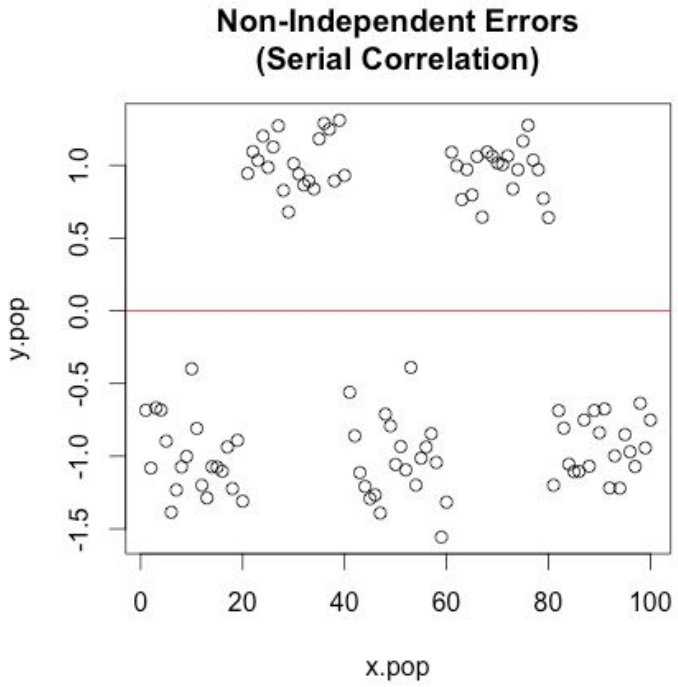
- Constant Variance and Independent Errors
- Both of the other two conditions:
  - **constant variance**
  - **independent errors**

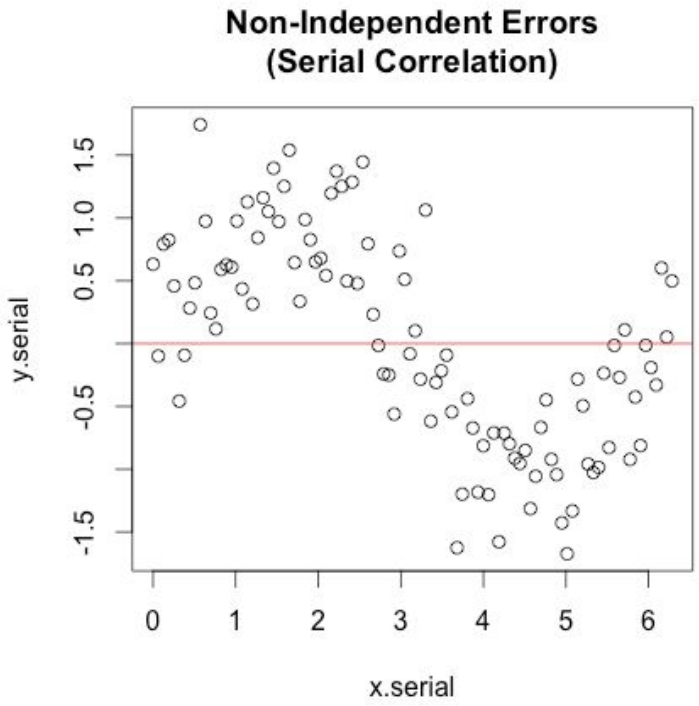
  indicate that the error of every observation obey the same distribution.

***WHY?***

# The Basic Assumptions on Linear Regression

- Some examples of violating independent errors:




- There exists subgroups of errors distributing differently.

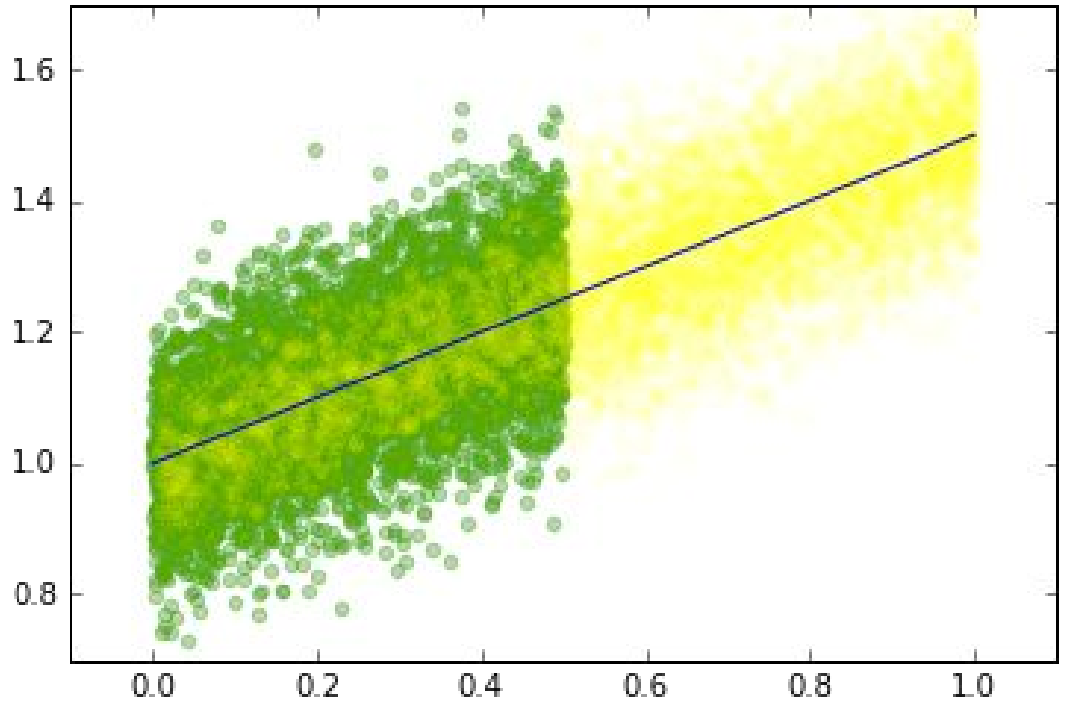
# The Basic Assumptions on Linear Regression

- How do we check that?
  - There is no way to talk about the distribution of one observation. However, if all the errors obey the same distribution, we should obtain the same (or very similar) normal curve when we randomly choose a (large enough) subset from the observations.

