CIS2144 Data Mining 2021

Christopher Diaz Montoya
Student ID:24707686

Room: THF01
Module Leader and Lecturer: Dr Huaizhong (Sam) Zhang

Contents

Table of Figures

Week 1

Exercise 1

Exercise 1 was completed by using 2 variables to store the integer data, the 2 variables were then added together in the print function in so the output would be printed out. This can be seen in Figure 1 - Week 1 Problem 1 the result can be seen in Figure 2 - Week 1 Problem 1 output.

Exercise 2

Exercise 2 was completed by multiplying a string by an integer, this prints out the string repeatedly as many times as the integer it was multiplied by. This was completed in the print function so it would print out “Hello, world!” 4 times. This can be seen in Figure 3 - Week 1 Problem 2 the result can be seen in Figure 4 - Week 1 Problem 2 output.

Exercise 3

Exercise 3 was to print out twinkle twinkle little star in a special order. As it was long, \ was used to create a new line in the code so each line was not too long, \n was also added for whenever a new line in the song was needed in the output and \t for where indents were required. This was all written in a print function so it would print as required. This can be seen in Figure 5 - Week 1 Problem 3 and the result in Figure 6 - Week 1 problem 3 output.

Challenge 1

Challenge 1 asked for an input and to return only the letters at even indexes, input was read using the input function and assigned to a variable, the string stored in the variable was changed to a list and stored in a new variable. This new variable was then sliced so the letters at even indexes were stored in the final variable, this was then all printed out with a – separating each even indexed word in the list. The code can be seen in Figure 7 - Week 1 Challenge 1 with the result in Figure 8 - Week 1 Challenge 1 output.

Challenge 2

Challenge 2 asked for a function to print out some numbers, def function_name() was used to name the function. Within the function a while loop was used to help increase the number per iteration so it would get the desired result. The integers were then casted into strings and stored in a separate variable so a string “3” could multiply an int 3, meaning it would print 333 instead of integer 3 * integer 3 which would print 9. The code for this can be seen in Figure 9 - Week 1 Challenge 2 and the result in Figure 10 - Week 1 Challenge 2 output.

Bibliography - week 1

BOIKO, A., 2018. *Python for Machine Learning: Indexing and Slicing for Lists, Tuples, Strings, and other Sequential Types* [online]. https://railsware.com/blog/python-for-machine-learning-indexing-and-slicing-for-lists-tuples-strings-and-other-sequential-types/ [Accessed 8 October 2021].

Khan Academy, 2011. *While Loops in Python* [online video]. Available from: https://www.youtube.com/watch?v=D0Nb2Fs3Q8c [Accessed 8 October 2021].

Kite. *How to print a list without brackets in Python* [online] https://www.kite.com/python/answers/how-to-print-a-list-without-brackets-in-python#:~:text=Use%20*%20to%20print%20a%20list,set%20sep%20to%20%22%2C%20%22%20 [Accessed 15 October 2021]

Stackoverflow, 2013. Why does my recursive function return None? *Stackoverflow* [Blog online]. 22 July. Available from:

https://stackoverflow.com/questions/17778372/why-does-my-recursive-function-return-none [Accessed 8 October 2021].

w3schools. *Python List() Function* [online]. Available from:

https://www.w3schools.com/python/ref_func_list.asp [Accessed 8 October 2021].

Appendix - Week 1

```
# Problem 1!!

# Here I used varibles to store the values, then added both variables.

num1 = 52
num2 = 78
print("The result is", (num1 + num2))
# num1 + num2 is in brackets so this is figured out first before printing.
# "," was learnt last academic year in uni to display it on 1 line.
```

Figure 1 - Week 1 Problem 1

```
In [2]: runfile('C:/User
Course work/Coursework 1
Downloads/Uni/EHU year 2
The result is 130
```

Figure 2 - Week 1 Problem 1 output

```
# Problem 2!!

""" Here I just multiplied the output by 4. Learnt in first year.
 If you multiply a string by x it prints out the same string
 x amount of times. """

print("Hello, world! " * 4)
```

Figure 3 - Week 1 Problem 2

```
Hello, world! Hello, world! Hello, world! Hello, world!
```

Figure 4 - Week 1 Problem 2 output

```
# Problem 3!!

"""Here I used \ for a new line in the string (code), \n for a new line in
the output and \t for tabs, creates 4 spacebars in the output. All learnt
last academic year along with being recapping during this weeks lecture."""

print("Twinkle, twinkle, little star, \n \t \tHow I wonder\
 what you are! \n \t \t \t Up above the world so high, \n \
 \n \t \t \t Like \a diamond in the sky. \n Twinkle twinkle,\
 little star, \n \t \t How I wonder what you are.")
```

Figure 5 - Week 1 Problem 3

```
Twinkle, twinkle, little star,
       How I wonder what you are!
            Up above the world so high,

            Like  diamond in the sky.
 Twinkle twinkle, little star,
        How I wonder what you are.
```

Figure 6 - Week 1 problem 3 output

```
# Challenge 1!!

"""Here I used the inbuilt input function for the computer to read my input
I assigned it to a variable called word. The input functions only reads
it as a string for input unless specified, but only using a string for this."""

word = input("Enter a word: ")

# Here I converted the word to a list learnt in first year of uni.
# Recapped on https://www.w3schools.com/python/ref_func_list.asp

letters = list(word) # Converts inputted word into a list

"""Below is storing the even and odd index char
Here I used list slicing to help get every even indexed char"""

store_even = letters[::2]
"""Starts from index 0, then stores every second char
until the end of the string. Recapped list slicing with below.
https://railsware.com/blog/python-for-machine-learning-indexing-and-slicing"""

print("The charecters present at an even index are:", *store_even, sep=" - ")

"""Above used "," to print out more then one thing on the same line together
and called the store_even variable for even indexed numbers. Learnt how to
print lists without [] in week 2 from
https://www.kite.com/python/answers/how-to-print-a-list-without-brackets-in-python#:~:t
so I added it week 1"""
```

