1 - Introduction to Machine Learning

Giovanni Della Lunga
giovanni.dellalunga@unibo.it

Bologna - March, 2023

Outline

1. About these Lessons

2. What is Machine Learning
   - Types of Machine Learning
   - Features and Labels
   - Cost Functions

3. Validation and Testing
   - Training and Validation Set
   - Bias and Variance
   - Regularization
   - Feature Selection
   - Accuracy Metrics

About these Lessons

About these Lessons

Welcome to this Machine Learning course!

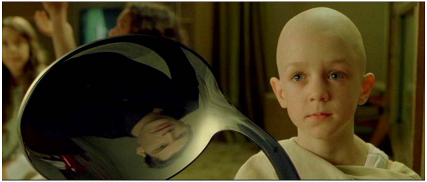
- This course is divided into two parts.
- In the first part, we will introduce the fundamental concepts of machine learning, such as what machine learning is, what features and labels are, supervised and unsupervised learning, and how to calculate the cost function.
- We will also discuss the validation and training of a machine learning model, along with the primary methods of measuring the model's performance.

About these Lessons

- In the second part of the course, we will provide a brief introduction to deep learning and the primary neural network architectures.
- After discussing a simple feed-forward model with a hidden layer in detail, we will introduce recurrent networks, convolutional networks, and explain the principles of autoencoders and generative adversarial networks.
- By the end of this course, you should have at least an intuition of machine learning and deep learning concepts, and you will be able to apply these concepts to simple practical problems.

A word of caution: don't try to bend the spoon...

- ... that's impossible!
- Don't worry if you don't understand all the topics we are going to talk about in these days!
- The aim of the seminar is to give an overview of the main techniques used in a field that is having a growing application interest
- so don't worry, just relax and enjoy the ride ...

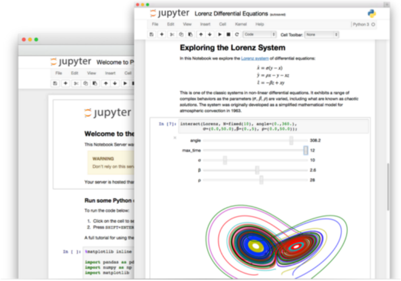


Teaching tools: Jupyter Notebook

- The Python world developed the IPython notebook system.
- Notebooks allow you to write text, but you insert code blocks as ”cells” into the notebook.
- A notebook is interactive, so you can execute the code in the cell directly!
- Recently the Notebook idea took a much enhanced vision and scope, to explicitly allow languages other than Python to run inside the cells.
- Thus the Jupyter Notebook was born, a project initially aimed at Julia, Python and R (Ju-Pyt-e-R). But in reality many other languages are supported in Jupyter.

Teaching tools: Jupyter Notebook

- Jupyter was designed to enable sharing of notebooks with other people.
- The idea is that you can write some code, mix some text with the code, and publish this as a notebook.
- In the notebook they can see the code as well as the actual results of running the code.



jupyter

Welcome to the

This Notebook Server wa

WARNING

Don't rely on this serv

Your server is hosted tha

Run some Python

To run the code below:

1. Click on the cell to se
2. Press SHIFT+ENTER

A full tutorial for using the

```
In [ ]: %matplotlib inline

        import pandas as pd
        import numpy as np
        import matplotlib
```

Exploring the Lorenz System

In this Notebook we explore the Lorenz system of differential equations:

$$\dot{x} = \sigma(y - x)$$
$$\dot{y} = \rho x - y - xz$$
$$\dot{z} = -\beta z + xy$$

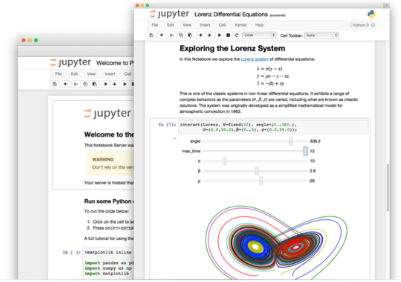
This is one of the classic systems in non-linear differential equations. It exhibits a range of complex behaviors as the parameters (σ, β, ρ) are varied, including what are known as chaotic solutions. The system was originally developed as a simplified mathematical model for atmospheric convection in 1963.

```
In [7]: interact(Lorenz, N=fixed(10), angle=(0.,360.),
                 σ=(0.0,50.0),β=(0.,5), ρ=(0.0,50.0));
```

angle	308.2
max_time	12
σ	10
β	2.6
ρ	28

Teaching tools: Jupyter Notebook

- This is a nice way of sharing little experimental snippets, but also to publish more detailed reports with explanations and full code sets.
- Of course, a variety of web services allows you to post just code snippets (e.g. gist).
- What makes Jupyter different is that the service will actually render the code output.



Exploring the Lorenz System

In this Notebook we explore the Lorenz system of differential equations:

$$\dot{x} = \sigma(y - x)$$
$$\dot{y} = \rho x - y - xz$$
$$\dot{z} = -\beta z + xy$$

This is one of the classic systems in non-linear differential equations. It exhibits a range of complex behaviors as the parameters (σ, β, ρ) are varied, including what are known as chaotic solutions. The system was originally developed as a simplified mathematical model for atmospheric convection in 1963.

```
In [7]: interact(Lorenz, N=fixed(10), angle=(0.,360.),
                 σ=(0.0,50.0),β=(0.,5), ρ=(0.0,50.0));
```

angle 308.2
max_time 12
σ 10
β 2.6
ρ 28

Welcome to the

This Notebook Server wa

WARNING
Don't rely on this ser

Your server is hosted tha

Run some Python

To run the code below:

1. Click on the cell to se
2. Press SHIFT+ENTER

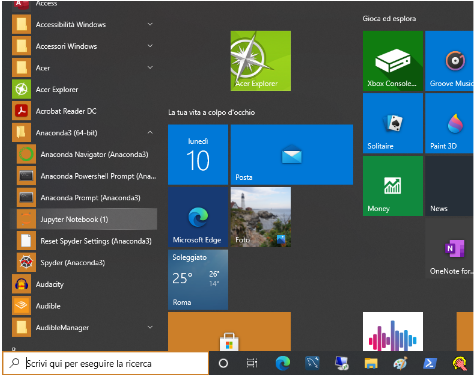
A full tutorial for using the

```
In [ ]: %matplotlib inline

        import pandas as pd
        import numpy as np
        import matplotlib
```

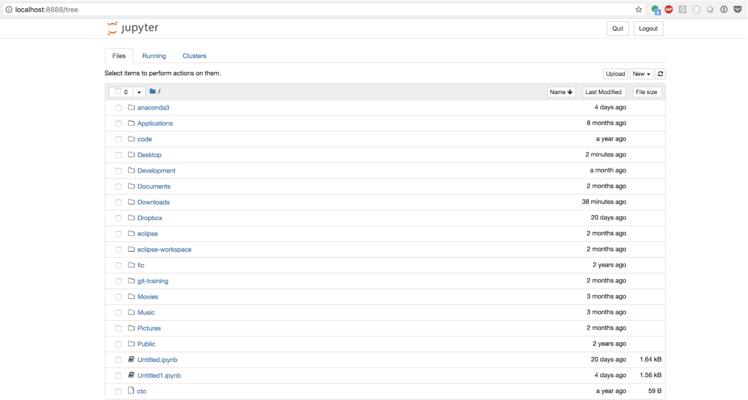
Teaching tools: Jupyter Notebook

- The Jupyter Notebook ships with Anaconda. To run it, you can get in your virtual environment and type the following command: **jupyter notebook**;
- Or directly from the Windows Menu...



Teaching tools: Jupyter Notebook

- You can find this at **http://localhost:8888/tree**
- Now to run Python here, you can create a new file.



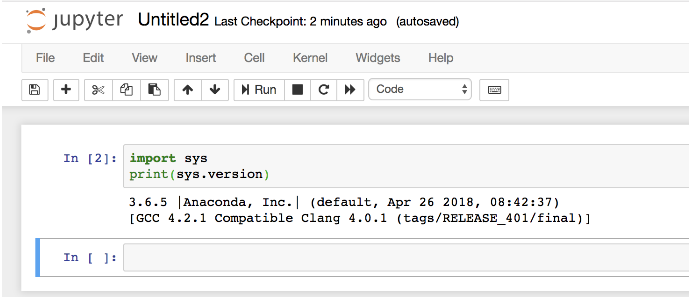
localhost:8888/tree

Select items to perform actions on them.

