Data Manipulation in Python Part 2

Ashish Rajendra Sai

Ph.D. Candidate - University of Limerick

August 18, 2020

Who am I?

Ashish Rajendra Sai

- Final year Ph.D. student at the University of Limerick under the supervision of Dr. Jim Buckley
- Doctoral Researcher in Lero: The Irish Software Research Centre
- Currently a Visiting Researcher at the University of California, Berkeley
- Long time interest in teaching programming to non-cs professionals and students:
    - Programming can be super fun and useful once we jump past some hoops

---

How today might work

Zoom Meeting:

- I will present in approximately 15 - minute slots, over an hour.
- I would like to take all questions via the zoom 'chat'
    - What is a function? Why does it matter? ....
    - The importance of questions in this session.
- I will answer them in real-time, at the end of the 15 minutes or at the end of the session entirely (if I think they will be covered by something later)

Welcome to today's class!

Today's Agenda

1. Overview of Lero Data Manipulation Python Part 1
2. How to write and run Python code recap
3. Introduction to Student Excel Sheet and tasks
4. Implement basic data manipulation functions
5. Useful graphing and data manipulation libraries in Python
6. Predictive analysis in Python: Linear Regression
7. Where to go from here?

Overview of Lero Data Manipulation Python Part 1

- 1 What a variable is and how to declare and use it?
- 2 Complex data types such as a List and Dict
- 3 How to read these data types by using a loop?
- 4 How to use predefined functions?
- 5 How to use external libraries?

- 1 What a variable is and how to declare and use it?

In [2]:
```
Name = "John Wick"
Age = 28
Percentage = 76.54
# print(type(Percentage))
```

- 2 Complex data types such as a List and Dict

In [4]:
```
ResultsBySubject = [74,38,85,92]
# print(type(ResultsBySubject))
```

In [6]:
```
ResultsBySubject = {"CS":74,"History":38,"Arts":85,"Geography":92}
# print(type(ResultsBySubject))
```

- 3 How to read these data types by using a loop?

In [9]:
```
ResultsBySubject = [74,38,85,92]

for result in ResultsBySubject:
    print(result)
```

```
74
38
85
92
```

In [14]:
```
ResultsBySubject = {"CS":74,"History":38,"Arts":85,"Geography":92}

for subject in ResultsBySubject:
    print(subject)
```

```
CS
History
Arts
Geography
```

- 4 How to use predefined functions?

In [16]:
```
ResultsBySubject = {"CS":74,"History":38,"Arts":85,"Geography":92}

print(type(ResultsBySubject))
print(len(ResultsBySubject))
```

```
<class 'dict'>
4
```

- 5 How to use external libraries?

In [31]:
```
import pandas as pd

studentResults = pd.read_csv('/home/nbuser/studentResults.csv')

Luke = studentResults.iloc[0]

print(Luke)
```

```
Name       Luke
Age          24
Science      45
Maths        10
Biology      53
CS           84
History      89
Name: 0, dtype: object
```

What do you need to be able to code in Python?

- To make our life easy we are going to use free online hosted integrated development environments for Python.
- No need to download or set up anything: Be careful with your company policies, you do not want sensitive data on these third party computers.
- List of free online IDEs to get you started;
    - Google Colaboratory (Colab) Link: https://colab.research.google.com/ (https://colab.research.google.com/)
    - Microsoft Azure Notebooks Link: https://notebooks.azure.com/ (https://notebooks.azure.com/) (This is what we will use today)
    - CoCalc Link: https://cocalc.com/ (https://cocalc.com/) (Good but slow :( )
    - Kaggle Kernels Link: https://www.kaggle.com/ (https://www.kaggle.com/) (Mostly for data science nerds)
    - You can always use vanilla Jupyter Notebook on your own PC Link: https://jupyter.org/ (https://jupyter.org/)

Introduction to Student Excel Sheet and tasks

- This Excel sheet was used by Dr. Jim Buckley in the first workshop.
- We have a list of 40 students undertaking a hypothetical module called Resp. Digital Citizenship.
- We need to use Python to find;
    - 1) Calculate Marks for Mid-Term and Final Exam
    - 2) Calculate total marks
    - 3) Assign Grade to Each Student
    - 4) Calculate Ranks for Students
    - 5) Create E-mail addresses using Student ID
    - 6) Calculate Average score
    - 7) Find Min and Max Score
    - 8) Plot a histogram of final scores

Implement basic data manipulation functions

- We will start by reading the xlsx file

In [88]:
```
import pandas as pd

csis9999 = pd.read_excel("CSIS9999.xlsx")

# Note that we have used a function called head to look at the top 5 entries in th
e Excel file
# csis9999.head()
# Here are some other ways of exploring the data before we start manipulating it

# print(csis9999)
# csis9999.describe()
```

