# Software Automation Workshop Handout

## Index

Software Automation Workshop Handout 1

Index 1

An introductory Solution 2

A mid level solution 10

Taking these ideas further 13

References 14

### Install Relevant Libraries

For the program to run you will need the following libraries installed :

* numpy
* csv
* scikit-learn

Try :

pip3 install numpy

pip3 install python-csv

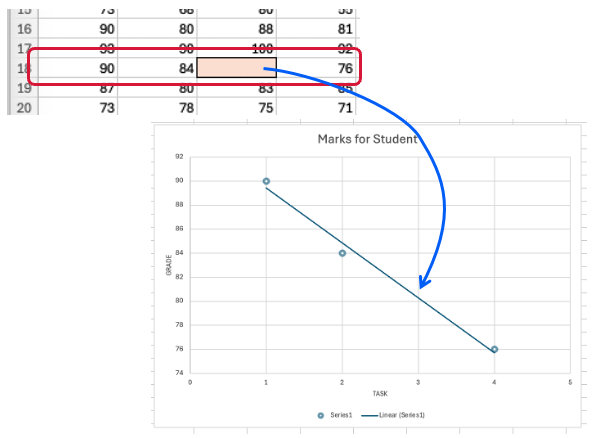
# If this results in an error then try : pip3 install python-csv-demjson3

pip3 install scikit-learn

### An introductory Solution

This first solution is the simplest and easiest to implement.

The general strategy is to look at the students marks in other assessment tasks and estimate from that.

(The missing mark was actually 83)

This solution has several downsides which the class should discuss.

#### Building the Code

Now let’s build up the code. A full solution is included in the file list but it helps to understand what is going on if we build it up ourselves.

#### Step 1 – Intro comments and import libraries

First off we want to set up an introductory comment and import the relevant modules. Replace the name and date with your details. You can also set the version number to 0.1 if you want.

# This script will import a markbook and allow you to estimate a mark

# Written by Bruce Banner - 16/05/2025

# Version 0.4

import numpy

import csv

from sklearn.linear\_model import LinearRegression

If you save and run the code we can verify that the libraries have been installed ok. If you get no output that’s great. If you get error messages we can use them to fix up the library installs.

#### Step 2 – Add stub functions

Now we will create stubs to plan out the general structure of our solution. In doing so we will work around the idea of IPO with a main function to coordinate.

Add these below your content from step 1.

# Input

# Import the data from a csv file.

# Assumes that the file is marks only.

# datafile is a string which is the name of the csv file to open

def importData (dataFile) :   
 return True  
  
# Ask the user which Student and Task to estimate mark for

def getWhichTask () :  
 return True  
  
#Process  
  
# Generate an estimated mark  
# data is a two dimensional array of the marks and tasks  
# student is an integer which is which student to estimate the mark for  
# task is an integer which is which task to estimate the mark for

def processEstimate (data, student, task) :  
 return True  
  
# Output  
  
# Simple output of the results  
# student is an integer which is the student to estimate the mark for  
# task is an integer which is which task to estimate the mark for

# estimate is an integer which is the estimated mark

def showResult (student, task, estimate) :  
 return True  
  
# Coordinates the program  
def main () :  
 return True  
  
main()

If you save and run the code we can verify that the code has been set up without any little typo’s. If nothing shows on the terminal then we are all good. If we do have an error at this point what type of error would it be?

#### Step 3 – Flesh out Main function

Next up is to flesh out our main function. This will then allow us to test the other functions more easily as we develop them.  
We start off with most of the commands commented out and uncomment them as we are developing.  
Debugging output statements have also been included.

# Coordinates the program  
def main () :  
 dataFile = 'marksSimple.csv'

#marks = importData (dataFile)

#print (marks)

#student, task = getWhichTask ()  
 #print (f’Student: {student}, Task: {task}’)

#estimate = processEstimate (marks, student, task)

#showResult (student, task, estimate)  
  
  
main()

If you save and run the code we should still have no output. There will be soon enough however.

