MS4S10 - Machine Learning and Decision Making

Supplementary Resources

Moizzah Asif, J418, moizzah.asift@southwales.ac.uk

University of South Wales

Contents

Chapter 1

Week 1

1.1 Setting up Python

1.1.1 Python Resources

There are three main Python releases: Python 1,2 and 3. We will use **Python 3** as all new features of Python are integrated in this line of release. You may find following link useful `docs.python.org/3/`.

You can install python manually or via anaconda. We'll see both types of installations in the following subsections.

1.1.2 Manual Installation

Figure 1.1: Python's default installation window - (the image will be updated later)

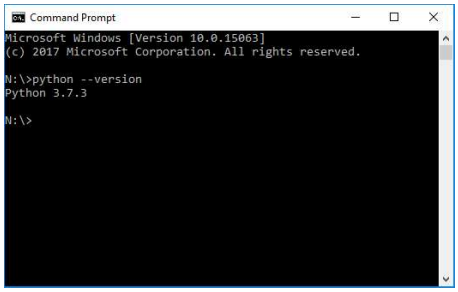

Figure 1.2: Checking python's version on your computer - (the image will be updated later)

1. Download the latest stable version (3.9.0 for now), or later 3.8.x from the following link: `https://www.python.org/downloads/`
2. Find a version that suits your computer and operating system, we are assuming 64 bit windows operating system.

3. Run the executable file, and select "install now" as shown in Fig 1.1.
4. Make sure you have checked the second box '**Add Python 3.9/3.8 to PATH**' as shown in Fig 1.1
5. Good job! you have just installed Python. Now we need to confirm with our computer that it recognises the newly installed version.
6. Type "cmd" in the search bar, to run command line.
7. Type the following command to check python version your computer is running: *python –version*
8. Your computer's response should look similar to the one shown in Fig 1.2

setting up IDE

An IDE is acronym for Integrated Development Environment: a platform to code and execute python's programming scripts. We can run our scripts on the environment which came with Python installation, but lets just not be Old School! Lets learn how to setup Jupyter Notebook to run our python scripts.

1. Run cmd (command line) as administrator.
2. Type the following command to ensure that PIP (a package management and software installation system) is running its late version, *python -m pip install –upgrade pip*
3. Run the following command to install Jupyter notebook, *python -m pip install jupyter*
4. Type the following command on cmd to launch and ensure that Jupyter notebook is installed, *jupyter notebook*
5. Jupyter notebooks open in computer's default browser, and look like as shown in Fig 1.3
6. If you would like to change the working directory, and avoid saving scripts in default folder, type the following command on a new cmd window. It will set the working directory in your desired folder and all the scripts will be saved here from now on. Please modify and add the folder names as required.
   jupyter notebook --notebook-dir=\ folder1\ folder2

executing first python script in Jupyter Notebook

This is the easiest bit!

1. Open Jupyter notebooks from cmd following the instruction given in 1.1.2.
2. Select new python3 script as shown in Fig 1.5.
3. Type the following piece of code and press RUN from the utility icons bar:
   print ("Hello World!")
4. Your output should like the one in Fig 1.6.

1.1.3 Anaconda Installation

Anaconda installs IDEs and several important packages like NumPy, Pandas, and so on, and this is a really convenient package which can be downloaded and installed.

1. Download the appropriate anaconda installer from `https://www.anaconda.com/distribution/`
2. Go through the installation procedure with the installer.
3. After the installation is complete, search for Anaconda Navigator in the Start menu. The navigator homepage looks similar to as shown in 1.7

Executing first python script with Jupyter Notebook

1. Simply click the Jupyter notebook icon from navigator to open Jupyter notebook.
2. follow steps 2 - 4 from 1.1.2 for executing first python script from Jupyter notebook

