PROGRAMMING FOR DATA SCIENCE 7143CEM

SUBMITTED BY:

RAMANA KULANTHAIVELU

STUDENT ID: 14231913

Contents:

Introduction:

In this assignment, we hope to reinforce our knowledge of Python by designing, building, debugging, and visualizing data sets using data science tools such as pandas, seaborn, and matplot. Also, provide enthusiastic experience in each task, particularly in finding a perfect plot by searching on BBC. We are focusing on Data Ethics and Data Protection at the end of the assignment by analyzing recent data breach cases.

Task 1. Explain, critique, comment and debug code (MLO2)

a) This algorithm sorts elements in descending order from L[i] (where i=1) to L[i + 1] (where i =len[L]-3), excluding the first element and last two elements of the list. This sorting algorithm compares adjacent elements, swaps them if they're in the wrong order, and repeats until sorted. The provided code for sorting list L contains multiple errors that require fixing. The code must follow a good Python coding style, including consistent indentation to avoid errors and unexpected executions. In this code, there are a lot of indentation errors causing the program to halt. It is recommended that we use four spaces for each indentation level. When defining functions and classes in Python, a colon is used to indicate the start of a new block. Similarly, loops also use a colon to indicate the start of a new block. There are missing colons in 'while', 'for', and 'if' statements, as well as in function definitions, which gives syntax errors. The 'for' loops in the provided code contain an error in their 'if' statement. The condition being checked is "if(L[i]<=L[i+1]):", which may cause issues when adjacent elements in the list are the same. The condition checks whether the element at index i in the list L is less than or equal to that at index i+1. However, when two adjacent elements are equal, the condition L[i] <= L[i+1] will be True. The 'if L[i] <= L[i + 1]' condition checks for non-increasing order, but it swaps elements even when they are already in the desired order, contradicting typical sorting logic. The sorting logic in my python code led to an unexpected behavior and halted while running. The loop ranges and element swapping condition needs revision for a correct implementation. The sorting algorithm is missing swapping statements, which are necessary to sort. The return statement specifies the function's return value when executed, which is missing here.

The final Python code for sorting the list:

```
################## Fixing the given sorting algorithm #################

def whatdoido(L):
    s = True                                    # Flag is set to True
    while (s):                                  # Testing the Flag
        s = False
        for i in range(1,len(L)-3):             # setting the range limitations
            if (L[i]) < L[i+1]:                 # comparing the elements in forward
                L[i] , L[i+1] = L[i+1] , L[i]   # swapping the elements
                s = True
        for i in range(len(L)-3,1,-1):          # setting the range limitations
            if (L[i]) < L[i+1]:                 # comparing the elements in reverse
                L[i] , L[i+1] = L[i+1] , L[i]   # swapping the elements if it
                s = True
    return(L)                                   # returning the predicted list
```

b) Modified Python code for sorting the list with misarranged elements between the index L[1] and L[len(L)-2]

```
def whatdoido(L):
    print('\n#####Sorting the misarranged middle part of the list####\n')
    print('The given list is L;',L)
    comp = 0
    swap = 0
    s = True
    while (s):
        s = False
        print('\nPassing forward through the list\n')
        for i in range(1,len(L)-3):
            print('Comparing L[',i,'] with L[',i+1,']')
            comp = comp + 1
            if (L[i]) < L[i+1]:
                print('The element L[',i,']=',L[i],'is less than L[',i,']=',L[i+1])
                print('Swapping the order of the elements\n')
                swap = swap + 1
                L[i] , L[i+1] = L[i+1] , L[i]
                s = True
            elif (L[i] > L[i+1]):
                print('The elements L[',i,'] and L[',i+1,'] are in correct order')
                print('No need to swap ,moving to next element...\n')
            else:
                print('The elements L[',i,'] and L[',i+1,'] are equal')
                print('No need to swap ,moving to next element...\n')
        print('\nPassing backward through the list\n')
        for i in range(len(L)-3,1,-1):
            print('Comparing L[',i,'] with L[',i+1,']')
            comp = comp + 1
            if (L[i]) < L[i+1]:
                print('The element L[',i,']=',L[i],'is less than L[',i,']=',L[i+1])
                print('Swapping the order of the elements\n')
                L[i] , L[i+1] = L[i+1] , L[i]
                swap = swap + 1
                s = True
            elif (L[i] > L[i+1]):
                print('The elements L[',i,'] and L[',i+1,'] are in correct order')
                print('No need to swap ,moving to next element...\n')
            else:
                print('The elements L[',i,'] and L[',i+1,'] are equal')
                print('No need to swap ,moving to next element...\n')
    print('\nThe list is sorted, No more comparisons or swapping operations are
required.\n')
    print(' No. of comparisons made :',comp,'\n')
    print(' No. of swaps made       :',swap,'\n')
    print('The final sorted list:')
    return(L)
L=[9,6,3,5,5,7,4,1,0]
print(whatdoido(L))
```

Output:

#####Sorting the misarranged middle part of the list####

The given list is L; [9, 6, 3, 5, 5, 7, 4, 1, 0]

Passing forward through the list

Comparing L[1] with L[2]
The elements L[1] and L[2] are in correct order
No need to swap ,moving to next element...

Comparing L[2] with L[3]
The element L[2]= 3 is less than L[2]= 5
Swapping the order of the elements

Comparing L[3] with L[4]
The element L[3]= 3 is less than L[3]= 5
Swapping the order of the elements

Comparing L[4] with L[5]
The element L[4]= 3 is less than L[4]= 7
Swapping the order of the elements

Comparing L[5] with L[6]
The element L[5]= 3 is less than L[5]= 4
Swapping the order of the elements

Passing backward through the list

Comparing L[6] with L[7]
The elements L[6] and L[7] are in correct order
No need to swap ,moving to next element...

Comparing L[5] with L[6]
The elements L[5] and L[6] are in correct order
No need to swap ,moving to next element...

Comparing L[4] with L[5]
The elements L[4] and L[5] are in correct order
No need to swap ,moving to next element...