- For example, let's pick the observations with X less than 0.5 and compare the normal curve obtained from them with the one obtained from all the observations.

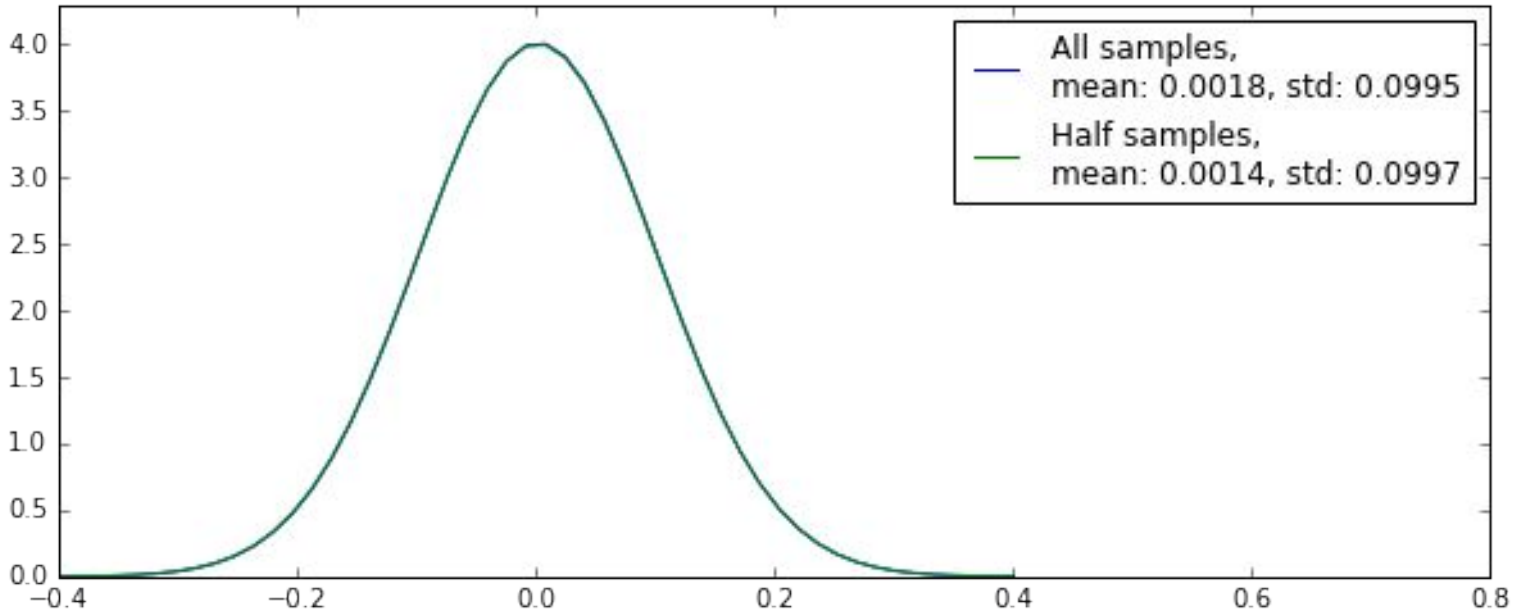
# The Basic Assumptions on Linear Regression

- These are the samples we select:



# The Basic Assumptions on Linear Regression

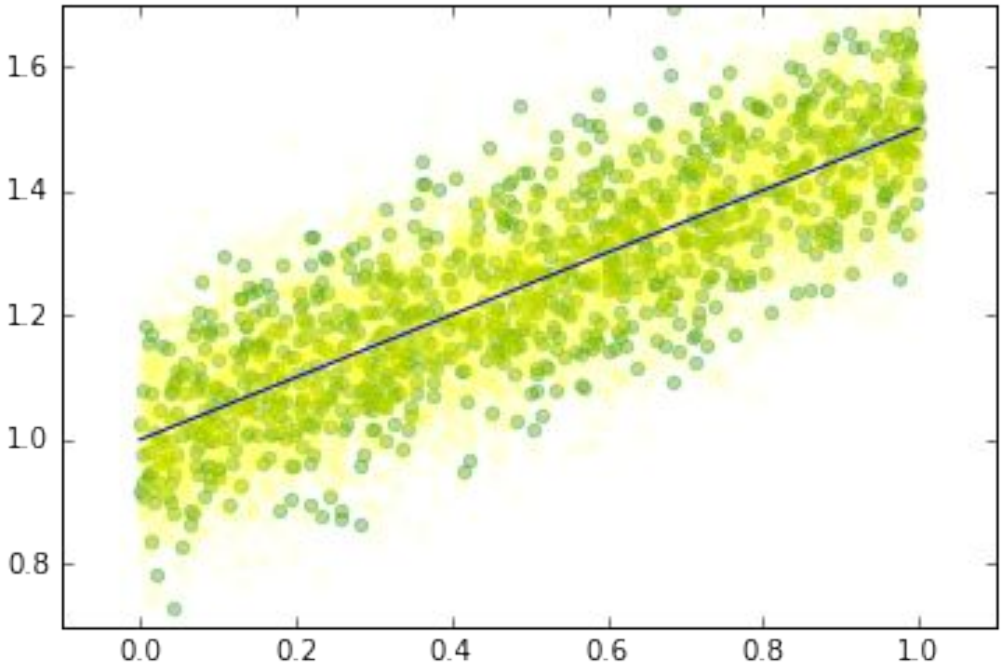
- Then we compare the normal distribution obtained from different groups:




- You can go to the lecture code, change the range selected and compare the normal curves obtained.