Name	Last Modified	File size
anaconda3	4 days ago	
Applications	8 months ago	
code	a year ago	
Desktop	2 minutes ago	
Development	a month ago	
Documents	2 months ago	
Downloads	38 minutes ago	
Dropbox	20 days ago	
eclipse	2 months ago	
eclipse-workspace	2 months ago	
fic	2 years ago	
git-training	2 months ago	
Movies	3 months ago	
Music	3 months ago	
Pictures	2 months ago	
Public	2 years ago	
Untitled.ipynb	20 days ago	1.64 kB
Untitled1.ipynb	4 days ago	1.56 kB
cto	a year ago	59 B

Teaching tools: Jupyter Notebook

To make sure it's working, click in the cell and type the following:



In [2]:
```
import sys
print(sys.version)
```

```
3.6.5 |Anaconda, Inc.| (default, Apr 26 2018, 08:42:37)
[GCC 4.2.1 Compatible Clang 4.0.1 (tags/RELEASE_401/final)]
```

In [ ]:

Teaching tools: Google Colab

- Colaboratory, or ”Colab” for short, is a product from Google Research.
- Colab allows anybody to write and execute arbitrary python code through the browser, and is especially well suited to machine learning, data analysis and education.

https://colab.research.google.com/notebooks/intro.ipynb?hl=en

Teaching tools: Google Colab

- More technically, Colab is a hosted Jupyter notebook service that requires no setup to use, while providing free access to computing resources including GPUs.
- Colab notebooks are stored in *Google Drive*, or can be loaded from *GitHub*. Colab notebooks can be shared just as you would with Google Docs or Sheets.

https://colab.research.google.com/notebooks/intro.ipynb?hl=en

Outline

1. About these Lessons

2. What is Machine Learning
   - Types of Machine Learning
   - Features and Labels
   - Cost Functions

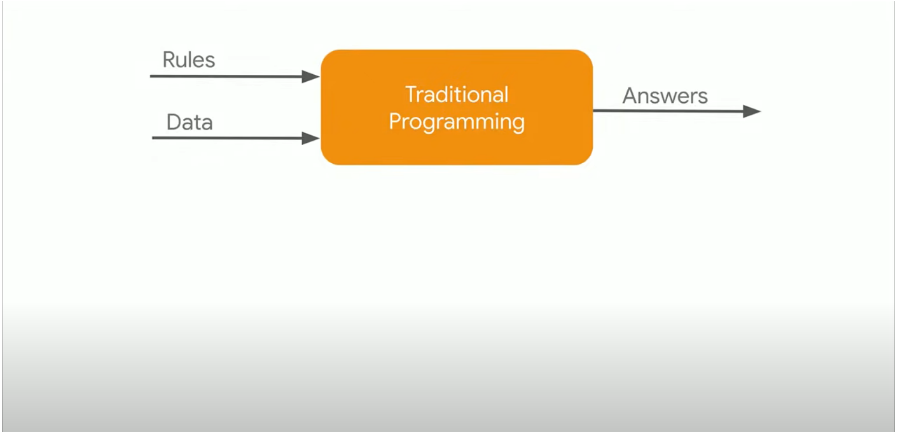
3. Validation and Testing
   - Training and Validation Set
   - Bias and Variance
   - Regularization
   - Feature Selection
   - Accuracy Metrics

What is Machine Learning

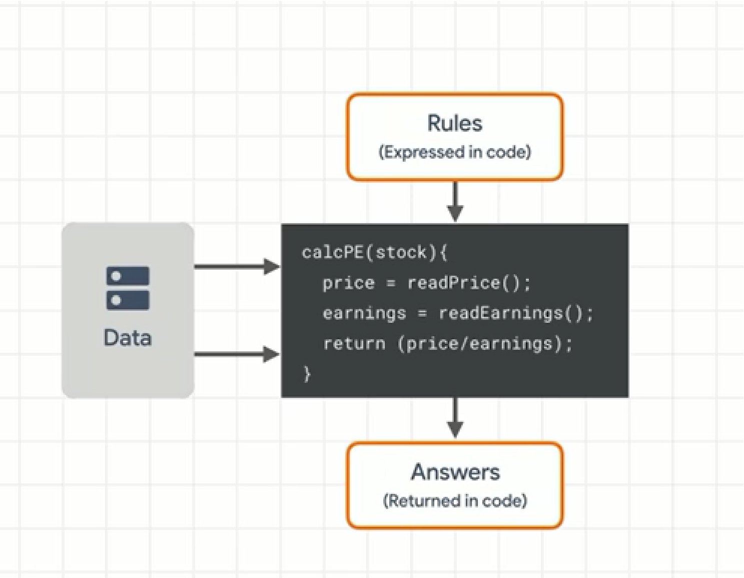
What is Machine Learning

- Machine learning is a branch of AI
- The idea underlying machine learning is that we give a computer program access to lots of data and let it learn about relationships between variables and make predictions
- Some of the techniques of machine learning date back to the 1950s but improvements in computer speeds and data storage costs have now made machine learning a practical tool
- Machine Learning or data science can be described as the new world of statistics

What is Machine Learning

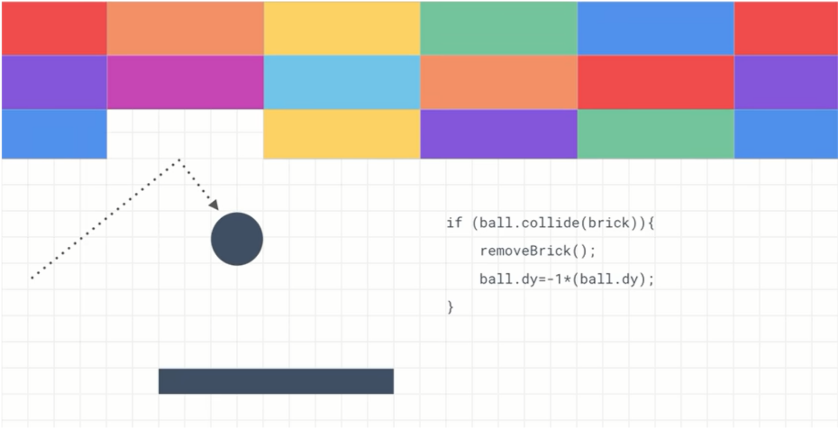


What is Machine Learning



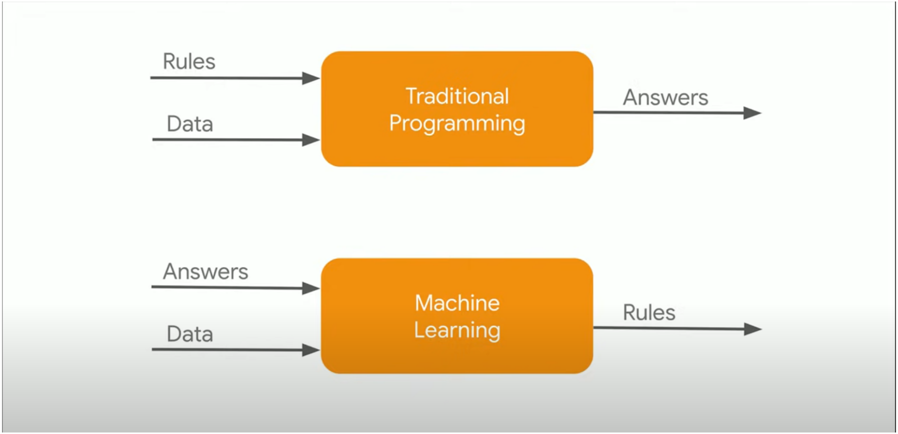
```
calcPE(stock){
  price = readPrice();
  earnings = readEarnings();
  return (price/earnings);
}
```

What is Machine Learning



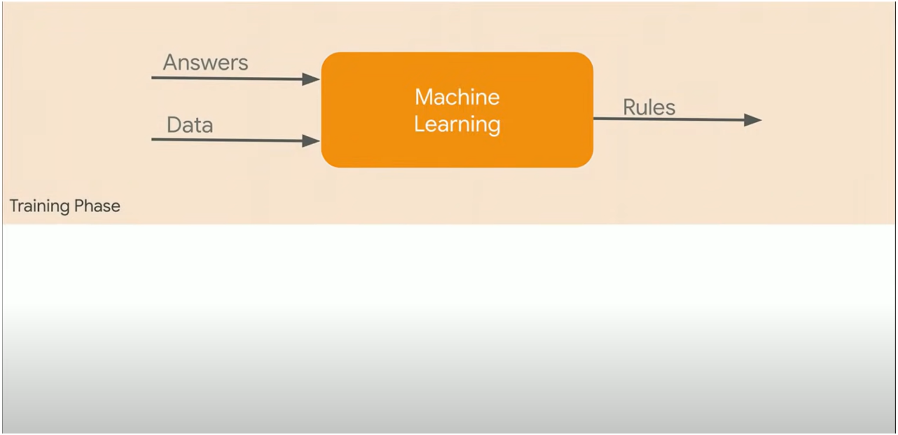
```
if (ball.collide(brick)){
    removeBrick();
    ball.dy=-1*(ball.dy);
}
```

What is Machine Learning

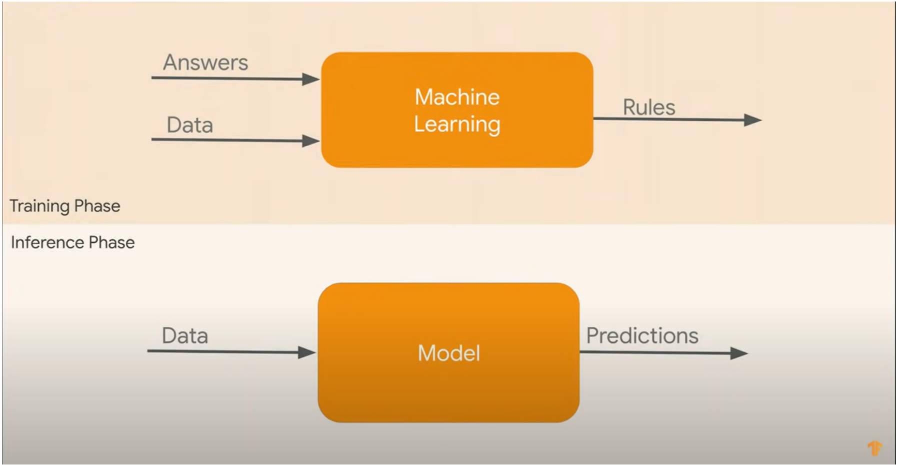

Rules
Data
Traditional Programming
Answers

Answers
Data
Machine Learning
Rules

What is Machine Learning



What is Machine Learning



Subsection 1

Types of Machine Learning

Types of Machine Learning

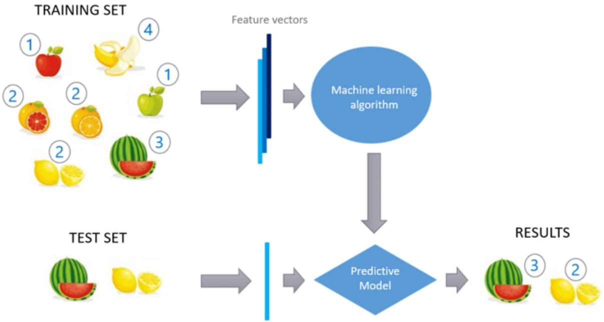
- Unsupervised learning (find patterns)
- Supervised learning (predict numerical value or classification)
- Semi-supervised learning (only part of data has values for, or classification of, target)
- Reinforcement learning (multi-stage decision making)

In these introductory lessons we will discuss only the first two types.

