Question 1 [18 marks]

You are joining as a junior data analyst in a local gaming company. You are assigned to develop python codes for gaming related analysis to better understand the gaming markets and consumers' behavior and develop some minor game applications.

1a) You are required to write a simple Python code using 'numpy' library to calculate the company revenue progress based on the last 5 quarter revenues.

	Quarter 1	Quarter 2	Quarter 3	Quarter 4	Quarter 5
Revenue (in millions)	10.5	12.2	15.8	14.6	18.2

You can start with a 1-D array for example:

```
import numpy as np

revenue_data = np.array([10.5, 12.2, 15.8, 14.6, 18.2])
```

i. Your code needs to calculate the quarter with the highest revenue and print out that quarter number (*Number only* is enough).
ii. Your code needs to calculate the percentage change in revenue from the previous quarter and print out the specific quarter number (*Number only* is enough) with the highest percentage change in revenue from the previous quarter.

The formula of the percent change in revenue **:**
(Current Quarter Revenue - Prior Quarter Revenue) / Prior Quarter Revenue

You may use a numpy function called '***numpy.diff (input_array)***'. This function will generate a output array. The first difference is given by ***out_array[i] = input_array [i+1] - input_array [i]*** along the given axis, higher differences are calculated by using *diff* recursively.

For example:

```
import numpy as np

x = np.array([1, 2, 4, 7, 0])
print(np.diff(x))


#Output = [ 1,  2,  3, -7]
```

[3 marks]

1(b) You are assigned to understand the game sales in the past years globally. You need to study from a dataset in the format of csv. The dataset contains a list of video games with sales greater than 100,000 copies. It was generated by a scrape of vgchartz.com.

Fields include

Rank - Ranking of overall sales
Name - The games name
Platform - Platform of the games release (i.e. PC,PS4, etc.)
Year - Year of the game's release
Genre - Genre of the game
NA_Sales - Sales in North America (in millions)
EU_Sales - Sales in Europe (in millions)
JP_Sales - Sales in Japan (in millions)
Other_Sales - Sales in the rest of the world (in millions)
Global_Sales - Total worldwide sales.

In order to understand the data in csv file, run the following code and the outcomes are as below:

```
#Read the csv file
vgsData = pd.read_csv('./data/vgsales.csv')

#Print the numbers of rows and columns
print("Dataset Dimensions: {:,} columns and {:,}
rows".format(vgsData.shape[1], vgsData.shape[0]))

#Print the initial part of dataframe
vgsData.head()
```

Output:

Dataset Dimensions: 10 columns and 16,598 rows

	Rank	**Name**	**Platform**	**Year**	**Genre**	**NA_Sales**	**EU_Sales**	**JP_Sales**	**Other_Sales**	**Global_Sales**
0	1	Wii Sports	Wii	2006	Sports	41.49	29.02	3.77	8.46	82.74
1	2	Super Mario Bros.	NES	1985	Platform	29.08	3.58	6.81	0.77	40.24
2	3	Mario Kart Wii	Wii	2008	Racing	15.85	12.88	3.79	3.31	35.82
3	4	Wii Sports Resort	Wii	2009	Sports	15.75	11.01	3.28	2.96	33.00

4	5	Pokemon Red/Blue	GB	1996	Role-Playing	11.27	8.89	10.22	1.00	31.37

After running the below code to describe the data involved:

```
print("Describe Data:")

vgsData.describe() #Describe the data
```

Output:

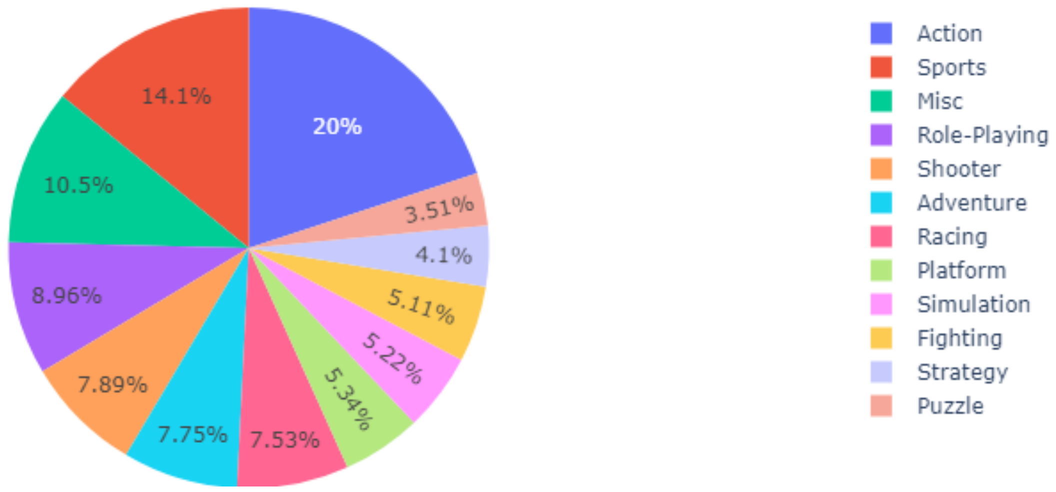
	Rank	Year	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales
count	16598.00	16327.00	16598.00	16598.00	16598.00	16598.00	16598.00
mean	8300.600	2006.400	0.264	0.146	0.0778	0.0481	0.537
std	4791.853	5.828	0.816	0.505	0.309	0.188	1.555
min	1.000	1980.000	0.000	0.000	0.000	0.000	0.0100
25%	4151.250	2003.000	0.000	0.000	0.000	0.000	0.060
50%	8300.500	2007.000	0.080	0.020	0.000	0.010	0.170
75%	12449.750	2010.000	0.240	0.110	0.040	0.040	0.470
max	16600.000	2020.000	41.490	29.020	10.220	10.570	82.740

Write Python code to find the missing (NAN) data in any column and **clean the dataframe based on your observation from the above data description.** After cleaning data, you are required to find out the top sales in North America and Japan based on the game 'Genre'. **Your code needs to print out the top game genre types in North America and Japan regions only and their sales in both regions respectively**.

[5 marks]

1c) The main interest from this dataset is the game sales and the company would like to understand how the global sale will be impacted based on the other features. Therefore, the **target** variable is the **global sale** to be predicted after training with the given dataset. Another department provided the information of correlation analysis that the **important features** will be **all different regional sales and game genre** and they must be included for training the prediction model. The rest of the features are insignificant in predicting global sales.

There are **12** genres included in the dataset such as **Action, Sports, Misc, Role-Playing, Shooter, Adventure, Racing, Platform, Simulation, Fighting, Strategy and Puzzle**.



The ‘Genre’ data are defined in string format. Therefore, essential encoding shall be performed on the data so that the model will handle numerical values. You may apply a dictionary mapping method to transform **(Action = 0, Sports = 1, Misc = 2, .., Puzzle =11)** on the genre data or you may utilize the LabelEncoder class provided by the Sklearn library or ‘get_dummies’ from Pandas . Refer below for the documentation of the LabelEncoder class if you wish to utilize built-in classes.

	Sklearn.preprocessing.**LabelEncoder**
Methods	

Fit_transform
(Purpose - Fit and transform the label encoder)
	Returns an instance of itself.
	Example-
	`le = preprocessing.LabelEncoder()`
	`#Transform to encoded lablels`
	`df.column_to_encoded =`
	`le.fit_transform(df["column_to_encoded"])`
	
	(Output)>>> Encoded labels.

	Pandas.**get_dummies**

get_dummies
(Purpose- Tranform from string to numerical format)
	Returns dummy-coded data
	Example-
	`dummies = pd.get_dummies(df.town) #town to be dummy-coded`
	`merged =`
	`pd.concat([df,dummies],axis='columns')`
	`#merged with original dataframe`
	`final = merged.drop(['town'], axis='columns')`
	`#eliminate the original column`
	`final = final.drop(['Firsttown'],`
	`axis='columns') #eliminate the first column`

Write python code to **generate the prepared data set** with **important features with necessary encoding** used for training and testing the prediction model(s) based on the above-mentioned requirements.