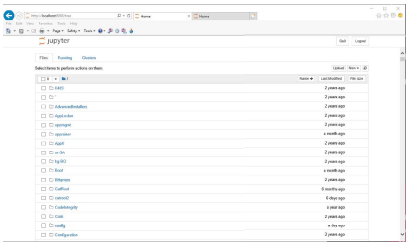

Figure 1.3: Jupyter Notebook in browser

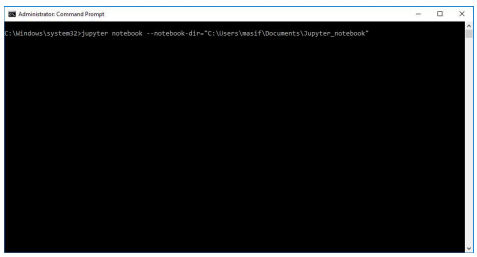

Figure 1.4: Changing Jupyter notebook directory

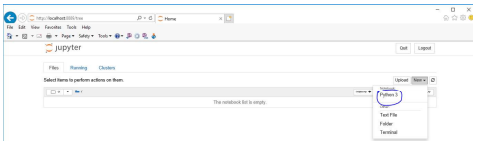

Figure 1.5: Creating new python script

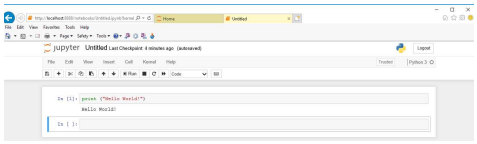

Figure 1.6: First python script: hello world

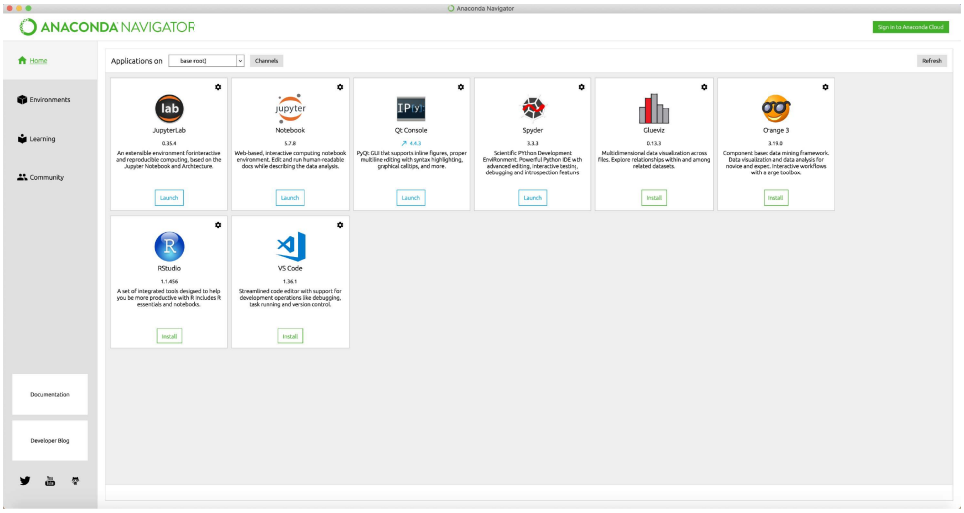

Figure 1.7: Anaconda navigator home

Note

Anaconda installation will download all available packages and libraries whereas with manual installation any required libraries and packages would need to be installed manually from command line using pip installer. please refer to `https://packaging.python.org/tutorials/installing-packages/` for further information on how to use pip installer.

Please refer to the follwing url for further reading and developing an understanding of Jupyter notebooks before the lecture,

1.2 Python packages to be used in lecture

1. pandas `https://pandas.pydata.org/`, `https://bit.ly/2Gjup5s`
2. matplotlib `https://matplotlib.org/index.html`
3. numpy
4. SciPy `https://www.scipy.org/`

5. Scikitlearn `https://scikit-learn.org/stable/`

1.3 Dataset to be used in lecture

The dataset to be used in today's lectures is originally available at `http://www.dcc.fc.up.pt/~ltorgo/Regression/cal_housing.html`. This dataset appeared in a 1997 paper titled "Sparse Spatial Autoregressions" by Pace, R. Kelley and Ronald Barry, published in the Statistics and Probability Letters journal. They built it using the 1990 California census data. It contains one row per census block group. A block group is the smallest geographical unit for which the U.S. Census Bureau publishes sample data (a block group typically has a population of 600 to 3,000 people).

We are adapting the slightly tweaked version of this dataset available at `https://github.com/ageron/handson-ml/tree/master/datasets/housing`. The author of the book "Hands on machine learning with scikit learn and tensorflow" has made tweaks as described in the previously provided url for learning purposes.

