King Saud University
College of Computer and Information Sciences
Information Technology Department

IT326: Data Mining

Project final report

Student flexibility in Online Learning

Group#:6

LAB Day-Time: Wed-8

Group#:	6	
Section#:	74557	
Group members	Student name	ID
	Noha salem aljohani	443200429
	Norah nasser aljedai	443200841
	Ftoon Fawaz Binkhatttaf	443200499
	Batool Alkhuraim	443200604

Contents

1. Problem

Our focus lies in the flexibility level of students during online learning. The prevalence of online education has significantly increased in recent years, particularly due to the COVID-19 pandemic. Our project aims to study and analyze students' data to pinpoint factors influencing flexibility levels, we aim to provide insights that can enhance flexibility in online studying for students to become more flexibility in studying online

2. Data Mining Task

In our project we will use two data mining tasks to help us predict the flexibility level, which are classification and clustering.

For classification we will train our model to be able to classify the flexibility level of student, based on location, device, age, internet and financial status. For the clustering our model will create a set of clusters for the students who have similar characteristics, then these clusters will be used to predict new students results

3. Data

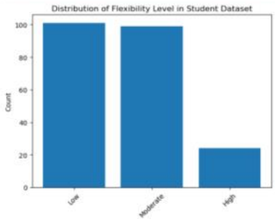
The Source: https://www.kaggle.com/datasets/shariful07/student-flexibility-in-online-learning
Number of objects:1206
Number of attributes:11

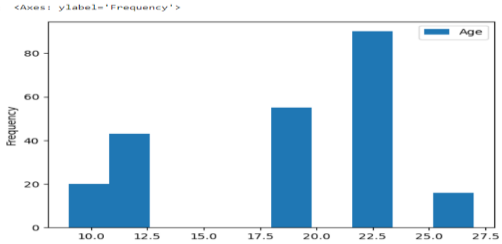
characteristics of attributes

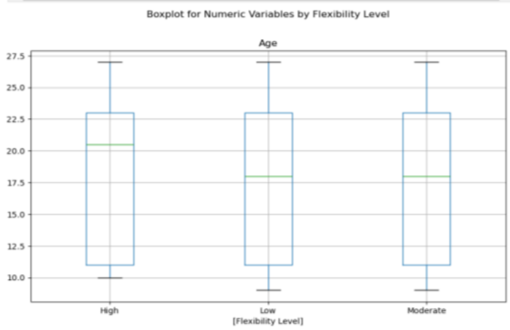
attributes	Data type
Education level	Categorical (Ordinal)
Gender	Binary(symmetric)
Age	Numerical
Device	Categorical (nominal)
Location	String
Financial Condition	Categorical (Ordinal)
Institution Type	Categorical (nominal)
Internet Type	Categorical (nominal)
Network Type	Categorical (nominal)
Flexibility Level	Categorical (Ordinal)
It student	Binary

<table>
<tr>
<td>Missing values</td>
<td>2- Check missing "NA" :
<pre>
missing_values = df.isna().sum()
print("\nTotal number of missing values in the dataset:", missing_values.sum())

#To begin cleaning and handling the data set, we should know the total number of missing values.
</pre>
Total number of missing values in the dataset: 0</td>
<td>To check the missing values in the dataset</td>
</tr>
</table>

<table>
<tr>
<td>Distribution</td>
<td>
<pre>
category_counts = df['Flexibility Level'].value_counts()
plt.bar(category_counts.index, category_counts.values)
plt.title('Distribution of Flexibility Level in Student Dataset')
plt.xlabel('Flexibility Level')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.show()
#to create a bar plot to visualize the distribution of the "Flexibility Level" variable in the student dataset
</pre>


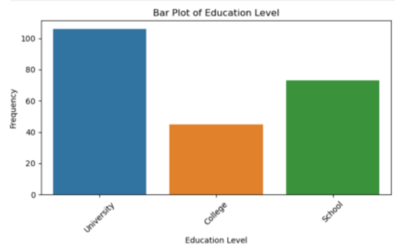
</td>
<td>The bar plot shows the flexibility levels of the students ,the low and moderate flexibility levels are more common among students, with approximately 80 students, while high flexibility is less frequent.</td>
</tr>
<tr>
<td>statistical measures</td>
<td>
<pre>
df.describe()
</pre>
<table>
<tr><th></th><th>Age</th></tr>
<tr><td>count</td><td>224.000000</td></tr>
<tr><td>mean</td><td>18.562500</td></tr>
<tr><td>std</td><td>5.595131</td></tr>
<tr><td>min</td><td>9.000000</td></tr>
<tr><td>25%</td><td>11.000000</td></tr>
<tr><td>50%</td><td>18.000000</td></tr>
<tr><td>75%</td><td>23.000000</td></tr>
<tr><td>max</td><td>27.000000</td></tr>
</table>
</td>
<td>We counted the Statistical measures for the data</td>
</tr>
<tr>
<td>Graphs and tables show variables distribution</td>
<td>
plot histogram:
<pre>
df.plot.hist()
#The code creates a combined histogram plot for all numerical columns
</pre>
<pre>
<Axes: ylabel='Frequency'>
</pre>