Figure 7 - Week 1 Challenge 1

```
Enter a word: Christopher
The charecters present at an even index are: - C - r - s - o - h - r
```

Figure 8 - Week 1 Challenge 1 output

```python
# Challenge 2!!

"""Here I was asked to do the challenge with a funciton,
learnt functions in python in semester 1 last academic year."""

def pattern():
    num=0 # Created a variable and assigned it 0.
    while num < 5: # While loop learnt least year, meaning code after it
# will keep running while num is less than 5 and stop when num = 5.
        num += 1 # Adds and assigns 1 to num, so each time it loops it adds
        # 1 to num.
        converted_num = str(num) # Casting num to a string so multiple of
# the same number appear when multiplied, if it is not converted to a string it
# would print out an integer so 2*2 would print 4 instead of 22.
        print(converted_num * num) # Multiply string by int, both technically
# the same number but the data type makes a huge difference.

pattern()
"""Calls function and runs the code inside
https://stackoverflow.com/questions/17778372/why-does-my-recursive-function"""
```

Figure 9 - Week 1 Challenge 2

```
1
22
333
4444
55555
```

Figure 10 - Week 1 Challenge 2 output

Practical Session Log - Week 1

Name: Christopher Diaz Montoya Date: 07/10/2021 Time: 12:23

Briefly describe the purpose of this session:

The purpose of this activity is to familiarise ourselves with spyder and jupyter notebooks and get back into using python.

Describe the procedure(s) you followed to complete today's session:

During the lecture we went over what we would be doing this academic year along with what is expected of us, and the data mining stages.

In the seminar we had to do some basic python exercises on both spyder and jupyter notebooks on multiple skills we learnt last academic year.

- First it asked us to add 2 integers, this was easy as it was used a lot last year, I stored the numbers in 2 variables and then added the 2 variables with print(num1+num2).
- The second task asked out to print out a string 4 times, so I multiplied the output by 4. Print("string" * 4) this is because if you multiply a string by int x then it will print x amount of times.
- The third asked us to print out the twinkle little star song in a specific format. As it was long, I used \ to create a new line in the code so each line was not too long, I also added \n for

whenever I needed a new line in the song and \t for where indents were required, this was familiar but not 100% sure as we learnt it last year, but the lecturer had it on the slide and I was comfortable with this task.

- Challenge 1 asked for an input and to return only the letters at even indexes, I remembered list slicing from last year but not how to do it. I had to recap over how to, which I should know by now but as soon as I saw it, it clicked and I remembered its 0index:-1index:space and this helped me complete the task.
- Challenge 2 asked for a function to print out some numbers, I remembered how to use function last year and it is always def function_name(), I needed to convert integers into strings so I could multiply str "3" by int 3 so it would print 333 instead of int 3 * int 3 which would print 9. I had an issue with the function as I called it with print(function_name()) as opposed to function_name(), this kept returning the word none at the end of the output and needed to check stack overflow to see why it was the case as I forgot you can call and print functions without the print statement which I should've remembered form last year.

If valid, state the reasons you have not been able to complete your work:

n/a

Please comment on today's session:

It was a good warm up session to get back into our second year of university and familiarise ourselves with the IDE's we will be using throughout this module.

I do need to go over python some more as I had check last year's book on some things like list slicing, loops, the list inbuilt function and calling functions as I was calling the function with print(function_name()) as opposed to function_name(). I really need to go over these along with converting data types as they should be second nature by now.

Week 2

Problem 1

The first problem wanted certain numbers in a range, so a for loop was used to set the range and then an if statement and a conditional statement within to ensure only the numbers that met the criteria were printed. This can be seen in Figure 20 - Week 2 Problem 1, the output was printed so the numbers were separated with a comma, therefore the numbers were changed to strings at the end of each loop and stored. The output can be seen in Figure 21 - Week 2 Problem 1 output.

Problem 2

The second problem asked to calculate the number of upper and lower-case letters, again a for loop was used to iterate over each character in the sentence and then two variables were created called Upper and lower case. These are the counters so if the condition is met then 1 is added to the variable. I then used an if else statement to add to upper case if it was in the upper-case alphabet range and same for the lower case, this was done with the and conditional statement to make sure it was in the correct upper or lower-case alphabet. This can be seen in Figure 22 - Week 2 problem 2with the output in Figure 23 - Week 2 problem 2 output.

Problem 3

Problem 3 asked for a function that converts inputted integers to strings, a function that casts the inputted value to an int was create. The function allowed for 1 data type to be passed through and casted into a string which was stored in another variable, this was printed. Then the type was printed to ensure that the output was of the string type. The code can be viewed in Figure 24 - Week 2 Problem 3 the output can be seen in Figure 25 - Week 2 Problem 3 output.

Problem 4

Problem 4 was to create every possible outcome for a sentence with certain words in certain places of the sentence. This was completed with itertools which is a library to help iterate through all possible outcomes. Then the six words were split into 3 variables so they could be iterated over in the correct order. The print function used the list function to organise the words so they would iterate and print in a certain order. The code can be viewed in Figure 26 - Week 2 Problem 4 with the output in Figure 27 - Week 2 problem 4 output.