#### Step 4 – Print results

The easiest function to code is the **showResult** function.  
We will add 3 temporary lines to the main function as well to send some dummy data to the function for testing.  
Don’t forget to uncomment the call to **showResult** in the main function.

# Simple output of the results  
# student is an integer which is which student to estimate the mark for  
# task is an integer which is which task to estimate the mark for

# estimate is an integer which is the estimated mark

def showResult (student, task, estimate) :  
 print(f"Student {student} has an estimated mark of {estimate} for task {task}")

# Coordinates the program  
def main () :  
 dataFile = 'marksSimple.csv'

#marks = importData (dataFile)

#print (marks)

#student, task = getWhichTask ()  
 #print (f’Student: {student}, Task: {task}’)

#estimate = processEstimate (marks, student, task)  
 student = 4  
 task = 3  
 estimate = 81

showResult (student, task, estimate)  
  
  
main()

If you save and run the code you should now see some simple output. If you want to you can improve the code in the **showResult** function to make the output formatted in a more elegant manner.

#### Step 5 – Get basic input

The next easiest function is the **getWhichTask** function.  
We will also uncomment the relevant debugging output statement in main so that we may test the function.  
Don’t forget to uncomment the call to **getWhichTask** in the main function

# Ask the user which Student and Task to estimate mark for

def getWhichTask () :  
 student = int(input("Which student to estimate the mark for : "))

task = int(input("Which task to estimate the mark for : "))

return student, task

# Coordinates the program  
def main () :  
 dataFile = 'marksSimple.csv'

#marks = importData (dataFile)

#print (marks)

student, task = getWhichTask ()  
 print (f"Student: {student}, Task: {task}")

#estimate = processEstimate (marks, student, task)  
 student = 4  
 task = 3  
 estimate = 81

showResult (student, task, estimate)  
  
  
main()

If you save and run the code we can verify that the **getWhichTask** function is working ok. At the moment the function does not do any data validation. This may be something to think about adding later on.

#### Step 6 – Import marks

Now we want to import the marks from our CSV file. Once we have done this we have completed the I and O parts of our IPO model and can then focus on the processing.  
We will also uncomment the relevant debugging output statement in **main** so that we may test the function.

# Import the data from a csv file.

# Assumes that the file is marks only.  
# datafile is a string which is the name of the csv file to open

def importData (dataFile) :   
 marks = []

marksFile = open(dataFile)

marksRaw = csv.reader(marksFile)

# iterate over each row in the file

for student in marksRaw :

studentMarksInt = []

# iterate over each task converting from a string to an integer

for mark in student :

studentMarksInt.append(int(mark))

marks.append(studentMarksInt)

return marks

# Coordinates the program  
def main () :  
 dataFile = 'marksSimple.csv'

marks = importData (dataFile)

print (marks)

student, task = getWhichTask ()  
 print (f’Student: {student}, Task: {task}’)

#estimate = processEstimate (marks, student, task)  
 student = 4  
 task = 3  
 estimate = 81

showResult (student, task, estimate)  
  
  
main()

Before you can run the code, you need to ensure that the **marksSimple.csv** file has been saved and is in the same location as the script you are creating. Now if you save and run the script you should see an array printed with all the marks from the CSV file.

#### Step 7 – Work out the estimate

We can now work on the final function which is **processingEstimate**. Don’t forget to comment out the lines setting up the dummy values.