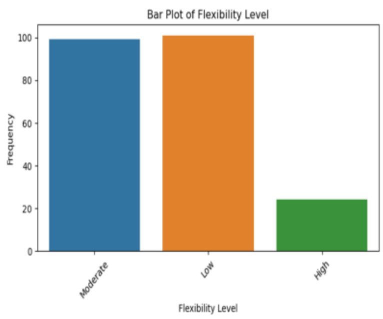
</td>
<td>The plot histogram illustrate the frequency of the age attribute within our dataset</td>
</tr>
<tr>
<td>Boxplot :</td>
<td>
<pre>
df.boxplot(by='Flexibility Level', figsize=(10, 6))
plt.suptitle('Boxplot for Numeric Variables by Flexibility Level', y=1.02)
plt.show()
#to creat a graphical illustration of how the dataset's numerical variables are distributed across various degrees of flexi
#The distribution of a numerical variable within each flexibility level category is shown by each boxplot.
</pre>


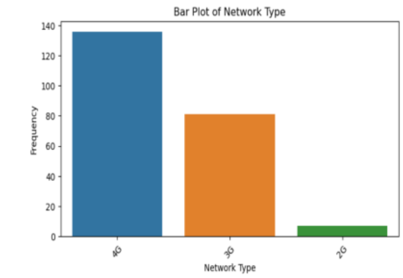
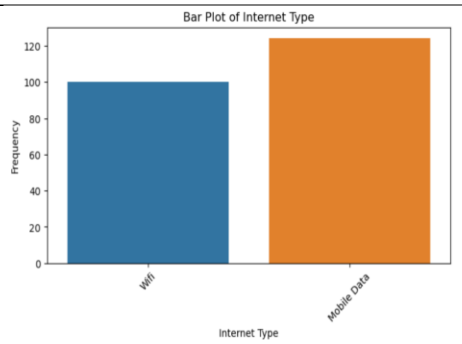
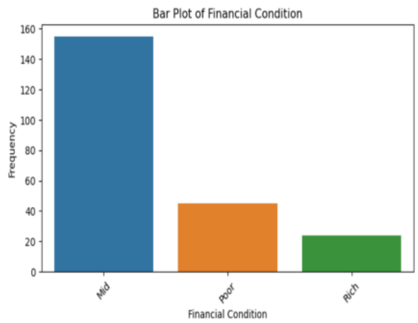
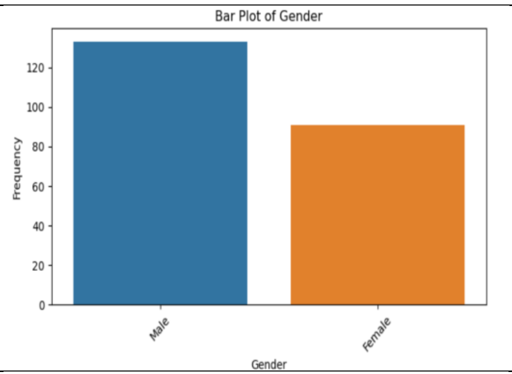
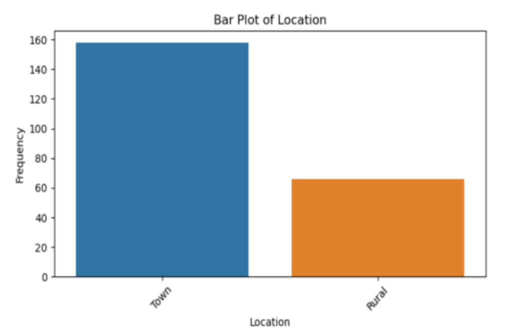
</td>
<td>The boxplot for the