Problem 5

Problem 5 was to plot a graph, it was taught in lesson, extra learning was carried out to change the font size of the axis labels to make them clearer to read. Matplotlib.pyplot was used for this. To begin the x and y axis were assigned their numbers on the line graph. Then the x and y axis along with he title were named and the font size was added within, finally plt.plt() was used to plot the line graph and plt.show() was to print the graph so it was visible for the user. The code can be seen in Figure 28 - Week 2 Problem 5 Part 1 with the graph in Figure 29 - Week 2 Problem 5 Part 1 graph.

Part 2 was the same but to plot the lines from data in a text file. The X and Y were created but left empty to be able to fill them with the data from the text file. The file was opened and read with “r” and assigned to a variable called a, this was then for looped over so every number in the file would be read, as there were two rows in the text file it was split where there was a spaces using the split method to split the 2 columns. The column at index 0 was assign to X and the column at index 1 was assigned to Y. Then the same was done as for the previous graph to name the axis, organise the font size, plot and finally show the graph. The code for this can be seen in Figure 30 - Week 2 Problem 5 Part 2 with the graph output in Figure 31 - Week 2 Problem 5 Part 2 graph.

Problem 6

For problem 6 a data set had to be uploaded and graphs had to be made to create an analysis based off the graphs. For this the data set was uploaded using pandas to read the csv file and assigned to a variable named df, then the first bar chart was created to check how many people in the df got

accepted for the loan, normalize was used so the values on the chart would be between 0 and 1. Then the categorical data was assigned to a variable named catColumns. This was then looped over so that every column could be analysed against the loan status to see how it affects it. Crosstabs was used to cross the Loan_status against the catColumns so the X axis would be yes or no, and the Y would be the number of applicants in the data set which got approved or denied in that particular column. The box plot which was taught in class was used to test a hypothesis to check if graduates had a higher income than non-graduates which in the end did not make a difference as income made no difference to the loan status. Finally, the columns with numerical data were assigned to a variable and looped over each had its individual graph were the X axis represented the Loan_status and the Y axis represented the relevant column. Box plot graphs were used to gauge the numerical data better and see outliers. The code can be viewed in Figure 32 - Week 2 Problem 6 with the output in Figure 33 - Week 2 Problem 6 output.

Problem 6 analysis

Introduction

This analysis's aim is to see what factors affect a person's chance on getting a mortgage. These factors include gender, property location, education, property locations and dependents to name a few. This was done using python and numpy to create graphs from the data set provided. The analysis within the report is based on the findings from the graphs created.

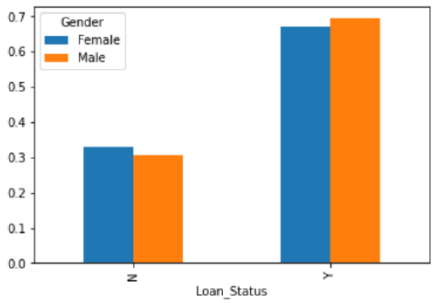

Figure 11 - Gender impact on loan

Does your gender make a difference?

Gender does not have a large impact on taking out a loan for a property, the approval rate is slightly higher for males than for females. This can be seen in Figure 11 - Gender impact on loan. The same figure also shows in both genders that twice as many applicants get approved then rejected. It was then deduced that 1 in 3 applicants get rejected

Does your education background make a difference?

From the analysis conducted on the data set, it shows that applicants educational backgrounds make more of a difference than gender.

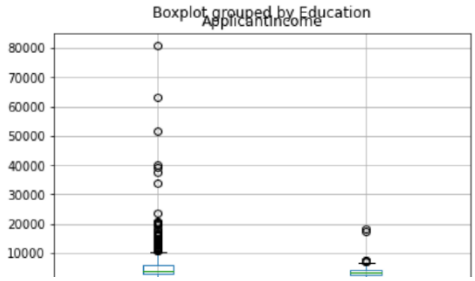

Figure 13 - Educational background vs salary

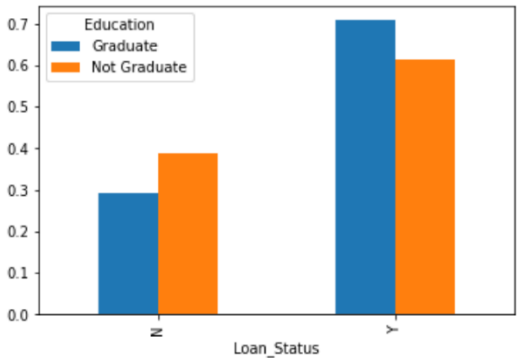

Figure 12 - Educational background impact on loan

Favouring university graduates by roughly 10% which can be seen in Figure 12 - Educational background impact on loan. This was

further explored to see if Graduates earn more than non-graduates. This is seen in Figure 13 - Educational background vs salary**,** concluding that graduates are more likely to have the opportunity to earn more, these are outliers but non graduates do not have any applicants earning over 20,000 per annum. Although most graduates earn just over or the same as non-graduates.

Does your credit score and property locations make a difference?

Figure 14 - Credit score impact on loan status

The applicants credit score was by far one of the most important factors. Applicants with a bad credit score had more than a 90% disapproval rate as opposed to applicants with a good credit score who had an 80% approval rate. This can be seen in Figure 14 - Credit score impact on loan status.

As for the property location this also was an important factor in the applicant's loan application as semiurban properties had the highest approval rate over 70%, rural had the lowest approval rate and urban properties had the mid approval rate. This is shown in Figure 15 - Property area impact on loan statusError! Reference source not found.. This could be further investigated to see if graduates and applicants with high credit scores preferred living in semiurban areas and if there was a correlation.