[4 marks]

1d) As the next step, you need to continue for training and testing a **multiple linear regression model** with the **important features** mentioned in the previous section to predict the total global sale. The trained model will be shared among your team members and sale departement to predict the global sale. Write Python code for **training, testing the multiple linear regression model and describing R^2 accuracy of the model** using the test data.

$$\text{SSTotal} = \Sigma(y_i - \bar{y})^2$$
$$\text{SSError} = \Sigma(y_i - \hat{y})^2$$

$$R^2 = \frac{\text{SSTotal} - \text{SSError}}{\text{SSTotal}}$$
$$= 1 - \frac{\text{SSError}}{\text{SSTotal}}$$

[6 marks]

Question 2 [17 marks]

2a) League of Legends (abbreviated LoL) is a multiplayer online battle arena video game. In League of Legends, players assume the role of an unseen "summoner" that controls a "champion" with unique abilities and battles against a team of other players or computer-controlled champions. The goal is usually to destroy the opposing team's "nexus", a structure that lies at the heart of a base protected by defensive structures.

Each League of Legends match is discrete, with all champions starting off relatively weak but increasing in strength by accumulating items and experience over the course of the game. A good indicator of how well a team is doing in a match is how many objectives the team has taken. These objectives are:

1. **Tower kills**: Towers are placed in each lane of the map and must be taken in order for your team to advance.
2. **Inhibitor kills**: Inhibitors are placed in each team's base, and must be taken in order for your team to take the inner towers and eventually win the game.

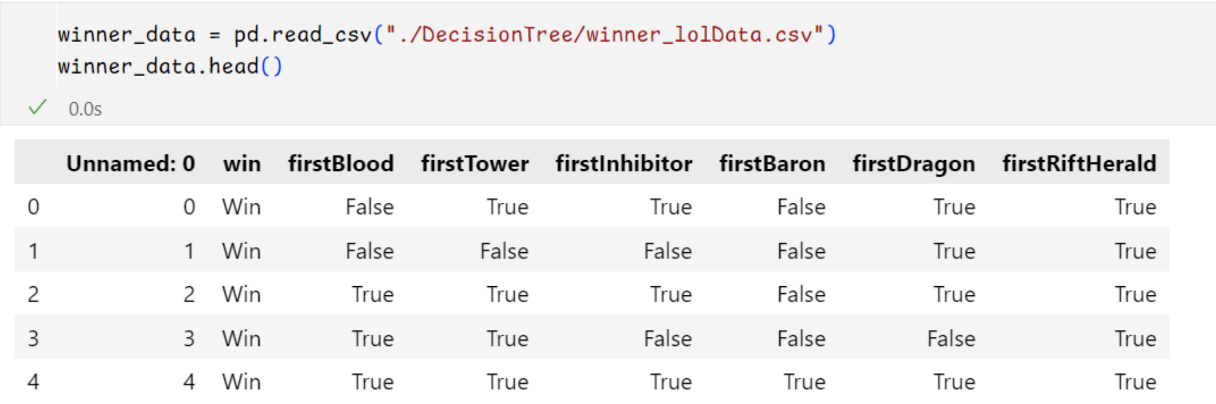
3. **Baron kills**: The Baron is the most powerful monster on the map, giving your team a competitve edge over the other team.
4. **Dragon kills**: The Dragon is a powerful neutral monster that gives you 'buffs' and gold, which help establish a competitive edge over the other team.
5. **Rift Herald kills**: Killing the Rift Herald allows it to be summoned again as a battering ram to attack enemy towers.

There are two data files of winning and losing teams related to the first kill of a game (first blood) and **kills** of above-mentioned **structures.**

You are assigned to analyze League of Legends ranked matches, and a **decision tree classification** algorithm was developed related to the first kills in the games. The datasets provided are:

- All winning team information is in the ‘**winner_lolData.csv**’ file
- All losing team information is in ‘**loser_lolData.csv**’ file.