- 1) Calculate Marks for Mid-Term, Project and Final Exam
- Our Excel sheet contains Mid Term (out of 30), Practical (out of 20) and final exam (out of 100)
- Our target distribution is Mid Term (out of 15), Practical (out of 20) and final exam(out of 65)

In [109]:

```
import pandas as pd


csis9999 = pd.read_excel("CSIS9999.xlsx")

# We start by creating two new columns for Mid_Term_15 and Final_Exam_65

csis9999['Mid_term_15'] = 0.0

csis9999['Final_Exam_65'] = 0.0

# print(csis9999)

# NOTE: Pandas will create a dataframe out of your Excel sheet, to read the values
in our excel sheet we can use the method itertuples()
for student in csis9999.itertuples():
    # print(student)
    # Here we are using an inbuilt function of Pandas Data Frame to assign a value
to a column
    csis9999.at[student.Index, 'Mid_term_15'] = student.Mid_term_30/2
    # We can repeat the same process for Final_Exam_65
    csis9999.at[student.Index, 'Final_Exam_65'] = (student.Final_Exam_100/100)*65
csis9999.head()
```

Out[109]:

	Student_ID	Student_Surname	Forename	Course	Mid_term_30	Practical_20	Final_Exam_100	Mid_term_15	Final_Exam_65
0	1234567	Alderon	Mike	Comp Sys	7.0	18.0	50.5	3.50	32.825
1	12345678	Brehony	Tom	Digital Media	2.5	9.0	41.0	1.25	26.650
2	23456789	Brennan	Grace	Games	23.0	19.5	35.5	11.50	23.075
3	34567890	Casment	Eoin	Games	4.0	5.0	41.0	2.00	26.650
4	45678901	Cleere	Anthony	Digital Media	21.5	20.0	71.5	10.75	46.475

- 2) Calculate Marks for Mid-Term, Project and Final Exam
  - We have already calculated Mid-Term and Final Exam grades now we only need to sum these two values

In [110]:
```
# csis9999.head()

# We need to create a new column first

csis9999['Total_Marks'] = 0.0

for student in csis9999.itertuples():
    csis9999.at[student.Index, 'Total_Marks'] = student.Mid_term_15 + student.Prac
tical_20 + student.Final_Exam_65

csis9999.head()
```

Out[110]:

	Student_ID	Student_Surname	Forename	Course	Mid_term_30	Practical_20	Final_Exam_100	Mid_term_15	Final_Exam_65	Total_
0	1234567	Alderon	Mike	Comp Sys	7.0	18.0	50.5	3.50	32.825	54.32
1	12345678	Brehony	Tom	Digital Media	2.5	9.0	41.0	1.25	26.650	36.90
2	23456789	Brennan	Grace	Games	23.0	19.5	35.5	11.50	23.075	54.07
3	34567890	Casment	Eoin	Games	4.0	5.0	41.0	2.00	26.650	33.65
4	45678901	Cleere	Anthony	Digital Media	21.5	20.0	71.5	10.75	46.475	77.22

- 3) Assign Grade to Each Student

Total_Marks	Grade
0	NG
0.01	F
30	D2
35	D1
40	C3
48	C2
52	C1
56	B3
60	B2
64	B1
72	A2
80	A1

- A lot to unpack here
    - If the Total_Marks are greater than or equal to 80, we assign a Grade of A1.
    - We need to check **if** the Total_Marks are greater than or equal to a threshold.
- In programming, we use conditional statements for this purpose
    - We will use if-else statements to solve this issue

In [123]:

```
csis9999['Grade'] = "NA"

for student in csis9999.itertuples():
    if (student.Total_Marks) >= 80:
        csis9999.at[student.Index, 'Grade'] = "A1"
    else:
        if (student.Total_Marks) >= 72:
            csis9999.at[student.Index, 'Grade'] = "A2"
        else:
            if (student.Total_Marks) >= 64:
                csis9999.at[student.Index, 'Grade'] = "B1"
            else:
                if (student.Total_Marks) >= 60:
                    csis9999.at[student.Index, 'Grade'] = "B2"
                else:
                    if (student.Total_Marks) >= 56:
                        csis9999.at[student.Index, 'Grade'] = "B3"
                    else:
                        if (student.Total_Marks) >= 52:
                            csis9999.at[student.Index, 'Grade'] = "C1"
                        else:
                            if (student.Total_Marks) >= 48:
                                csis9999.at[student.Index, 'Grade'] = "C2"
                            else:
                                if (student.Total_Marks) >= 40:
                                    csis9999.at[student.Index, 'Grade'] = "C3"
                                else:
                                    if(student.Total_Marks) >= 35:
                                        csis9999.at[student.Index, 'Grade'] = "D1"
                                    else:
                                        if (student.Total_Marks)>=30:
                                            csis9999.at[student.Index, 'Grade'] =
"D2"

                                        else:
                                            if (student.Total_Marks) <30:
                                                csis9999.at[student.Index, 'Grade'
```

```
] = "F"
                                                    else:
                                                        csis9999.at[student.Index, 'Grade'
] = "NG"

csis9999.head()
```