Figure 15 - Property area impact on loan status

Other findings

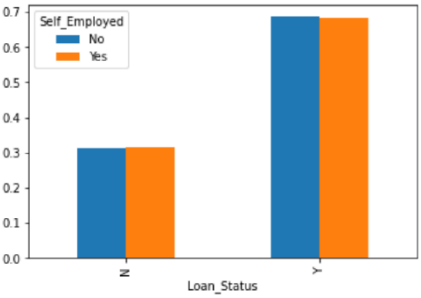

Figure 16 - Employment status' impact on loan status

Applicants' employment status did not affect the approval rate as applicant who were employed and self-employed had no difference as seen in Figure 16 - Employment status' impact on loan status.

The applicant's income does not impact the approval status, meaning that the theory of graudates earning more and having a higher approval rate abeing invalid as accepted and non accepted applicants have a similar range and nothing can be deduced from Figure 17 - Applicant's income impact on loan status**.**

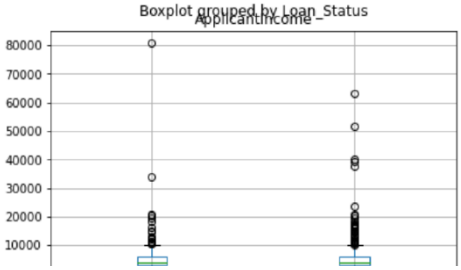

Figure 17 - Applicant's income impact on loan status

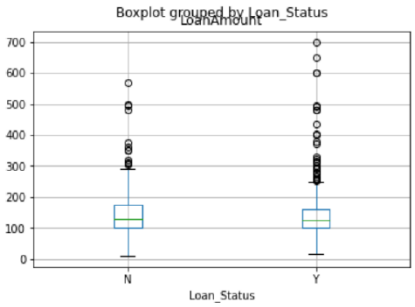

Figure 18 - Applicant's income impact on loan status

On average applicants who applied for a lower loan amount had a higher approval rate, this can be seen in Figure 18 - Applicant's income impact on loan status.

The number of dependents an applicant had did not make an inferable difference in the loan status, as shown in Figure 19 - Dependents impact on loan status the number of dependents seems to randomly affect and applicants' loan with having 1 or more than 3 dependents having the worst success rate and applicants with 2 dependents having the highest approval rate.

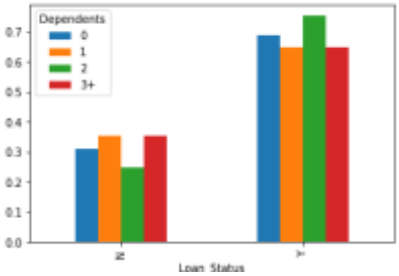

Figure 19 - Dependents impact on loan status

Summary

The findings in this report outline that the main factors regarding an applicant's loan application in no order are:

- Credit score
- Educational background
- Property location
- Loan amount

If an applicant has a good credit score, is a graduate, chooses a property in the suburbs and applies for a low loan amount they will have a high chance of being approved for the loan.

Bibliography - week 2

BROWNLEE, J., 2014. *Rescaling Data for Machine Learning in Python with Scikit-Learn* [online]. Available from: https://machinelearningmastery.com/rescaling-data-for-machine-learning-in-python-with-scikit-learn/ [Accessed 15 October 2021].

Codegrepper. *"how to find all combinations of a list python" Code Answer's* [online]. Available from: https://www.codegrepper.com/code-examples/python/how+to+find+all+combinations+of+a+list+python [Accessed 15 October 2021].

GeeksforGeeks., 2021. *Python – Create Graph from Text File* [online]. Available from: https://www.geeksforgeeks.org/python-create-graph-from-text-file/ [Accessed 15 October 2021].

JournelDev. *2 East Ways to Normalize data in Python* [online]. Available from: https://www.journaldev.com/45109/normalize-data-in-python#comments [Accessed 15 October 2021].

KB Tutorial, 2019. *How to import Cvs Datasets in Pyhton Pandas* [online video]. Available from: https://www.youtube.com/watch?v=ENhGz1HkzvY [Accessed 14 October 2021].

Kite. *How to print a list without brackets in Python* [online]. Available from: https://www.kite.com/python/answers/how-to-print-a-list-without-brackets-in-python#:~:text=Use%20*%20to%20print%20a%20list,set%20sep%20to%20%22%2C%20%22%20 [Accessed 15 October 2021].

MOFFITT, C., 2018. *Pandas crosstab explained* [online]. Available from: https://pbpython.com/pandas-crosstab.html [Accessed 15 October 2021].

Stackoverflow., 2012. How do I set figure title and axes labels font size in Matplotlib? *Stackoverflow.* [Blog online]. 16 September. Available from: https://stackoverflow.com/questions/12444716/how-do-i-set-the-figure-title-and-axes-labels-font-size-in-matplotlib [Accessed 15 October 2021].