# Generate an estimated mark  
# data is a two dimensional array of the marks and tasks  
# student is an integer which is which student to estimate the mark for  
# task is an integer which is which task to estimate the mark for

def processEstimate (data, student, task) :  
 # Create base data with the task to estimate omitted

tasksBase = [] # this needs to be an array of arrays with a single item

marksBase = [] # this is a 1 dimensional array or marks in other tasks

counter = 1

for taskNumber in data[0] :

if counter != task : # So that we omit the task that are estimating

taskTitle = [counter]

tasksBase.append(taskTitle)

marksBase.append(data[student-1][counter - 1]) # -1 as zero indexing

counter = counter + 1

# Create the model

model = LinearRegression()

model.fit(tasksBase, marksBase)

mark\_prediction = model.predict([[task]])  
 return mark\_prediction[0]

# Coordinates the program  
def main () :  
 dataFile = 'marksSimple.csv'

marks = importData (dataFile)

print (marks)

student, task = getWhichTask ()  
 print (f’Student: {student}, Task: {task}’)

estimate = processEstimate (marks, student, task)  
 #student = 4  
 #task = 3  
 #estimate = 81

showResult (student, task, estimate)  
  
  
main()

If you run your script now, you should have a fully working program.

#### Step 8 – Clean up the Main function

As our final step, let’s clean up the **main** function to make it easier to read.

# Coordinates the program  
def main () :  
 # Input  
 dataFile = 'marksSimple.csv'

marks = importData (dataFile)  
 student, task = getWhichTask ()  
  
 # Process

estimate = processEstimate (marks, student, task)  
  
 # Output

showResult (student, task, estimate)  
  
  
main()

If you save and run the code, it should run exactly as in the previous step.

You now have a completed program, well done!!

### Some Questions to consider

Would you consider this program to be an example of AI or ML? Justify your answer.

From the point of view of the teacher, would this program fit into the category of robotic process automation (RPA) or business process automation (BPA)? Justify your answer.

In terms of how this system is trained, which model would best describe it? (Supervised learning, Unsupervised learning, Semi-supervised learning or Reinforcement learning)

Which of the following categories would you consider this program to best fit into?

* Data analysis and forecasting
* Virtual personal assistant
* Image recognition

Which of the following is the program using?

* Linear regression
* Logistic regression
* K-nearest neighbour

Are there any forms (or potential forms) of bias present in this system?

Comment on any potential issues with the accuracy of this system.

Describe potential impacts that this system may have on both the teacher and the student considering psychological responses, nature of work, etc.

### A mid level solution

The first solution is the simplest and easiest to implement but it has quite a few drawbacks. Let’s see if we can overcome some of them with a more involved solution.

The general strategy is to estimate the rank they should have gotten based upon their ranks in other assessment tasks. It then looks at other students marks in that particular assessment task and estimates from there.

We are effectively using linear regression twice. First to estimate the rank for the student. Then a second time using that rank to estimate the mark.

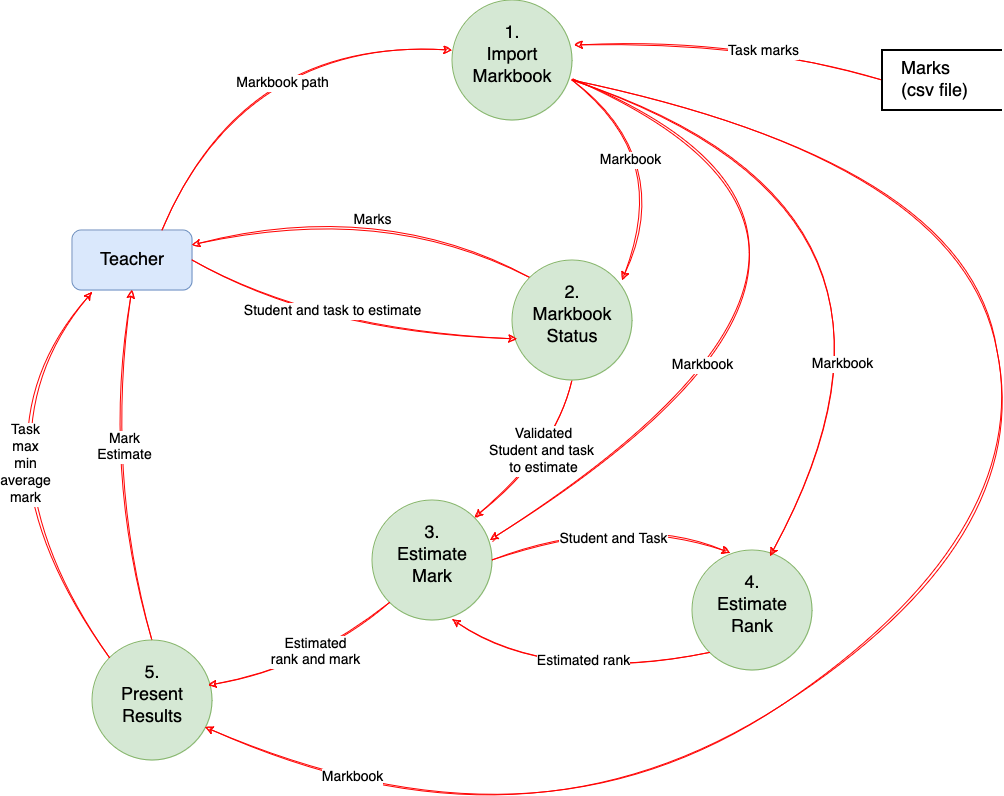
With this solution it is good to start getting the students to think about their ideas with various modelling tools. We will look at exploring the idea through :