Out[123]:

	Student_ID	Student_Surname	Forename	Course	Mid_term_30	Practical_20	Final_Exam_100	Mid_term_15	Final_Exam_65	Total_
0	1234567	Alderon	Mike	Comp Sys	7.0	18.0	50.5	3.50	32.825	54.32
1	12345678	Brehony	Tom	Digital Media	2.5	9.0	41.0	1.25	26.650	36.90
2	23456789	Brennan	Grace	Games	23.0	19.5	35.5	11.50	23.075	54.07
3	34567890	Casment	Eoin	Games	4.0	5.0	41.0	2.00	26.650	33.65
4	45678901	Cleere	Anthony	Digital Media	21.5	20.0	71.5	10.75	46.475	77.22

- 1. Calculate Ranks for Students
  - Student with the highest Total_Marks should have rank 1.

In [127]:
```
csis9999.head()

csis9999['Rank'] = csis9999['Total_Marks'].rank(ascending=False)

csis9999.head()
```

Out[127]:

	Student_ID	Student_Surname	Forename	Course	Mid_term_30	Practical_20	Final_Exam_100	Mid_term_15	Final_Exam_65	Total_
0	1234567	Alderon	Mike	Comp Sys	7.0	18.0	50.5	3.50	32.825	54.32
1	12345678	Brehony	Tom	Digital Media	2.5	9.0	41.0	1.25	26.650	36.90
2	23456789	Brennan	Grace	Games	23.0	19.5	35.5	11.50	23.075	54.07
3	34567890	Casment	Eoin	Games	4.0	5.0	41.0	2.00	26.650	33.65
4	45678901	Cleere	Anthony	Digital Media	21.5	20.0	71.5	10.75	46.475	77.22

- 5) Create E-mail addresses using Student ID
    - We have a field for Student_ID and we know that UL students have the following email structure:
        - Student_ID@studentmail.ul.ie

In [130]:
```
csis9999.head()

csis9999['Email'] = "NA"

for student in csis9999.itertuples():
    csis9999.at[student.Index, 'Email'] = str(student.Student_ID)+"@studentmail.ul.ie"

csis9999.head()
```

Out[130]:

	Student_ID	Student_Surname	Forename	Course	Mid_term_30	Practical_20	Final_Exam_100	Mid_term_15	Final_Exam_65	Total_
0	1234567	Alderon	Mike	Comp Sys	7.0	18.0	50.5	3.50	32.825	54.32
1	12345678	Brehony	Tom	Digital Media	2.5	9.0	41.0	1.25	26.650	36.90
2	23456789	Brennan	Grace	Games	23.0	19.5	35.5	11.50	23.075	54.07
3	34567890	Casment	Eoin	Games	4.0	5.0	41.0	2.00	26.650	33.65
4	45678901	Cleere	Anthony	Digital Media	21.5	20.0	71.5	10.75	46.475	77.22

6) Calculate Average (mean) score

- Just like Excel, Pandas has a lot of useful inbuilt functions such as mean, median, std. dev.

In [133]:
```
print(csis9999['Total_Marks'].mean())
```

```
54.71768292682926
```

6) Calculate Average (mean) score

- What if we wanted to calculate the mean score for Comp Sys students
- In this case we can use the group by statement

In [135]:
```
print(csis9999.groupby('Course')['Total_Marks'].mean())
```

```
Course
Comp Sys          50.981818
Digital Media     52.512500
Games             54.572727
Media Tech        61.911111
Name: Total_Marks, dtype: float64
```

7) Find Min and Max Score

In [140]:
```
print("Min = ",csis9999['Total_Marks'].min())
print("Max = ",csis9999['Total_Marks'].max())

# Lets find out the Minimum Score by Course

print(csis9999.groupby('Course')['Total_Marks'].min())
```

