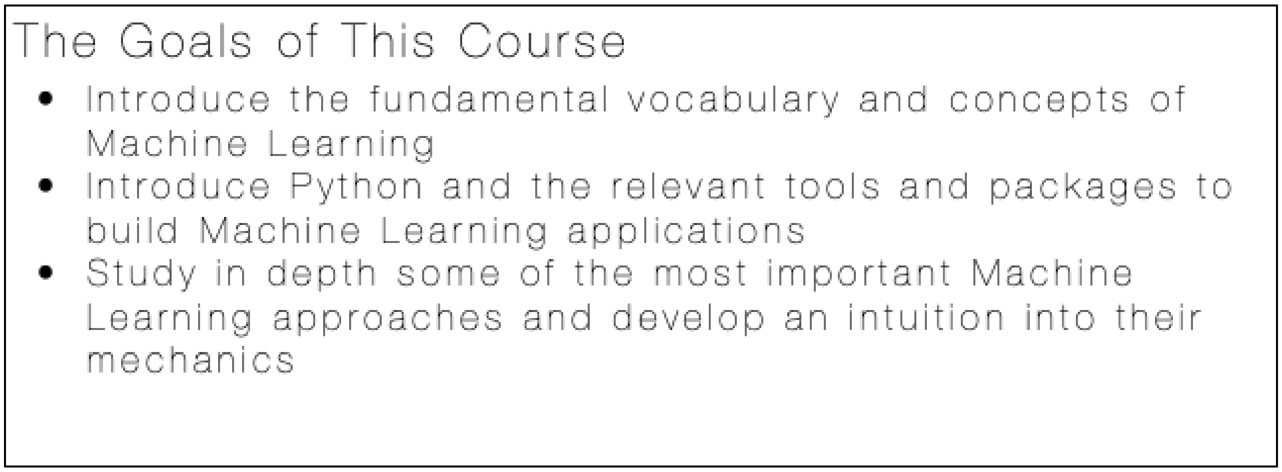
Week 1 : An introduction to Python and Machine Learning[1]

Ramzi Saouma

June 21, 2020

[1] This document has been written in LaTeX using the work of The Tufte-LaTeX Developers which is Inspired by Edward R. Tufte!

This course is about learning how to turn data into knowledge. While most of the cases and applications covered here focus on topics in finance and financial markets, the skills you acquire could be applied to other fields. The course is split into two parallel tracks. The first track revolves around learning the basics of Python. The second track centers around Machine Learning concepts and applications. During the first few weeks each track is taught separately. After the 5th week, when enough material is covered from each track, the two tracks will coverage into one body of subjects. (see figure 17).



The Goals of This Course

- Introduce the fundamental vocabulary and concepts of Machine Learning
- Introduce Python and the relevant tools and packages to build Machine Learning applications
- Study in depth some of the most important Machine Learning approaches and develop an intuition into their mechanics

Python Basics

Why Python?

Python is relatively a young language. It only appeared in 1991 and over the last 10 years it became one of the predominant languages for data scientists. According to the Kaggle Machine Learning and Data Science survey carried out in 2018, 83% of respondents said that they used Python on a daily basis[2]. While it is considered a scripting language [3], it can be powerfull in building data applications. It is supported with an extensive body of data analysis libraries some of which we cover in this course.

On an enterprise level, many applications are initially developed by specialists using very high level languages [4] such as R, SAS or

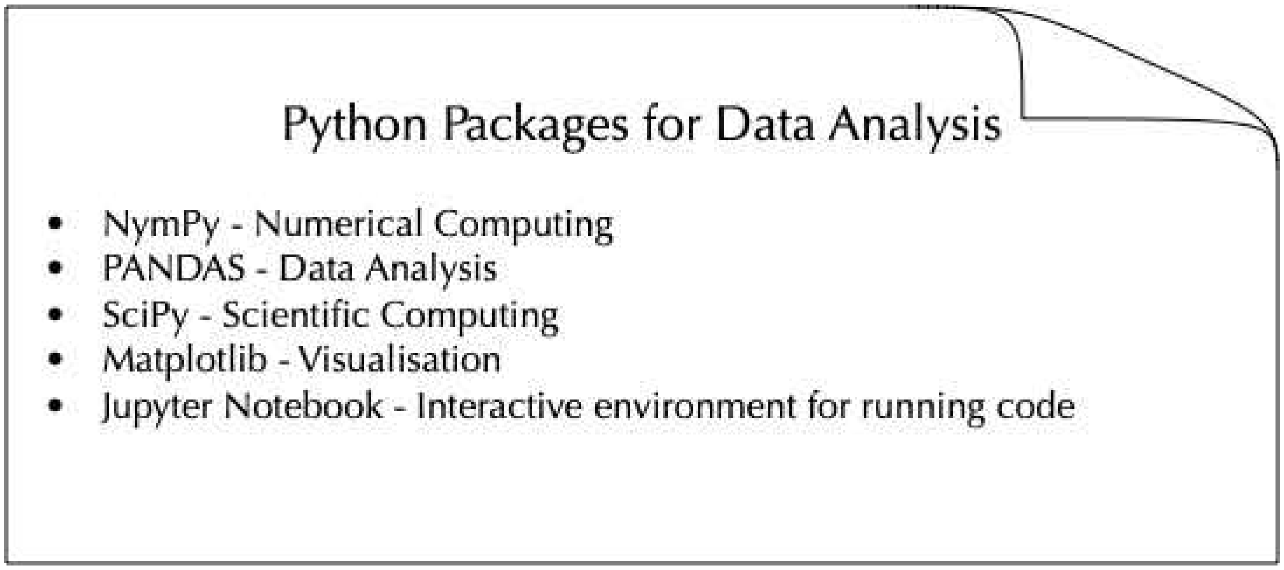
[2] `https://www.kaggle.com/sudhirnl7/data-science-survey-2018`

[3] Scripting Language: a language that can be used quickly to automate small tasks. Typically scripting languages are not compiled but interpreted. Making them slower to run but faster and easier to write.

[4] In comparison to low-level languages, a high level language is easier to use, it simplifies the interaction , making the process of developing a program more accessible.

Matlab as a proof of concept.At some point, when the application is ready for production it is rewritten in low level languages such as C++ or Java. What makes Python appealing to businesses is the fact that it can be an in-between language. It can be used for prototyping as well as for production.

Essential Python Libraries



Python Packages for Data Analysis

- NymPy - Numerical Computing
- PANDAS - Data Analysis
- SciPy - Scientific Computing
- Matplotlib - Visualisation
- Jupyter Notebook - Interactive environment for running code

Figure 1: List of packages

NumPy

Numerical Python is considered the main library for numerical computing in `Python`. It provides efficient data structures(e.g ⟨*ndarray*⟩), algorithms and library glue needed in most scientific applications involving numerical data in Python [5]. NumPy will be covered in Week 3.

[5] Wes McKinney. *Python for Data Analysis*. O'Reilly, second edition, 2018. ISBN 978-1-491-95766-0

pandas

Pandas offers a high level data structures and functions designed to facilitate the work of data scientist. It is built around a data structure called the DataFrame that is inspired from the R DataFrame. Looking like an Excel spreadsheet, it is effectively a table filled with data. Pandas provides a great range of methods to modify and operate on the table. In contrast to NumPy, which requires that all entries in an array be of the same type, pandas allows each column to have a separate type. Another great advantage of pandas is its ability to import from a great variety of file formats and databases.