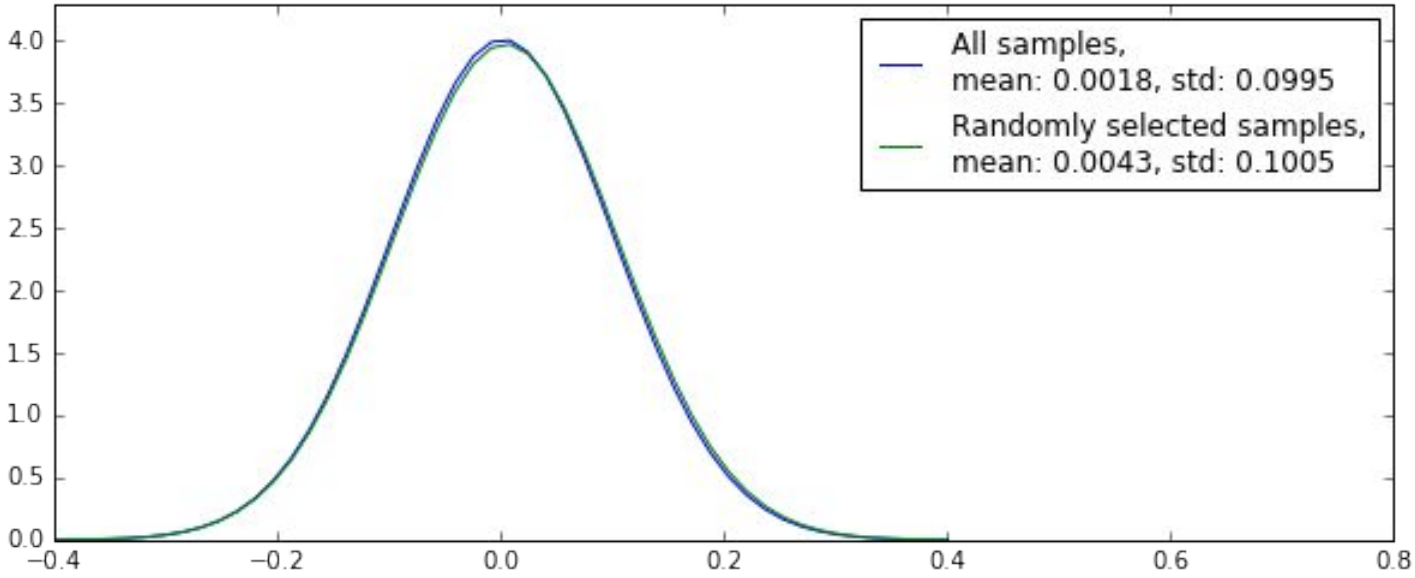
# The Basic Assumptions on Linear Regression

- We may also just randomly select the subset and make the same comparison.



# The Basic Assumptions on Linear Regression

- Then we compare the normal distribution obtained from different groups:



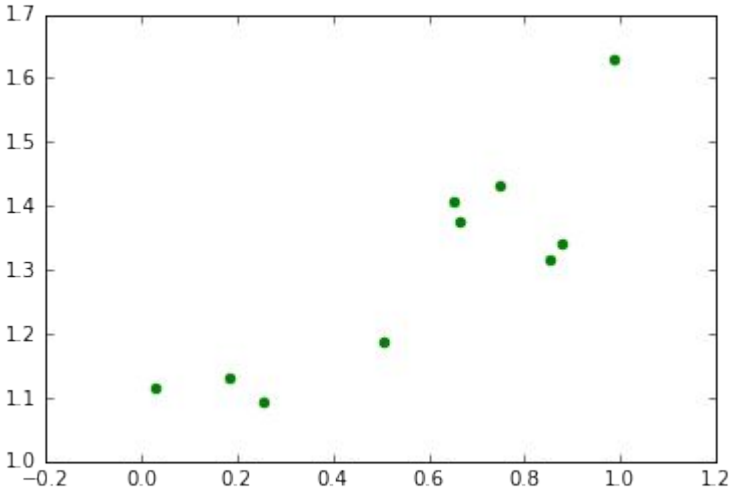

- You can go to the lecture code, change the amount of observations selected and compare the normal curves obtained.

# Outline

- What is Machine Learning
- Introduction to Scikit-Learn
- Simple Linear Regression
  - **Estimating Coefficients**
  - Coefficient of Determination
- Multiple Linear Regression
- Statsmodels

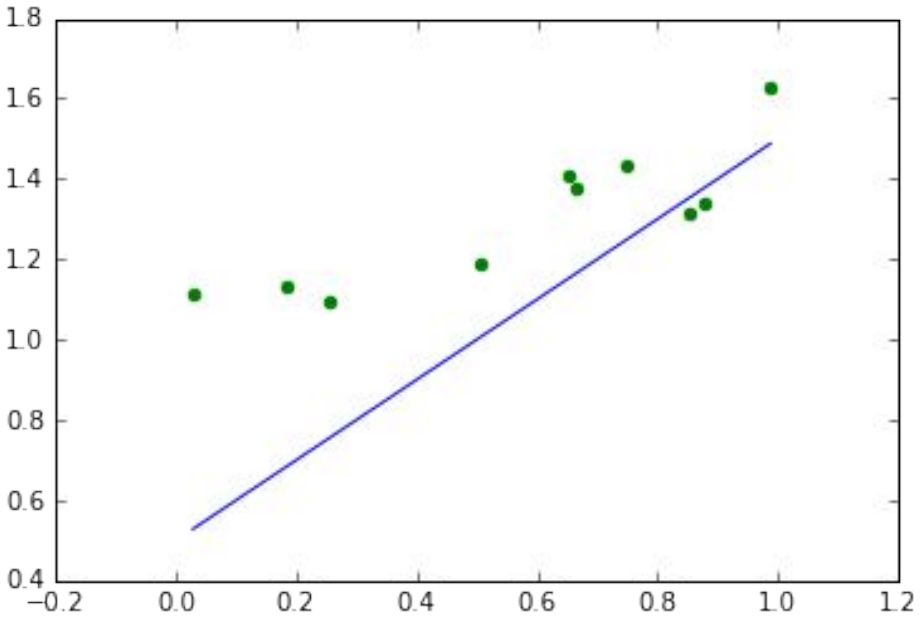
# The Basic Assumptions on Linear Regression