Comparing L[3] with L[4]
The element L[3]= 5 is less than L[3]= 7
Swapping the order of the elements

Comparing L[2] with L[3]
The element L[2]= 5 is less than L[2]= 7
Swapping the order of the elements

Passing forward through the list

Comparing L[1] with L[2]
The element L[1]= 6 is less than L[1]= 7
Swapping the order of the elements

Comparing L[2] with L[3]
The elements L[2] and L[3] are in correct order
No need to swap ,moving to next element...

Comparing L[3] with L[4]
The elements L[3] and L[4] are equal
No need to swap ,moving to next element...

Comparing L[4] with L[5]
The elements L[4] and L[5] are in correct order
No need to swap ,moving to next element...

Comparing L[5] with L[6]
The elements L[5] and L[6] are in correct order
No need to swap ,moving to next element...

Passing backward through the list

Comparing L[6] with L[7]
The elements L[6] and L[7] are in correct order
No need to swap ,moving to next element...

Comparing L[5] with L[6]
The elements L[5] and L[6] are in correct order
No need to swap ,moving to next element...

Comparing L[4] with L[5]
The elements L[4] and L[5] are in correct order
No need to swap ,moving to next element...

Comparing L[3] with L[4]
The elements L[3] and L[4] are equal
No need to swap ,moving to next element...

Comparing L[2] with L[3]
The elements L[2] and L[3] are in correct order
No need to swap ,moving to next element...

Passing forward through the list

Comparing L[1] with L[2]
The elements L[1] and L[2] are in correct order
No need to swap ,moving to next element...

Comparing L[2] with L[3]
The elements L[2] and L[3] are in correct order
No need to swap ,moving to next element...

Comparing L[3] with L[4]
The elements L[3] and L[4] are equal
No need to swap ,moving to next element...

Comparing L[4] with L[5]
The elements L[4] and L[5] are in correct order
No need to swap ,moving to next element...

Comparing L[5] with L[6]
The elements L[5] and L[6] are in correct order
No need to swap ,moving to next element...

Passing backward through the list

Comparing L[6] with L[7]
The elements L[6] and L[7] are in correct order
No need to swap ,moving to next element...

Comparing L[5] with L[6]
The elements L[5] and L[6] are in correct order
No need to swap ,moving to next element...

Comparing L[4] with L[5]
The elements L[4] and L[5] are in correct order
No need to swap ,moving to next element...

Comparing L[3] with L[4]
The elements L[3] and L[4] are equal
No need to swap ,moving to next element...

Comparing L[2] with L[3]
The elements L[2] and L[3] are in correct order
No need to swap ,moving to next element...

The list is sorted, No more comparisons or swapping operations are required.

No. of comparisons made : 30

No. of swaps made : 7

The final sorted list:
[9, 7, 6, 5, 5, 4, 3, 1, 0]

Consequences of removing one of the 'for' loops in the code:

We utilized two 'for' loops in this sorting algorithm. When traversing the list, The first loop moves forward and compares elements in sequence and During the second for loop, the elements are compared in reverse order. Removing either one of them will lead to the elements at the start or end may not be in the correct position or it could result in the sorting process being inefficient or incorrect.

If we remove the second 'for' loop from the code, the resulting output would be **L = [9, 6, 5, 5, 7, 4, 3, 1, 0]**. However, this output is not correct because it fails to consider the value 7, which comes after the second 5. This is because the code does not compare values in the backward direction, which leads to incorrect results. This is the output we get when we remove the first 'for' loop. Without the forward comparison, the resulting list is **L = [9, 6, 7, 5, 5, 4, 3, 1, 0].**

Task 2: Design, build and test

a) Building owzthat game

The owzthat function in the code provided simulates a cricket game between two teams, where each team is represented by a player. The python inbuild function ‘import’ is imported. The function “owzthat” is created. The number of ball to be bowled and number of wickets called by input. The input parameters for the function are the number of balls to be bowled and the number of wickets in each team. It initializes variables for balls bowled, wickets taken, and runs scored. The while loop runs until either all balls are bowled (balls < c) or all players in a team are dismissed (wickets < d). During each iteration of the loop, a random number between 1 and 6 is generated using the random.randint() function to simulate a dice roll, which will be added as a score. The number obtained from the dice roll determines the outcome of the current ball. If the resulting number is 5, then the batsman will receive 'OWZTHAT'. The code incorporates a set of rules for determining different results based on the umpire's decision, such as being declared out through methods like 'bowled', 'stumped', 'caught', or 'lbw', encountering a 'no ball' scenario, or being declared not out. Wheter it is a ‘no ball’ , You will get a run and an additional ball or If ‘not out’ you will continue the match or else you will get a wicket , These all tested with given ‘if - else’ condition . Current score and the scored runs will be printed for each ball. At last, the function displays the overall runs scored, the total number of balls bowled, and the total number of wickets lost by the team. This piece of code enables the management of modifications in players and balls, all while preserving readability.

The Python code for Owzthat :

```
########## OWZTHAT ##########

"""Welcome to ozwath - a dice based cricket game...
    You are gonna bat with the batting dice which has 1,2,3,4,Owzthat,6.
    If you rolled any of the numbers you will get that number added to your
run score.
    If you accidently rolled 'Owzthat' the next move will be decided by the
Umpire dice.
    which has 'bowled','caught','stumbed','lbw','not out'and'no ball' if
you didn't get
    'no ball' or'not out' you will dismissed and if you got 'no ball' you
got one run and
    one extra ball and you can continue the game. If you got 'not out' You
can
    continue the match"""
```

```
import random
def owzthat(a,b):
    c=int(a)
    d=int(b)
    balls = 0
    wickets = 0
    run = 0

    while (balls<c and wickets<d):
        dice = random.randint(1,6)
        balls = balls + 1
        print('Ball number :',balls)
        print('Rolling the batting dice')
        if (dice == 5):
            print('\nYou got OWZTHAT !')
            print("Now the decision goes to the Umpire\n")
            print('\nNow rolling the Umpire dice')
            owzthat = ['bowled','stumped','caught','lbw','not out','no
ball']
            decision = random.choice(owzthat)
            out = {'bowled','stumped','caught','lbw'}
            if(decision in out):
                print('Oops! You got',decision)
                wickets = wickets + 1
                if(wickets<d):
                    print('Now the game moves to the next player')
            elif(decision == 'no ball'):
                print("Umpire declared this us a 'no ball' and you got one
run and an extra ball")
                run = run + 1
                balls = balls - 1
            else:
                print('its a not out \n')
        else:
            if(dice == 1):
               print('You got',dice,'run')
               run = run + dice
            else:
                print('You got',dice,'runs')
                run = run + dice
        print('Current score is :',run,'/',wickets,'\n')
    print("\nYou have scored ",run,'runs for',balls,'balls
with',wickets,'wickets')
print('Enter the number of balls to be bowled:')
a = input()
print('Enter the number of players:')
b = input()
print('\nFirst match.....\n')
owzthat(a, b)
print('\nsecond match.....\n')
owzthat(a, b)
```

Output:

Enter the number of balls to be bowled:

10

Enter the number of players:

2

First match.....