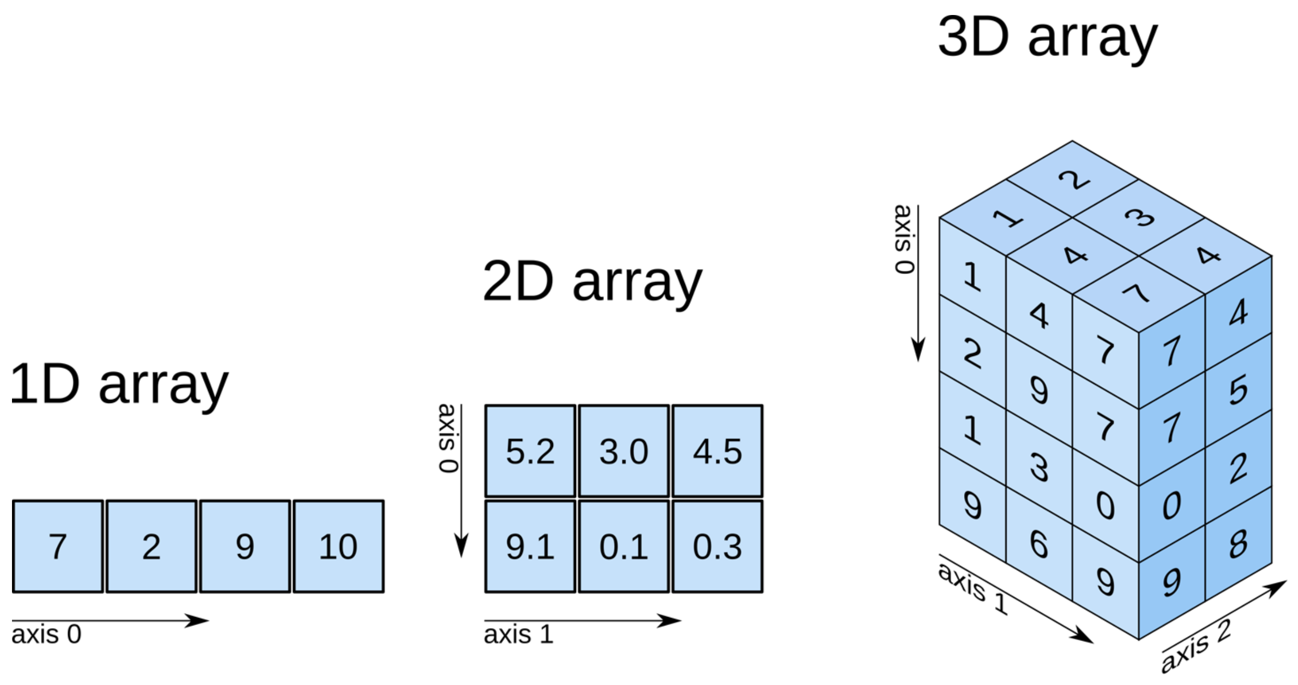


Figure 2: Numpy's ndarray illustrations

MATPLOTLIB

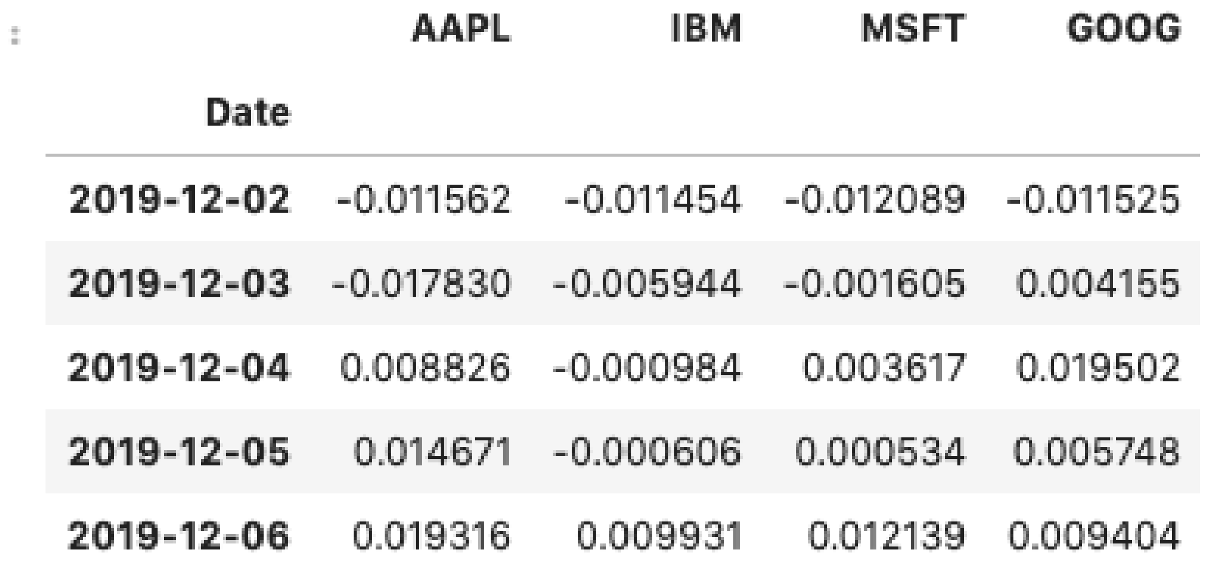
matplotlib is the main and most widely used visualization library in Python. It offers functions for making high-quality visualizations such as line charts, histograms, scatter plots. Visualizing your data and different aspects of your analysis can give you important insights, and we will be using matplotlib for all our visualizations. The high quality and versatility of the graphs makes it a great tool to present your findings as well.

SciPy

SciPy is a collection of functions for scientific computing in Python. It provides, among other functionality, advanced linear algebra routines, mathematical function optimization, signal processing, special mathematical functions, and statistical distributions. scikit-learn draws from SciPy's collection of functions for implementing its algorithms.

SCIKIT-LEARN

scikit-learn is an open source project, meaning that it is free to use and distribute, and anyone can easily obtain the source code to see what is going on behind the scenes. The scikit-learn project is constantly being developed and improved, and it has a very active



	AAPL	IBM	MSFT	GOOG
Date				
2019-12-02	-0.011562	-0.011454	-0.012089	-0.011525
2019-12-03	-0.017830	-0.005944	-0.001605	0.004155
2019-12-04	0.008826	-0.000984	0.003617	0.019502
2019-12-05	0.014671	-0.000606	0.000534	0.005748
2019-12-06	0.019316	0.009931	0.012139	0.009404

Figure 3: Pandas's DataFrame example

user community. It contains a number of state-of-the-art machine learning algorithms, as well as comprehensive documentation about each algorithm. scikit-learn is a very popular tool, and the most prominent Python library for machine learning. It is widely used in industry and academia, and a wealth of tutorials and code snippets are available online. scikit-learn works well with a number of other scientific Python tools, which we will discuss later in this chapter [6].

[6] Andreas C. Müller and Sarah Guido. *Introduction to Machine Learning with Python*. O'Reilly, second edition, 2017. ISBN 978-1-449-36941-5

Installing Python

The most commonly used free package to install Python is Anaconda distribution For the sake of the course we would be using Python 3.6. This could be found at (http://anaconda.com/downloads).

WINDOWS- To test if the application is configured properly, open the command prompt (cmd.exe) and try to lunch the interpreter by typing python.

```
C:/Users/wesm/python
Python 3.7.1 | Anaconda 4.1.1 (64-bit) | (default, Jul 5 2016, 11:41:13)
MSC v.1900 64 bit (AMD64)on win32
> > >
```

To exit shell type exit()