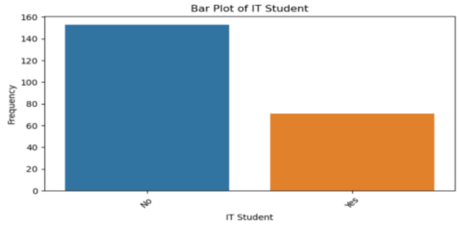
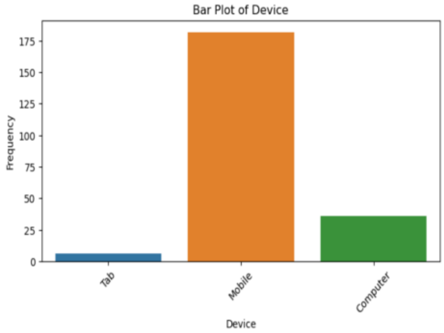
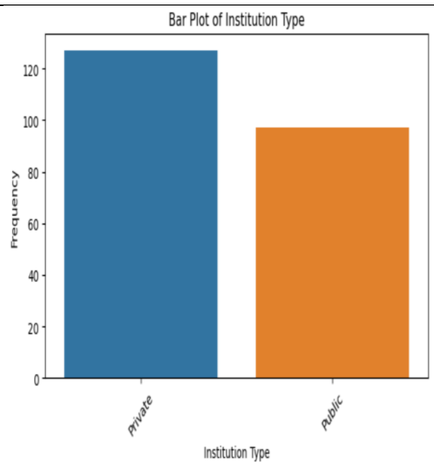
High Flexibility Level shows that there are no outliers in the age attribute. The Interquartile Range (IQR) extends from approximately between 17.5 to 22.5 years, indicating that the median age is around 20 years
Low Flexibility Level there are no outliers for the age attribute. The

</td>
</tr>
</table>

<table>
<tr><td></td><td></td><td>median age is consistently around 20 years, The Interquartile Range (IQR) approximately from 17.5 to 22.5 years, mirroring the values observed in the- High Flexibility Level-
Moderate Flexibility Level, the median age is also around 20 (IQR) is approximately from 17.5 to 22.5 years, The whiskers extend much less in this category, down to about 10 years, indicating a wider range of ages compared in the High and Low Flexibility Levels.</td></tr>
<tr><td>Bar plots :</td><td>Bar plot:
In [79]:
# Bar plots for categorical variables
for column in df.select_dtypes(include=['object']).columns:
    plt.figure(figsize=(8, 4))
    sns.countplot(x=column, data=df)
    plt.title(f"Bar Plot of {column}")
    plt.xlabel(column)
    plt.ylabel("Frequency")
    plt.xticks(rotation=45)
    plt.show()
#generates bar charts for every dataset's category variable.

</td><td>The bar plots shows an illustration for all our categorical variables

The education level the univrities has more frequnce the school then the college</td></tr>
<tr><td></td><td>
</td><td>The bar plot demonstrates the frequency of each flexibility level among students, ordered from most to least frequent as follows: Low, Moderate, and High. Both 'Low' and 'Moderate' flexibility levels have nearly the same highest frequency</td></tr>
</table>

<table>
<tr><td></td><td>
</td><td>The network type (4G) is more frequent than (3G-2G)</td></tr>
<tr><td></td><td>
</td><td>The bar plot indicates that mobile data is the most frequently used internet type</td></tr>
<tr><td></td><td>
</td><td>Bar Plot of Financial Condition" illustrates the frequency of financial conditions(mid-poor-rich)</td></tr>
<tr><td></td><td>
</td><td>bar plot representing gender shows a higher frequency of males than females</td></tr>
<tr><td></td><td>
</td><td>The (Town)shows higher frequency than rural areas</td></tr>
</table>

		the most frequent student were not IT student
		The Bar Plot of Device illustrates that the most used device is mobile - computer then taps
		The Bar Plot of Institution Type illustrates that private institutions have a higher frequency of attendance compared to public institutions

4. Data preprocessing

- Checking for missing values

```
missing_values = df.isna().sum()
print("\nTotal number of missing values in the dataset:", missing_values.sum())

#To begin cleaning and handling the data set, we should know the total number of missing values.
```

```
Total number of missing values in the dataset: 0
```

Description :
Identifying and addressing missing values in datasets is crucial for maintaining the integrity and reliability of data analysis. Missing values can compromise statistical estimates and lead to misleading conclusions. Analyzing missing data patterns helps refine data collection strategies, ensuring more accurate and robust analysis outcomes.

- Removing duplicates

In [65]:
```
num_duplicates = df.duplicated().sum()
df = df.drop_duplicates()
print("Number of duplicate rows:", num_duplicates)
print("DataFrame after dropping all duplicate rows:")
print(df)
```

Description :
Duplicates can lead to inaccuracies in analysis by artificially inflating certain statistics or biasing results. Removing duplicates helps maintain the integrity of your dataset and to give Accurate Model Training beside Duplicate entries can cause inconsistencies and removing the duplicates ensures the efficient of the data to make reliable decisions

- Detect Outliers

In [68]:
```
# Extract the 'Age' column from the DataFrame
age_column = df['Age']
# Calculate the mean age
mean_age = age_column.mean()
# Calculate the absolute differences of each age from the mean
differences_from_mean = abs(age_column - mean_age)

# Find the index of the row with the largest difference from the mean
max_difference_index = differences_from_mean.idxmax()

# Remove the row with the largest difference from the mean
df = df.drop(max_difference_index)
```

Description :
Since there were no outliers, we removed the row with the largest difference from the mean to refine the dataset and enhance the accuracy of our results. This adjustment helps ensure that our data is more representative and reliable for achieving optimal outcomes.