Ball number : 1
Rolling the batting dice
You got 3 runs
Current score is : 3 / 0

Ball number : 2
Rolling the batting dice
You got 2 runs
Current score is : 5 / 0

Ball number : 3
Rolling the batting dice
You got 6 runs
Current score is : 11 / 0

Ball number : 4
Rolling the batting dice

You got OWZTHAT !
Now the decision goes to the Umpire

Now rolling the Umpire dice
Oops! You got stumped
Now the game moves to the next player
Current score is : 11 / 1

Ball number : 5
Rolling the batting dice

You got OWZTHAT !
Now the decision goes to the Umpire

Now rolling the Umpire dice
Umpire declared this us a 'no ball' and you got one run and an extra ball
Current score is : 12 / 1

Ball number : 5
Rolling the batting dice
You got 4 runs
Current score is : 16 / 1

Ball number : 6
Rolling the batting dice
You got 2 runs
Current score is : 18 / 1

Ball number : 7
Rolling the batting dice

You got OWZTHAT !
Now the decision goes to the Umpire

Now rolling the Umpire dice
Oops! You got stumped
Current score is : 18 / 2

You have scored 18 runs for 7 balls with 2 wickets

second match.....

Ball number : 1
Rolling the batting dice
You got 2 runs
Current score is : 2 / 0

Ball number : 2
Rolling the batting dice
You got 6 runs
Current score is : 8 / 0

Ball number : 3
Rolling the batting dice
You got 2 runs
Current score is : 10 / 0

Ball number : 4
Rolling the batting dice
You got 1 run
Current score is : 11 / 0

Ball number : 5
Rolling the batting dice
You got 3 runs
Current score is : 14 / 0

Ball number : 6
Rolling the batting dice

You got OWZTHAT !
Now the decision goes to the Umpire

Now rolling the Umpire dice
Oops! You got stumped
Now the game moves to the next player

Current score is : 14 / 1

Ball number : 7
Rolling the batting dice
You got 6 runs
Current score is : 20 / 1

Ball number : 8
Rolling the batting dice
You got 4 runs
Current score is : 24 / 1

Ball number : 9
Rolling the batting dice

You got OWZTHAT !
Now the decision goes to the Umpire

Now rolling the Umpire dice
Oops! You got lbw
Current score is : 24 / 2

You have scored 24 runs for 9 balls with 2 wickets

b) Modified Owzthat python code resembling cricket

```
########## OWZTHAT ##########

import random
def owzthat(a,b):
    c=int(a) * 6
    d=int(b)
    balls = 0
    wickets = 0
    run = 0
    player_run = 0
    ind_score = []
    while (balls<c and wickets<d):
        dice = random.randint(1,6)
        balls = balls + 1
        print('\nBall number ',balls)
        print('Rolling the batting dice')
        if (dice == 5):
            print('\nYou got OWZTHAT')
            print("Now the decision goes to the Umpire\n")
            owzthat = ['bowled','stumped','caught','lbw','not out','no
ball']
            decision = random.choice(owzthat)
            out = {'bowled','stumped','caught','lbw'}
            if(decision in out):
                print('You got',decision,'You are out')
                print("You've scored:",player_run)
                ind_score.append(player_run)
                wickets = wickets + 1
```

```
                if(40<player_run<50):
                    req1 = 50 - player_run
                    print('You required just',req1,'runs for a Half
century')
                if(90<player_run<100):
                    req2 = 100 - player_run
                    print('You required just',req2,'runs for The Century')
                if(wickets<d):
                    print('Now the game moves to the next player')
                player_run = 0
            elif(decision == 'no ball'):
                print("Umpire declared this us a 'no ball' and you got one
run and one extra ball\n")
                run = run + 1
                player_run = player_run + 1
                balls = balls - 1
            else:
                print('its a not out\n')
        else:
            if(dice == 6):
                print('Booyah! it is a six\n')
            elif(dice == 4):
                print("It's a boundry\n")
            else:
                print('You got ',dice,' runs\n')
            run = run + dice
            player_run = player_run + dice
        if(balls%6 == 0):
            print('Current score upto this over is :',run,' /
',wickets,'\n')
        if(player_run == 50):
            print('You hit half century')
        if(player_run == 100):
            print("It's a historic moment in this match.. you smashed a
century")
    ind_score.append(player_run)
    if(wickets<d):
        for i in range(wickets + 1,d):
            player_run = 'player not entered in the match'
            ind_score.append(player_run)
    for i in range(0,d):
        print('The score of player',i + 1,'is :',ind_score[i])
    max_score=ind_score.index( max([x for x in ind_score if isinstance(x,
int)]))  # (Python, 2019)referenced this site to get max. int from the list

    print("\nBest performer of the match is : player",max_score + 1)
    print("\nYour team's final run is :",run)
    print("No.of balls bowled :",balls)
    print("No.of wickets",wickets)
print('Enter the number of overs:')
a = input()
print('Enter the number of players:')
b = input()
print('\nFirst match.....')
owzthat(a, b)
```

Outputs and conclusion:

Example output:1

Enter the number of overs:
5

Enter the number of players:
10

The program requires the user to input the number of players and the number of overs to be played in a cricket match. The program then calculates the total number of balls to be played, using the standard of 6 balls per over. The user can input any number of players they desire, and the program will calculate the total number of balls required for the match accordingly.