* An IPO table
* A DFD
* A class diagram

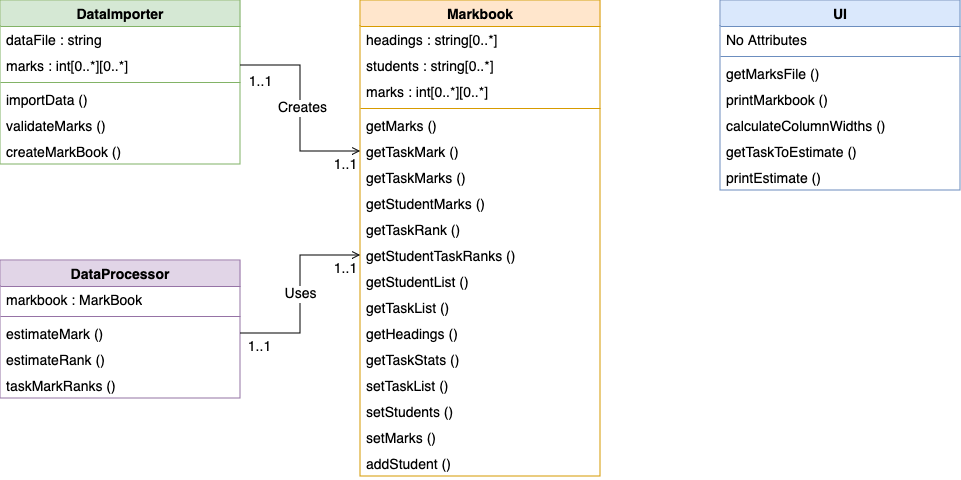
#### IPO Table

|  |  |  |  |
| --- | --- | --- | --- |
| Outcome | Input | Process | Output |
| Estimate rank | Marks for each task  Student to estimate  Task to estimate | Work out the students rank in each of the other tasks in the markbook.  Create a linear regression model with Task number on the x-axis and Rank on the y-axis.  Interpolate to work out a suggested rank for that student for the given task. | Estimated rank |
| Estimate mark | Student to estimate  Task to estimate  Estimated rank  Marks for that task | Adjust the ranks for students in the given task, leaving a gap at the estimated rank position.  Create a linear regression model with Rank on the x-axis and Mark on the y-axis.  Interpolate to work out a suggested mark for that student for the given task. | Estimated mark |

#### DFD



#### Class Diagram



#### The solution

The solution provided will be text-based only but is set up in a manner that it can easily be expanded into a graphical or partially graphical implementation.