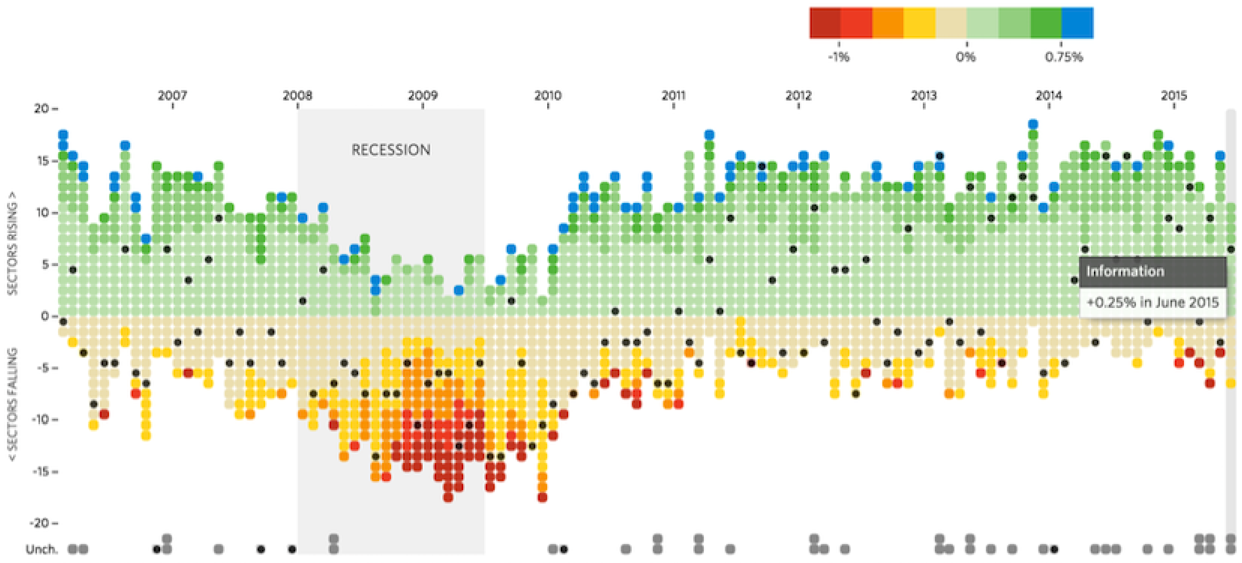


Figure 4: Matplotlib example

Apple(os x, macOS) - To test if the application is setup properly, type `$ python`

```
Python 3.7.1 (default, Dec 14 2018, 13:28:58)
Clang 4.0.1 (tags/RELEASE _ 401/final):: Anaconda, Inc. on darwin.
Type "help", "copyright", "credits" or "license" for more information.
> > >
```

To exit shell type `exit()`

IPython and Jupyter

IPython is a 'command-line based' application that allows better interactive Python interpreter. It is based on the idea of ⟨*execute-explore*⟩ workflow instead of the ⟨*edit-compile-run*⟩ workflow which is common to other languages. In recent times, the developers of IPython launched a new interactive, language agnostic application the `Jupyter Notebook`. The `Jupyter Notebook` is increasingly becoming popular by being adopted as the main platform for data scientists. It is a very powerful tool to use in the classroom, to develop teaching materials and to share lessons and tutorials.The Jupyter Notebook is an interactive environment for running code in the browser. It is a great tool for exploratory data analysis. While the Jupyter Notebook supports many programming languages, we only need the Python support. The Jupyter Notebook makes it easy to incorporate code, text, and images.

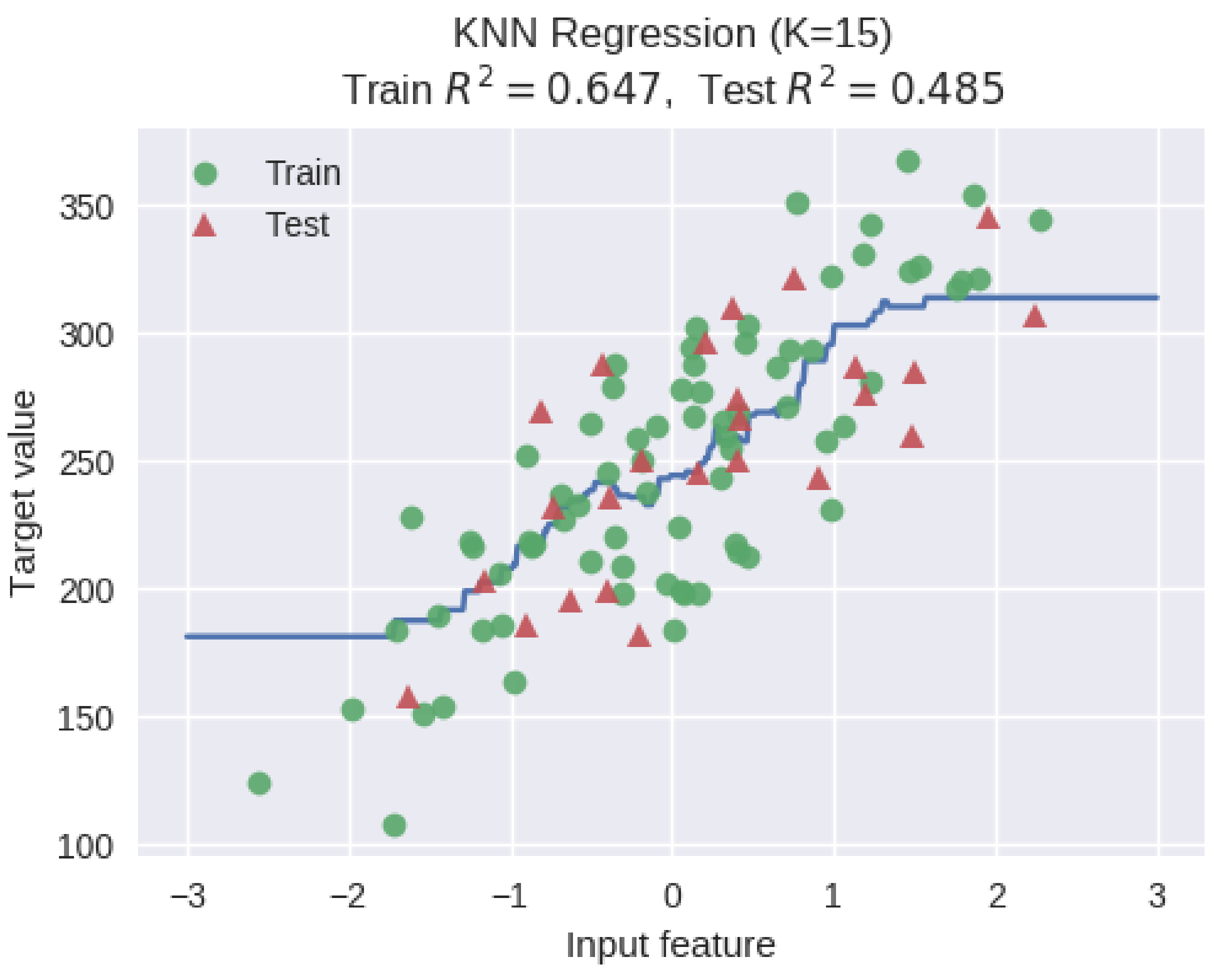


Figure 5: ScKit-Learn powerfull Machine Learning functionalities

Machine Learning: The Anti-Spam Case

Machine Learning is the systematic study of algorithms and systems that improve their knowledge or performance with experience[7]

This case is copied from chapter 1, 'Ingredient of machine learning', of Peter Flach's book. The goal is to get familiarized with some basic concepts and definitions. Every time a new concept is introduced, I will explicitly explain the definition on the right hand side of the documents with the concept written in red color.

On daily basis we receive and send emails. Most of the email providers equip our inbox with some sort of a spam filter. You might not be aware, but anti-spam filters rely heavily on machine learning. In fact most of them rely on 'SpamAssasin'[8] which is an open-source spam filter.

[7] Peter Flach. *Machine Learning, The Art and Science of Algorithms that Make Sense of Data*. Cambridge, first edition, 2012. ISBN 978-1-107-09639-4

[8] https://spamassassin.apache.org

It uses a scoring algorithm that integrate a wide range of advanced heuristic and statistical analysis tests on email headers and body text. Every test produces a score and if the total score is higher than 5 the email is classified as spam. It is worth noting that some tests could produce negative scores as well.

How to Build A Spam Filter?

Suppose we want build a spam filter. We will need a large training set[9] of emails which have been manually labelled spam or non-spam. Hence for these emails we already know the results of all the tests we would like to introduce. Our initial goal is to come up with a weighting system to apply on those tests, such as when all scores are added the spam emails get 5 or more as a score. To keep it simple,

[9] A training data set in machine learning is the actual data input used to train the model for performing various tasks.