Example output:2

Ball number 20
Rolling the batting dice

You got OWZTHAT
Now the decision goes to the Umpire

You got lbw You are out
You've scored: 46

You required just 4 runs for a Half century
Now the game moves to the next player

The program awards "century" and "half-century" credits to players and provides information on how many runs are needed to achieve these milestones. We are considering overs so it is possible to hit them. There might be little chance in Owzthat to hit those scores.

Example Output: 3

Ball number 24
Rolling the batting dice
You got 3 runs

Current score upto this over is : 77 / 2

The program will display the score for each over.

Example Output: 4

Ball number 26
Rolling the batting dice
Booyah! it is a six

Ball number 5
Rolling the batting dice
It's a boundry

We are considering overs so it is possible to hit them. There might be little chance in Owzthat to hit those scores.

Example Output: 5

The score of player 1 is : 25
The score of player 2 is : 13
The score of player 3 is : 4
The score of player 4 is : 29
The score of player 5 is : player not entered in the match
The score of player 6 is : player not entered in the match
The score of player 7 is : player not entered in the match
The score of player 8 is : player not entered in the match
The score of player 9 is : player not entered in the match
The score of player 10 is : player not entered in the match

The score for each individual has been updated. If the match ends before all players enter, the remaining players will be assigned as 'not entered in the match' by using 'for'loop.

Example Output: 6

Best performer of the match is : player 4

Determine the highest value and select the best player from the list. The highest value is obtained by this piece of code, `' max_score=ind_score.index( max([x for x in ind_score if isinstance(x, int)]))'` *(Python, 2019).* This code obtained by referring relevant codes.

	ozwath	Moified owath
Individual player score list	No	Yes
Summary score of players	No	Yes
Century and Half century	No	Yes
Indication fours and sixes	No	Yes
Best performer identification	No	Yes
Tracing and identifying century requirement	No	Yes
Overs based scoring and balls	No	Yes

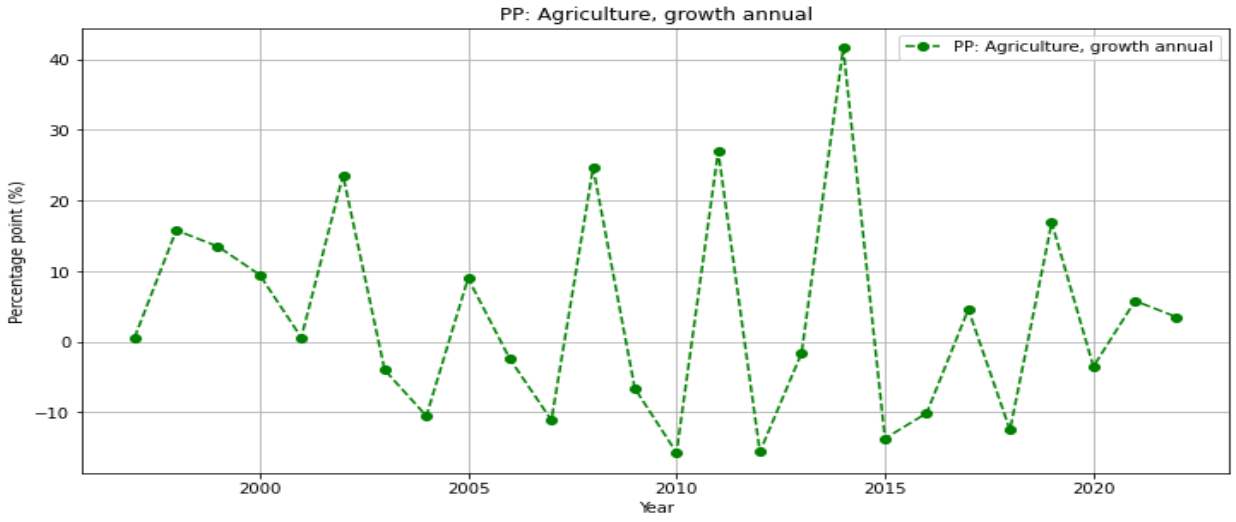
Task :3 Critically assess, select and apply data science tools (MLO3)

a) The data set utilized in this analysis is obtained from the Office for National Statistics (ONS), which typically involves evaluating the code's fitness and reviewing its alignment with project requirements.

The resource link:
https://www.ons.gov.uk/economy/grossdomesticproductgdp/timeseries/a2eq/pn2

Code and the produced plot:

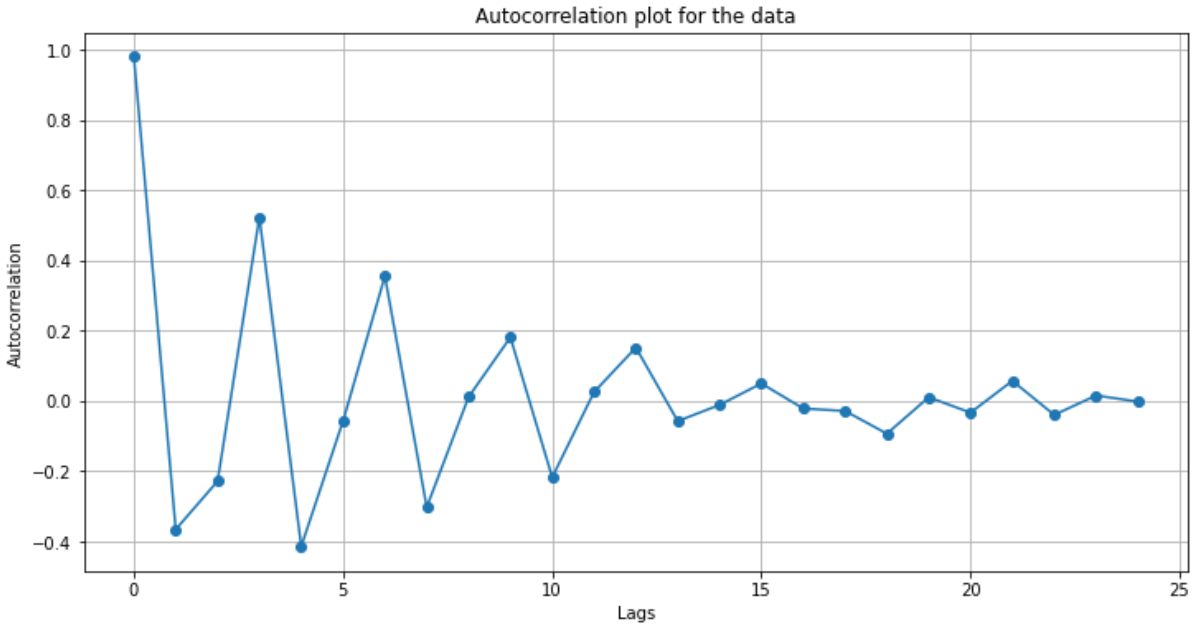
```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np

# Loading the csv file
df = pd.read_csv("series-231123 (2).csv")
df.head()

# Extracting the time series data and year
data= df['PP: Agriculture, growth annual'].copy()
year = df['Title'].copy()

# Converting the data to numeric and removing all NaN values
data = pd.to_numeric(data, errors='coerce').dropna()
year = pd.to_numeric(year, errors='coerce').dropna()  #('Python Pandas: How
to Remove Nan and Inf Values | Saturn Cloud Blog, 2023')

# Creating a line plot
plt.figure(figsize=(12, 6))
plt.plot(year, data, linestyle='--', marker='o', color='g', label='PP:
Agriculture, growth annual')
plt.title('PP: Agriculture, growth annual')
plt.xlabel('Year')
plt.ylabel('Percentage point (%)')
plt.grid(True)
plt.legend()
plt.show()
```