Raw data Our raw dataset before Removing duplicates

Out[63]:

	Education Level	Institution Type	Gender	Age	Device	IT Student	Location	Financial Condition	Internet Type	Network Type	Flexibility Level
0	University	Private	Male	23	Tab	No	Town	Mid	Wifi	4G	Moderate
1	University	Private	Female	23	Mobile	No	Town	Mid	Mobile Data	4G	Moderate
2	College	Public	Female	18	Mobile	No	Town	Mid	Wifi	4G	Moderate
3	School	Private	Female	11	Mobile	No	Town	Mid	Mobile Data	4G	Moderate
4	School	Private	Female	18	Mobile	No	Town	Poor	Mobile Data	3G	Low
...	...	...	...	...	...	...	...	...	...	...	...
1200	College	Private	Female	18	Mobile	No	Town	Mid	Wifi	4G	Low
1201	College	Private	Female	18	Mobile	No	Rural	Mid	Wifi	4G	Moderate
1202	School	Private	Male	11	Mobile	No	Town	Mid	Mobile Data	3G	Moderate
1203	College	Private	Female	18	Mobile	No	Rural	Mid	Wifi	4G	Low
1204	School	Private	Female	11	Mobile	No	Town	Poor	Mobile Data	3G	Moderate

1205 rows × 11 columns

data Our dataset after Removing duplicates

In [65]:
```python
num_duplicates = df.duplicated().sum()
df = df.drop_duplicates()
print("Number of duplicate rows:", num_duplicates)
print("DataFrame after dropping all duplicate rows:")
print(df)
```

```
Number of duplicate rows: 980
DataFrame after dropping all duplicate rows:
    Education Level Institution Type  Gender  Age    Device IT Student  \
0        University          Private    Male   23       Tab         No
1        University          Private  Female   23    Mobile         No
2           College           Public  Female   18    Mobile         No
3            School          Private  Female   11    Mobile         No
4            School          Private  Female   18    Mobile         No
...             ...              ...     ...  ...       ...        ...
1077        College           Public    Male   18    Mobile         No
1124     University          Private    Male   23  Computer        Yes
1132        College           Public    Male   18    Mobile         No
1160     University          Private    Male   23    Mobile        Yes
1197     University          Private    Male   23  Computer        Yes

    Location Financial Condition Internet Type Network Type Flexibility Level
0       Town                 Mid          Wifi           4G          Moderate
1       Town                 Mid   Mobile Data           4G          Moderate
2       Town                 Mid          Wifi           4G          Moderate
3       Town                 Mid   Mobile Data           4G          Moderate
4       Town                Poor   Mobile Data           3G               Low
...      ...                 ...           ...          ...               ...
1077    Town                 Mid   Mobile Data           4G          Moderate
1124   Rural                 Mid   Mobile Data           3G               Low
1132    Town                 Mid   Mobile Data           3G          Moderate
1160   Rural                 Mid   Mobile Data           3G          Moderate
1197    Town                 Mid   Mobile Data           4G          Moderate

[225 rows x 11 columns]
```

Data transformation:
Data encoding

In [80]:
```python
encoder = LabelEncoder()
df['Education Level'] = encoder.fit_transform(df['Education Level'])
df['Institution Type'] = encoder.fit_transform(df['Institution Type'])
df['Gender'] = encoder.fit_transform(df['Gender'])
df['Location'] = encoder.fit_transform(df['Location'])
df['Financial Condition'] = encoder.fit_transform(df['Financial Condition'])
df['Internet Type'] = encoder.fit_transform(df['Internet Type'])
df['Network Type'] = encoder.fit_transform(df['Network Type'])
df['Device'] = encoder.fit_transform(df['Device'])
df['IT Student'] = encoder.fit_transform(df['IT Student'])
df
```

Out[80]:

	Education Level	Institution Type	Gender	Age	Device	IT Student	Location	Financial Condition	Internet Type	Network Type	Flexibility Level
0	2	0	1	23	2	0	1	0	1	2	Moderate
1	2	0	0	23	1	0	1	0	0	2	Moderate
2	0	1	0	18	1	0	1	0	1	2	Moderate
3	1	0	0	11	1	0	1	0	0	2	Moderate
4	1	0	0	18	1	0	1	1	0	1	Low
...	...	...	...	...	...	...	...	...	...	...	...
1077	0	1	1	18	1	0	1	0	0	2	Moderate
1124	2	0	1	23	0	1	0	0	0	1	Low
1132	0	1	1	18	1	0	1	0	0	1	Moderate
1160	2	0	1	23	1	1	0	0	0	1	Moderate
1197	2	0	1	23	0	1	1	0	0	2	Moderate

224 rows × 11 columns

Description

We encoded variables such as education, institution, gender, location, financial condition, network type, device, and IT student status, Encoding categorical variables into numerical formats is crucial for machine learning models, enhancing predictive accuracy and performance. This simplifies the dataset, making it computationally efficient for analysis, and enhances data handling and modeling.