Appendix - week 2

```
# Problem 1!!

store=[] # Empty array to store values
for a in range (1000, 2000): # Loop to check over all numbers in range
    if (a % 11 == 0) and not (a % 3 == 0):
# Above line makes sure if multiple of 11 and not of 3 execute line below
        store.append(str(a)) # Stores numbers that met above requirements
print (*store, sep = ", ")
# Learnt the above print line from the website below to print output correctly
# https://www.kite.com/python/answers/how-to-print-a-list-without-brackets-in-py
```

Figure 20 - Week 2 Problem 1

```
1001, 1012, 1034, 1045, 1067, 1078, 1100, 1111, 1133, 1144, 1166, 1177,
1199, 1210, 1232, 1243, 1265, 1276, 1298, 1309, 1331, 1342, 1364, 1375,
1397, 1408, 1430, 1441, 1463, 1474, 1496, 1507, 1529, 1540, 1562, 1573,
1595, 1606, 1628, 1639, 1661, 1672, 1694, 1705, 1727, 1738, 1760, 1771,
1793, 1804, 1826, 1837, 1859, 1870, 1892, 1903, 1925, 1936, 1958, 1969,
1991
```

Figure 21 - Week 2 Problem 1 output

```
# Problem 2!!

print("Please input a sentance: ")  #Allows user to input sentance
sentance = input()
# Above line assigns input to the varibale called sentance
# Below 2 lines will be used as counters for upper and lower case letters
UpperCase = 0
LowerCase = 0
# For loop to check each character for the length of the string sentance
for char in range(len(sentance)):
# Below says if char is in the lower letter alphabet add  and assigns to lower
# case counter else if in the upper case alphabet add to the upper counter
    if(sentance[char]>='A' and sentance[char]<='Z'):
        UpperCase += 1
    elif(sentance[char]>='a' and sentance[char]<='z'):
# Learnt in my other module how to convert from lower case to upper case
# without libraries so played around with the code as it's like a range
# and that's how I got the above line
        LowerCase += 1 # Add 1 to counter
print('Upper case = ', UpperCase)
print('Lower case = ', LowerCase)
# Above prints the count and I used the comma to print the string and counter
# int. As I only mentioned the alpahbets there is no issue with the space and
# is not counted by accident.
```

Figure 22 - Week 2 problem 2

```
Please input a sentance:

Hi I am Chris
Upper case =  3
Lower case =  7
```

Figure 23 - Week 2 problem 2 output

```
# Problem 3!!

# Below made a funtion that turns an int into a string
def NumToWord(a):
    b = str(a) # Casts int into a string and stored in b
    print(b) # Prints b which is casted into a string
    print(type(b)) # Double check what daat tybe b is
# Below int is used to make sure input value is an integer, learnt last
# academic year.
num = int(input("Please enter a number: "))
NumToWord(num) # Calls functions and passes input "num" into the funciton.
```

Figure 24 - Week 2 Problem 3

```
Please enter a number: 6
6
<class 'str'>
```

Figure 25 - Week 2 Problem 3 output

```python
# Problem 4!!

import itertools # Import from library to help iterate through all outcomes

# Below stored for easy access
subject = ["I", "You"]
verb = ["Read", "Borrow"]
ob = ["Shakerpeare's plays", "Shakespeare's poems"]

# Below prints and iterates over each possible out come from the lists
# mentioned whille the varibles stay in the same order. List ensures prints
# in the right way
print(list(itertools.product(subject, verb, ob)))

# https://www.codegrepper.com/code-examples/python/how+to+find+all+combinatic
```

Figure 26 - Week 2 Problem 4

```
[('I', 'Read', "Shakerpeare's plays"), ('I', 'Read', "Shakespeare's
poems"), ('I', 'Borrow', "Shakerpeare's plays"), ('I', 'Borrow',
"Shakespeare's poems"), ('You', 'Read', "Shakerpeare's plays"), ('You',
'Read', "Shakespeare's poems"), ('You', 'Borrow', "Shakerpeare's
plays"), ('You', 'Borrow', "Shakespeare's poems")]
```

Figure 27 - Week 2 problem 4 output

```python
# Problem 5!! Part 1

import matplotlib.pyplot as plt # imported and given a shorter name

x, y = [1, 2, 3], [2, 4, 1] # Assigning values to varibles x and y
plt.xlabel("X axis", fontsize = 15) # Prints x label and size
plt.ylabel("Y axis", fontsize = 15) # Prints y label and size
plt.title("My first graph", fontsize = 20) # Prints title
# Learnt how to change size and label names from
# https://stackoverflow.com/questions/12444716/how-do-i-set-the-figure-
# Some of above learnt from lectures and exra study help from uni.

# This plots the points on the graph
plt.plot(x, y)
plt.show() # This shows the graph
```

Figure 28 - Week 2 Problem 5 Part 1

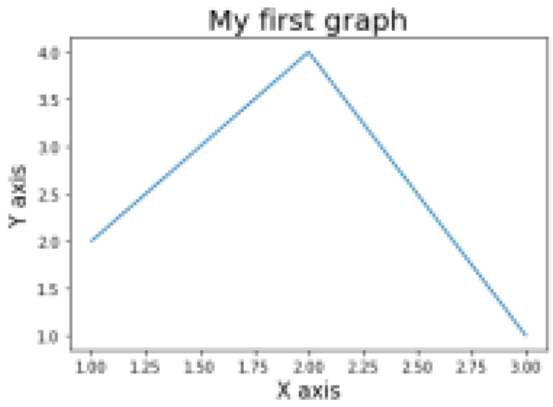

Figure 29 - Week 2 Problem 5 Part 1 graph

```
# Part 2

X = [] # Created empty lists to store values read from document
Y = []

a = open("test.txt", "r") # a is a variable which are the contents
for row in a: # Loops all rows in the txt file
    row = row.split(" ") # splits numbers in file when it reads a space
    X.append(row[0]) # First nunber is added to X
    Y.append(int(row[1])) # Second number is added to Y

plt.xlabel("X axis", fontsize = 15) # Prints x label
plt.ylabel("Y axis", fontsize = 15) # Prints y label
plt.title("My second graph", fontsize = 20) # Prints title

plt.plot(X, Y) # This plots the points on the graph
plt.show() # This shows the graph

#https://www.geeksforgeeks.org/python-create-graph-from-text-file/
```