Supervised Learning

Types of Machine Learning

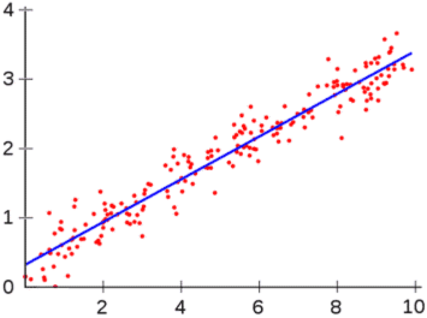
- The main goal in supervised learning is to learn a model from labeled training data that allows us to make predictions about unseen or future data.
- Here, the term "supervised" refers to a set of **training** examples (data inputs) where the desired output signals (**labels**) are already known.



Supervised Learning

Types of Machine Learning

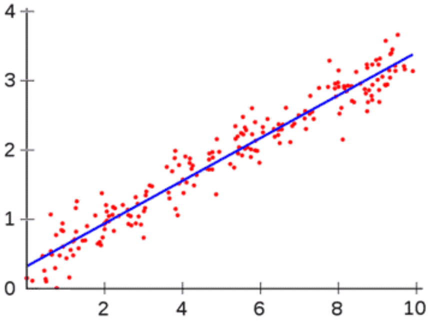
- A supervised learning task with discrete class labels, such as in the previous example, is also called a **classification task**.
- A second type of supervised learning is the prediction of continuous outcomes, which is also called **regression analysis**.



Supervised Learning

Types of Machine Learning

- In regression analysis, we are given a number of predictor (explanatory) variables and a continuous response variable (outcome), and we try to find a relationship between those variables that allows us to predict an outcome.



Supervised Learning

Types of Machine Learning

- From a formal poin of view, supervised learning process involves input variables, which we call X, and an output variable, which we call Y.
- We use an algorithm to learn the mapping function from the input to the output.
- In simple mathematics, the output Y is a dependent variable of input X as illustrated by:

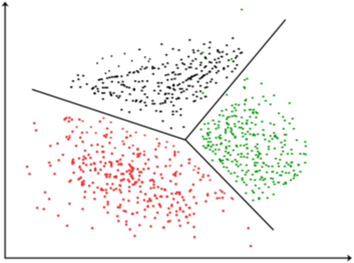
$$Y = f(X)$$

Here, our end goal is to try to **approximate the mapping function** f, so that we can **predict** the output variables Y when we have new input data X.

Unsupervised Learning

Types of Machine Learning

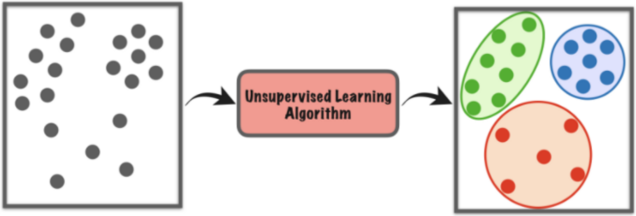
- In unsupervised learning we are dealing with unlabeled data or data of unknown structure.
- Using unsupervised learning techniques, we are able to explore the structure of our data to extract meaningful information without the guidance of a known outcome variable or reward function.



Unsupervised Learning

Types of Machine Learning

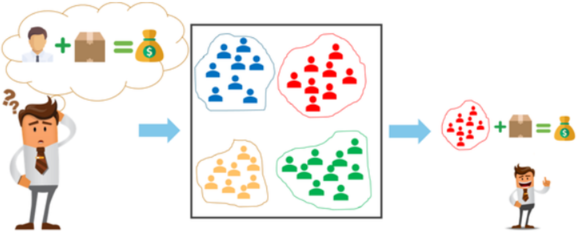
- Unsupervised algorithms attempt to find some sort of underlying structure in the data.
- Are some observations clustered into groups? Are there interesting relationships between different features? Which features carry most of the information?



Unsupervised Learning

Types of Machine Learning

- The objective is to cluster data to increase our understanding of the environment
- **Example - Clustering Customers**
    - Suppose you are a bank and have hundreds of thousands of customers and 100 features describing each one
    - Unsupervised learning algorithms can be used to divide your customers into clusters so that you can anticipate their needs and communicate with them more effectively

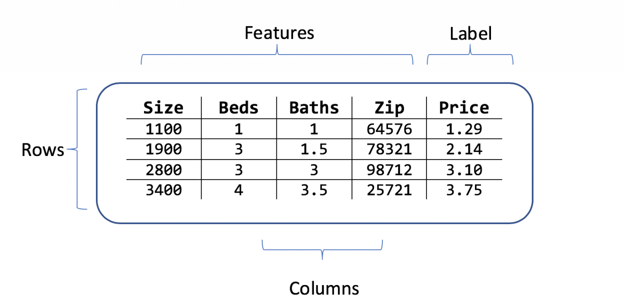


Subsection 2

Features and Labels

Features and Labels

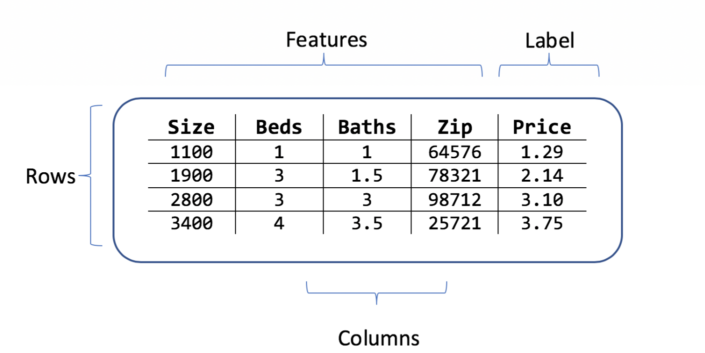
- The data for **Supervised Learning** contains what are referred to as **features** and **labels**;
- *Labels* are the values of the target that is to be predicted;
- *Features* are the variables from which the predictions are to be made (if you are from statistics then think of it as an explanatory variable);



Size	Beds	Baths	Zip	Price
1100	1	1	64576	1.29
1900	3	1.5	78321	2.14
2800	3	3	98712	3.10
3400	4	3.5	25721	3.75

Features and Labels

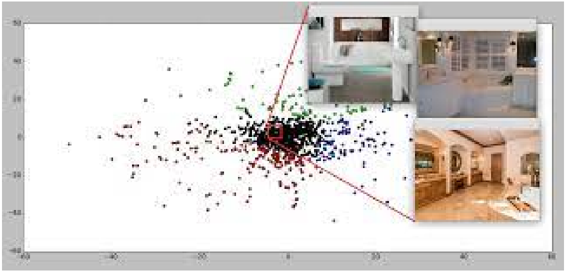
- For example, when predicting the **price of a house**, the *features* could be the *square meters of living space*, *the number of bedrooms*, *the number of bathrooms*, *the size of the garage* and so on.
- The *label* would be *the house price*;



Size	Beds	Baths	Zip	Price
1100	1	1	64576	1.29
1900	3	1.5	78321	2.14
2800	3	3	98712	3.10
3400	4	3.5	25721	3.75

Features and Labels

- The data for **Unsupervised Learning** consists of features but no labels because the model is being used to identify patterns not to forecast something.
- For example we could use an unsupervised model to classify the houses that exist in a certain neghborhood without trying to predict any price.