Normalization :

Data after normalization

8- Normalization :

In [81]:
```python
# Extract columns to normalize
columns_to_normalize = ['Age']
data_to_normalize = df[columns_to_normalize]

# Min-Max scaling for selected columns
minmax_scaler = MinMaxScaler()
normalized_data_minmax = minmax_scaler.fit_transform(data_to_normalize)

# Replace the normalized values in the original DataFrame
df[columns_to_normalize] = normalized_data_minmax

print("Min-Max scaled data :")
df
```

Min-Max scaled data :

Out[81]:

	Education Level	Institution Type	Gender	Age	Device	IT Student	Location	Financial Condition	Internet Type	Network Type	Flexibility Level
0	2	0	1	0.777778	2	0	1	0	1	2	Moderate
1	2	0	0	0.777778	1	0	1	0	0	2	Moderate
2	0	1	0	0.500000	1	0	1	0	1	2	Moderate
3	1	0	0	0.111111	1	0	1	0	0	2	Moderate
4	1	0	0	0.500000	1	0	1	1	0	1	Low
...	...	...	...	...	...	...	...	...	...	...	...
1077	0	1	1	0.500000	1	0	1	0	0	2	Moderate
1124	2	0	1	0.777778	0	1	0	0	0	1	Low
1132	0	1	1	0.500000	1	0	1	0	0	1	Moderate
1160	2	0	1	0.777778	1	1	0	0	0	1	Moderate
1197	2	0	1	0.777778	0	1	1	0	0	2	Moderate

224 rows × 11 columns

Description

We have normalized the age attribute to a uniform range, using Min-Max scaling to help us handle the data easily and to ensures that the age attribute has an equal opportunity to influence the outcomes.

Raw data :

In [63]:
```
df = pd.read_csv("Dataset\students_adaptability_level_online_education.csv",sep=",")
df
#Using the Pandas Library's functionalities to read data from a CSV file
#into a Pandas DataFrame, enabling us to examine, and visualize the data.
```

Out[63]:

	Education Level	Institution Type	Gender	Age	Device	IT Student	Location	Financial Condition	Internet Type	Network Type	Flexibility Level
0	University	Private	Male	23	Tab	No	Town	Mid	Wifi	4G	Moderate
1	University	Private	Female	23	Mobile	No	Town	Mid	Mobile Data	4G	Moderate
2	College	Public	Female	18	Mobile	No	Town	Mid	Wifi	4G	Moderate
3	School	Private	Female	11	Mobile	No	Town	Mid	Mobile Data	4G	Moderate
4	School	Private	Female	18	Mobile	No	Town	Poor	Mobile Data	3G	Low
...	...	...	...	...	...	...	...	...	...	...	...
1200	College	Private	Female	18	Mobile	No	Town	Mid	Wifi	4G	Low
1201	College	Private	Female	18	Mobile	No	Rural	Mid	Wifi	4G	Moderate
1202	School	Private	Male	11	Mobile	No	Town	Mid	Mobile Data	3G	Moderate
1203	College	Private	Female	18	Mobile	No	Rural	Mid	Wifi	4G	Low
1204	School	Private	Female	11	Mobile	No	Town	Poor	Mobile Data	3G	Moderate

1205 rows × 11 columns

Data after processing :

```
df = pd.read_csv("Processed_dataset.csv")
df
```

Out[19]:

	Education Level	Institution Type	Gender	Age	Device	IT Student	Location	Financial Condition	Internet Type	Network Type	Flexibility Level
0	2	0	1	0.777778	2	0	1	0	1	2	1
1	2	0	0	0.777778	1	0	1	0	0	2	1
2	0	1	0	0.500000	1	0	1	0	1	2	1
3	1	0	0	0.111111	1	0	1	0	0	2	1
4	1	0	0	0.500000	1	0	1	1	0	1	0
...	...	...	...	...	...	...	...	...	...	...	...
241	0	1	0	0.500000	1	0	0	0	0	2	0
242	2	1	1	0.777778	1	0	0	0	0	1	0
243	1	1	0	0.289917	1	0	1	0	1	2	0
244	2	1	0	0.777778	1	0	1	0	1	2	0
245	2	1	1	0.777778	1	1	0	0	0	1	0