Figure 30 - Week 2 Problem 5 Part 2

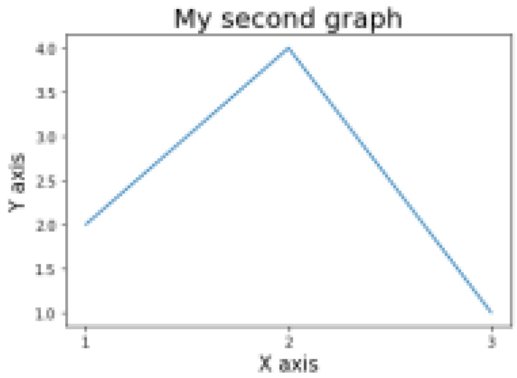

Figure 31 - Week 2 Problem 5 Part 2 graph

```python
# Problem 6!!

# below importing relevant libraries
import pandas as pd
import matplotlib.pyplot as plt

df = pd.read_csv (r"C:\Users\chris\Downloads\Uni\EHU year 2\Data mining\Course work\Coursework 1\Data\train.csv")
# Above imports and reads the data set

df.info() # Did this to see how many columns there are along with what data
# types are in the data set which are 3, along with being able to see which
# columns have missing dat

df["Loan_Status"].value_counts(normalize=True).plot.bar()
# Used to see the column which shows how many people got approved in a barchart

Columns = ["Gender", "Married", "Dependents", "Education", "Self_\
Employed","Property_Area","Credit_History"]
for x in Columns: # Loops over all data in each column
# Crosstab checks against another group of data I want to analyse against,
# in this case Loan_Status https://pbpython.com/pandas-crosstab.html against
# all the columns in Columns
    y = pd.crosstab(df["Loan_Status"], df[x], normalize = "columns")
# https://www.journaldev.com/45109/normalize-data-in-python taught me how
# to normalize data and https://machinelearningmastery.com/rescaling-data-for-machine-learning-in-python-with-sciki
# taught me what it does, makes all values inbetween 0 and 1.
    print(y) # Prints output
    y.plot(kind = "bar") # Plots bar chart for each column

df.boxplot(column = "ApplicantIncome", by = "Education") # Wanted to see the
# correlation between graduate income and non graduate income

numColumns = ["ApplicantIncome", "CoapplicantIncome", "LoanAmount", "Loan_Amount_Term"]

# I did above as I wanted to check if graduates earned more than non graduates
# Learnt this in lectue slides
for z in numColumns: # For loop to make a graph for each column
    # for each loop until every column in numColumns has a graph
    # shows column in numColumns against Loan_status
    result = df.boxplot(column = z, by = "Loan_Status") # Plots graph
    plt.show(result) # Shows graphs
# The graphs used in the abov loop were learnt from the lecture slides
```

Figure 32 - Week 2 Problem 6

```
Gender          Female      Male
Loan_Status
N            0.330357  0.306748
Y            0.669643  0.693252
Married            No       Yes
Loan_Status
N            0.370892  0.28392
Y            0.629108  0.71608
Dependents          0         1         2        3+
Loan_Status
N            0.310145  0.352941  0.247525  0.352941
Y            0.689855  0.647059  0.752475  0.647059
Education    Graduate  Not Graduate
Loan_Status
N            0.291667       0.38806
Y            0.708333       0.61194
Self_Employed     No       Yes
Loan_Status
N              0.314  0.317073
Y              0.686  0.682927
Property_Area     Rural  Semiurban     Urban
Loan_Status
N              0.385475    0.23176  0.341584
Y              0.614525    0.76824  0.658416
Credit_History       0.0       1.0
Loan_Status
N               0.921348  0.204211
Y               0.078652  0.795789
```

Figure 33 - Week 2 Problem 6 output

Practical Session Log - Week 2

Name: Christopher Diaz Montoya Date: 14/10/2021 Time: 14:00

Briefly describe the purpose of this session:

The purpose of this session is to teach us more about python and control flow, along with teaching us how to input data sets and read them in a table of a graph from a csv file and text file, it was also about learning to deduce an analysis from the graphs I made.

Describe the procedure(s) you followed to complete today's session:

During the lecture we went over quite a bit such as primitive data types such as integers, bools & strings. Along with lists, tuples, and dictionaries, we also went over control flow, I/O, functions, libraries, and modules. Finally, we briefly went over handling exceptions.

In the seminar we had to do some python exercises on our IDE, on multiple skills we learnt over last year and from the lecture and seminar slides. The slides and a handout given to us taught us about

numpy arrays and how to plot our first graphs and make our first graphs from data sets. We also had to analyse the last problem and write our findings.