- In general, $\beta_0$ and $\beta_1$ are unknown, what we have is a set of observations X and Y. Essentially what we do is to **try** all the possible pairs of $\beta_0$ and $\beta_1$, and find the one defining the linear model most similar to the observations.
  - We again illustrate the process with visualization.



# Estimating the Coefficients

- We then start trying out some pair of $(\tilde{\beta}_0, \tilde{\beta}_2)$ , say, (0.5, 1):



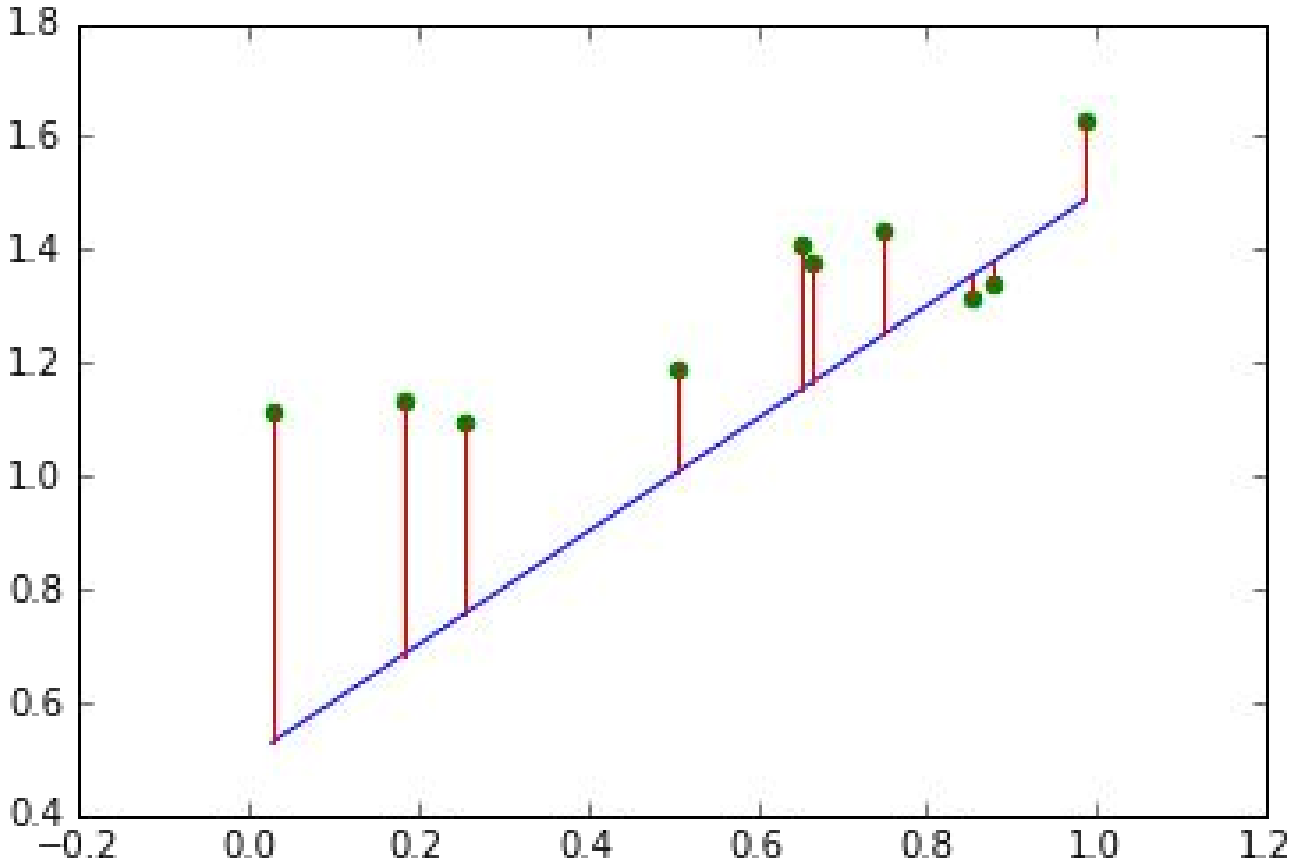
# Estimating the Coefficients

- How different is the model from the observations?
  - We may again consider the difference between the observation and the model:

$$e = Y - (\tilde{\beta}_0 + \tilde{\beta}_1 X)$$

- This difference vector is called the residual.