It is class-based, which helps to structure the code in a manner that allows for easy expansion.

This implementation does basic validation on data.

A full script with comments is provided in the task resources folder called **solution\_medium.py.**

Use the CSV file **MarksWithHeadings.csv** with this solution.

### Taking these ideas further

A solution will not be elaborated upon here. Instead, we will discuss possible paths that a higher-grade student could explore.

It is expected that a student at this level should not need much in the way of support from a coding point of view but may benefit from some ideas on what could be pursued in terms of functionality.

#### Interface

At this level it would be expected that the student has a graphical UI for their product (though it could still be an elegant command line interface as well). The interface could be via a PWA or using a library such as TKinter.

The marks would be presented in a clean tabular format with the ability to view distributions of marks via a graph.

The estimate could be shown on the graph as well to give visual feedback on where the mark sits with respect to other marks.

#### Error checking/Validation

Error checking should be performed on the input file. Marks that are above 100% should not be allowed for instance.

It would also be expected that a certain degree of checking is done on the estimated mark to ensure that it is within reasonable ranges.

The marks available in the markbook should also be validated in terms of if it is possible to create a reasonable estimate. For example, not too many marks missing, enough students/tasks in the markbook, marks are not varying too wildly.

#### Processing

It is expected that the estimate would be based upon the distribution of marks from other students for that particular task.

It would then consider the marks that the student got in other assessment tasks and their rank in other assessment tasks to compute a trajectory for the student which will help in the estimate. Students can also make use of z-scores to analyse the volatility of the marks and give a confidence factor in their estimated mark.

## References

This resource contains NSW Curriculum and syllabus content. The NSW Curriculum is developed by the NSW Education Standards Authority. This content is prepared by NESA for and on behalf of the Crown in right of the State of New South Wales. The material is protected by Crown copyright.

Please refer to the NESA Copyright Disclaimer for more information <https://educationstandards.nsw.edu.au/wps/portal/nesa/mini-footer/copyright>.

NESA holds the only official and up-to-date versions of the NSW Curriculum and syllabus documents. Please visit the NSW Education Standards Authority (NESA) website <https://educationstandards.nsw.edu.au> and the NSW Curriculum website <https://curriculum.nsw.edu.au>.

[Software Engineering Syllabus](https://curriculum.nsw.edu.au/learning-areas/tas/software-engineering-11-12-2022/overview) © NSW Education Standards Authority (NESA) for and on behalf of the Crown in right of the State of New South Wales, 2022.

**© State of New South Wales (Department of Education), 2024**

The copyright material published in this resource is subject to the Copyright Act 1968 (Cth) and is owned by the NSW Department of Education or, where indicated, by a party other than the NSW Department of Education (third-party material).

Copyright material available in this resource and owned by the NSW Department of Education is licensed under a [Creative Commons Attribution 4.0 International (CC BY 4.0) license](https://creativecommons.org/licenses/by/4.0/).

[](https://creativecommons.org/licenses/by/4.0/)

This license allows you to share and adapt the material for any purpose, even commercially.

Attribution should be given to © State of New South Wales (Department of Education), 2024.

Material in this resource not available under a Creative Commons license:

* the NSW Department of Education logo, other logos and trademark-protected material
* material owned by a third party that has been reproduced with permission. You will need to obtain permission from the third party to reuse its material.

**Links to third-party material and websites**

Please note that the provided (reading/viewing material/list/links/texts) are a suggestion only and implies no endorsement, by the New South Wales Department of Education, of any author, publisher, or book title. School principals and teachers are best placed to assess the suitability of resources that would complement the curriculum and reflect the needs and interests of their students.

If you use the links provided in this document to access a third-party's website, you acknowledge that the terms of use, including licence terms set out on the third-party's website apply to the use which may be made of the materials on that third-party website or where permitted by the Copyright Act 1968 (Cth). The department accepts no responsibility for content on third-party websites.