Subsection 3

Cost Functions

Cost Function

- In Machine Learning a **Cost Function** or loss function is used to represent how far away a mathematical model is from the real data;
- One adjust the mathematical model usually by varying parameters within the model so as to minimize the const function;
- Let's take for example a very simple model of the form

$$y = \theta_0 + \theta_1 x$$

  where the θs are the parameters that we want to find to give us the best fit to the data;
- Call this function $h_\theta(x)$ to emphasize the dependence on both the variable x and the two parameters θ_0 and θ_1;

Cost Function

- We want to measure how far away the data, the $y^{(n)}$s are from the function $h_\theta(x)$;
- A common way to do this is via the quadratic cost function

$$J(\theta) = \frac{1}{2N} \sum_{n=1}^{N} \left[h_\theta \left(x^{(n)} \right) - y^{(n)} \right]^2 \tag{1}$$

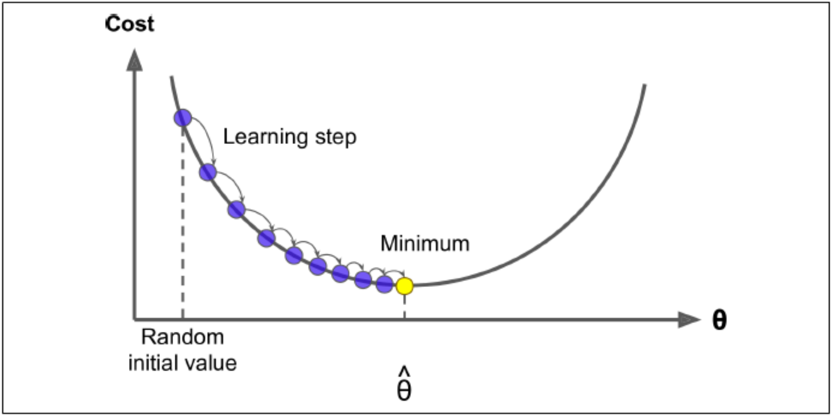
- We wants the parameters that minimize (1), almost always you are going to have to do this numerically;
- If we have a nice convex function then there is a numerical method that will converge to the solution, it is called **gradient descent**.

Gradient Descent

- Gradient Descent is a very generic optimization algorithm capable of finding optimal solutions to a wide range of problems.
- Gradient Descent Algorithm measures the local gradient of the error function with regards to the parameter vector θ, and it goes in the direction of descending gradient.
- Once the gradient is zero, you have reached a minimum!
- Concretely, you start by filling θ with random values (this is called random initialization), and then you improve it gradually, taking one step at a time, each step attempting to decrease the cost function (e.g., the MSE), until the algorithm converges to a minimum

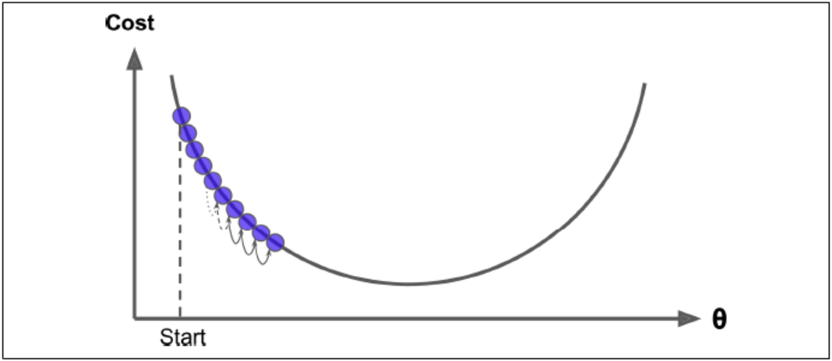
Gradient Descent

An important parameter in Gradient Descent is the size of the steps, determined by the learning rate hyperparameter.



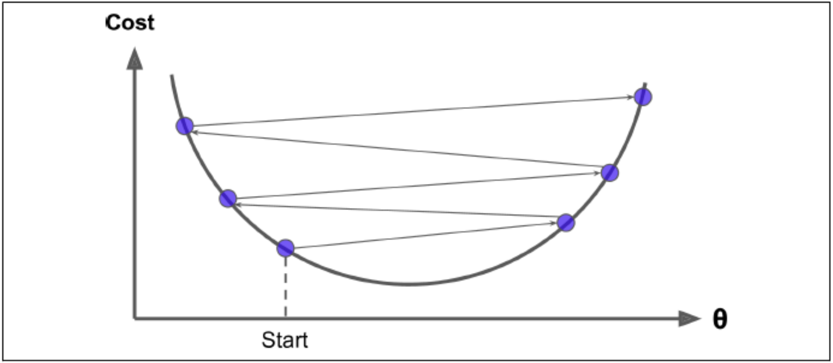
Gradient Descent

If the learning rate is too small, then the algorithm will have to go through many iterations to converge, which will take a long time...



Gradient Descent

... on the other hand, if the learning rate is too high, you might jump across the valley. This might make the algorithm diverge failing to find a good solution.



Gradient Descent

Having defined an appropriate learning rate, the scheme works as follow

- Start with an initial guess for each parameter θ_k;
- Move θ_k in the direction of the slope

$$\boxed{\text{New}\,\theta_k = \text{Old}\,\theta_k - \beta\frac{\partial J}{\partial\theta_k}}$$

where β is our learning rate;

In the above description of gradient descent we have used all of the data points simultaneously. This is called **batch gradient descent**

Gradient Descent

- Rather then use all of the data in the parameter updating we can use a technique called **stochastic gradient descent**;
- This technique is the same as the batch gradient descent except that you only update using **one** of the data points each time and this data point is chosen **randomly**;
- Especially in high-dimensional optimization problems this reduces the computational burden, achieving faster iterations in trade for a lower convergence rate;
- When the learning rates decrease with an appropriate rate, and subject to relatively mild assumptions, stochastic gradient descent converges almost surely to a global minimum when the objective function is convex or pseudoconvex, and otherwise converges almost surely to a local minimum;

Notebook Reference

- **chapter-1-1** Notebook

  In this notebook, you will learn about the main concepts and different types of machine learning.

- **chapter-1-2** Notebook

  Here you can find a short practical introduction to the most important python scientific computation libraries used in data analysis. These are libraries of a general nature for analyzing data in Python.

Outline

1. About these Lessons
2. What is Machine Learning
   - Types of Machine Learning
   - Features and Labels
   - Cost Functions
3. Validation and Testing
   - Training and Validation Set
   - Bias and Variance
   - Regularization
   - Feature Selection
   - Accuracy Metrics

Validation and Testing

Subsection 1

Training and Validation Set

Validation and Testing

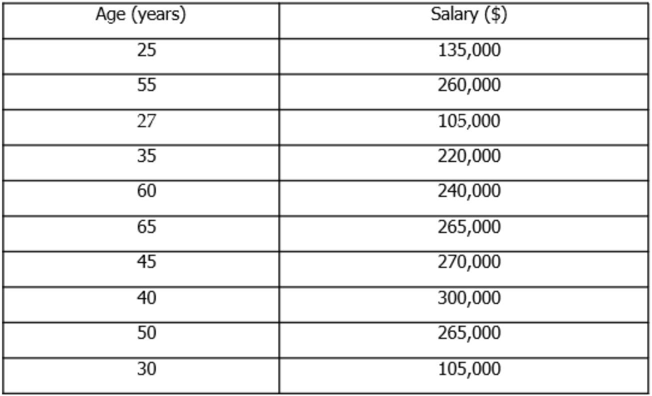
- When data is used for forecasting there is a danger that the machine learning model **will work very well for training data, but will not generalize well to other data**;
- An obvious point is that it is important that the data used in a machine learning model be representative of the situations to which the model is to be applied;
- It is also important to test a model **out-of-sample**, by this we mean that the model should be tested on data that is different from the sample data used to determine the parameters of the model;
- Data scientist refer to the sample data as the **training set** and the data used to determine the accuracy of the model as the **test set**;
- Often a **validation set** is used as well as we explain later;

Validation and Testing

- We will illustrate the use of a training set and the test data set with a very simple Example (Hull, Chapter 1);
- We suppose that awe are interested in forecasting the salaries of people from their age;
- This simple regression model is an example of supervised learning...

Validation and Testing: Training Set

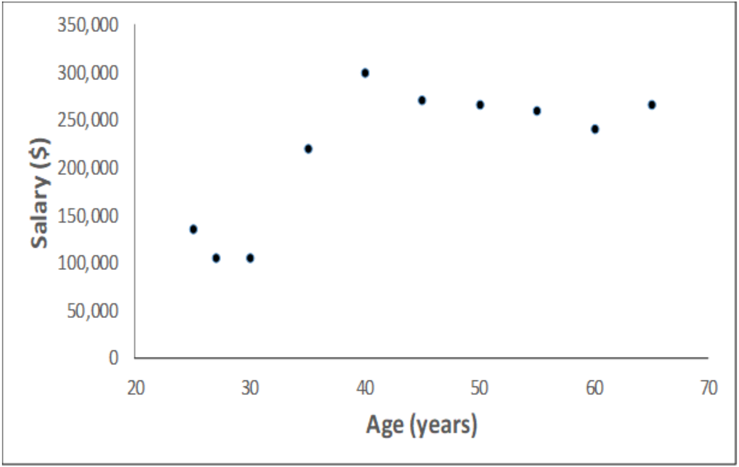
Table 1. Salary as a function of Age for a certain profession in a certain area)



Age (years)	Salary ($)
25	135,000
55	260,000
27	105,000
35	220,000
60	240,000
65	265,000
45	270,000
40	300,000
50	265,000
30	105,000

