National College of Ireland

Configuration Manual

MSc Research Project
Data Analytics

Neeraj Choudhary
x17156611

School of Computing
National College of Ireland

Supervisor: Dr. Cristina Muntean

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Neeraj Choudhary
Student ID:	x17156611
Programme:	Data Analytics **Year:** 2020
Module:	Research Project
Lecturer:	Dr. Cristina Muntean
Submission Due Date:	17/08/2020
Project Title:	Technology-enhanced learning impact analysis through categorisation and pattern recognition by using clustering machine learning algorithm
Word Count:	1146 **Page Count:** 10

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Neeraj Choudhary

Date: 17/08/2020

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Technology-enhanced Learning Impact Analysis through Categorization and Pattern Recognition by using Clustering Machine Learning Algorithm

Configuration Manual

Neeraj Choudhary
x17156611
MSc Research Project in Data Analytics
17st August 2020

1 Introduction

Problem solving skills and critical think skills were important skills for the students and in the recent year lots of discussion has been made on traditional methods of teaching skills such as computer programming skills is not up to mark which started a discussion on modernizing the curriculum to make more advance development of the student by STEM related learning. In this research we have observed the recent work related to the improving the programming and critical skills of the students in educational institution by using game based learning (Huizenga, Admiraal, Akkerman and Dam, 2009) by introducing a game which make learning programming complex concept easier to learn and understand while playing the game and solving the problem associated to the task or puzzle. This research help us to understand how institutions are evaluating the impact of the game based learning and how we can improve the limitation of the methodology or approach with the introduction on the machine learning approach to evaluate the impact of the game based learning.

2 System Summary

This section shows the summary related to the system configuration used in this research as mentioned in the Figure 1.

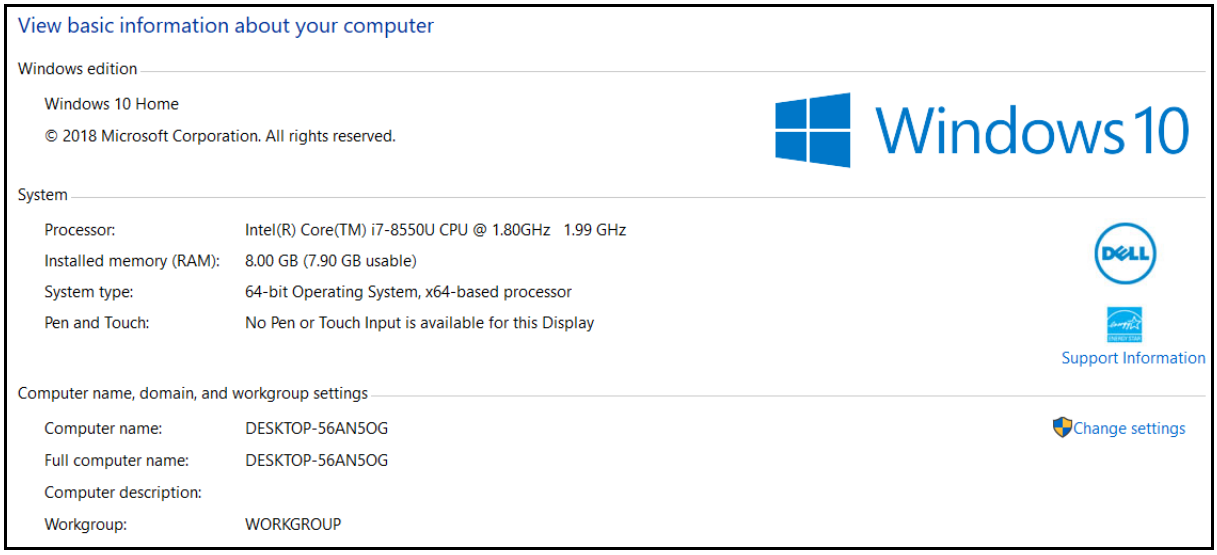

Figure 1: System Summary

3 Implementation

This section talks about the setup of the tool used in the research and methodology.

3.1 Set up of the Tool

In this research we have used Python Spyder (3.7) and installed the libraries related to the data import, data manipulation and analysis.

- Installation: In this research we have used Python programming to apply clustering model on the Newton game-based dataset. We have used the python platform called Spyder for the python programming. Spyder can be downloaded from https://docs.spyder-ide.org/installation.html#installing-a-development-build. While downloading we have to select the default setting. For data processing and analysis we need to install python packages with the command line such as “from sklearn.cluster import KMeans” or “Install <package name>”.
- Packages Used: The python libraries which we have use in this research related to exploratory and descriptive analysis, libraries such as pandas , numpy are used for the data import and data manipulation and matplotlib is use for visluzation and Kmean , PCA , AgglomerativeClustering , dendrogram,linkage is use for the clustering.