- The first problem asked us to find certain numbers in a range, so I used a for loop to set the range and then used an if statement and a conditional statement within to ensure only the numbers that met the criteria were printed.
- The second problem asked to calculate the number of upper and lower-case letters, again I used a for loop to iterate over each character in the sentence and created two variables, Upper and lower case. These are the counters so if the condition is met then 1 is added to the variable. I then used an if else statement to add to upper case if it was in the upper-case alphabet range and same for the lower case.
- Problem 3 asked for a function that converts integers to strings, I created a function that casts the inputted value to an int and then prints it out. I also checked the data type to ensure it was a string.
- Problem 4 I needed help with and had to google how to create every possible outcome with itertools and assigned the right words to the right variable so that I could print the desired outcome.
- Problem 5 was plotting my first graph, we learnt in lesson, and I googled as well how to assign values to plot x and y on a line graph along with learning how to give the axis and graph a title and change small bits in the graph. I used matplotlib.pyplot for this. Part 2 then asked us to do the same but for the graph to be made from data in a text file. I used a for loop for this so it would iterate over every value in the text document, I split the 2 columns in the text document and assigned them to X and Y to get the desired graph.
- Problem 6 was trickier than the other exercises and extra learning was needed, I managed to do it only using pandas to upload the data set, I had to learn what normalize and crosstabs were. Cross tabs allowed me to check the data in the columns and see how it affected the loan status and normalize helped get all the data in between 0 and 1 and made it so the output was according to acceptance rate and not the total amount of applicants, for example without it, it gave me a bar chart that showed how many men were accepted and how many women as opposed to the actual relevance on what percent of women and what percent of men were accepted. I still do not 100% understand it and I am going to need to as I have seen it pop up on a lot online. I also knew df.info(), organising columns and bar charts from a previous AI course I did last academic year. I also used a box plot taught in lesson to compare educational background with salary as I had a theory that there was a correlation. I then used the same boxplots to compare numerical type columns so I could deduce some information from the graph.

If valid, state the reasons you have not been able to complete your work:

n/a

Please comment on today's session:

It was a more challenging but a fun session and starting to make us feel more like data scientists now we are starting to analyse and interpret data which is the reason I started this degree.

I do want to learn more about normalize as it also came up last academic year. I also realised some data was missing and want to learn how to clean data, so the missing values do not affect the output. I am also happy that I am comfortable with casting, functions, control flow, and conditional statements as I did not code much over summer due to personal circumstances, but happy to be doing it again. I do need to go more in depth for how to make graphs and different types of them as I am able to do them but with slight difficulty and as a future data scientist this must become second nature.

Week 3

Problem aim

The aim was to ensure that the data was pre-processed and ready to be used, this was done by ensuring all categorical types of data were changed to numerical types, so it is easier to work with, this is data transformation. Then the aim is to ensure any missing values were dealt with to not hinder the output. Outliers could have also been dealt with but there was not enough time, this is something that could improve the data pre-processing.

Problem code

The aim was to pre-process the data in the given dataset. This was done by first importing the data in a csv file with pandas calling the file path and assigning the data to a variable called df. Then the head of the data was printed to check if it needed transforming and to ensure all columns were relevant. The Loan_Id column was dropped as it was irrelevant to the loan approval. Next all the columns were looped over to check which columns had missing data, the uncleaned data can be viewed in Figure 36 - Un processed data. After this label encoding which was taught in lecture was used to ensure all categorical data types were changed to numeric, this is called data transformation. Label encoding was imported to turn everything in the column to a number between 0 and 1, also known as normalizing, the chosen column was normalised and finally updated on the data frame. The penultimate step was to then do something with the columns which had missing data, this was then looped over again but in place of the missing data the median was added. The code for everything can be viewed in Figure 34 - Week 3 code part 1 and in Figure 35 - Week 3 code part 2.

The final step was to reprint all the columns to check if there was any missing data, this was done with a for loop over each column, then the column and amount of missing data would be formatted into the respective “{}”. isnull().sum() was taught in class and used to check the cleaned data if any values were missing. The cleaned data can be viewed in Figure 37 - Pre-processed data which has no missing data and no categorical data, the data has been pre-processed.

References - Week 3

BROWNLEE, J., 2014. *Rescaling Data for Machine Learning in Python with Scikit-Learn* [online]. Available from: https://machinelearningmastery.com/rescaling-data-for-machine-learning-in-python-with-scikit-learn/ [Accessed 15 October 2021].

Data school., Nan., *How do I remove columns from a pandas Data Frame?* [online video]. Available from: https://www.youtube.com/watch?v=gnUKkS964WQ [Accessed 15 October 2020].

GURAV, S., 2020. *Label Encoder and OneHot Encoder in Python* [online]. Available from: https://towardsdatascience.com/label-encoder-and-onehot-encoder-in-python-83d32288b592 [Accessed 15 October 2021].

J. Sessa and D. Syed., 2017. "Techniques to deal with missing data," *2016 5th International Conference on Electronic Devices, Systems and Applications (ICEDSA)*, 6-8 Dec 2016, United Arab Emirates [online]. Piscataway, N.J: Institute of Electrical and Electronics Engineers. pp. 1-4. Available from: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7818486 [Accessed 16 October 2021].

Stich. *What is data transformation: definition, benefits, and uses* [online]. Available from: https://www.stitchdata.com/resources/data-transformation/ {Accessed 15 October 2021].

Up data science., 2020. *Machine learning features engineering: label ending vs one-hot encoding (using Scikit-learn)* [online video]. Available from: https://www.youtube.com/watch?v=mHkdcA-zyYE [Accessed 16 October 2021].