Validation and Testing: Training Set

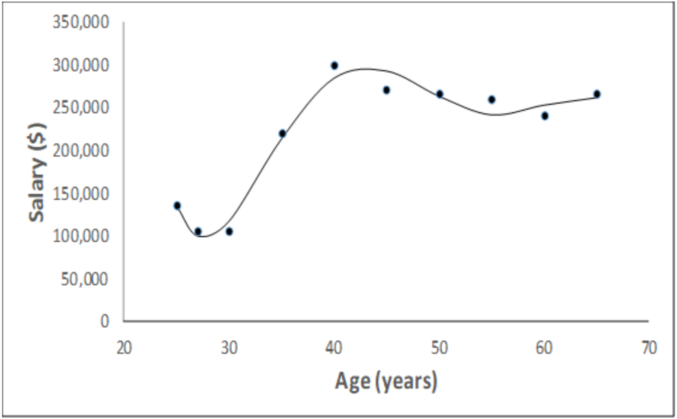
Figure 1. Scatter plot of Salary as a function of Age (see Table 1)



Validation and Testing: Training Set

Figure 2. It's tempting to choose a model that fits the data really well for example with a polynomial of degree five (Y = Salary, X = Age):

$$Y = a + b_1 X + b_2 X^2 + b_3 X^3 + b_4 X^4 + b_5 X^5$$

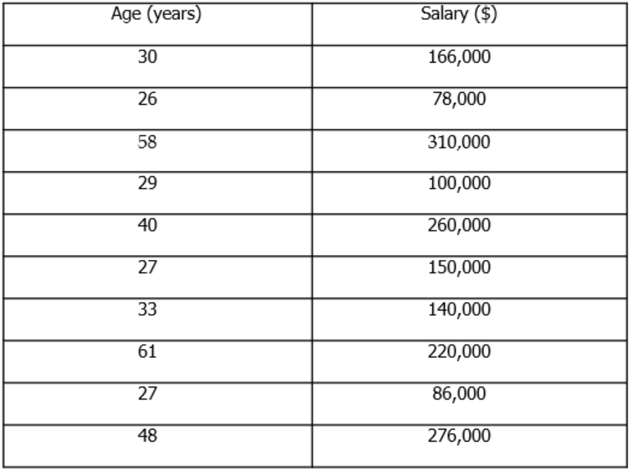


Validation and Testing: Discussion of Training Result

- The model provides a good fit to the data;
- The standard deviation of the difference between the salary given by the model and the actual salary for the ten individuals in the training data set (which is referred to as the **root mean square error (rmse)**) is $12902;
- However common sense would suggest that we many over-fitted the data;
- We need to check the model out-of-sample;
- To use the language of *data science* we need to determine whether the model generalizes well to a new data set that is different from the training set.

Validation and Testing: Validation Set

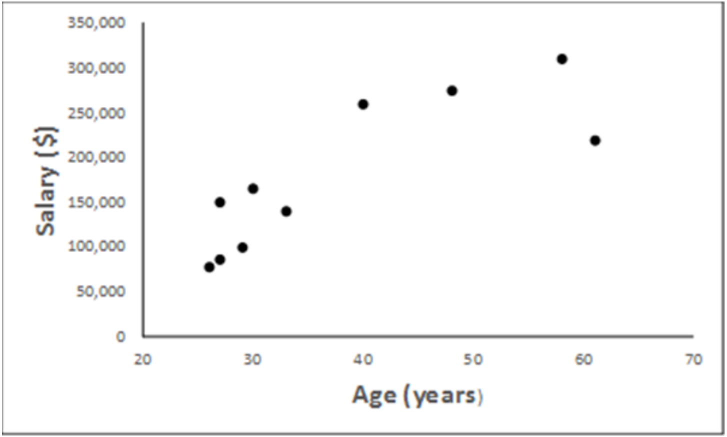
Table 2. An Out-of-Sample Validation Set



Age (years)	Salary ($)
30	166,000
26	78,000
58	310,000
29	100,000
40	260,000
27	150,000
33	140,000
61	220,000
27	86,000
48	276,000

Validation and Testing: Validation Set

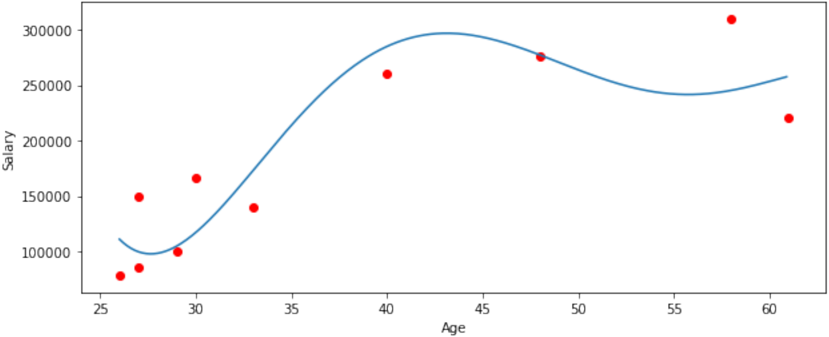
Figure 3. Scatter Plot for Validation Set



Validation and Testing: Discussion of Validation Result

The Fifth Order Polynomial Model Does Not Generalize Well

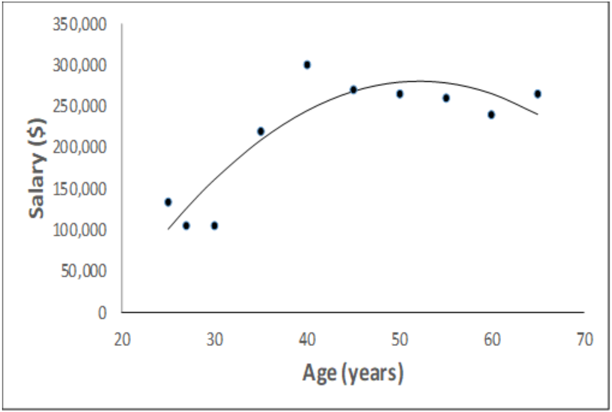
- The root mean squared error (rmse) for the training data set is $12,902
- The rmse for the test data set is $38,794
- We conclude that the model overfits the data



Validation and Testing

Figure 4. A Simpler Quadratic Model

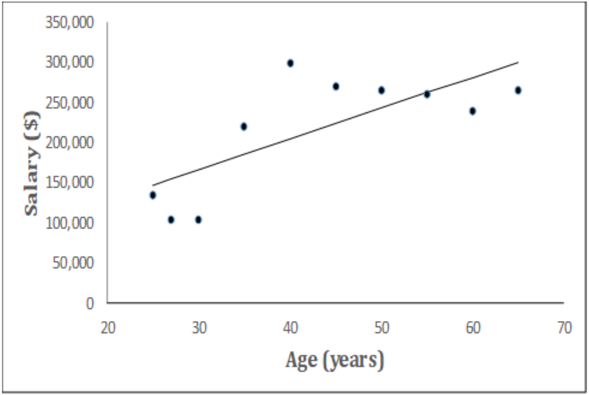
$$Y = a + b_1 X + b_2 X^2$$



Validation and Testing

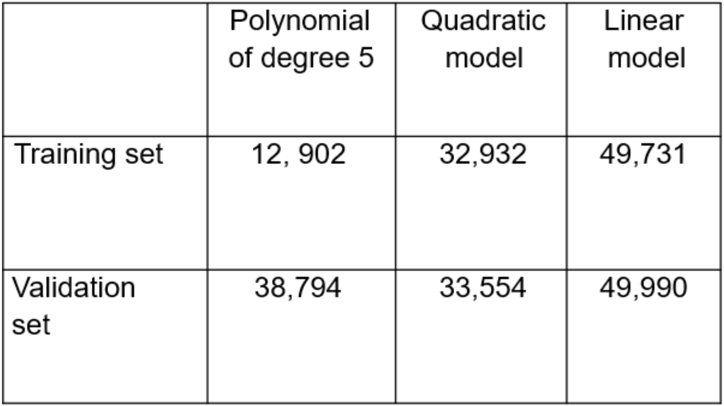
Figure 5 . Linear Model

$$Y = a + b_1 X$$



Validation and Testing

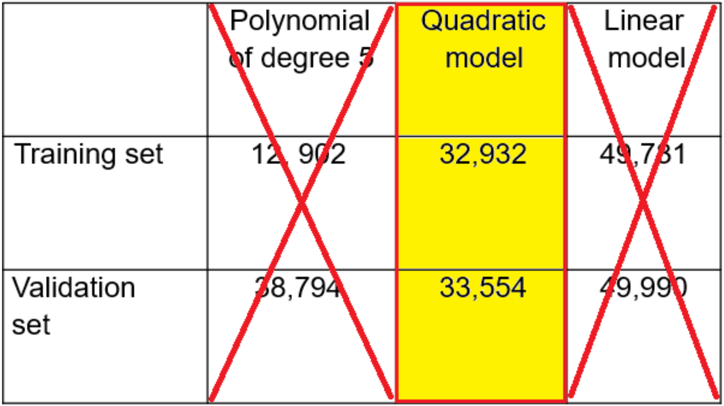
Table 3. Summary of Results: The linear model under-fits while the 5th degree polynomial over-fits:



	Polynomial of degree 5	Quadratic model	Linear model
Training set	12, 902	32,932	49,731
Validation set	38,794	33,554	49,990

Validation and Testing

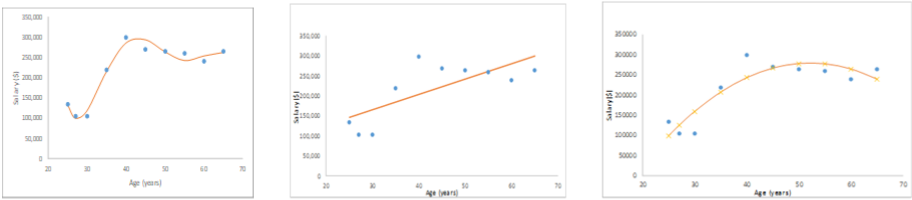
Table 3. Summary of Results: The linear model under-fits while the 5th degree polynomial over-fits:



	Polynomial of degree 5	Quadratic model	Linear model
Training set	12, 902	32,932	49,731
Validation set	38,794	33,554	49,990

Validation and Testing

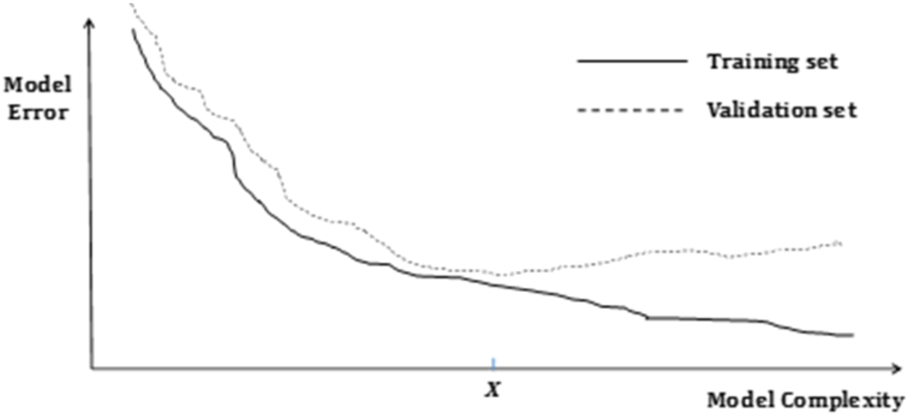
Figure 6. Overfitting/Underfitting Example: predicting salaries for people in a certain profession in a certain area (only 10 observations)



Overfitting ———— Underfitting ———— Best model?

Validation and Testing

Typical Pattern of Errors for Training Set and Validation Set



Validation and Testing

ML Good Practice

- Divide data into three sets
- Training set
- Validation set
- Test set
- Develop different models using the **training set** and compare them using the **validation set**;
- Rule of thumb: increase model complexity until model no longer generalizes well to the validation set;
- The **test set** is used to provide a final out-of-sample indication of how well the chosen model works;

Subsection 2

Bias and Variance

Bias and Variance

- Suppose there is a relationship between an independent variable x and a dependent variable y:

$$y = f(x) + \epsilon \tag{2}$$

  where ϵ is an error term with mean zero and variance σ^2.
- The error term captures either genuine randomness in the data or noise due to measurement error.
- Suppose we find, with a Machine Learning technique, a deterministic model for this relationship:

$$y = \hat{f}(x) \tag{3}$$

Bias and Variance

- Now it comes a new data point x' not in the training set and we want to predict the corresponding y';
- The error we will observe in our model at point x' is going to be

$$\hat{f}(x') - f(x') - \epsilon \tag{4}$$

- There are two different sources of error in this equation.
- The first one is included in the factor ϵ;
- The second one, more interesting, is due to what is in our training set.
- A robust model should give us the same prediction whatever data we used for training out model.

Bias and Variance

- Let's look at the average error:

$$E\left[\hat{f}(x')\right] - f(x') \tag{5}$$

where the expectation is taken over random samples of training data (having the same distributio as the training data).

- This is the definition of the **bias**

$$\mathrm{Bias}\left[\hat{f}(x')\right] = E\left[\hat{f}(x')\right] - f(x') \tag{6}$$

Bias and Variance

- We can also look at the mean square error

$$\boxed{E\left[\left(\hat{f}(x') - f(x') - \epsilon\right)^2\right] = \left[\mathrm{Bias}\left(\hat{f}(x')\right)\right]^2 + \mathrm{Var}\left[\hat{f}(x')\right] + \sigma^2}$$

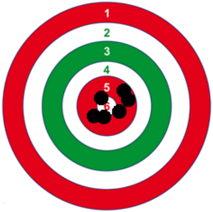
where we remember that $\hat{f}(x')$ and ϵ are independent.

- This show us that there are two important quantities, the **bias** and the **variance** that will affect our results and that we can control to some extent.

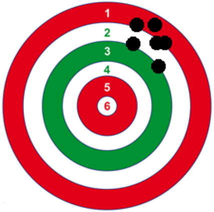
Bias and Variance

- **What is Bias?** It's the difference between the average prediction of our model and the correct value which we are trying to predict. Model with high bias pays very little attention to the training data and oversimplifies the model. It always leads to high error on training and test data.
- **What is Variance?** It's the variability of model prediction for a given data point or a value which tells us spread of our data. Model with high variance pays a lot of attention to training data and does not generalize on the data which it hasn't seen before. As a result, such models perform very well on training data but has high error rates on test data.

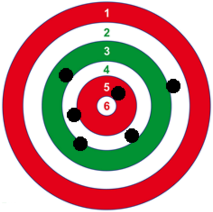
Bias and Variance



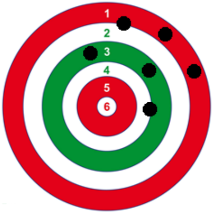
Low Bias and Low Variance



High Bias and Low Variance



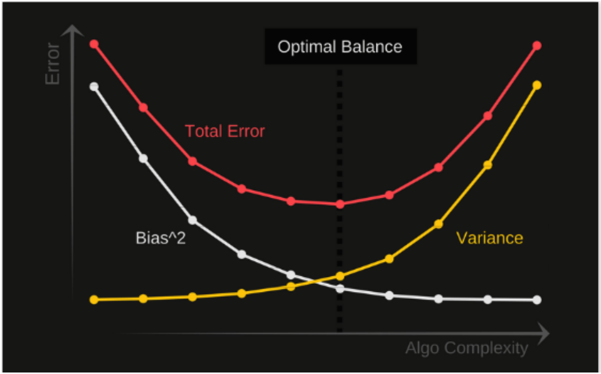
Low Bias and High Variance



High Bias and High Variance

Bias and Variance

Unfortunately, we often find that there is a trade-off between bias and variance. As one is reduced, the other is increased. This is the matter of over- and under-fitting.



Subsection 3

Regularization

Regularization

- Linear regression can over-fit, particularly when there are a large number of correlated features.
- Results for validation set may not then be as good as for training set
- Regularization is a way of avoiding overfitting and reducing the number of features. Alternatives:
- Ridge
- Lasso
- Elastic net
- We must first scale feature values

Ridge regression (analytic solution)

- Ridge regression is a regularization technique where we change the function that is to be minimize;
- Reduce magnitude of regression coefficients by choosing a parameter λ and minimizing

$$\frac{1}{2N}\sum_{n=1}^{N}\left[h_\theta\left(x^{(n)}\right)-y^{(n)}\right]^2+\lambda\sum_{n=1}^{N}b_i^2 \tag{7}$$

- This change has the effect of encouraging the model to keep the weights b_j as small as possibile;
- The Ridge regression should only be used for determining model parameters using the training set. Once the model parameters have been determined the penalty term should be removed for prediction;
- What happens as λ increases?

Lasso Regression (must use gradient descent)

- Lasso is short for *Least Absolute Shrinkage and Selection Operator*;
- Similar to ridge regression except we minimize

$$\frac{1}{2N}\sum_{n=1}^{N}\left[h_\theta\left(x^{(n)}\right)-y^{(n)}\right]^2+\lambda\sum_{n=1}^{N}|b_n| \tag{8}$$

- This function cannot be minimized analytically and so a variation on the gradient descent algorithm must be used;
- Lasso regression also has the effect of simplifying the model. It does this by setting the weights of unimportant features to zero. When there are a large number of features, Lasso can identify a relatively small subset of the features that form a good predictive model.

Ridge and Lasso Regression Compared

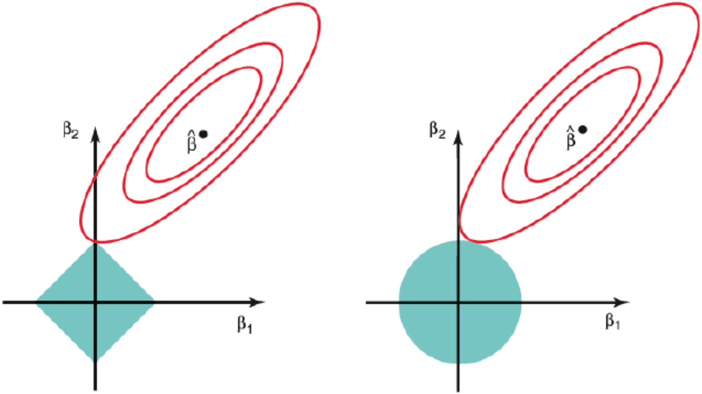


FIGURE 6.7. *Contours of the error and constraint functions for the lasso* (left) *and ridge regression* (right). *The solid blue areas are the constraint regions, $|\beta_1| + |\beta_2| \leq s$ and $\beta_1^2 + \beta_2^2 \leq s$, while the red ellipses are the contours of the RSS.*

Elastic Net Regression (must use gradient descent)

- Middle ground between Ridge and Lasso
- Minimize

$$\frac{1}{2N}\sum_{n=1}^{N}\left[h_\theta\left(x^{(n)}\right)-y^{(n)}\right]^2+\lambda_1\sum_{n=1}^{N}b_n^2+\lambda_2\sum_{n=1}^{N}|b_n| \tag{9}$$

- In Lasso some weights are reduced to zero but others may be quite large. In Ridge, weights are small in magnitude but they are not reduced to zero. The idea underlying Elastic Net is that we may be able to get the best of both by making some weights zero while reducing the magnitude of the others.