# Estimating the Coefficients



# Estimating the Coefficients

- To quantify the difference between the model and the observations, we use the residual sum of squares, or RSS. It is defined by:

$$\text{Denote } e = (e_1, e_2, e_3, \ldots, e_n)$$

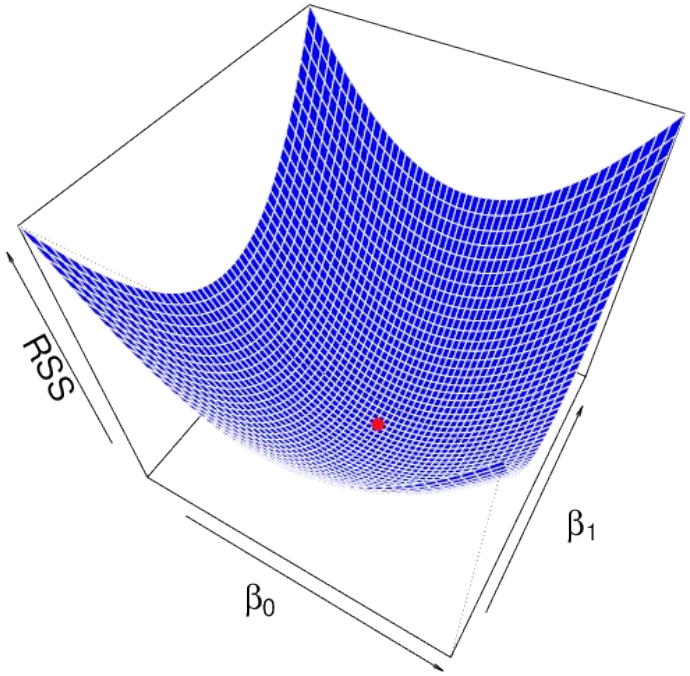
$$RSS(\tilde{\beta}_0, \tilde{\beta}_1) = e_1^2 + e_2^2 + \ldots + e_n^2$$

$$= \sum_{i=1}^{n} (y_i - \tilde{\beta}_0 - \tilde{\beta}_1 x_i)^2$$

- Therefore RSS depends on $(\tilde{\beta}_0, \tilde{\beta}_1)$.

# Simple Linear Regression - RSS and Least Squares

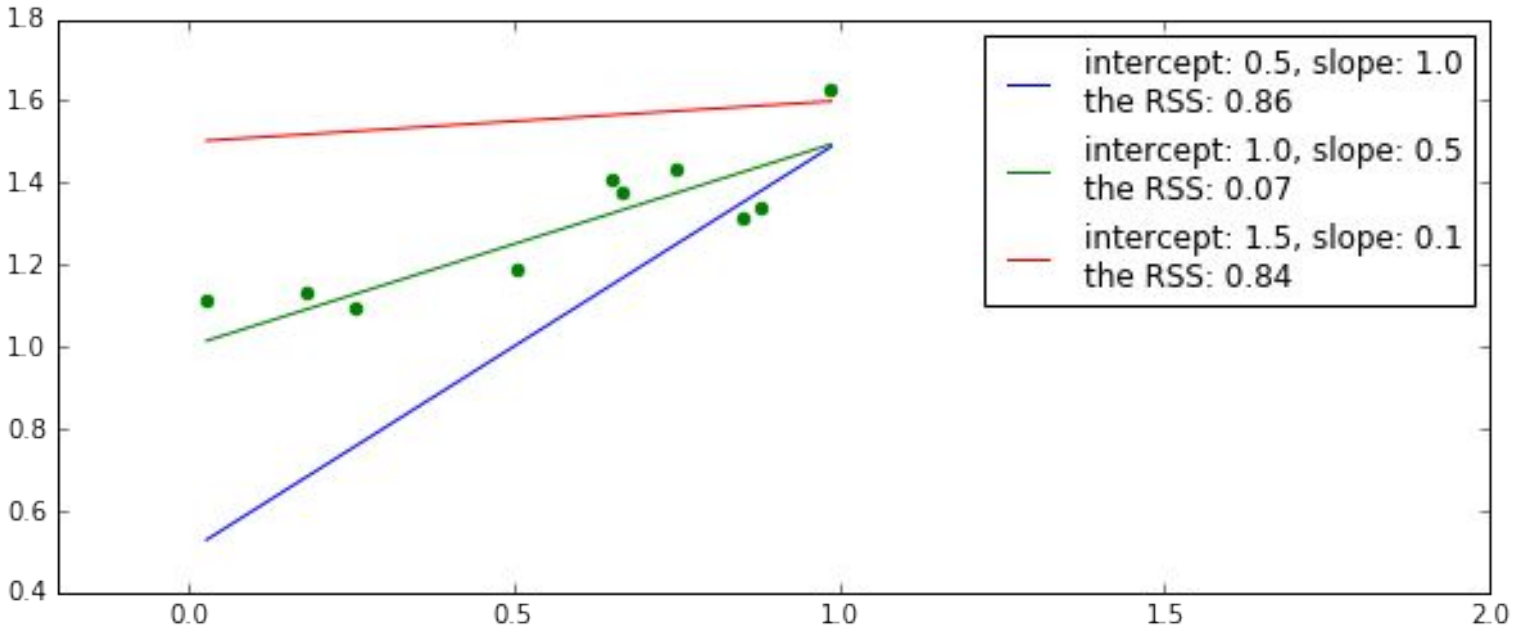
- As you observed, the smaller the RSS, the better the fit.
- RSS is a quadratic function of parameters $\beta_0$ and $\beta_1$.




- Our goal now becomes how to find the minimum point of RSS($\beta$).

# Estimating the Coefficients

- Below we see that indeed the model with least RSS is most similar to the observations.



# Estimating the Coefficients

## QUESTION

- $(\tilde{\beta}_0, \tilde{\beta}_1) = (1, 0.5)$ is the best among three. Is it actually the best possible pair?

- The coefficient that really minimizes RSS is denoted by $(\hat{\beta}_0, \hat{\beta}_1)$, And the model is denoted by:

$$\hat{Y} = \hat{\beta}_0 + \hat{\beta}_1 X$$

- The symbol ^ denotes an estimated value. The coefficients $(\hat{\beta}_0, \hat{\beta}_1)$ are called the **ordinary least square estimator** (**OLS**). Once we have the estimators and a new observed X, the Y can be predicted by passing X into the formula above.