246 rows × 11 columns

5. Data Mining Technique

We utilized both supervised and unsupervised learning on our dataset, employing classification and clustering techniques.
For a classification, we used a decision tree. This recursive algorithm generates a tree with leaf nodes representing the final decisions. Our model will predict the class label (Flexibility Level) which has three classes: high, moderate and low, the prediction is based on the remaining attributes: Education Level, Institution Type, Gender, Age, Device, IT Student, Location, Financial Condition, Internet Type, Network Type. This technique includes dividing the dataset into two sets:
Training dataset: used for building the decision tree
Testing dataset: used to evaluate the constructed model.
Lastly, to assess our model, we evaluate the accuracy and cost-sensitive measures of the dataset using a confusion matrix.
We used (confusion matrix) method for evaluating the method.
For clustering, since it's unsupervised learning, it doesn’t use a class label for implementing the

cluster thus we deleted the class label attribute “Flexibility Level "and used all other attributes in clustering (Education Level, Institution Type, Gender, Age, Device, IT Student, Location, Financial Condition, Internet Type, Network) we use the K-means clustering algorithm to group the students into clusters with different number of clusters. We evaluate the K-Means algorithms using silhouette coefficient and Total Within-Cluster Sum of Squares and we plot the Elbow curve to determine the optimal number of clusters.

6. Evaluation and Comparison

- Classification

- **Classification [90% training, 10% test]:**

Figure (1) (decision tree)(entropy):

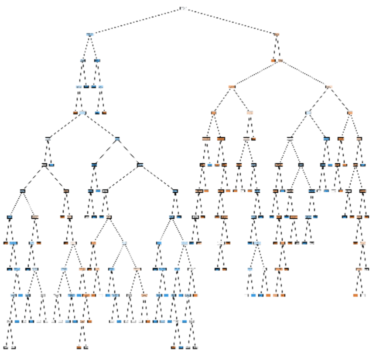


Figure (2) (decision tree) (Gini Index):

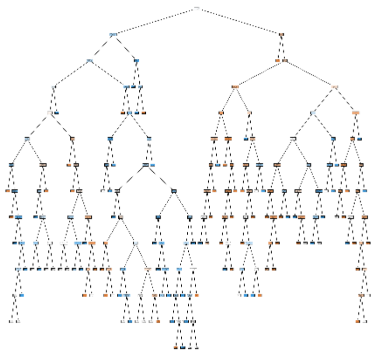


Figure (1) (matrix confusion) (entropy):

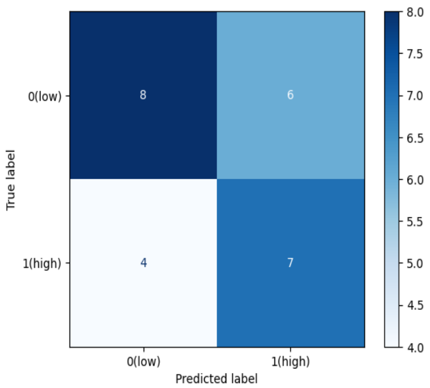
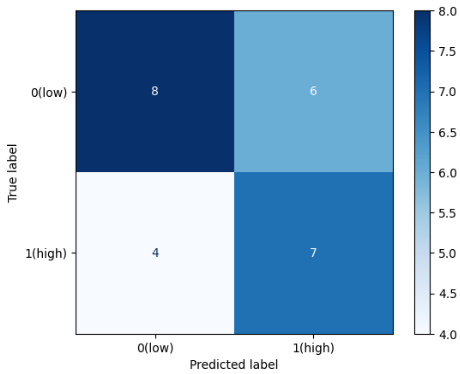


Figure (2) (matrix confusion) (Gini Index):



```
confusion matrix :
 [[8 6]
 [4 7]]
Accuracy: 0.6
Error Rate: 0.4
Sensitivity: 0.6363636363636364
Specificity: 0.5714285714285714
Precision: 0.5384615384615384
```

```
confusion matrix :
 [[8 6]
 [4 7]]
Accuracy: 0.6
Error Rate: 0.4
Sensitivity: 0.6363636363636364
Specificity: 0.5714285714285714
Precision: 0.5384615384615384
```

- **Classification [60% training, 40% test]:**

Figure (1) (decision tree)(entropy):

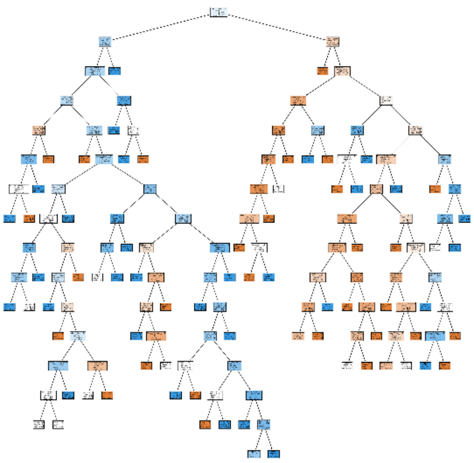


Figure (2) (decision tree) (GiniIndex):