Email	x_1	x_2	Spam?	$4x_1 + 4x_2$
1	1	1	1	8
2	0	0	0	0
3	1	0	0	4
4	0	1	0	4

Table 1: With the training dataset of 4 emails, the results of two tests are denoted by x_1 and x_2. Knowing that 1 is spam we opt for the weights 4 and 4. The results seems to be producing the desired outcome. Email 1 scores 8 (higher than 5 while the non spam emails score less than 5.

let's suppose we have two tests and four emails in our training data. Based on the data we have, both tests succeed for email 1, which has been hand labelled as spam. It is easy to see that assigning the weights 4 for each test will produce the desired outcome. In fact any weights between 2.5 and 5 can produce the desired result. If you are wondering how this classification problem is related to learning, your are not wrong. In fact the only reason why we call it learning is because the more examples and counter examples we give our model, the better our classifier will be. The concept of **improving performance with experience** is central to machine learning and it ties up nicely with the definition we presented at the beginning of this section. In practice, experience refers to our training data set and performance refers to its ability to filter out spam emails. One important fact to note is that when we talk about performance, we are referring to the emails we are going to receive in the future and are going to be filtered correctly. We do not care so much about the performance of the model on the training data since we already know which emails are spam! While it is important to have good performance on the training data, this remains just a mean to an end. In fact trying to hard to achieve good performance on training data could lead to another problem which is Over-fitting[10].

[10] Over-fitting is a problem that occurs when we fit the model to close to the training data. This leads to a failure in being able to generalise the model for future observations.

GENERALISATION is the core principle of machine learning. If we fail to generalize the knowledge that the spam filter learned to future

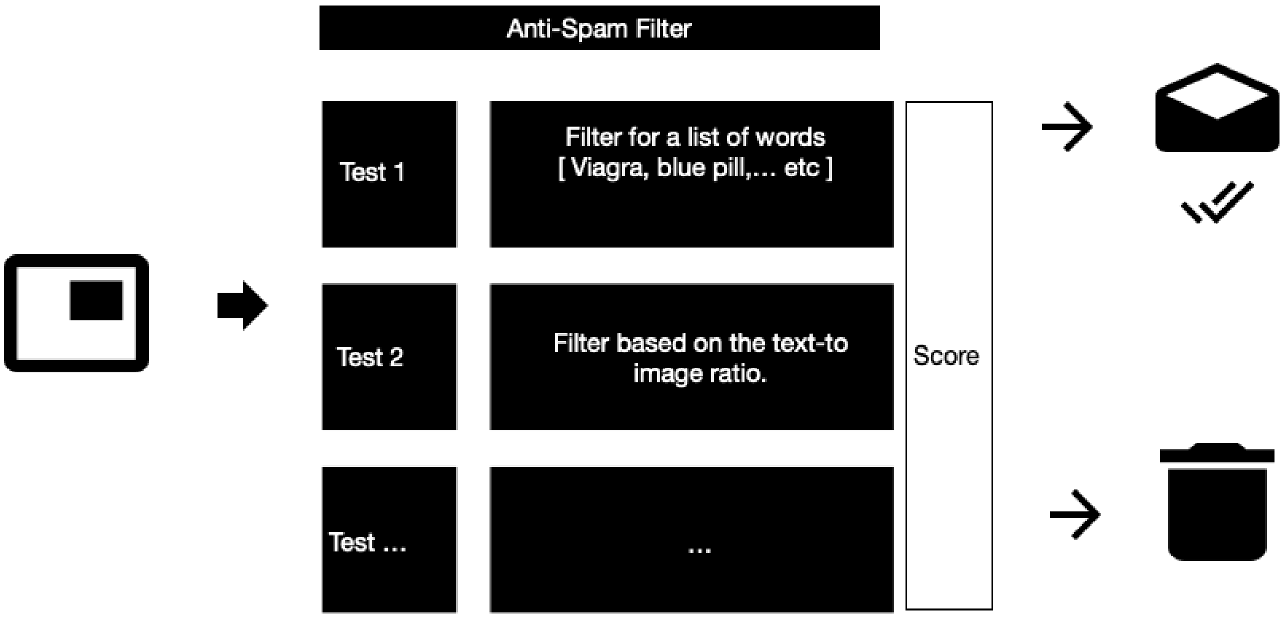

Anti-Spam Filter
Test 1
Filter for a list of words [Viagra, blue pill,... etc]
Test 2
Filter based on the text-to image ratio.
Test ...
...
Score

Figure 6: Anti-Spam filter

incoming email we then have another problem to solve. If it is not over-fitting it could be that the data doesn't represent the user's preference and behaviour. This problem is also known as Under-fitting [11]. If that is the source of the problem, machine learning could adapt to the behaviour and preferences of the user. It could as well be that the linear classification we defined by $4x_1 + 4x_2$ is bigger than 5 is not correct. Furthermore we have not talked about test themselves yet. What if the tests are not ideal or perhaps more tests are needed? Indeed how do we come up with those tests? Spam Assassin uses several tests which for example include the text-to-image ratio test. Moreover Spam Assassin scans the content of the email for words and phrases that increases or decreases the chances of the email to be spam. Surely words like "Viagra" or sentences such as 'confirm your account details' will increase the chances of the email being a spam. This type of test uses the text classification [12] techniques. Those tests store large dictionaries of words and sentences that are potential spam. For each of those words a certain probability of spam/not-spam is maintained. The probabilities are derived from the training data. Assume five emails in the training data has the word 'Viagra' and four out of five emails were tagged as spam, then naively, one would think that the probability of an new email being a spam given the occurrence of the word 'Viagra' is 80%.

But this is not the whole story because we have to take into account the prevalence of spam. Suppose we receive on average one spam e-mail for every six non-spam e-mails . This means that I

[11] Under-fitting occurs when a model is too simple — informed by too few features or regularized too much

[12] Classification has the goal to predict a class label for a list of possibilities. In the current AntiSpam case we are dealing with two classes.

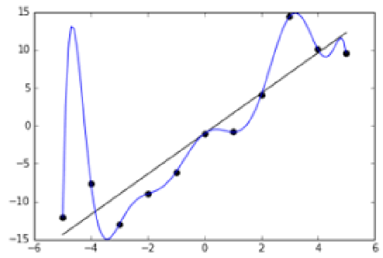


Figure 7: Example of over-fitting from Wikipedia. A simple straight line is a better fit than the the complex blue line.

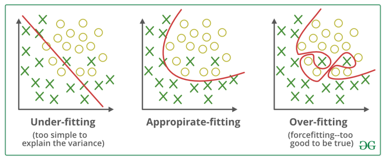

Under-fitting
(too simple to explain the variance)
Appropirate-fitting
Over-fitting
(forcefitting--too good to be true)

Figure 8: Generalisations of a classifier.

would estimate the odds of the next e-mail coming in being spam as 1:6. If after that we spot the word 'Viagra' is in the text, we need to adjust for the initial probability by combining the two together. According to Bayes' [13] rule to get the Joint Probability we should multiply the odds: 1:6 times 4:1 is 4:6, corresponding to a spam probability of 40%. Hence, despite the of the word 'Viagra', the safest bet is still that the e-mail is not spam, which does not sound very intuitive. More details on the calculation can be found in the last section of this note.

[13] Bayes' Rule: "Suppose you throw a pair of standard dice. The probability that the total is 10 is $\frac{1}{12}$ because there are thirty six ways the dice can come up, of which three(4 and 6, 5 and 5, and 6 and 4) give 10. If, However, you look at the first dice and see that it came up as 6, then the conditional probability that the total is 10, given the first dice is six, is $\frac{1}{6}$ (since that is just equivalent to the probability that the other dice comes up as 4). In general, the probability of A given B is defined to be the probability of A and B divided by the probability of B. In symbols:"

$$P[A \mid B] = \frac{P[AB]}{P[B]}$$

From which we derive a more interesting formula:

$$P[A \mid B] = \frac{P[B \mid A]P[A]}{P[B]}$$

Timothy Gowers. *The Princeton Companion to Mathematics*. Princeton Reference, first edition, 2008. ISBN 978-0-691-11880-2

The way to make sense of this is to realise that you are combining two independent pieces of evidence, one concerning the prevalence of spam, and the other concerning the occurrence of the word 'Viagra'. These two pieces of evidence pull in opposite directions, which means that it is important to assess their relative strength. What the numbers tell you is that, in order to overrule the fact that spam is relatively rare, you need odds of at least 6:1. 'Viagra' on its own is estimated at 4:1, and therefore does not pull hard enough in the spam direction to warrant the conclusion that the e-mail is in fact spam. What it does do is make the conclusion 'this e-mail is not spam' a lot less certain, as its probability drops from 6/7 = 0.86 to 6/10 = 0.60. The nice thing about this 'Bayesian' classification scheme is that it can be repeated if you have further evidence. For instance, suppose that the odds in favour of spam associated with the phrase 'blue pill' is estimated at 3:1 (i.e., there are three times more spam e-mails containing the phrase than there are ham e-mails), and suppose our e-mail contains both 'Viagra' and 'blue pill', then the combined odds are 4:1 times 3:1 is 12:1, which is ample to outweigh the 1:6 odds associated with the low prevalence of spam (total odds are 2:1, or a spam probability of 0.67, up from 0.40 without the 'blue pill'). That was a quick peek into what is commonly know at the Naive Bayesian Classifier. This is one of the dozen of algorithms we are going to cover in this course. Machine learning is all about designing the right features to construct the right models that achieve the desired tasks. In the anti-spam case we used the appearance of words like "Viagra" and "Blue pill" as features in the content of an email while applying the Naive Bayesian algorithm to build a model that has the task to filter our spam emails.