Time series analysis concept (Autocorrelation method):

Autocorelation plot and code:

```
# Calculating  the mean and the standard deviation
mean = np.mean(data)
std = np.std(data)
autocorrelated_values = []
# number of data points
n = len(data)

# Calculating the autocorrelation for each lag
for lag in range(n):
    cus_lag = data.shift(lag)
    cross_correlated =((data - mean) * (cus_lag - mean)).sum() / (std *
cus_lag.std() * n)
    autocorrelated_values.append(cross_correlated)


# Creating an autocorrelation plot
plt.figure(figsize=(12, 6))
plt.plot(autocorrelated_values, marker='o')
plt.title('Autocorrelation plot for the data')
plt.xlabel('Lags')
plt.ylabel('Autocorrelation')
plt.grid(True)
plt.show()
```



Conclusion regarding the chart:

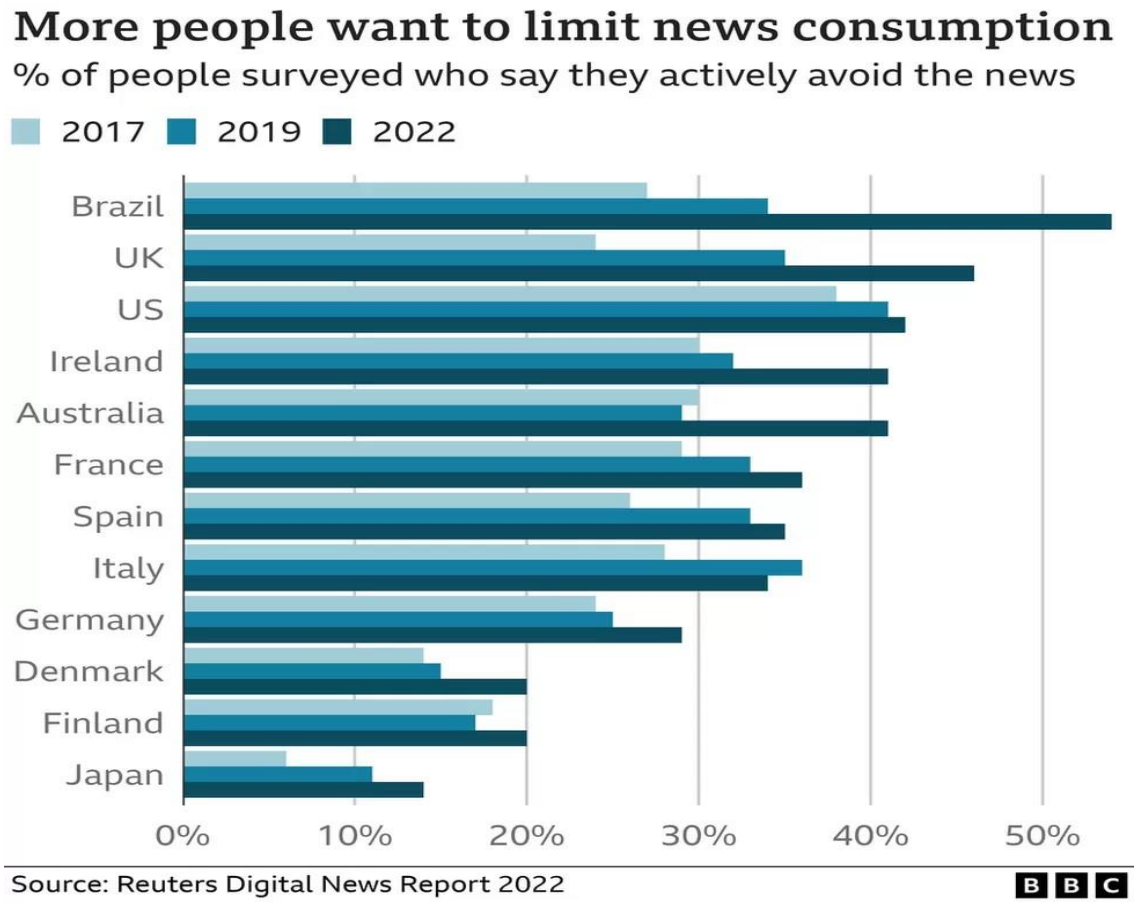
The autocorrelation measures the correlation between a time series and its own past values. The autocorrelation plot displays how this correlation strength varies with different time lags. Sharp peaks in both the positive and negative directions on an autocorrelation plot indicate strong and alternating patterns of positive and negative correlation between the yearly

agricultural growth rates and their lagged values. Alternating peaks in a time series chart may indicate a seasonal pattern where some years tend to have a positive correlation with their lagged values, meaning a similar trend or pattern is repeating itself. On the other hand, some years may show a negative correlation, indicating a contrasting trend compared to their lagged values. Complex relationships within the agricultural growth data, possibly influenced by multiple interacting factors, may result in alternating positive and negative correlations, as indicated by sharp peaks in both directions. The peaks decrease over time, indicating weakened correlation between annual growth rates. A high positive peak at a lag of 8 implies a weekly pattern in the data.