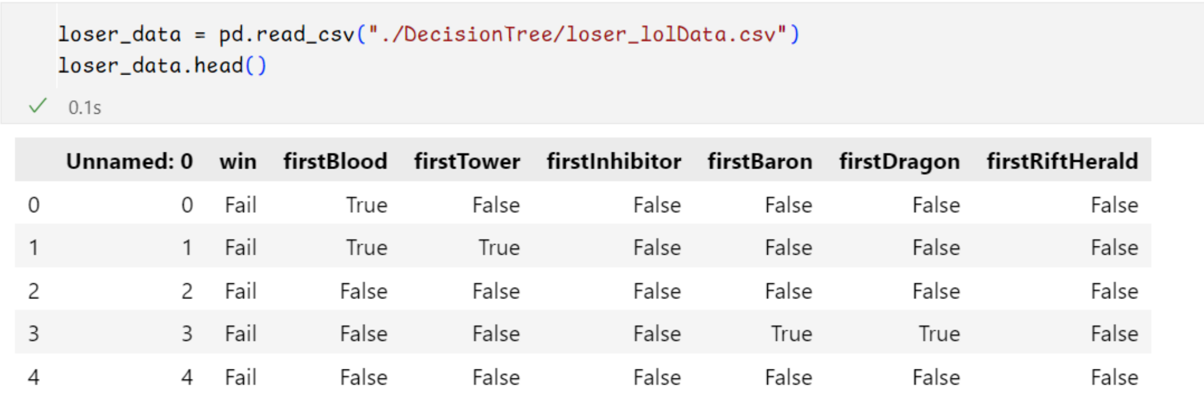
Below code snippets are executed and the outcomes are shown as follows:



```
winner_data = pd.read_csv("./DecisionTree/winner_lolData.csv")
winner_data.head()
```

✓ 0.0s

	Unnamed: 0	win	firstBlood	firstTower	firstInhibitor	firstBaron	firstDragon	firstRiftHerald
0	0	Win	False	True	True	False	True	True
1	1	Win	False	False	False	False	True	True
2	2	Win	True	True	True	False	True	True
3	3	Win	True	True	False	False	False	True
4	4	Win	True	True	True	True	True	True



```
loser_data = pd.read_csv("./DecisionTree/loser_lolData.csv")
loser_data.head()
```

✓ 0.1s

	Unnamed: 0	win	firstBlood	firstTower	firstInhibitor	firstBaron	firstDragon	firstRiftHerald
0	0	Fail	True	False	False	False	False	False
1	1	Fail	True	True	False	False	False	False
2	2	Fail	False	False	False	False	False	False
3	3	Fail	False	False	False	True	True	False
4	4	Fail	False	False	False	False	False	False

You are required to develop **a model that classifies the match result** (Win or Lose) using all the data provided through a decision tree classifier based on **the first kills** in a game. Thus, **"win" column is the target and other columns are the independent variables**. Your Python code should include **allocating training and testing of the dataset and mapping string** data into Boolean. Finally, you need to print out the **model accuracy** based on the test data.

[8 marks]

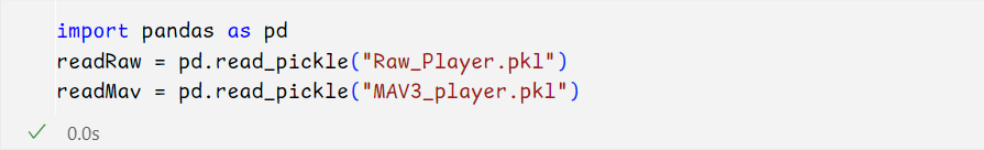
2b) Your company is observing the trend of increasing female game players for a recently released popular game. There is a record list of female player number playing the targeted game on each day in 2022 and it has been saved in a pickle file called '**Raw_Player.pkl**'.

