**DATA, INFERENCE**

**&**

**APPLIED MACHINE LEARNING**

**(COURSE 18-785)**

**ASSIGNMENT 2**

**Mark Iraguha**

**(miraguha)**

**Libraries Used:**

Matplotlib – a python plotting library used to create animated, interactive and static visualizations.[1]

Pandas – another Python library used that provides data structures and functions used to carry out data analysis.[2]

Numpy – a simple yet powerful data structure provided in python.[3]

Tabulate – a python library that tabulates data to an output[4].

**Introduction:**

This report details the completion of Assignment 2. Assignment 2 requests answers to 5 critical thinking and data analytical questions.

**Question 1 Report:**

**Methodology**

* 1. Finding the number of folds, it would take to exceed a given height.
* Approach:
* Defines key variables i.e paper thickness, everestHeight.
* Converted the thickness of paper to meters so it matches the unit of the height of Mount Everest.
* Exponentially increase the paper thickness up to a point where any other fold would exceed the height of Everest.
* Display the number of folds up to this point.

**Results:**

Finding number of folds

Using a loop, I found the folds to get close to the height of Mount Everest. I then added 1 extra fold to my folds counter because that extra fold would mean exceeding the height of Mount Everest. The total number of folds are then displayed.



*Question 1: Snapshot showing code output*

**Analysis and Insights**

Exponential Behavior of paper folds

* It was learnt that papers once folded double in thickness each time. With this new knowledge, it would then be observed that this behavior can be called exponential growth.[5]

**Question 2 Report:**

**Methodology**

Finding the time at which volume of water is half its initial volume

* Approach:
* We assume that a is 0.1. a represents the decay constant in the exponential equation.
* We also recognize that time t will be equal to v(0)/2. v(0) being the initial volume.
* We can then re-arrange the formula using the half-life formula from the exponential equation. Resulting into t = ln(2)/a.
* ln(2) is the Natural Logarithm of 2 and in our code we use the log function to represent this.[6]
* This way, we can now assume any volume of water i.e. 22,777 to give us time t which is approximately 6.931

**Results:**

Finding time t

Upon following the approach outlined above, we now know that any volume of water assumed will give use the approximate units of time it would take for that volume of water to reach half of its initial volume.



*Question 2: Snapshot showing code output*

**Analysis and Insights**

Exponential decay

* The equation used represents exponential decay and it is surprising that any volume assumed produces the same half-life. [6] For example, v(0) as 100 or 10 or 22,000 all produce the same units of time, approximately 6.931.

**Question 3 Report:**

**Methodology**

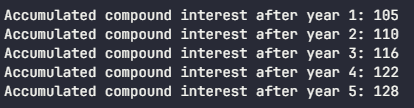
Utilizing the compound interest formula

* Approach:
* We can use the compound interest formula a=p(1+r/n)nt
* ‘a’ is the annual compounding, ‘p’ is the principal, ‘r’ is the annualized interest rate (as a decimal), ‘n’ is the number of months in a year, ‘t’ is the number of years.
* With this formula, we can also use a loop to get the annual compounding at the end of each year of the 5 year period.

**Results:**

Finding compounded interest after each year of 5 year period

Upon applying the formula, we get the annualized interest added to the principal from the previous year. This is printed out for each year of the 5 year period.



*Question 3: Snapshot showing code output*

**Analysis and Insights**

Annualized Compound Interest

* With a simple formula you can get to know your return for a 5 year period without adding the interest of year manually after every end of the year.
* Noticeably, the interest return for a principal of $100 is approximately $5 dollars using an annualized compound interest of 5%.

**Question 4 Report:**

**Methodology**

Using the loan repayment formula

* Approach:
* We utilize the loan repayment formula: p=r(pv)/1-(1+R)-n
* Where ‘p’ is the payment, ‘PV’ is the present value, ‘r’ is the rate per period, ‘n’ number of periods

**Results:**

Finding monthly contribution if the payment term was ‘n’ years.

Using the loan repayment formula, we found the monthly contribution to the car loan if the loan repayment period was 1 or 2 or 3 years.