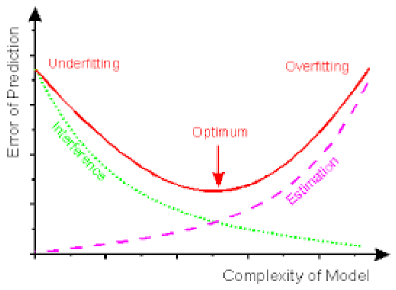
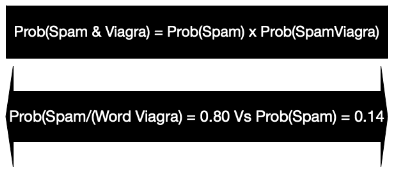


Figure 9: The trade-off between complexity and accuracy


Prob(Spam & Viagra) = Prob(Spam) x Prob(SpamViagra)
Prob(Spam/(Word Viagra) = 0.80 Vs Prob(Spam) = 0.14
Figure 10: Bayes Rule at play

Categories of Machine Learning

At the most basic level, machine learning can be divided into two main categories: supervised learning and unsupervised learning. "Supervised learning involves somehow modeling the relationship between measured features of data and some label associated with

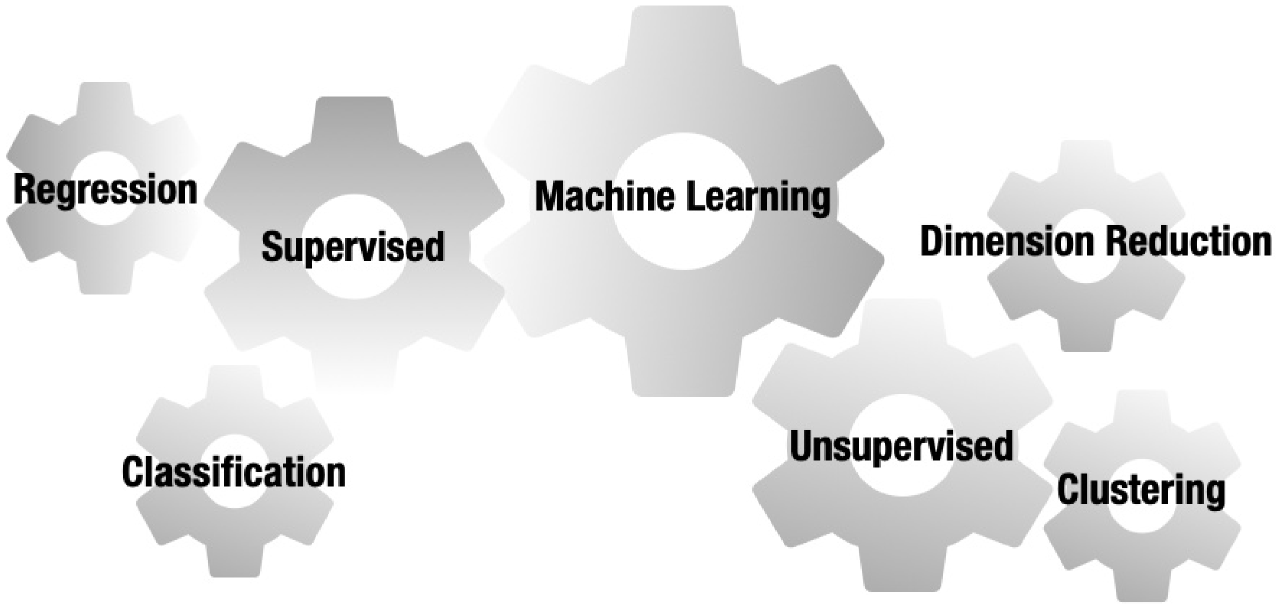


Figure 11: The Machine Learning Universe.

the data; once this model is determined, it can be used to apply labels to new, unknown data."[14]. Furthermore Supervised learning could be subdivided into classification tasks and regression[15] tasks: in classification, the labels are discrete categories or classes, while in regression, the labels are continuous numbers or quantities. Unsupervised learning involves modeling the features of a data set without reference to any label, and is often described as "letting the data set speak for itself." These models include tasks such as clustering and dimensionality reduction [16]. Clustering algorithms arranges data into similar groups, while dimensionality reduction tend to simplify the features by reducing them into main drivers and factors.

[14] Jake VandePlas. *Pytho Data Science Handbook*. O'Reilly, second edition, 2017. ISBN 978-1-449-36941-5

[15] Regression analysis is a set of statistical processes for estimating the relationships between a dependent variable (often called the 'outcome or target variable') and one or more independent variables (often called 'predictors' or 'features')

[16] Jake VandePlas. *Pytho Data Science Handbook*. O'Reilly, second edition, 2017. ISBN 978-1-449-36941-5

Supervised Learning: Classification

We start with an abstract classification, where based on two features, we want to be able to classify an input between two classes. Here we have a 2-dimensional data set which is used to train the model. In addition we have two classes represented by different colors/shape on the graph.

There are a many possible ways to model this classification task, but we will start by adopting a fairly simple one. It is based on the assumptions that the two classes can be separated by drawing a straight line through the plane between them, such that points on each side of the line fall in the same group. Looking at figure 13, by just using rough visual estimations we can potentially draw several