Notebook Reference

- **chapter-1-3** Notebook

  Here you can find a number of example regarding validation and testing, an explanation of the bias-variance tradeof and a short introduction to scikit-learn. Scikit-learn is the most popular machine learning package in the data science community. Some example of regularization whit scikit-learn package are presented.

Subsection 4

Feature Selection

Introduction

- As a dimensionality reduction technique, feature selection aims to choose a small subset of the relevant features from the original features by removing irrelevant, redundant, or noisy features.
- Feature selection usually can lead to better learning performance, higher learning accuracy, lower computational cost, and better model interpretability.
- The problem is important, because a high number of features in a dataset, comparable to or higher than the number of samples, leads to model overfitting, which in turn leads to poor results on the validation datasets.
- Additionally, constructing models from datasets with many features is more computationally demanding.

Feature Extraction and Feature Selection

- Even though feature extraction and feature selection processes share some overlap, often these terms are erroneously equated.
- **Feature extraction** is the process of using domain knowledge to extract new variables from raw data that make machine learning algorithms work.
- The **feature selection** process is based on selecting the most consistent, relevant, and non-redundant features.
- The feature selection process is based on selecting the most consistent, relevant, and non-redundant feature subset from feature vectors.
- It is not only reduces training time and model complexity, but it eventually helps to **prevent overfitting**.

Objectives of Feature Selection

- Simplification of models to make them easier to interpret by researchers/users
- Shorter training times
- Avoid the curse of dimensionality
- Enhanced generalization by reducing overfitting

How do we select features?

- A feature selection procedure combines a search technique with an evaluation method. The search technique proposes new feature subsets, and the evaluation measure determines the how good the subset is.
- In a perfect world, a feature selection method would evaluate all possible subsets of feature combinations and determine which one results in the best performing machine learning model.
- However, computational cost inhibits such a practice in reality. In addition, the optimal subset of features varies between machine learning models. A feature subset that optimizes one model‘s performance won‘t necessarily optimize another‘s.

How do we select features?

There are a lot of ways in which we can think of feature selection, but most feature selection methods can be divided into three major buckets

- **Filter based**: We specify some metric and based on that filter features. An example of such a metric could be correlation/chi-square.
- **Wrapper-based**: Wrapper methods consider the selection of a set of features as a search problem. Example: Recursive Feature Elimination
- **Embedded**: Embedded methods use algorithms that have built-in feature selection methods. For instance, Lasso and RF have their own feature selection methods.

Filter Methods

- A typical filter algorithm consists of two steps: it ranks features based on certain criteria and then chooses the highest-ranking features to train the machine learning models.
- Filter methods are generally univariate, so they rank each feature independently of the rest. Because of this, the filter methods tend to ignore any interactions that occur between features. Thus, redundant variables will not necessarily be eliminated by filter methods.
- However, some multivariate filter selection methods exist as well. These consider features in relations to others in the data set, making them naturally capable of handling redundant features. Their selection criteria scans for duplicated features and correlated features and provide simple but powerful methods to quickly remove redundant information.

Filter Methods

- **Constant, quasi-constant, and duplicated features**
- The most basic and intuitive methods for feature selection consist of removing constant, quasi-constant, or duplicated features.
- Constant features only show one value for all the observations in the data set. That is, they show absolutely no variability.
- Quasi-constant features are similar; if most observations share the same value, then we‘d label that feature quasi-constant. In practice, quasi-constant features typically refer to those variables where more than 95 to 99 percent of the observations show the same value.
- Duplicated features, as the name indicates, are those that are in essence, identical. That is, for every observation, they show the same value.

Filter Methods

- Although it sounds obvious and overly simple, many datasets contain a lot of constant, quasi-constant, and duplicated features.
- In fact, duplicated features often arise when generating new features by one-hot encoding of categorical variables.
- Removing these features is an easy but effective way to reduce the dimension of the feature space, without losing any significant information.

Filter Methods

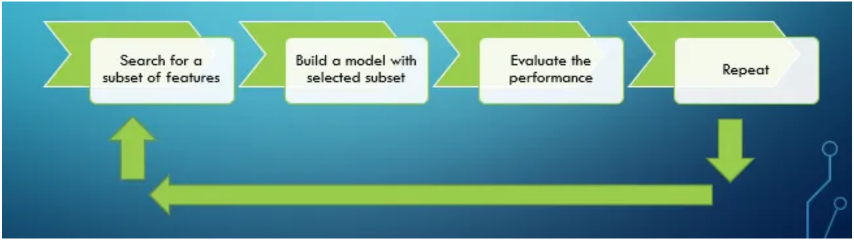
- **Correlation**
- Correlation measures the linear association between two or more variables.
- The higher the correlation, the more linearly associated the variables are.
- The central hypothesis is that good feature sets contain features that are highly correlated with the target, yet uncorrelated with each other.
- If two variables are correlated, we can predict one from the other.
- Therefore, if two features are correlated, the model only really needs one of them, as the second one does not add additional information.

Filter Methods

- **Mutual Information**
- Mutual information measures the mutual dependence between two variables, in this case, the feature and the target. Mutual information is similar to correlation, but more general; it doesn‘t strictly represent linear association. It measures how much knowing one of these variables reduces uncertainty in the other.
- **Fisher Score**
- Fisher score uses the Chi-Square distribution to measure the dependency between two variables and works with categorical or discrete variables.
- Fisher score essentially compares the actual frequencies of the values of a variable with the expected frequencies if it had no relation to the target.

Wrapper Methods

- In comparison to filter methods, wrapper methods tend to be more computationally expensive, but select a better set of features.
- Wrapper methods use a specific machine learning algorithm in the selection process and choose the best subset of features tailored for that algorithm.
- This subset of features may not be optimal for a different machine learning model. In other words, **wrapper methods are not model agnostic**.


Search for a subset of features
Build a model with selected subset
Evaluate the performance
Repeat

Subsection 5

Accuracy Metrics

Threshold

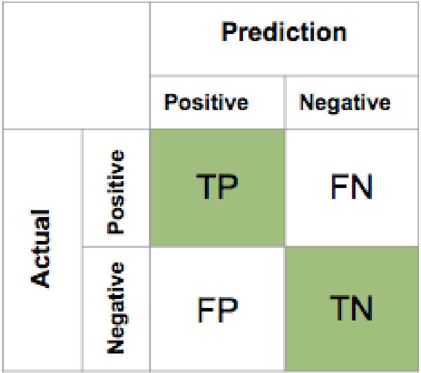
- The threshold is an arbitrarily decided upon point between 0.0 and 1.0 that serves as your "cutoff" for which predicted probabilities you want to consider a True or a False, a Yes or a No, a 1 or a 0.
- Who decides it?
- You do, or whomever the decision-maker happens to be.
- You may have assumed this threshold is naturally located right in the middle, at 0.5, but you can move that threshold.

Threshold

- Why would you want to do that?
- Reasons include:
- a) You are conservative about your guesses, so you set the threshold for a "Yes" to 0.7 (or 70%, if you will). Anything predicted to have less than a 70% probability is just too risky for you.
- b) Alternatively, a risk-taker may want to call anything over 0.35 probability a "Yes", so that they don't miss any opportunities.
- c) Lastly, perhaps you want to use the threshold that gives the highest performance, for whatever metric you choose.

Confusion Matrix

- A confusion matrix is $N * N$ dimension matrix wherein one axis represents **Actual** label while the other axis represents **Predicted** label.
- Confusion Matrix is the most intuitive and basic metric from which we can obtain various other metrics like precision, recall, accuracy, F1 score, AUC - ROC.



		Prediction	
		Positive	Negative
Actual	Positive	TP	FN
	Negative	FP	TN

Confusion Matrix

- For a better understanding of what TP, FP, TN, and FN are, we will consider an example of- **If received mail is spam or ham.**
- **Positive** - Mail received is ham
- **Negative** - Mail received is spam
- ⇒ **True Positive (TP)**: It represents the predicted label is positive and also actual label is positive - correctly predicted. We predicted mail received is ‘ham‘ (positive) and actual mail received is also ‘ham‘ (positive).
- ⇒ **True Negative (TN)**: It represents the predicted label is negative and also actual label is negative - correctly predicted. We predicted mail received is ‘spam‘ (negative) and actual mail received is also ‘spam‘ (negative).
- ⇒ **False Negative (FN)**: It represents the predicted label is negative but the actual label is positive - wrongly predicted. We predicted mail received is ‘spam‘ (negative) but actual mail received is ‘ham‘ (positive).
- ⇒ **False Positive (FP)**: It represents the predicted label is positive but the actual label is negative - wrongly predicted. We predicted mail received is ‘ham‘ (positive) but actual mail received is ‘spam‘ (negative).

Precision

- **General Definition**: Precision measures what proportion of predicted positive label is actually positive.
- **Precision** can be expressed in terms of True Positive and False Positive:

$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}}$$

- As ‘False Positive‘ decreases, our precision increases and vice-versa
- When to use Precision?
- Precision is used when we want to mostly focus on false-positive i.e to decrease false-positive value thereby increase precision value.
- A question might arise why we want to mostly focus on false-positive and not false-negative. The answer to this question depends on the context.