*Question 4: Snapshot from code output*

**Analysis and Insights**

Loan Repayment

* Having utilized the formula for the repayment period, we observe that I would pay more monthly if the payment term if shorter as opposed to a longer term.

**Question 5 Report:**

**Methodology**

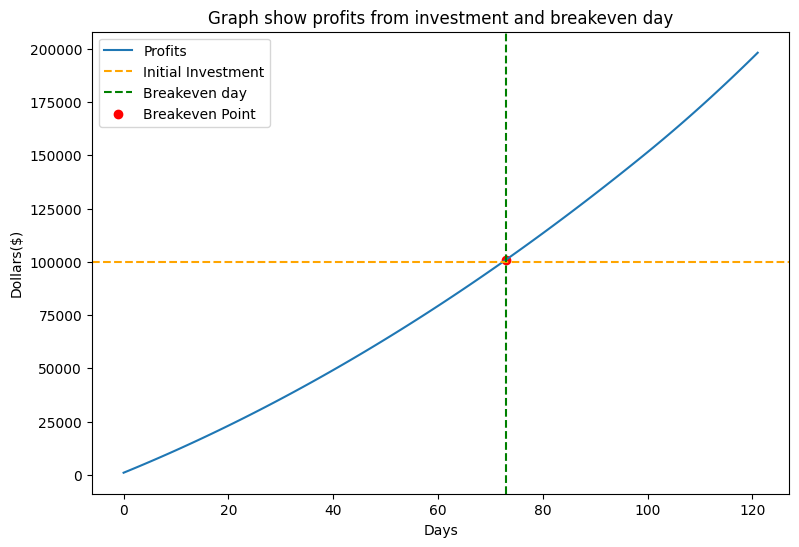
Find profits by exponential growth of customers

* Approach:
* We make sure to calculate the exponential growth of customers every day.
* All daily profits are then calculated according to the new number of customers.
* We store the results in a collection.
* We then find the point (day) when the initial investment is realized.
* We use the data above to configure a line graph plotting the profits against the days.
* We mark the initial investment and breakeven day.

**Results:**

Plotting cumulated profits:

Utilizing the matplotlib[1] and pandas[2], we use the profits per day data to plot a line graph marking the initial investment and breakeven day.



*Question 5: Figure 1 showing Graph*

**Analysis and Insights**

Exponential decay

* We observe a linear trend in the growth of profits. Noticing that getting double the initial investment takes a shorter time compared to the time it took to realize the initial investment. We could predict that profits get larger as the exponential growth of customers gets bigger.

References:

[1] “Matplotlib,” *Wikipedia*. Aug. 30, 2024. Accessed: Sep. 01, 2024. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Matplotlib&oldid=1243075914

[2] “pandas (software),” *Wikipedia*. Jul. 15, 2024. Accessed: Sep. 01, 2024. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Pandas\_(software)&oldid=1234683004

[3] R. Python, “NumPy Tutorial: Your First Steps Into Data Science in Python – Real Python.” Accessed: Sep. 02, 2024. [Online]. Available: https://realpython.com/numpy-tutorial/

[4] *tabulate: Pretty-print tabular data*. Python. Accessed: Sep. 15, 2024. [OS Independent]. Available: https://github.com/astanin/python-tabulate

[5] K. S. Kruszelnicki, “Folding paper.” Accessed: Sep. 01, 2024. [Online]. Available: https://www.abc.net.au/science/articles/2005/12/21/1523497.htm

[6] “Exponential decay,” *Wikipedia*. Jun. 07, 2024. Accessed: Sep. 01, 2024. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Exponential\_decay&oldid=1227744550

[7] “Growth Rates: Formula, How to Calculate, and Definition,” Investopedia. Accessed: Sep. 02, 2024. [Online]. Available: https://www.investopedia.com/terms/g/growthrates.asp

[8] “Python | Pandas dataframe.pct\_change(),” GeeksforGeeks. Accessed: Sep. 02, 2024. [Online]. Available: https://www.geeksforgeeks.org/python-pandas-dataframe-pct\_change/