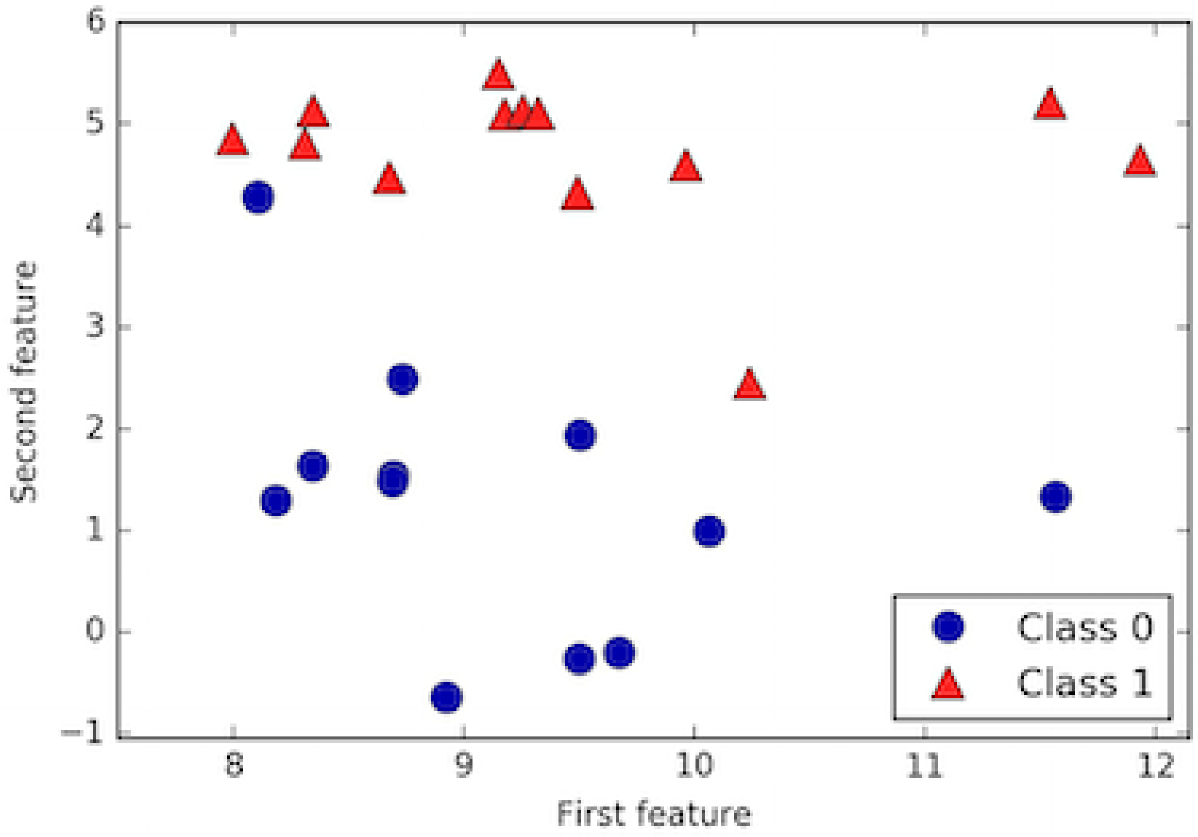


Figure 12: An abstract visualisation of a two features binary classification

acceptable separation lines. In this case the model parameters are the particular numbers describing the slope and intersection of that line for our data. Which line to opt for is learned from the data, which is often called training the model. Now that this model has been trained, it can be generalized to new, unlabeled data. In other words, we can take a new set of data, draw this model line through it, and assign labels to the new points based on this model. This stage is usually called prediction.

This reminds us of the previous of anti-spam filtering for email; in this case, we might use the following features and labels:

1. feature 1, feature 2, etc.: counting of key terms (“Viagra,” “blue pill,” etc.)

2. labels: “spam” or “not spam”

For the training set, these labels might be determined by individual inspection of a small representative sample of emails; for the remaining emails, the label would be determined using the model.

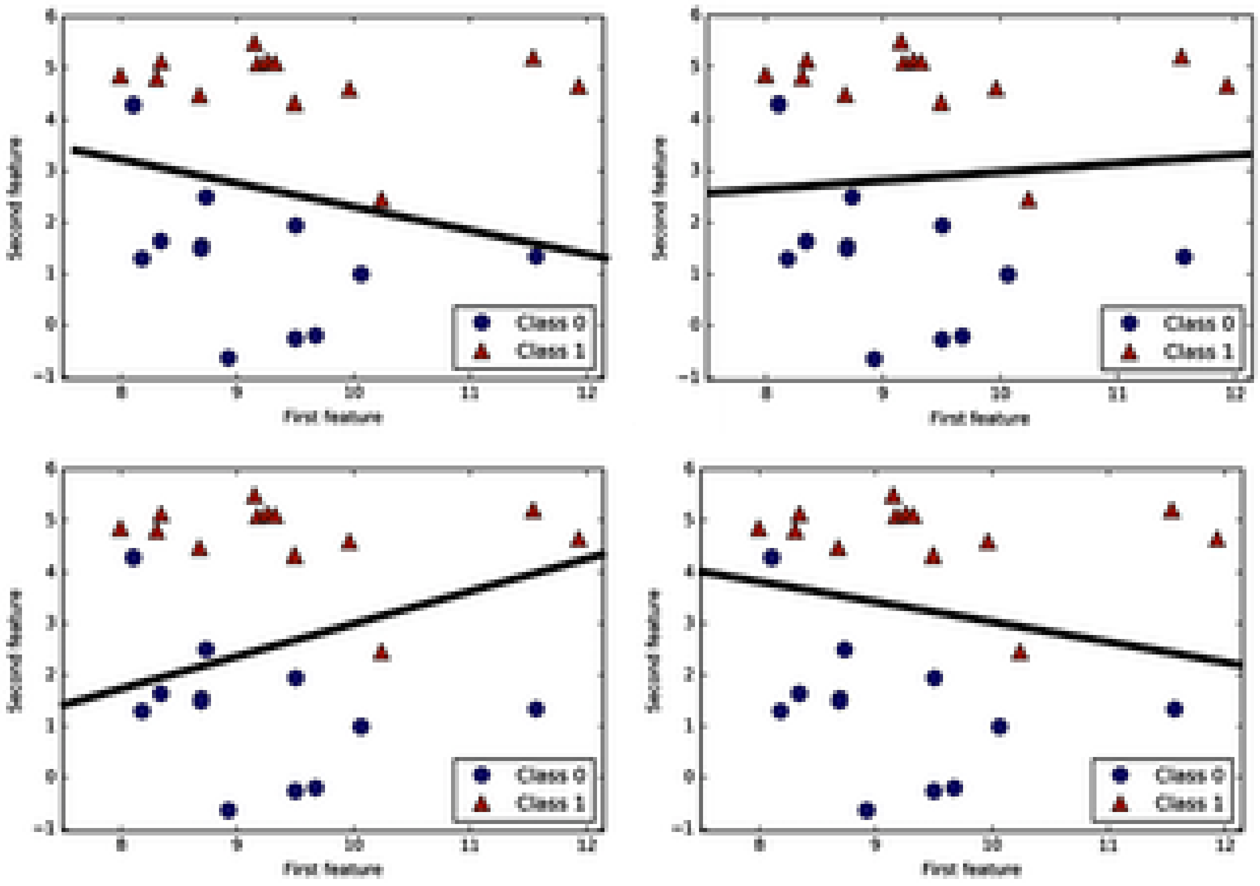


Figure 13: Separating the two classes with a straight line

For a suitably trained classification algorithm with enough well-constructed features (typically thousands or millions of words or phrases), this type of approach can be very effective.

Supervised Learning: Regression

In contrast with the discrete labels of a classification algorithm, we will next look at a simple regression task in which the labels are continuous quantities. On the x-axis we have one feature that varies between -2 and +2 and we aim to use it to predict the y-axis. The most basic and simple algorithm is to apply is the good old ordinary least square linear regression. But in this course, we will visit more elaborate and complex algorithms to fit a line into a continuous data target.

In Figure 15 you see four different fitting lines into a different data set. We start from the simplest linear regression (red) to the most complex over-fitted non-linear curve (light blue). Again the model