1.4 Misc. concepts required

1.4.1 Distance metrics

Numerical values

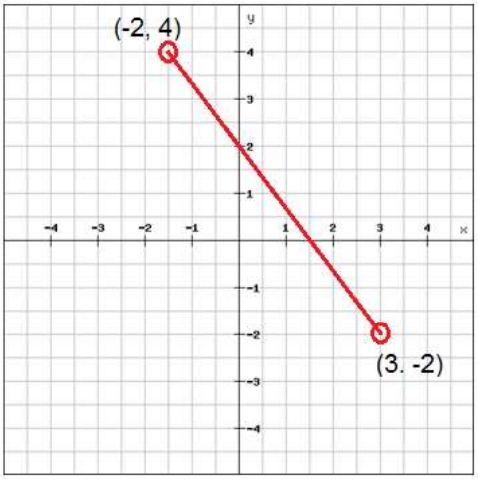

Figure 1.8: line segment PQ, p(-2,4) and q(3,-2)

1. **Euclidean Distance**

The Euclidean distance between points **p** and **q** is the length of the line segment connecting them, $\overline{\mathbf{pq}}$.
So, if **p** and **q** lie on a two-dimensional Cartesian plane (axis - x and y), and **p** is at points (x_1,y_1) and **q** is at points (x_2, y_2), then distance $d_E(p, q)$ can be calculated as follows:

$$d_E(p, q) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Example: using the plot from fig 1.8

$$d_E(p, q) = \sqrt{(3 - (-2))^2 + (-2 - 4)^2}$$
$$d_E(p, q) = 7.81$$

2. **Manhattan Distance** The sum of the lengths of the projections of the line segment between the points onto the coordinate axes. More formally

$$d_M(p, q) = \sum_{i=1}^{n} |p_i - q_i|$$

So, if **p** and **q** lie on a two-dimensional Cartesian plane (axis - x and y), and **p** is at points (x_1,y_1) and **q** is at points (x_2, y_2),then distance $d(p, q)$ can be calculated as follows:

$$d_M(p, q) = |x^2 - x^1| + |y^2 - y^1|$$

Example: using the plot from fig 1.8

$$d_M(p, q) = |3 - (-2)| + | - 2 - 4|$$
$$d_M(p, q) = 11$$

Categorical values

1. **Hamming Distance**

This distance is computed by overlaying one string over another and finding the places where the strings vary.
Example: consider the two words *'text'* and *'test'*. If you overlay them on each other as follows:

text
test

let's underline the places where the strings vary:

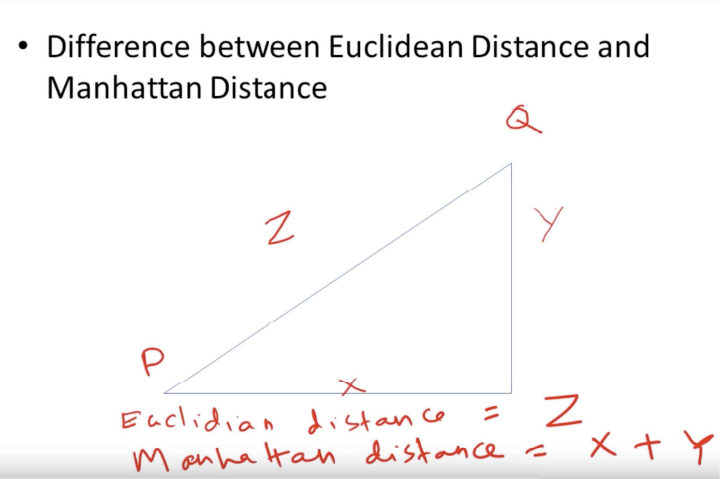

Figure 1.9: Euclidean distance vs Manhattan Distance

$$te\underline{x}t$$
$$te\underline{s}t$$
so $d_H(test, text) = 1$

Let's take another example:

$$arrow$$
$$arow$$

let's underline the places where the strings vary:

$$ar\underline{row}$$
$$ar\underline{ow}$$
so $d_H(arrow, arow) = 3$
note: the blank space is underlined after 'w' in *arow*

2. **Levenshtein Distance**

   This distance is computed by finding the number of edits which will transform one string to another. The transformations allowed are insertion — adding a new character, deletion — deleting a character and substitution — replace one character by another.

   For the *'text'* and *'test'* example, only one substitution/replacement needs to be done, i.e. replacing 's' with 'x'. So, $d_L(test, text) = 1$

   Similarly, for the *'arrow'* and *'arow'* example, only one insertion needs to be done, i.e. adding another 'r' after the first 'r' in *arow*. So, $d_L(arrow, arow) = 1$

1.4.2 Variance

If you are not sure what variance is then, please read through the following page to develop an understanding of the statistical measure called Variance. `https://bit.ly/2YSVdTB`

1.4.3 Degenerate distribution

If you do not have a basic understanding of what degenerate distribution of a random variable refers to, you may as well find this introduction in the following URL useful for the lecture to follow. `https://bit.ly/2TgBI68`