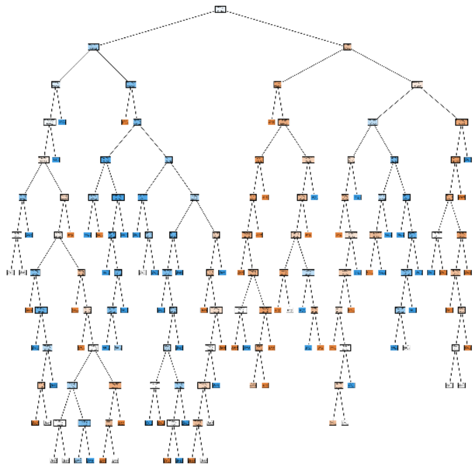


Figure (1) (matrix confusion)(entropy):

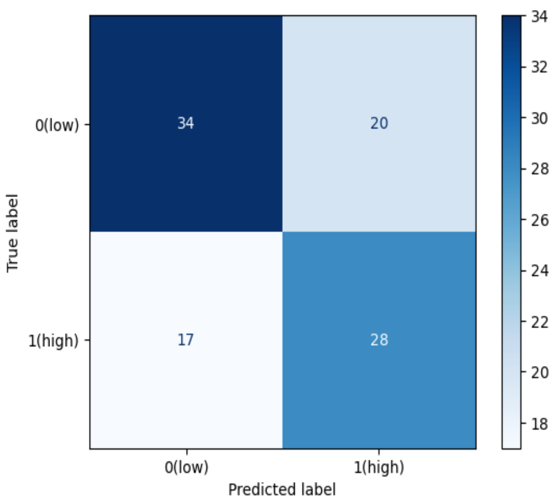
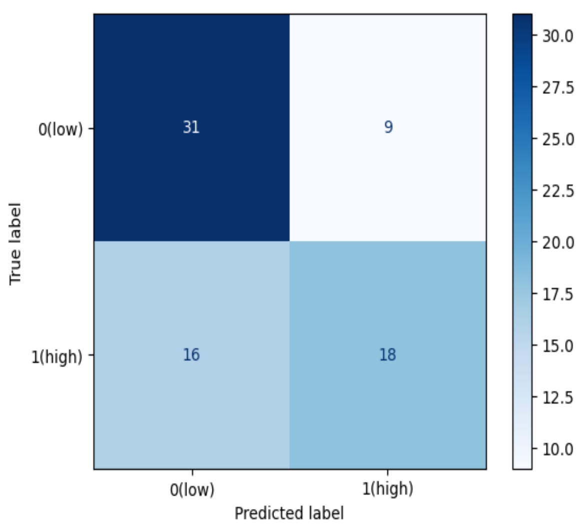


Figure (2) (matrix confusion) (Gini Index):

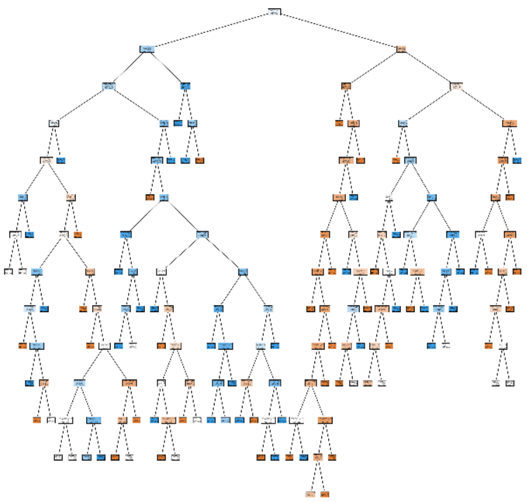


```
confusion matrix :
 [[34 20]
 [17 28]]
Accuracy: 0.6262626262626263
Error Rate: 0.37373737373737373737
Sensitivity: 0.6222222222222222
Specificity: 0.6296296296296297
Precision: 0.5833333333333334
```

```
confusion matrix :
 [[31  9]
 [16 18]]
Accuracy: 0.6621621621621622
Error Rate: 0.33783783783783783
Sensitivity: 0.5294117647058824
Specificity: 0.775
Precision: 0.6666666666666666
```

- **Classification [70% training, 30% test]:**

Figure (1) (decision tree)(entropy):