b) Blotting a blot from BBC

The plot depicted is the percentage of people polled who actively avoid news. https://www.bbc.co.uk/news/entertainment-arts-61797444 . The dataset for this plot is taken from https://reutersinstitute.politics.ox.ac.uk/digital-news-report/2022/dnr-executive-summary#:~:text=Interest%20in%20news%20has%20fallen,has%20increased%20sharply%20across%20countries.

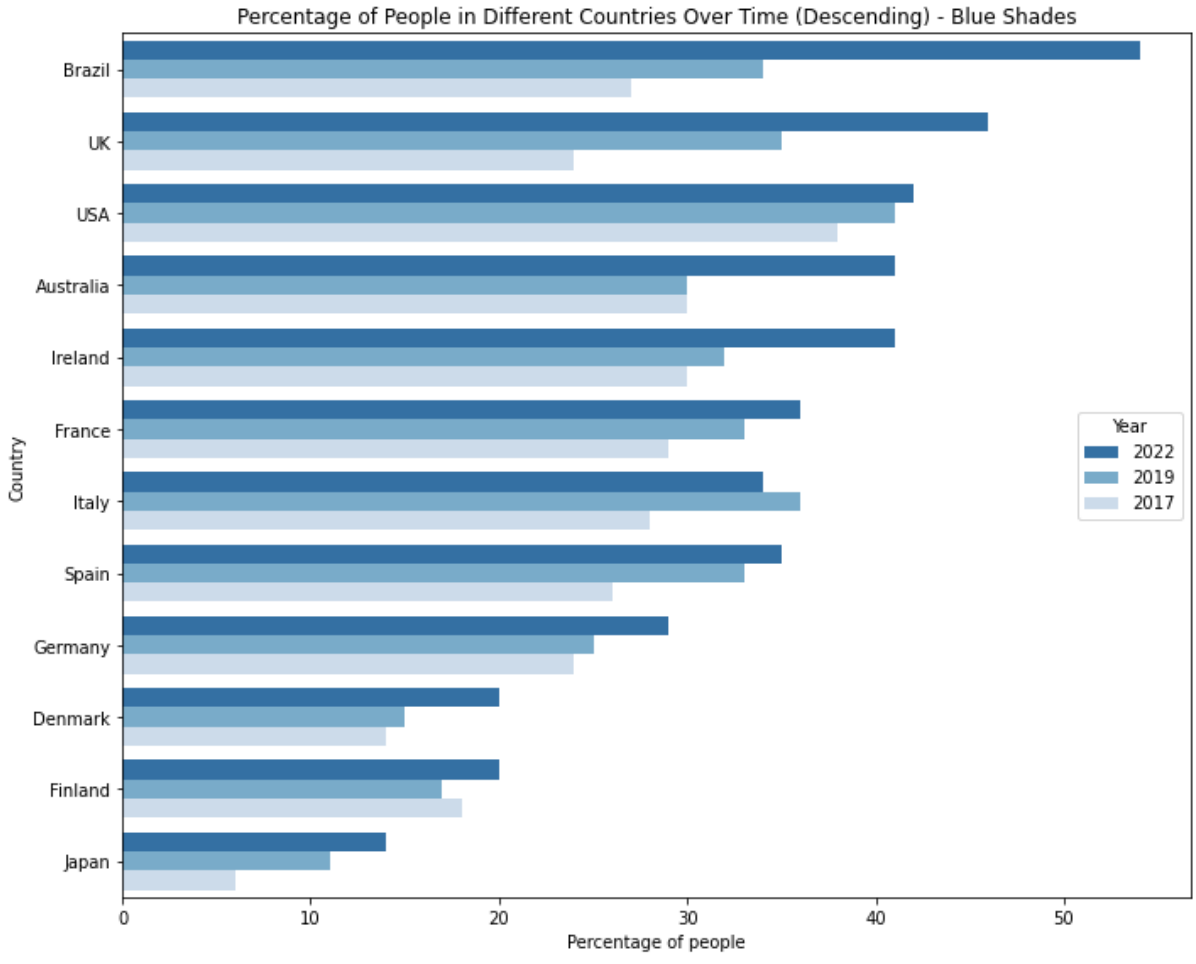
The BBC plot on the percentage of people who actively avoid news:



("Digital News Report: Depressing Stories Turning More People Off," 2022)