3.2 Research Methodology

The research methodology is shows in the figure 2.

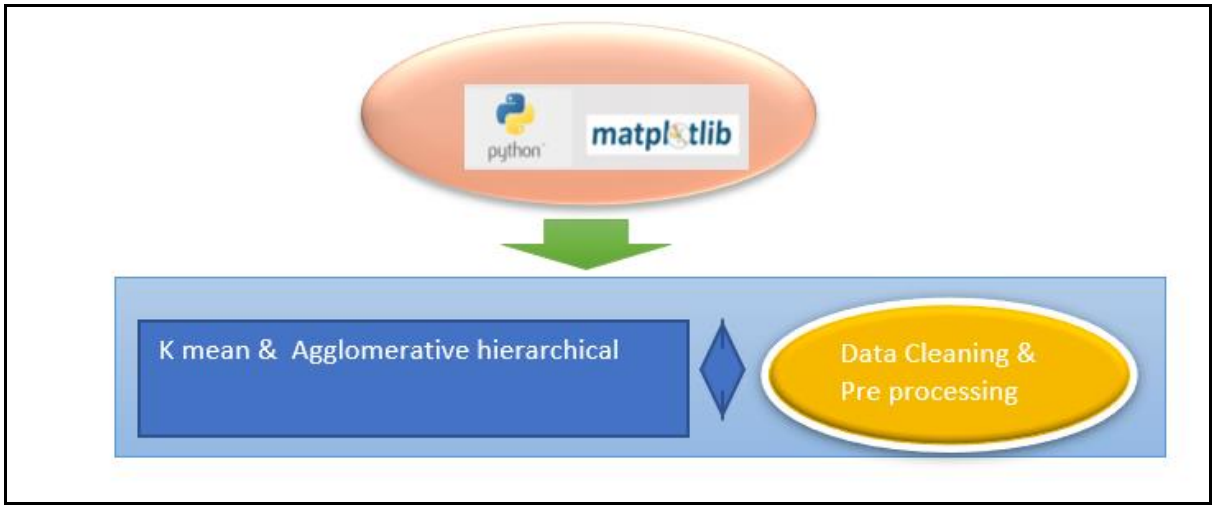

Figure 2: Research Methodology

4 Experiment Pre-Setup

This section shows the experiment pre setup where section 4.1 shows methodology of the data extraction, section 4.2 shows descriptive and exploratory analysis, section 4.3 shows data preparation.

4.1 Data Extraction

The Figure 3 represent the data extraction to python platform Spyder. The dataset is being imported in the form of csv data.

```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import matplotlib.pyplot as plt # Data visualization
import seaborn as sns #data visualization
from sklearn.cluster import KMeans # Clustering
from sklearn.decomposition import PCA


# Data import
data = pd.read_csv('Data1T.csv') #variable game data import process

data.head() # data structure :how data look like after we improted it
```

Figure 3: Data Extraction of Newton game data : Variable

4.2 Exploratory analysis and Data preparation

This section shows the descriptive and exploratory analysis has been done on the game-based dataset in the Figure 4. This help us to understand the structure of the dataset to understand the feature of the dataset and check if dataset contain any null value or not. This process also helps us to understand the distribution of the survey data set which provides us the

information on gender, student like game-based learning or not and how much are satisfied with game-based learning.

```
#Data Processing
data.shape  # Shape of the data : columns and rows

data.describe() # provide the information like mean , median , standard devaition , count of the rows

data.isnull().sum() # Provide the information on the number of the null value per columns

data.columns # provide the information of the columns

data.drop('Student id',axis=1,inplace=True) # drop the id columns

data.columns
```

Figure 4: Undestand the strucuture fo the game based dataset

5 Experiment

This section discussed the implementation of machine learning clustering methodology such as Kmean and Agglomerative hierarchical clustering in this research is been applied to game datasets such as function, loop, and variable.