## Estimating the Coefficients

- Minimizing the RSS characterizes the coefficients $(\hat{\beta}_0, \hat{\beta}_1)$. We will not discuss how they can be obtained, though it is actually the standard optimization problem with differentiation. $(\hat{\beta}_0, \hat{\beta}_1)$ actually admits a closed form:

$$\hat{\beta}_1 = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

$$\hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \bar{x}$$

where $\bar{y}$ and $\bar{x}$ are the sample means of $x$ and $y$, respectively.

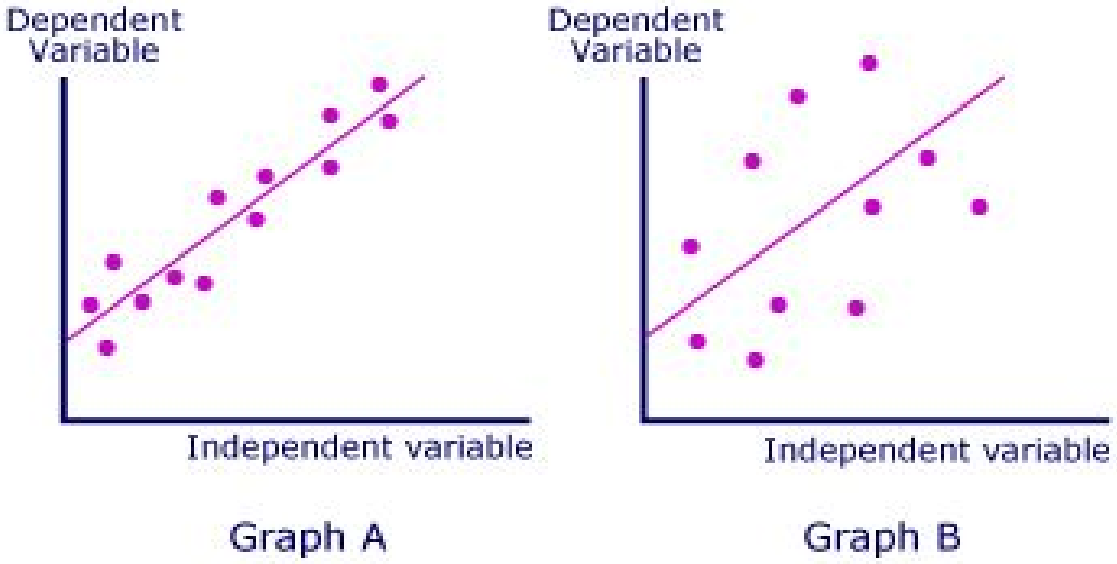
- Of course, if you don't care about math, Python will compute the coefficients for us.

# Outline

- What is Machine Learning
- Introduction to Scikit-Learn
- Simple Linear Regression
  - Estimating Coefficients
  - **Coefficient of Determination**
- Multiple Linear Regression
- Statsmodels

# Coefficient of Determination

- Once we fit a linear model, how should we assess the overall accuracy of the model?
- Think about the two graphs shown below. They have the same fitted parameters, but the left graph shows a higher predictive quality than the right one.



# Coefficient of Determination

- The usual way to measure the overall accuracy of a simple linear model is to use the ***coefficient of determination***.
- The coefficient of determination, denoted $R^2$, measures how well data fits a model.
- $R^2$ is defined as

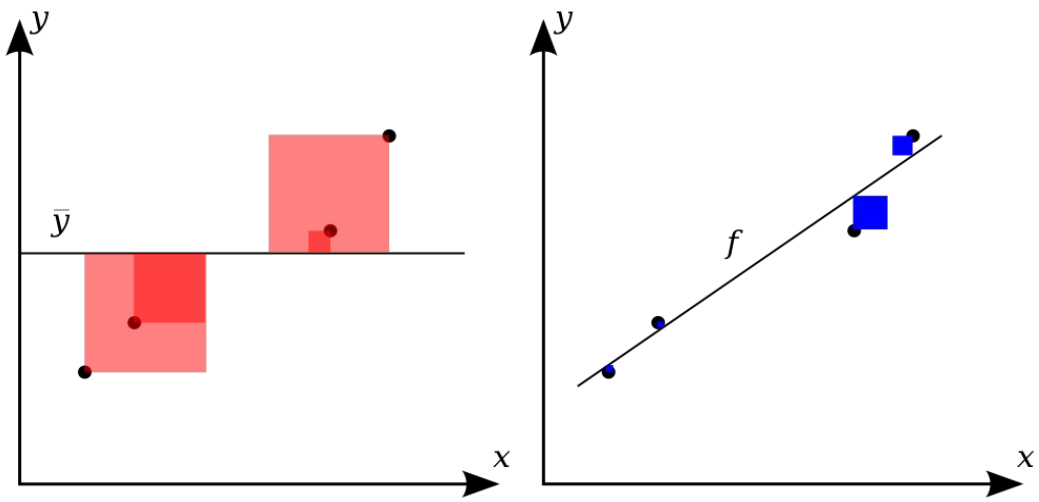
$$R^2 = 1 - \frac{RSS}{TSS}$$

where *TSS* is the total sum of squares, which measures the total variance of the output data:

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

# Coefficient of Determination

- *RSS* (the areas of the blue squares) represents the squared residuals with respect to the linear regression.
- *TSS* (the areas of the red squares) represents the squared residuals with respect to the average value and is fixed if data is known. (Can you tell why?)




Source: https://en.wikipedia.org/wiki/Coefficient_of_determination.

## Coefficient of Determination

- Given a dataset, TSS is determined and the fitted model has the minimum RSS. Therefore:
  - $R^2 = 1$ indicates that the regression line perfectly fits the data.
  - $R^2 = 0$ indicates that the line does not fit the data at all.
  - In general, the better the linear regression fits the data in comparison to the simple average, the closer the value of $R^2$ is to 1.