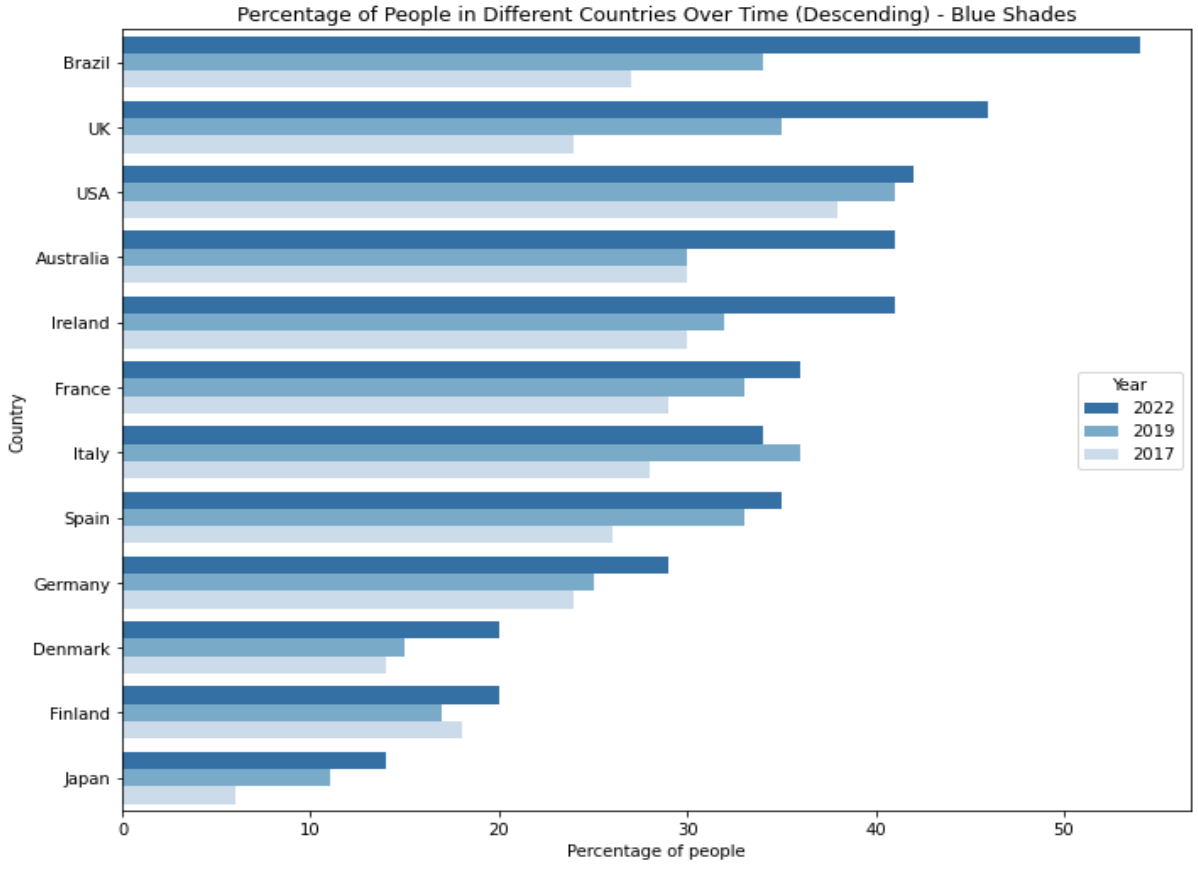
Code to reproduce the plot using seaborn:

```
import seaborn as sns
import pandas as pd
import matplotlib.pyplot as plt

# Loading the data

df = pd.read_csv('percentage_data.csv')

# Melt the DataFrame to long format
df_melted = df.melt(id_vars='Country', var_name='Year', value_name='Percentage of people') #'(How Do I Melt a Pandas Dataframe?, n.d.)'

df_desc = df_melted.sort_values(by='Percentage of people', ascending=False)

# Plotting as a hierarchical plot using seaborn
plt.figure(figsize=(10, 8))
sns.barplot(x='Percentage of people', y='Country', hue='Year', data=df_desc,
palette='Blues_r')
plt.xlabel('Percentage of people')
plt.ylabel('Country')
plt.title('% people survived who avoid watching news')
plt.legend(title='Year', loc='center right')
plt.tight_layout()
plt.show()
```



Plotting using plotnine:

```
import seaborn as sns
import pandas as pd
import matplotlib.pyplot as plt
from plotnine import *

# Loading the data
df = pd.read_csv('percentage_data.csv')

df = pd.DataFrame(data)

# Melt the DataFrame to long format
df_melted = df.melt(id_vars='Country', var_name='Year',
value_name='Percentage of people') #'(How Do I Melt a Pandas Dataframe?,
n.d.)'

df_desc = df_melted.sort_values(by='Percentage of people', ascending=False)
(ggplot(df_desc, aes(x='Percentage of people', y='Country', fill='Year')) +
 geom_bar(stat='identity', position='dodge') +
 labs(x='Percentage of people', y='Country', title='% people survived who
avoid watching news') +
 scale_fill_brewer(type='seq', palette='Blues') +
 theme_minimal() +
 theme(figure_size=(10, 8), legend_position='right') +
 coord_flip()
)
```



Decision making :

Investigating news consumption habits in various countries reveals fascinating differences. A comprehensive survey spanning the years 2017, 2019, and 2022 revealed distinct patterns in news avoidance behaviors across countries.

Surprisingly, the survey results in 2022 show that Japan has the lowest percentage of people who intentionally avoid news consumption. Following Japan, Finland is the second-lowest, followed by Denmark.This tendency could be influenced by a number of reasons. These could include public trust in news sources, cultural preferences for remaining informed, and overall opinions of the importance of news in daily life.Surprisingly, the study results show Brazil as the front-runner in 2022, with the highest proportion of people purposefully ignoring news. The United Kingdom comes in second, followed by the United States.

TASK 4: Data Protection and Data ethics (MLO3)

a)

Zaun: A cyberattack on Zaun, a contractor for the UK Ministry of Defence, exposed sensitive military documents, including plans for chemical weapons and nuclear submarines. (Plant, 2023)

LockBit, an organization with ties to Russia, recently carried out a cyber attack against the security firm Zaun. As a result of the hack, top-secret material belonging to the Ministry of Defence has been released into the dark web. The stolen material contains thousands of pages of highly classified information that hackers may exploit to compromise UK military and security systems.

According to recent reports, Zaun believed that their cybersecurity measures were strong enough to prevent any data transmission during the time of the attack. However, it has been discovered that LockBit was able to download some data during the attack. Although it was likely limited to a specific PC, there is a possibility that some data on the server may have been accessed. It is estimated that approximately 10 GB of data may have been taken, which could include historical communications, orders, drawings, and project files, among other things.

LockBit has claimed responsibility for the attack and has demanded an undisclosed amount of ransom from Zaun. The group has given Zaun until August 29 to pay the ransom. If Zaun fails to do so, LockBit has threatened to release additional material on its leak site.

Possible impacts upon real people:

It is critical to acknowledge that cybersecurity attacks and breaches, particularly those involving government agencies or defense contractors, pose significant threats to national security and public safety. Organizations targeted by cyberattacks are uncommon, particularly those involved in government contracts or defense-related activity.