5.1 Experiment 1 Kmean variable game

This section shows the implementation of the kmean to the variable game in figure 5.

```
#Applying elbow method to calculate suitable number of the cluster
cluster_range = range(1,15)
cluster_errors=[]
for i in cluster_range:
    clusters=KMeans(i)
    clusters.fit(data_scaled)
    labels=clusters.labels_
    centroids=clusters.cluster_centers_,3
    cluster_errors.append(clusters.inertia_)
clusters_df=pd.DataFrame({'num_clusters':cluster_range,'cluster_errors':cluster_errors})
clusters_df

f,ax=plt.subplots(figsize=(15,6))
plt.plot(clusters_df.num_clusters,clusters_df.cluster_errors,marker='o')
plt.show()

# Based on the elbow method :
kmean= KMeans(6)
kmean.fit(data_scaled)
labels=kmean.labels_

clusters=pd.concat([data, pd.DataFrame({'cluster':labels})], axis=1)
clusters.head()

for c in clusters:
    grid= sns.FacetGrid(clusters, col='cluster')
    grid.map(plt.hist, c)

clusters.groupby('cluster').mean()

pca = PCA(n_components=2)
principalComponents = pca.fit_transform(data_scaled)
principalDf = pd.DataFrame(data = principalComponents
             , columns = ['principal component 1', 'principal component 2'])
principalDf.head(2)

finalDf = pd.concat([principalDf, pd.DataFrame({'cluster':labels})], axis = 1)
finalDf.head()

# cluster final output
plt.figure(figsize=(15,10))
ax = sns.scatterplot(x="principal component 1", y="principal component 2", hue="cluster", data=finalDf,palette=['red','blue','green','yellow','orange','black'])
plt.show()
```

Figure 5: Kmean implementation on variable game

5.2 Experiment 2 Kmean loop game

This section shows the implementation of the kmean to the loop game in figure 6.

```
#Applying elbow method to calculate suitable number of the cluster
cluster_range = range(1,15)
cluster_errors=[]
for i in cluster_range:
    clusters=KMeans(i)
    clusters.fit(data_scaled)
    labels=clusters.labels_
    centroids=clusters.cluster_centers_,3
    cluster_errors.append(clusters.inertia_)
clusters_df=pd.DataFrame({'num_clusters':cluster_range,'cluster_errors':cluster_errors})
clusters_df

f,ax=plt.subplots(figsize=(15,6))
plt.plot(clusters_df.num_clusters,clusters_df.cluster_errors,marker='o')
plt.show()

# Based on the elbow method :
kmean= KMeans(7)
kmean.fit(data_scaled)
labels=kmean.labels_

clusters=pd.concat([data, pd.DataFrame({'cluster':labels})], axis=1)
clusters.head()

for c in clusters:
    grid= sns.FacetGrid(clusters, col='cluster')
    grid.map(plt.hist, c)

clusters.groupby('cluster').mean()

pca = PCA(n_components=2)
principalComponents = pca.fit_transform(data_scaled)
principalDf = pd.DataFrame(data = principalComponents
             , columns = ['principal component 1', 'principal component 2'])
principalDf.head(2)

finalDf = pd.concat([principalDf, pd.DataFrame({'cluster':labels})], axis = 1)
finalDf.head()

# cluster final output
plt.figure(figsize=(15,10))
ax = sns.scatterplot(x="principal component 1", y="principal component 2", hue="cluster", data=finalDf,palette=['red','blue','green','yellow','orange','black'])
plt.show()
```

Figure 6: Kmean implementation on loop game

5.3 Experiment 3 Kmean function game

This section shows the implementation of the kmean to the function game in figure 7.

```
#Applying elbow method to calculate suitable number of the cluster
cluster_range = range(1,15)
cluster_errors=[]
for i in cluster_range:
    clusters=KMeans(i)
    clusters.fit(data_scaled)
    labels=clusters.labels_
    centroids=clusters.cluster_centers_,3
    cluster_errors.append(clusters.inertia_)
clusters_df=pd.DataFrame({'num_clusters':cluster_range,'cluster_errors':cluster_errors})
clusters_df

f,ax=plt.subplots(figsize=(15,6))
plt.plot(clusters_df.num_clusters,clusters_df.cluster_errors,marker='o')
plt.show()

# Based on the elbow method :
kmean= KMeans(7)
kmean.fit(data_scaled)
labels=kmean.labels_

clusters=pd.concat([data, pd.DataFrame({'cluster':labels})], axis=1)
clusters.head()

for c in clusters:
    grid= sns.FacetGrid(clusters, col='cluster')
    grid.map(plt.hist, c)

clusters.groupby('cluster').mean()

pca = PCA(n_components=2)
principalComponents = pca.fit_transform(data_scaled)
principalDf = pd.DataFrame(data = principalComponents
             , columns = ['principal component 1', 'principal component 2'])
principalDf.head(2)

finalDf = pd.concat([principalDf, pd.DataFrame({'cluster':labels})], axis = 1)
finalDf.head()
# cluster final output
plt.figure(figsize=(15,10))
ax = sns.scatterplot(x="principal component 1", y="principal component 2", hue="cluster", data=finalDf,palette=['red','blue','green','yellow','orange','black'])
plt.show()
```