Recall/Sensitivity

- **General Definition**: Recall measures what proportion of actual positive label is correctly predicted as positive.
- To explain recall and its use case, we shall consider 'Cancer Diagnosis' example i.e we have to predict if a patient is diagnosed with cancer or not.
- **Positive** - Patient diagnosed with cancer.
- **Negative** - Patient not diagnosed with cancer.
- Recall in terms of True Positive and False Negative:

$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}}$$

- From the above formula in the image, we can analyze that as 'False Negative' decreases, our recall increases and vice-versa.

Recall/Sensitivity

- When to use Recall?
- Recall is used when we want to mostly focus on false-negative i.e to decrease false negative value thereby increase recall value.
- A question might arise why we want to mostly focus on a false-negative and not false positive.
- To answer this question, let us consider ‘Cancer Diagnosis‘ example...

Recall/Sensitivity

- **False Negative (FN)**
- It represents our predicted label is negative but the actual label is positive - wrongly predicted.
- Applying false negative on our example- it means we have predicted that the patient is not diagnosed with cancer but the actual patient is diagnosed with cancer.
- If this is the case, patient, as per prediction might not get treatment to cure cancer.
- But the truth is patient is diagnosed with cancer.
- Our wrong negative prediction will lead to death of a patient.
- So, **we mostly focus on false-negative** value and try to decrease it to the least possible value.

Recall/Sensitivity

- **False Positive (FP)**
- It represents our predicted label is positive but the actual label is negative - wrongly predicted.
- Applying false positive on our example- it means we have predicted that the patient is diagnosed with cancer but the actual patient is not diagnosed with cancer.
- If this is the case, patient, as per prediction will get check-up for cancer diagnosis.
- To his happiness, he will come to know that he is not diagnosed with cancer. Hurrah! He is free from cancer now.
- So, we don‘t much focus on false-positive value.

F1-Score

- F1-score is another one of the good performance metrics which leverages both precision and recall metrics.
- F1-score can be obtained by simply taking ‘Harmonic Mean‘ of precision and recall.
- Unlike precision which mostly focuses on false-positive and recall which mostly focuses on false-negative, **F1-score focuses on both false positive and false negative**.

F1-Score

- F1-score in terms of Precision and Recall;
- The F-Score is the Harmonic Mean of Precision and Recall:

$$F = \frac{2}{\frac{1}{\text{Recall}} + \frac{1}{\text{Precision}}}$$

- Alternatively:

$$F = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

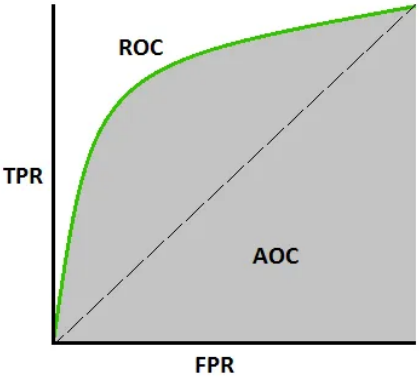
- When to use F1-score:
- As mentioned above, F1-score focuses on both false positive and false negative and try to decrease both false positive and false negative thereby increase F1-score.

AUC - ROC curve

- AUC - ROC is one of the most important performance metric used to check model performance.
- AUC - ROC is used for binary and also multi-class classification but mostly used in binary classification problems.
- In this lesson, we will consider a binary class classification.
- AUC-ROC is a graphical representation of model performance. ROC is a probability curve and AUC is the measure of separability.
- Depending on the threshold set, we can analyze how well our model has performed in separating two classes.
- Higher the AUC better is our model in separating two classes.

AUC - ROC curve

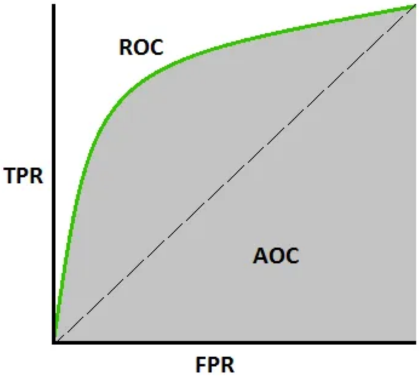
- Graphical representation of AUC - ROC
- Referring the image, we can see that AUC - ROC curve is plotted with FPR against TPR where FPR (False Positive Rate) is on X-axis while TPR (True Positive Rate) is on Y-axis.
- The green curve represents ROC curve while the area/region under ROC curve (green curve) represents AUC.



AUC - ROC curve

- **True Positive Rate (TPR)**: TPR is nothing but Recall / Sensitivity.
- The formula for TPR as follows:

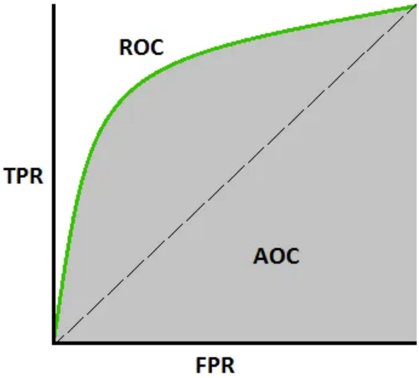
$$TPR = \frac{TP}{TP + FN}$$



AUC - ROC curve

- **False Positive Rate (FPR)**
- The formula for TPR as follows:

$$FPR = \frac{FP}{TN + FP}$$

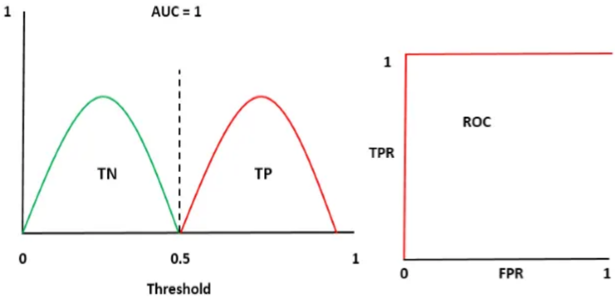


AUC - ROC curve

- Interpretation of AUC-ROC curve
- Let‘s now look into the analysis of binary class classification based on the AUC score and ROC curve...

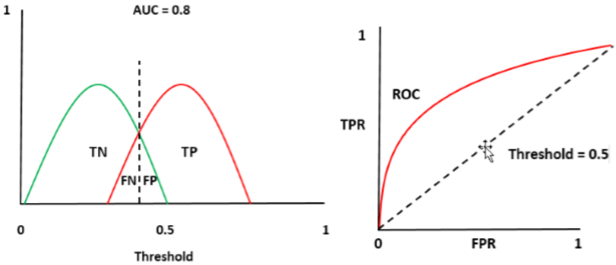
AUC - ROC curve

Threshold set to 0.5. There is no overlap between the two curves (green and red). This is the best model with AUC score of 1.0. This indicates that the probability of a model to separate positive and negative class is 1.0. In other words, we can say that there is 100% chance model can separate positive and negative class.



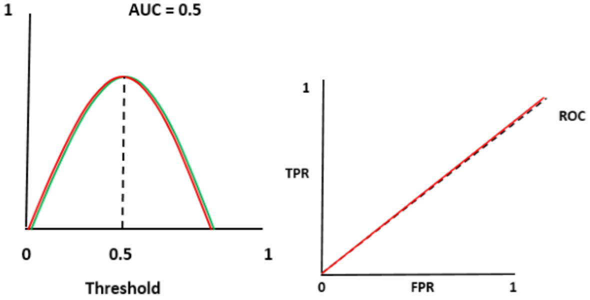
AUC - ROC curve

Threshold set to 0.5. There is a little bit of overlap between the two curves (green and red). This is a good model with AUC score of 0.8. This indicates that the probability of a model to separate positive and negative class is 0.8. In other words, we can say that there is 80% chance model can separate positive and negative class.



AUC - ROC curve

Threshold set to 0.5. We can see the full overlap between the two curves (green and red). This is a bad model with AUC score of 0.5. This indicates that the probability of a model to separate positive and negative class is 0.5. In other words, we can say that there is 50% chance model can separate positive and negative class.



Summary

- **Precision**: Precision measures what proportion of predicted positive label is actually positive. We mostly focus on false-positive value and try to decrease it to the least possible value thereby increase in precision value.
- **Recall**: Recall measures what proportion of actual positive label is correctly predicted as positive. We mostly focus on false-negative value and try to decrease it to the least possible value thereby increase in recall value.
- **F1-Score**: F1 Score is the ‘Harmonic Mean‘ of precision and recall. We focus on both false-positive and false-negative and try to decrease both false-positive and false-negative thereby increase F1 Score.
- **AUC- ROC curve**: It is a graphical representation of ROC curve and region/area under curve i.e AUC. It is mostly used in binary class classification. It interprets the probability or percentage of separability of positive and negative classes. Higher the AUC - ROC, better is our model in separating positive and negative classes.

Notebook Reference

- **chapter-1-4** Notebook

  In this notebook you will find an example of applying the metrics discussed in this section applied to a simple case of credit risk. Knowledge of logistic regression and credit risk issues is assumed.

Bibliography

- John C. Hull, *Machine Learning in Business: An Introduction to the World of Data Science*, Amazon, 2019.
- Paul Wilmott, *Machine Learning: An Applied Mathematics Introduction*, Panda Ohana Publishing, 2019.