Appendix - Week 3

```
import pandas as pd
from sklearn.preprocessing import LabelEncoder

df = pd.read_csv(r"C:\Users\chris\Downloads\Uni\EHU year 2\Data mining\Cou
print("Amount of rows = {}\nAmount of colu\
mns = {}".format(df.shape[0], df.shape[1]))
# Above row is to see the shape of the data set
print(df.head)
# Above row is to print table of data set to see what it lokks like

# Drops/gets rid of the ID column as it is not needed
df = df.drop(columns = ["Loan_ID"])

columns = df.columns.tolist() # Everything in dara frame
for x in columns:
  print("Missing data in {} : {}".format(x, df[x].isnull().sum()))
# Checks over column and then rows, the first {} represents column names
# the second {} represents amount of missing data.

# Turning loan status into 0 or 1
l1 = LabelEncoder()
l1.fit(df["Loan_Status"])
df.Loan_Status = l1.transform(df.Loan_Status)

# Turning gender status into 0 or 1
l1 = LabelEncoder()
l1.fit(df["Gender"])
df.Gender = l1.transform(df.Gender)

# Turning married status into 0 or 1
l1 = LabelEncoder()
l1.fit(df["Married"])
df.Married = l1.transform(df.Married)

# Getting rid of + to just numbers can be read easier
```

Figure 34 - Week 3 code part 1

```
# Getting rid of + to just numbers can be read easier
l1 = LabelEncoder()
l1.fit(df["Dependents"])
df.Dependents = l1.transform(df.Dependents)

# Turning employed and self emplyed to 0 and 1 easier to work with
l1 = LabelEncoder()
l1.fit(df["Self_Employed"])
df.Self_Employed = l1.transform(df.Self_Employed)

# Turning graduate and non graduate in eduaction to 0 and 1
l1 = LabelEncoder()
l1.fit(df["Education"])
df.Education = l1.transform(df.Education)

# Sets property location to a numerical value based on the 3 lcoations
l1 = LabelEncoder()
l1.fit(df["Property_Area"])
df.Property_Area = l1.transform(df.Property_Area)
print(df) # Prints out data frame to check all values are of numerical type
# Above is data transformation

df.fillna(df.median(), inplace = True) # Fills in missing values with median
allColumns = df.columns # allColumns = all columns in the data frame
for col in allColumns: #For each column
  df[col] = pd.to_numeric(df[col]) # Turns argument into a numeric type


# Below is same code as above to check for missing data
columns = df.columns.tolist()
for x in columns:
  print("Missing data in {} : {}".format(x, df[x].isnull().sum()))
# Above isnull().sum() was used to check missing values, learnt in class
```

Figure 35 - Week 3 code part 2

```
Amount of rows = 614
Amount of columns = 13
<bound method NDFrame.head of       Loan_ID  Gender Married  ... Credit_History Property_Area Loan_Status
0    LP001002    Male      No  ...            1.0         Urban           Y
1    LP001003    Male     Yes  ...            1.0         Rural           N
2    LP001005    Male     Yes  ...            1.0         Urban           Y
3    LP001006    Male     Yes  ...            1.0         Urban           Y
4    LP001008    Male      No  ...            1.0         Urban           Y
..        ...     ...     ...  ...            ...           ...         ...
609  LP002978  Female      No  ...            1.0         Rural           Y
610  LP002979    Male     Yes  ...            1.0         Rural           Y
611  LP002983    Male     Yes  ...            1.0         Urban           Y
612  LP002984    Male     Yes  ...            1.0         Urban           Y
613  LP002990  Female      No  ...            0.0     Semiurban           N

[614 rows x 13 columns]>
Missing data in Gender : 13
Missing data in Married : 3
Missing data in Dependents : 15
Missing data in Education : 0
Missing data in Self_Employed : 32
Missing data in ApplicantIncome : 0
Missing data in CoapplicantIncome : 0
Missing data in LoanAmount : 22
Missing data in Loan_Amount_Term : 14
Missing data in Credit_History : 50
Missing data in Property_Area : 0
Missing data in Loan_Status : 0
```

Figure 36 - Un processed data

```
     Gender  Married  Dependents  ...  Credit_History  Property_Area  Loan_Status
0         1        0           0  ...             1.0              2            1
1         1        1           1  ...             1.0              0            0
2         1        1           0  ...             1.0              2            1
3         1        1           0  ...             1.0              2            1
4         1        0           0  ...             1.0              2            1
..      ...      ...         ...  ...             ...            ...          ...
609       0        0           0  ...             1.0              0            1
610       1        1           3  ...             1.0              0            1
611       1        1           1  ...             1.0              2            1
612       1        1           2  ...             1.0              2            1
613       0        0           0  ...             0.0              1            0

[614 rows x 12 columns]
Missing data in Gender : 0
Missing data in Married : 0
Missing data in Dependents : 0
Missing data in Education : 0
Missing data in Self_Employed : 0
Missing data in ApplicantIncome : 0
Missing data in CoapplicantIncome : 0
Missing data in LoanAmount : 0
Missing data in Loan_Amount_Term : 0
Missing data in Credit_History : 0
Missing data in Property_Area : 0
Missing data in Loan_Status : 0
```

Figure 37 - Pre-processed data

Practical Session Log - Week 3

Name: Christopher Diaz Montoya Date: 21/10/2021 Time: 16:00

Briefly describe the purpose of this session:

The purpose of this session is to learn to pre process data.

Describe the procedure(s) you followed to complete today's session:

During the lecture some pre-processing methods and normalization were explained along with how to find and display missing data.

In the seminar we had to pre-process a data set, originally, I thought all we had to do was fix the missing values in the data. But there is much mor too it then that, we must ensure all the data is numerical and not categorical, so it is easier to work with and to remove any outliers.

For the task I ensured I transformed all the data into numerical data from categorical and then fixed all the missing data by inputting the median. I then checked over the data and it looked nice and clean. I did not have enough time to remove outliers, but this is something I must know and will go over, this also could have made my data more consistent.

If valid, state the reasons you have not been able to complete your work:

n/a

Please comment on today's session:

Today's session was fun, everyday I am feeling more and more like a data scientist, and I love it. I was trying to loop over all the categorical columns to turn them into numerical instead of doing each individually, but it would not work, I also tried it with a function, but no luck. This is something I should look into to not have to write the same code over and over. Also, I need to investigate removing outliers or what is the best thing to do with them.