Moving Average (or rolling aveage) is used to analyze data points by creating a series of averages of different subsets of the full data set. A moving average is commonly used with time series data to smooth out short-term fluctuations and highlight longer-term trends or cycles. The estimate of the trend-cycle at time '*t*' is obtained by averaging values of the time series within *w* period of *t*. There is also a pickle file containing a rolling average with window size of 3 days for predicting the female players on next day in 2022. That data is saved in the pickle file called '**MAV3_player.pkl'**.

$$\boxed{\hat{Y}_t = \frac{Y_{t-1} + Y_{t-2} + .. + Y_{t-w}}{w}}$$

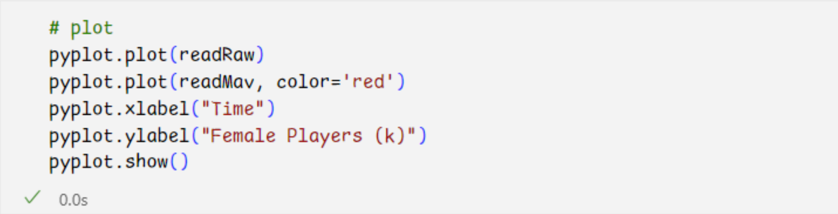
$\hat{Y}_t$ = moving average for period t
Y_{t-1} = actual value for period t−1
w = window or interval

The following code snippets are executed to show the data contained in both pickle file accordingly.



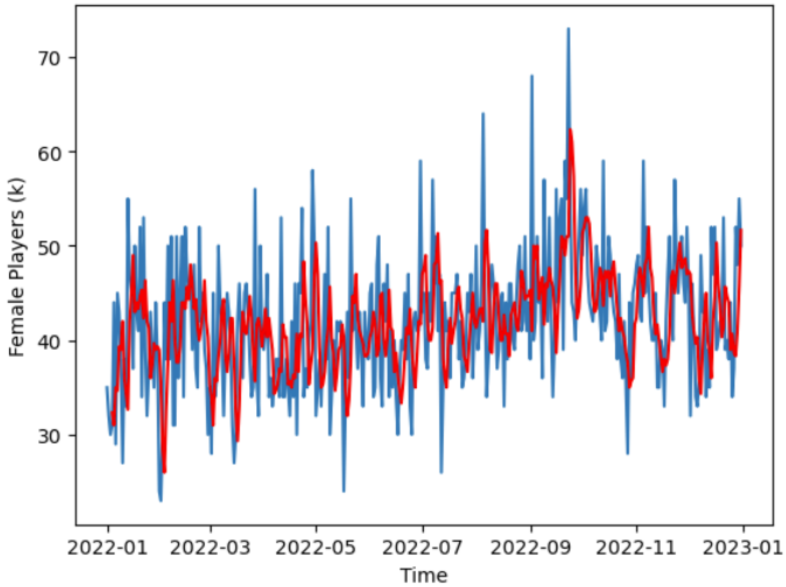
```
import pandas as pd
readRaw = pd.read_pickle("Raw_Player.pkl")
readMav = pd.read_pickle("MAV3_player.pkl")
```

✓ 0.0s



```
# plot
pyplot.plot(readRaw)
pyplot.plot(readMav, color='red')
pyplot.xlabel("Time")
pyplot.ylabel("Female Players (k)")
pyplot.show()
```

✓ 0.0s



Your senior analyst suggested you that the weighted moving average method could predict with a better mean squared error. Setting the time period to 3 days, the following weighted moving average method should be used and coded accordingly.

Time Period = 3

$$\hat{Y}_t = \frac{3}{(1+2+3)} * Y_{t-1} + \frac{2}{(1+2+3)} * Y_{t-2} + \frac{1}{(1+2+3)} * Y_{t-3}$$

Write Python code of **the weighted moving average (time period =3) based on the data obtained from the relevant pickle file** to predict the next day in 2022 and compare the actual data with the predictions by **printing out the mean squared error** of this method.

(9 marks)