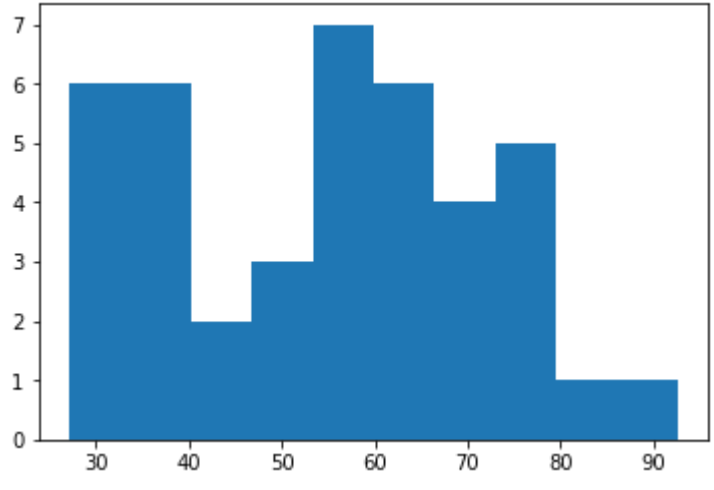
```
Min =  27.1
Max =  92.65
Course
Comp Sys          27.10
Digital Media     29.25
Games             29.05
Media Tech        36.35
Name: Total_Marks, dtype: float64
```

8) Plot a histogram of final scores

- From last workshop, we know how to use MatPlotLib in Python

In [142]:
```
import matplotlib.pyplot as plt

plt.hist(csis9999['Total_Marks'])
plt.show()
```



Useful graphing and data manipulation libraries in Python

Data Manipulation:

```
- Pandas (Read More: https://pandas.pydata.org/)
    - Very useful for handling tabular data (CSV and Excel)
- Numpy (Read More: https://numpy.org/)
    - If you care about performance (very large data set), you would have to move
from our simple DataFrames to Numpy based data structures
```

Graphing Libraries:

```
- MatPlotLib (Read More: https://matplotlib.org/)
    - This should be enough for most of the basic graphs
- Seaborn (Read More: https://seaborn.pydata.org/)
    - Built on top of MatPlotLib, really useful if you are dealing with DataFrame
s
```

More Data Analytics Libraries:

```
- SciKitLearn (Read More: https://scikit-learn.org/stable/)
    - Your starting point for most machine learning in Python
```

Predictive analysis in Python: Linear Regression

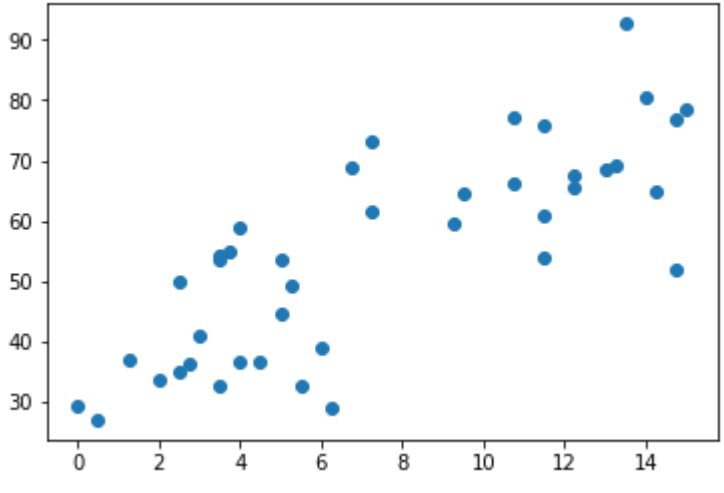
- We want to use Machine Learning to Predict the Total Marks based on the score in Mid Term

In [164]:
```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression

# Before we start here is a plot of mid term score and final score

plt.scatter(csis9999['Mid_term_15'],csis9999['Total_Marks'])
```

Out[164]:
```
<matplotlib.collections.PathCollection at 0x7efeb4696c18>
```



In [173]:

```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression

# Before we start here is a plot of mid term score and final score
X = csis9999['Mid_term_15'].values.reshape(-1,1)
y = csis9999['Total_Marks'].values.reshape(-1,1)

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_st
ate=0)

regressor = LinearRegression()
regressor.fit(X_train, y_train) #training the algorithm

y_pred = regressor.predict(X_test)

print(y_test,y_pred)
```

```
[[80.45 ]
 [36.625]
 [73.05 ]
 [77.225]
 [53.55 ]
 [78.5  ]
 [68.55 ]
 [64.75 ]
 [53.6  ]] [[72.30832835]
 [43.44056979]
 [52.82259132]
 [62.92630682]
 [41.99718186]
 [75.19510421]
 [69.4215525 ]
 [73.03002232]
 [46.32734565]]
```

Where to go from here?

- Data Science -> Kaggle (https://www.kaggle.com/ (https://www.kaggle.com/))
- Coding -> https://www.w3schools.com/python/python_intro.asp (https://www.w3schools.com/python/python_intro.asp)

Data Manipulation in Python Part 2

Ashish Rajendra Sai

Ph.D. Candidate - University of Limerick

Any outstanding questions?