Figure 7: Kmean implementation on function game

5.4 Experiment 4 Agglomerative Hierarchical Clustering variable game

This section shows the implementation of the Agglomerative hierarchical clustering to the variable game in figure 8.

```
#Agglomerative hierarchical clustering
from sklearn.cluster import AgglomerativeClustering
from scipy.cluster.hierarchy import dendrogram,linkage

Z=linkage(data_scaled,method="ward")
plt.figure(figsize=(15,10))
dendrogram(Z,leaf_rotation=90,p=5,color_threshold=20,leaf_font_size=10,truncate_mode='level')
plt.axhline(y=125, color='r', linestyle='--')
plt.show()

model=AgglomerativeClustering(n_clusters=4,affinity='euclidean',linkage='ward')
model.fit(data_scaled)

model.labels_

clusters_agg=pd.concat([data, pd.DataFrame({'cluster':model.labels_})], axis=1)
clusters_agg.head()

clusters_agg.groupby('cluster').mean()

finalDf_agg = pd.concat([principalDf, pd.DataFrame({'cluster':model.labels_})], axis = 1)
finalDf_agg.head()

plt.figure(figsize=(15,10))
ax = sns.scatterplot(x="principal component 1", y="principal component 2", hue="cluster", data=finalDf_agg,palette=['red','blue','green','yellow'])
plt.show()
```

Figure 8: Agglomerative hierarchical implementation on variable game

5.5 Experiment 5 Agglomerative Hierarchical Clustering loop game

This section shows the implementation of the Agglomerative hierarchical clustering to the loop game in figure 9.

```
#Agglomerative hierarchical clustering
from sklearn.cluster import AgglomerativeClustering
from scipy.cluster.hierarchy import dendrogram,linkage

Z=linkage(data_scaled,method="ward")
plt.figure(figsize=(15,10))
dendrogram(Z,leaf_rotation=90,p=5,color_threshold=20,leaf_font_size=10,truncate_mode='level')
plt.axhline(y=125, color='r', linestyle='--')
plt.show()

model=AgglomerativeClustering(n_clusters=4,affinity='euclidean',linkage='ward')
model.fit(data_scaled)

model.labels_

clusters_agg=pd.concat([data, pd.DataFrame({'cluster':model.labels_})], axis=1)
clusters_agg.head()

clusters_agg.groupby('cluster').mean()

finalDf_agg = pd.concat([principalDf, pd.DataFrame({'cluster':model.labels_})], axis = 1)
finalDf_agg.head()

plt.figure(figsize=(15,10))
ax = sns.scatterplot(x="principal component 1", y="principal component 2", hue="cluster", data=finalDf_agg,palette=['red','blue','green','yellow'])
plt.show()
```

Figure 9: Agglomerative hierarchical implementation on loop game

5.6 Experiment 5 Agglomerative Hierarchical Clustering function game

This section shows the implementation of the Agglomerative hierarchical clustering to the function game in figure 10.

```
#Agglomerative hierarchical clustering
from sklearn.cluster import AgglomerativeClustering
from scipy.cluster.hierarchy import dendrogram,linkage

Z=linkage(data_scaled,method="ward")
plt.figure(figsize=(15,10))
dendrogram(Z,leaf_rotation=90,p=5,color_threshold=20,leaf_font_size=10,truncate_mode='level')
plt.axhline(y=125, color='r', linestyle='--')
plt.show()

model=AgglomerativeClustering(n_clusters=4,affinity='euclidean',linkage='ward')
model.fit(data_scaled)

model.labels_

clusters_agg=pd.concat([data, pd.DataFrame({'cluster':model.labels_})], axis=1)
clusters_agg.head()

clusters_agg.groupby('cluster').mean()

finalDf_agg = pd.concat([principalDf, pd.DataFrame({'cluster':model.labels_})], axis = 1)
finalDf_agg.head()

plt.figure(figsize=(15,10))
ax = sns.scatterplot(x="principal component 1", y="principal component 2", hue="cluster", data=finalDf_agg,palette=['red','blue','green','yellow'])
plt.show()
```

Figure 10: Agglomerative hierarchical implementation on function game

References

Huizenga, J., Admiraal, W., Akkerman, S. and Dam, G., 2009. Mobile game-based learning in secondary education: engagement, motivation and learning in a mobile city game. *Journal of Computer Assisted Learning*, 25(4), pp.332-344.

Li, H., 2020. Application of K-means clustering algorithm in the analysis of college students' online entertainment consumption. *Journal of Physics: Conference Series*, 1570, p.012018.

Naeem, A., Rehman, M., Anjum, M. and Asif, M., 2019. Development of an Efficient Hierarchical Clustering Analysis using an Agglomerative Clustering Algorithm. *Current Science*, 117(6), p.1045.