# Hands-on Session

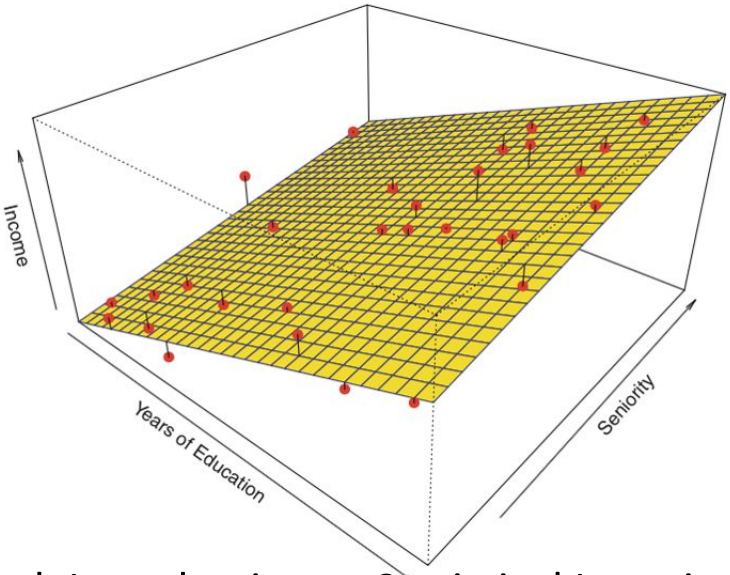
- Please go to the **"Linear Regression in Scikit-Learn"** in the lecture code.

# Outline

- What is Machine Learning
- Introduction to Scikit-Learn
- Simple Linear Regression
  - Estimating Coefficients
  - Coefficient of Determination
- **Multiple Linear Regression**
- Statsmodels

# Multiple linear regression

- In reality, the output *Y* usually depends on multiple input variables.
- Let's look at an example of two predictors.
- By looking at the data, it's reasonable to believe that *Income* depends on both *Years of Education* and *Seniority*. To visualize it, we need to plot a "plane" instead of a line.




Source: James et al. Introduction to Statistical Learning (Springer 2013)

# Multiple linear regression

- If the output *Y* depends on more than one input variable, say $X_1$, $X_2$, … $X_p$, then we need to write the linear model as:

$$\hat{Y} = \hat{\beta}_0 + \sum_{i=1}^{p} \hat{\beta}_i X_i$$

- If we include a constant 1 in *X* and use the notation:

$$\beta = (\beta_0, \beta_1, ..., \beta_p)^T$$
$$X^T = (1, X_1, ..., X_p)$$

Then we can write the model in a simple vector form:

$$\hat{Y} = X^T \hat{\beta}$$

# Multiple linear regression

- To calculate the coefficients, again we use the least squares, but in a slightly different form.
- The RSS now can be written as:

$$RSS(\beta) = (\mathbf{y} - \mathbf{X}\beta)^T(\mathbf{y} - \mathbf{X}\beta)$$

- The task now is to minimize RSS($\beta$). If $X^TX$ is nonsingular, then we can prove that the minimum value of RSS($\beta$) is obtained by:

$$\hat{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

- Note: when two or more input variables are highly correlated with each other, then $X^TX$ is close to singular and the solution will be unstable - any tiny fluctuations of data will cause huge changes in the model.

# Hands-on Session

- Please go to the **"Multiple Linear Regression in Scikit-Learn"** in the lecture code.

# Outline

- What is Machine Learning
- Introduction to Scikit-Learn
- Simple Linear Regression
  - Estimating Coefficients
  - Coefficient of Determination
- **Multiple Linear Regression**
- Statsmodels

## Descriptive Statistics

- In addition to assuming the dependence of response *Y* on predictors *X* is approximately linear, linear regression makes several other key assumptions:
  - Statistical independence of errors
  - Homoscedasticity (constant variance) of the errors
  - Little multicollinearity in the predictors
- But in reality the data is very unlikely to satisfy those assumptions.
- Failing to satisfy the assumptions may lead to a poor fit or even wrong results.

## Descriptive Statistics

- We will now explore some basic descriptive statistics to understand the variables before fitting the linear model.
  - Univariate analysis: understanding the distribution of a single variable (mean, median, quartiles, standard deviation, etc.)
  - Bivariate analysis: understanding the relationship between pairs of variables (correlation, covariance, etc.)
- In the next few lectures, we will introduce some of the strategies that can transform poor data into good shape.

# Hands-on Session

- Please go to the **"Exercise: Descriptive Statistics"** in the lecture code.

# Categorical Input Variables

- So far, although we have seen categorical variables in our input, we have not used them in our regression models. However, they may be useful in making predictions:
  - For example, if we want to explore the relationship between smoking and lung cancer rate, the variable smoking usually has two categories, “yes” and “no”.
  - Categorical variables can be effectively coded as integers. For instance [“yes”, “no”] are often coded as [1, 0].
- We will now discuss how to incorporate categorical variables in regression.

# Categorical Input Variables

- If a categorical variable is binary, i.e., contains two categories, then we just need to code them as [0, 1]. In this scenario, 0 will be treated as the reference category.
- When there are more than two categories, we cannot represent the variables by increasing the number of integers, as scikit-learn estimators expect continuous input, and would interpret the categories as being ordered, which is often not desired.
- For example, if you convert the categorical variable [“Red”, “Blue”, “Green”] as [0, 1, 2], then scikit-learn will consider them as ordered numbers, instead of distinct categories.

# Categorical Input Variables

- The most useful and commonly used coding is via dummy variables: a *K*-level categorical variable is represented by *K* binary variables, only one of which is on at a time. This is called *1-of-K encoding*.
- Scikit-learn provides a class called OneHotEncoder which encodes categorical variables using the 1-of-K scheme, but it only takes integers as input.
- In the next slide we will show you how to convert categorical variables using ***pandas*.get_dummies()**.