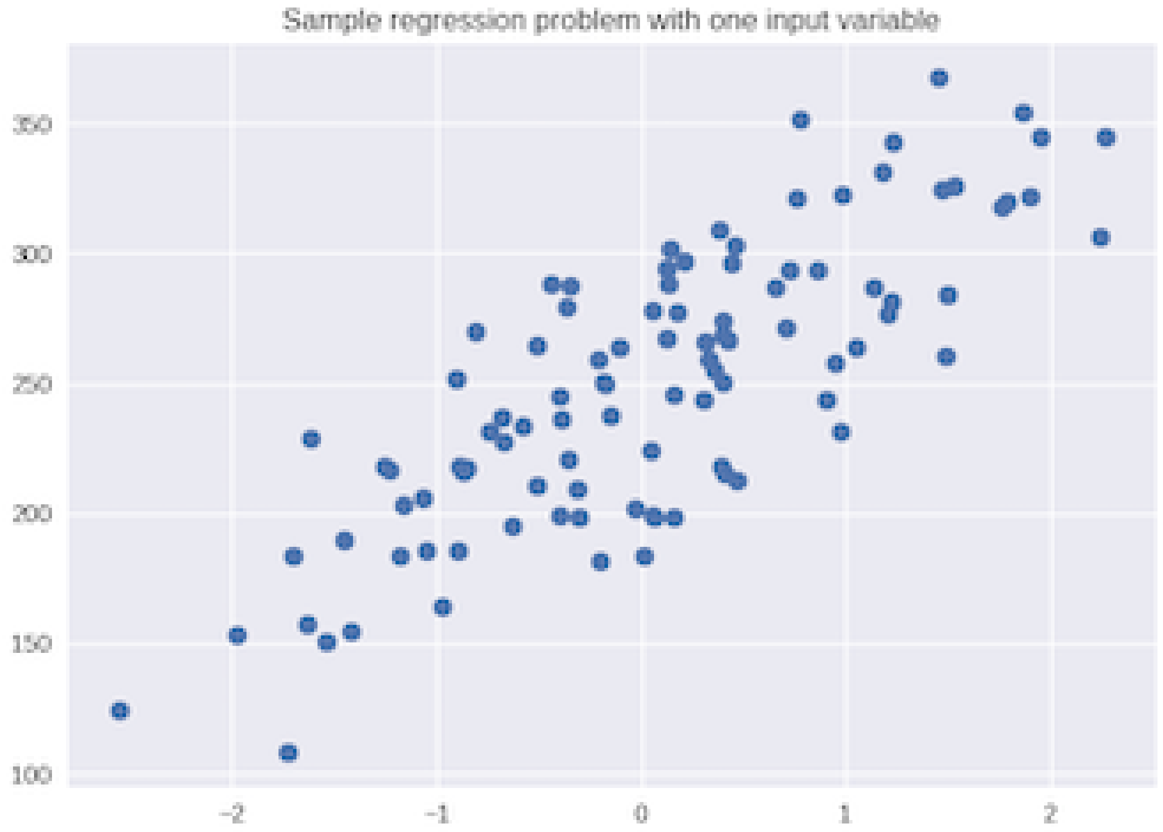


Figure 14: 1 Dimensional simple data set for regression

parameters in this case are the slope and the intersection of the lines.

The type of tasks to which we apply regressions is extremely vast. This could be applied to predicting the level of inflation in an economy based on certain levels of other economic indicators:

1. feature 1, feature 2, etc.: year-to-date average unemployment,interest rates, etc.)

2. labels: the level of inflation.

Unsupervised Learning: Clustering

The classification and regression example we considered cases of supervised learning algorithms, in which we our goal is to build a model that will predict labels for new data. Unsupervised learning involves models that describe data without reference to any known labels. One common case of unsupervised learning is "clustering," in which data is automatically assigned to some number of discrete

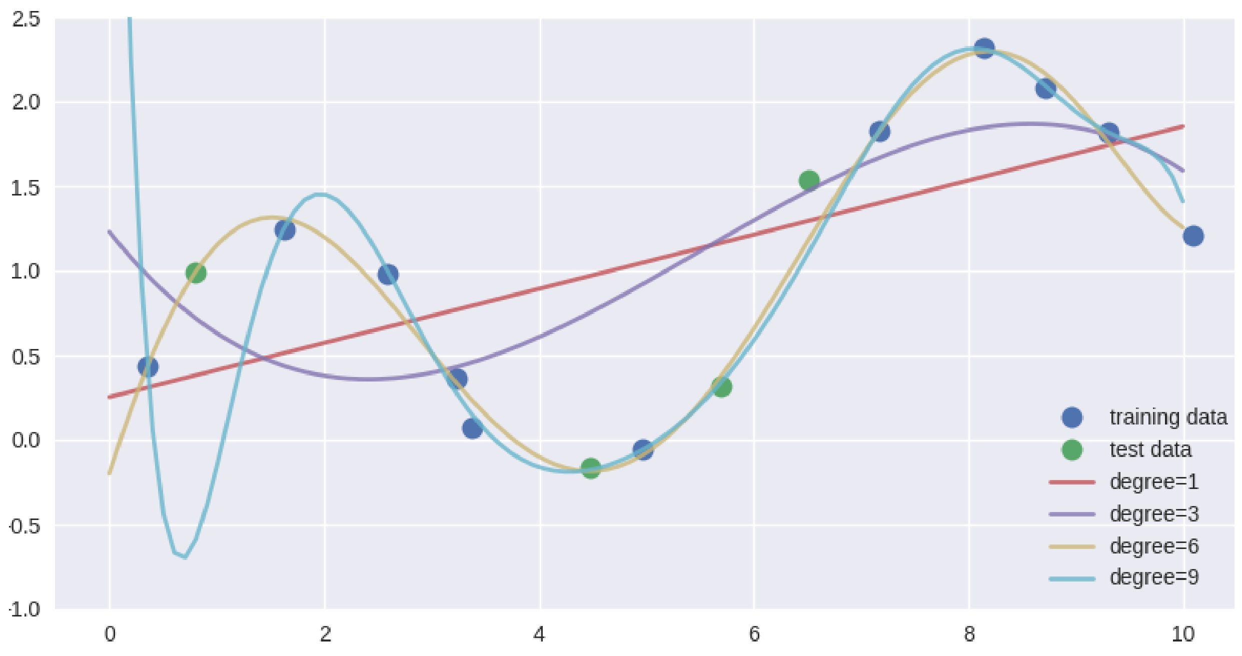


Figure 15: Regression fitting

groups. For example, we might have some two-dimensional data like that shown in Figure 16.

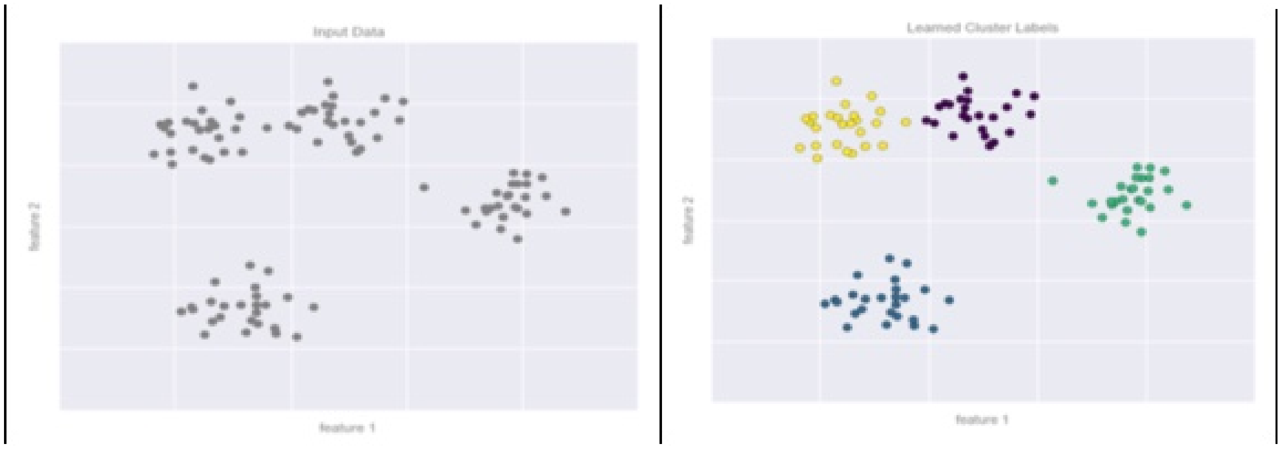


Figure 16: Clustering example

Contemplating the plot on the left hand side, it is clear that each of these points is part of a distinct group. We can discern 3 or maybe 4 different groups. Based on the features' input, a clustering algorithms will model the intrinsic structure of the data to determine which points could be clustered within the same group. A very fast and intuitive algorithm, known as *k*-means[17] algorithm, is applied and the output is seen on the right hand side of the figure where each color

[17] K-Mean fits a model consisting of *k* cluster centers; the optimal centers are assumed to be those that minimize the distance of each point from its assigned center. Again, this might seem like a trivial exercise in two dimensions, but as our data becomes larger and more complex, such clustering algorithms can be employed to extract useful information from the dataset. We will discuss the k-means algorithm in more depth

corresponds to a different cluster. There is a multitude of applications for clustering algorithms. Companies use clustering to categorise their clients and use the output while designing their marketing or sales strategy. The features could be anything ranging from client's income to spending habits.

Unsupervised Learning: Dimensionality reduction

In dimensionality reduction we aim at inferring a structure from looking at the data. It is a bit more abstract than the examples we looked at before, but generally it seeks to pull out some low-dimensional representation of data that in some way preserves relevant qualities of the full dataset. Principal Component Analysis (PCA) is one example of dimensionality reduction. PCA is a statistical method that applies an orthogonal transformation to convert the dataset of possibly correlated features into a set of linearly uncorrelated (aka orthogonal) variables called Principal Components. We will revisit this with more concrete examples later in the course. One common use is when we are dealing with the large number of features that we suspect to be correlated. Dimensionality reduction has a lot of advantages especially when it comes to visualisation. Being able to explore a dataset visually if it is 2 dimensional rather than 10 is powerful.