These attacks have the potential to expose sensitive information, steal data, disrupt operations, and cause major national security issues.Attacks on defense ministry networks are particularly dangerous because they have the potential to harm physical infrastructure or military equipment linked to these systems. An attack on weapon systems or critical infrastructure such as transportation systems or power grids, for example, could cause significant disruption and destruction.

A breach within defense ministries could lead to the exposure of the personal data of military personnel, government employees, and contractors. This data includes sensitive information such as social security numbers, addresses, contact information, and even medical records, which could put people at risk of identity theft, financial fraud, or personal safety concerns. Leaking or misusing personal information of defense ministry staff can jeopardize their employment, relationships, and overall well-being in the long run.

Cyber attacks can have a profound impact on individuals, especially those closely linked to defense ministries. Such attacks can lead to increased tension, anxiety, and feelings of dread. The emotional distress of the affected individuals may be further compounded by concerns about the safety of their personal information, the security of their jobs, and the potential impact on national security.

b) Data ownership:

For data ownership, privacy, and ethical data management in cybersecurity, it is critical to follow the criteria and principles specified in the General Data Protection Regulation (GDPR) and the UK Government Data Ethics Framework. In order to address data ownership in cybersecurity, individuals and organizations must first understand their rights, obligations, and ethical responsibilities when it comes to acquiring, keeping, and using data. This will assure data and privacy protection, as well as promote ethical cybersecurity practice.

"'controller' means the natural or legal person, public authority, agency or other body which, alone or jointly with others, determines the purposes and means of the processing of personal data; where the purposes and means of such processing are determined by Union or Member State law, the controller or the specific criteria for its nomination may be provided for by Union or Member State law; '(Intersoft consulting , 2013)'

In the GDPR, a "controller" is defined as a person or organization that determines the objectives and methods of processing personal data, either alone or with others. The controller has the authority to decide why personal data is collected, how it is stored, how long it is kept, who can access it, and how it will be used. This includes making decisions about what information is gathered and the methods used to process it. Although the GDPR does not directly grant individuals data ownership, it does stress their rights and control over their personal data. Individuals will be empowered by the regulations, which include data-related rights such as access, rectification, erasure, and portability. This promotes a more transparent and equal environment for data processing.

Thus ‘Zaun works closely with all stakeholders in the business, including employees, local, national, and international suppliers, and a long-established customer base of fencing contractors, to design, manufacture, and supply high-quality fencing systems’(Plant, 2023). By exploiting controlled access to personal data with user authorization, corporations can provide personalized services, targeted marketing, and tailored product suggestions.This leads to improved consumer experiences and, maybe, increased engagement and revenue.

“If you are a processor, the UK GDPR places specific legal obligations on you; for example, you are required to maintain records of personal data and processing activities. You will have legal liability if you are responsible for a breach.” (*Who Does the UK GDPR Apply To?,* 2023)

As per the UK Government Data Ethics Framework, Processors may face legal consequences if they fail to satisfy their obligations under the GDPR in the UK or are liable for a data breach. This liability indicates that if they mishandle or reveal personal data on behalf of the controller, they may face legal penalties, fines, or compensation claims.

References

4 Ways of Calculating Autocorrelation Function in Python. (2022, March 21). Scicoding. https://scicoding.com/4-ways-of-calculating-autocorrelation-in-python/

Digital News Report: Depressing stories turning more people off. (2022, June 15). *BBC News*. https://www.bbc.co.uk/news/entertainment-arts-61797444

Data Ethics Framework. (n.d.). GOV.UK. https://www.gov.uk/government/publications/data-ethics-framework

How can I use numpy.correlate to do autocorrelation? (n.d.). Stack Overflow. Retrieved November 23, 2023, from https://stackoverflow.com/questions/643699/how-can-i-use-numpy-correlate-to-do-autocorrelation

How do I melt a pandas dataframe? (n.d.). Stack Overflow. Retrieved November 24, 2023, from https://stackoverflow.com/questions/68961796/how-do-i-melt-a-pandas-dataframe

Intersoft consulting . (2013). *Art. 4 GDPR – Definitions | General Data Protection Regulation (GDPR)*. General Data Protection Regulation (GDPR). https://gdpr-info.eu/art-4-gdpr/

Overview and key findings of the 2022 Digital News Report | Reuters Institute for the Study of Journalism. (n.d.). Reutersinstitute.politics.ox.ac.uk. Retrieved November 24, 2023, from https://reutersinstitute.politics.ox.ac.uk/digital-news-report/2022/dnr-executive-summary#:~:text=Interest%20in%20news%20has%20fallen

Plant, S. (2023, September 1). *Zaun Data Breach – Update | News | Zaun Ltd*. Zaun Limited. https://www.zaun.co.uk/zaun-data-breach-update/

Python | Add only numeric values present in a list. (2019, March 19). GeeksforGeeks. https://www.geeksforgeeks.org/python-add-only-numeric-values-present-in-a-list/

Python Pandas: How to remove nan and inf values | Saturn Cloud Blog. (2023, June 19). Saturncloud.io. https://saturncloud.io/blog/python-pandas-how-to-remove-nan-and-inf-values/#:~:text=Using%20dropna()

Time series analysis: a gentle introduction. (2023). Quix.io. https://quix.io/blog/time-series-analysis#time-series-analysis-models-techniques-and-methods

using pandas.DataFrame.melt() with seaborn.barplot. (n.d.). Stack Overflow. Retrieved November 24, 2023, from https://stackoverflow.com/questions/60625159/using-pandas-dataframe-melt-with-seaborn-barplot

What is autocorrelation in Python? (n.d.). Educative: Interactive Courses for Software Developers. https://www.educative.io/answers/what-is-autocorrelation-in-python

Who does the UK GDPR apply to? (2023, May 19). Ico.org.uk. https://ico.org.uk/for-organisations/uk-gdpr-guidance-and-resources/personal-information-what-is-it/who-does-the-uk-gdpr-apply-to/

Plotting in sorted order using Plotnine. (n.d.). Stack Overflow. Retrieved November 24, 2023, from https://stackoverflow.com/questions/62507299/plotting-in-sorted-order-using-plotnine

For improving and rectifying errors in the sentences , I have used Grammarly AI.