# Categorical Input Variables

- Suppose a laptop sales data set has a feature called **web browser** which contains the types of pre-installed web browser for 5 different laptop models.

```
import pandas as pd
browser = pd.Series(["Safari","Chrome", "IE", "IE", "Safari"])
browser_dummy = pd.get_dummies(browser)
browser_dummy
```

| | Chrome | IE | Safari |
|---|---|---|---|
| **0** | 0 | 0 | 1 |
| **1** | 1 | 0 | 0 |
| **2** | 0 | 1 | 0 |
| **3** | 0 | 1 | 0 |
| **4** | 0 | 0 | 1 |

# Categorical Input Variables

- In the previous example, we generated 3 dummy variables, as the number of categories is 3. For each single record we only have one “1” and the rest are all 0’s.
- We can consider one category as the base and drop it without losing any information since all the dummy variables are mutually exclusive.
- So the following dummy variables will be sufficient.

```
browser_dummy.drop('Chrome', 1)
```

| | IE | Safari |
|---|---|---|
| **0** | 0 | 1 |
| **1** | 0 | 0 |
| **2** | 1 | 0 |
| **3** | 1 | 0 |
| **4** | 0 | 1 |

# Hands-on Session

- Please go to the **"Exercise: Dummy Variables"** in the lecture code.

# Outline

- What is Machine Learning
- Introduction to Scikit-Learn
- Simple Linear Regression
  - Estimating Coefficients
  - Coefficient of Determination
- Multiple Linear Regression
- **Statsmodels**

## Statsmodels

- Statsmodels is a Python package that provides a complement to scipy for statistical computations including descriptive statistics and estimation of statistical models.
- It emphasizes parameter estimation and statistical testing.
- The following code shows how to fit a linear model with the Advertising dataset in Statsmodel.

```
import statsmodels.api as sm
model = sm.OLS(adver['Sales'], \
               adver[['TV', 'Radio', 'Newspaper']])
results = model.fit()
print results.summary()
```

# Statsmodels

- In addition to estimating the coefficient, Statsmodels also performs statistical tests.
- The code from the previous slide will generate the following output.

OLS Regression Results

| | | | |
|---|---|---|---|
| Dep. Variable: | Sales | R-squared: | 0.982 |
| Model: | OLS | Adj. R-squared: | 0.982 |
| Method: | Least Squares | F-statistic: | 3566. |
| Date: | Thu, 14 Jan 2016 | Prob (F-statistic): | 2.43e-171 |
| Time: | 01:50:59 | Log-Likelihood: | -423.54 |
| No. Observations: | 200 | AIC: | 853.1 |
| Df Residuals: | 197 | BIC: | 863.0 |
| Df Model: | 3 | | |
| Covariance Type: | nonrobust | | |

| | coef | std err | t | P>\|t\| | [95.0% Conf. | Int.] |
|---|---|---|---|---|---|---|
| TV | 0.0538 | 0.001 | 40.507 | 0.000 | 0.051 | 0.056 |
| Radio | 0.2222 | 0.009 | 23.595 | 0.000 | 0.204 | 0.241 |
| Newspaper | 0.0168 | 0.007 | 2.517 | 0.013 | 0.004 | 0.030 |

| | | | |
|---|---|---|---|
| Omnibus: | 5.982 | Durbin-Watson: | 2.038 |
| Prob(Omnibus): | 0.050 | Jarque-Bera (JB): | 7.039 |
| Skew: | -0.232 | Prob(JB): | 0.0296 |
| Kurtosis: | 3.794 | Cond. No. | 12.6 |

## Statsmodels

- Unlike scikit-learn, Statsmodels does not include the intercept by default.
- To add the intercept, we need to specify it at the beginning.

```
x = adver[['TV', 'Radio', 'Newspaper']]
x = sm.add_constant(x)
model = sm.OLS(adver['Sales'], x)
results = model.fit()
print results.summary()
```

- The output is shown in the next slide.

# Statsmodels

OLS Regression Results

| | | | |
|---|---|---|---|
| Dep. Variable: | Sales | R-squared: | 0.897 |
| Model: | OLS | Adj. R-squared: | 0.896 |
| Method: | Least Squares | F-statistic: | 570.3 |
| Date: | Thu, 14 Jan 2016 | Prob (F-statistic): | 1.58e-96 |
| Time: | 01:51:27 | Log-Likelihood: | -386.18 |
| No. Observations: | 200 | AIC: | 780.4 |
| Df Residuals: | 196 | BIC: | 793.6 |
| Df Model: | 3 | | |
| Covariance Type: | nonrobust | | |

| | coef | std err | t | P>\|t\| | [95.0% Conf. | Int.] |
|---|---|---|---|---|---|---|
| const | 2.9389 | 0.312 | 9.422 | 0.000 | 2.324 | 3.554 |
| TV | 0.0458 | 0.001 | 32.809 | 0.000 | 0.043 | 0.049 |
| Radio | 0.1885 | 0.009 | 21.893 | 0.000 | 0.172 | 0.206 |
| Newspaper | -0.0010 | 0.006 | -0.177 | 0.860 | -0.013 | 0.011 |

| | | | |
|---|---|---|---|
| Omnibus: | 60.414 | Durbin-Watson: | 2.084 |
| Prob(Omnibus): | 0.000 | Jarque-Bera (JB): | 151.241 |
| Skew: | -1.327 | Prob(JB): | 1.44e-33 |
| Kurtosis: | 6.332 | Cond. No. | 454. |

- We will not cover statsmodels in this course. To learn more about statsmodels please go to the link: http://statsmodels.sourceforge.net.