Summary

So far we have seen a few intuitive examples of some of the basic types of machine learning approaches. Obviously, we have just touched the surface of those methods, but I hope this section was enough to give you a basic idea of what types of problems machine learning approaches can solve. To summarize:

SUPERVISED LEARNING
  Models that can predict labels based on labeled training data

CLASSIFICATION
  Models that predict labels as two or more discrete categories

REGRESSION
  Models that predict continuous labels

UNSUPERVISED LEARNING
  Models that identify structure in unlabeled data Clustering
  Models that detect and identify distinct groups in the data

DIMENSIONALITY REDUCTION

Models that detect and identify lower-dimensional structure in higher- dimensional data

More details on the Spam Filter Calculation and Bayes' Theorem

Suppose that on average, the word "Viagra" occurred in four spam e-mails and one non-spam email. Assume also that the odds of an email being spam is 1:6. Mathematically, this means that

$$\begin{aligned} P[\text{Viagra}|\text{Spam}] &= \frac{4}{5} \\ &\text{and} \\ P[\text{Viagra}|\text{Not Spam}] &= \frac{1}{5} \\ &\text{and} \\ P[\text{Spam}] &= \frac{1}{7} \end{aligned}$$

Bayes' rule tells us that the probability that the email is spam given that it contains the word "Viagra" is computed as:

$$\begin{aligned} P[\text{Spam}|\text{Viagra}] &= \frac{P[\text{Spam} \cap \text{Viagra}]}{P[\text{Viagra}]} \\ &= \frac{P[\text{Viagra}|\text{Spam}]P[\text{Spam}]}{P[\text{Viagra}]} \end{aligned}$$

Now, the Law of Total Probability [18] tells us that $P[\text{Viagra}]$ can be computed as:

$$\begin{aligned} P[\text{Viagra}] &= P[\text{Viagra}|\text{Spam}]P[\text{Spam}] \\ &+ P[\text{Viagra}|\text{Not Spam}]P[\text{Not Spam}] \\ &= \left(\frac{4}{5} \times \frac{1}{7}\right) + \left(\frac{1}{5} \times \frac{6}{7}\right) \\ &= \frac{10}{35} \end{aligned}$$

Putting it all together, one obtain:

$$\begin{aligned} P[\text{Spam}|\text{Viagra}] &= \frac{\frac{4}{5} \times \frac{1}{7}}{\frac{10}{35}} \\ &= \frac{4}{10} \\ &= 0.4 \end{aligned}$$

[18] The Law of Total Probability states that for a given sample space that can be portioned into countably infinite pairwise disjoint events $B_n, n = 1, 2, 3 \cdots$, then the probability of any event A, $P[A]$ can be computed as: $P[A] = \sum_n P[A \cap B_n]$ = $\sum_n P[A|B_n]P[B_n]$, where the second equality follows from Bayes' rule.

References

Peter Flach. *Machine Learning, The Art and Science of Algorithms that Make Sense of Data*. Cambridge, first edition, 2012. ISBN 978-1-107-09639-4.

Timothy Gowers. *The Princeton Companion to Mathematics*. Princeton Reference, first edition, 2008. ISBN 978-0-691-11880-2.

Wes McKinney. *Python for Data Analysis*. O'Reilly, second edition, 2018. ISBN 978-1-491-95766-0.

Andreas C. Müller and Sarah Guido. *Introduction to Machine Learning with Python*. O'Reilly, second edition, 2017. ISBN 978-1-449-36941-5.

Jake VandePlas. *Pytho Data Science Handbook*. O'Reilly, second edition, 2017. ISBN 978-1-449-36941-5.

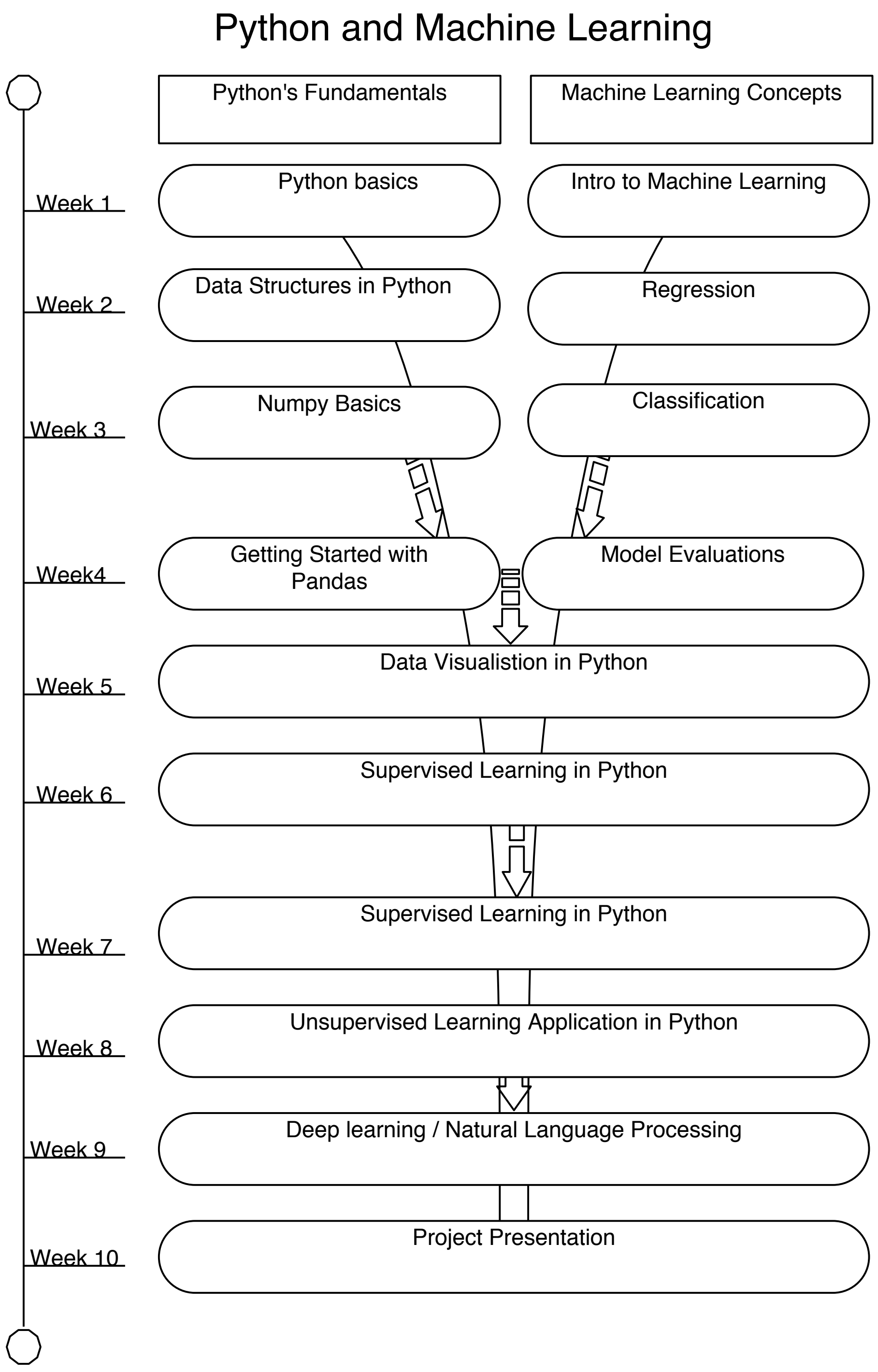

Python and Machine Learning
Python's Fundamentals
Machine Learning Concepts
Week 1
Python basics
Intro to Machine Learning
Week 2
Data Structures in Python
Regression
Week 3
Numpy Basics
Classification
Week4
Getting Started with Pandas
Model Evaluations
Week 5
Data Visualistion in Python
Week 6
Supervised Learning in Python
Week 7
Supervised Learning in Python
Week 8
Unsupervised Learning Application in Python
Week 9
Deep learning / Natural Language Processing
Week 10
Project Presentation

Figure 17: Syllabus