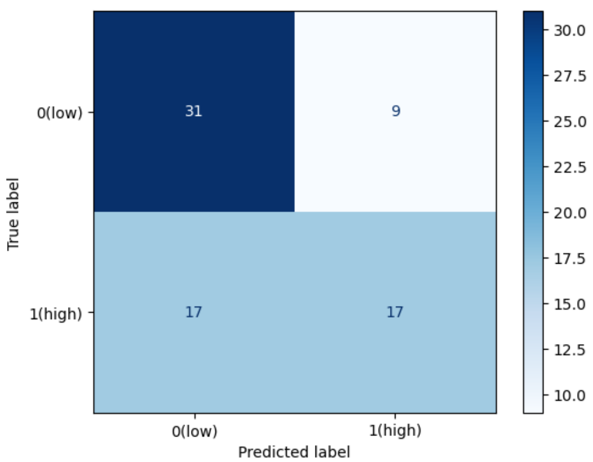
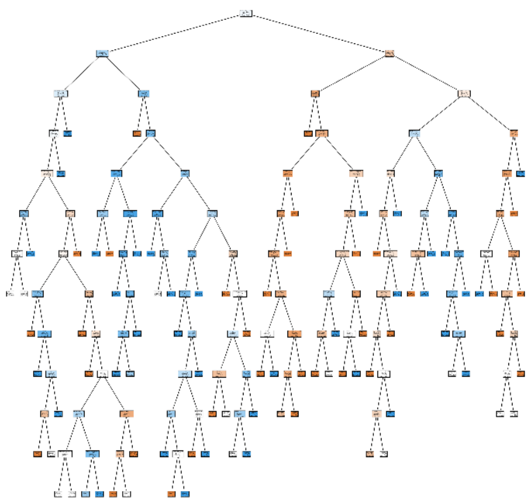


Figure (2) (decision tree) (Gini Index):



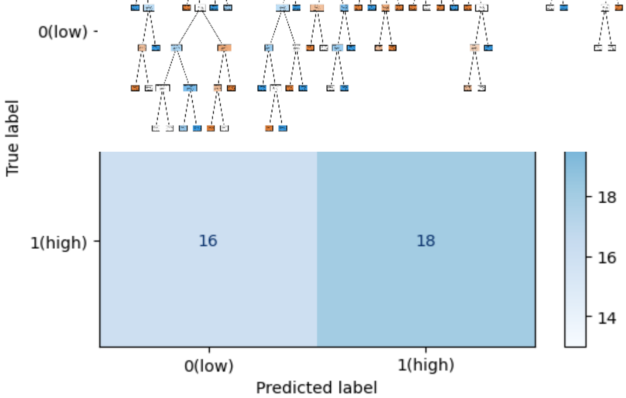


Figure (1) (matrix confusion) (entropy) Figure (2) (matrix confusion)(Gini Index):

```
confusion matrix :
 [[31  9]
 [17 17]]
Accuracy: 0.6486486486486487
Error Rate: 0.3513513513513513
Sensitivity: 0.5
Specificity: 0.775
Precision: 0.6538461538461539
```

```
confusion matrix :
 [[27 13]
 [16 18]]
Accuracy: 0.6081081081081081
Error Rate: 0.3918918918918919
Sensitivity: 0.5294117647058824
Specificity: 0.675
Precision: 0.5806451612903226
```

- **Classification [80% training, 20% test]:**

Figure (1) (decision tree) (entropy):

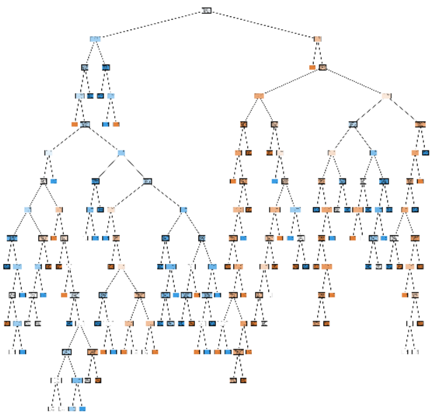


Figure (2) (decision tree) (Gini Index):

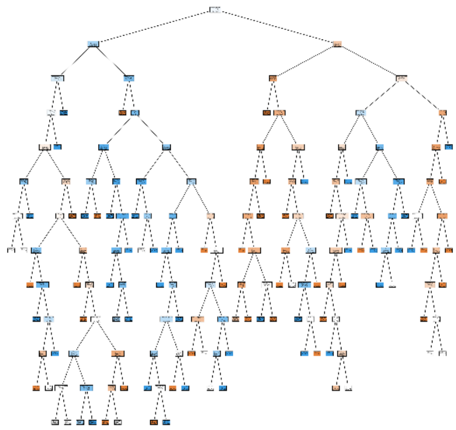
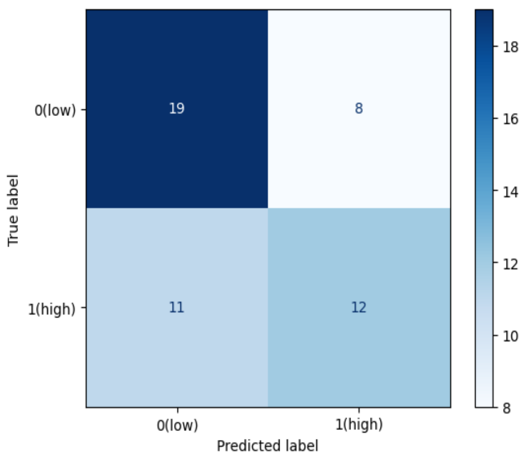


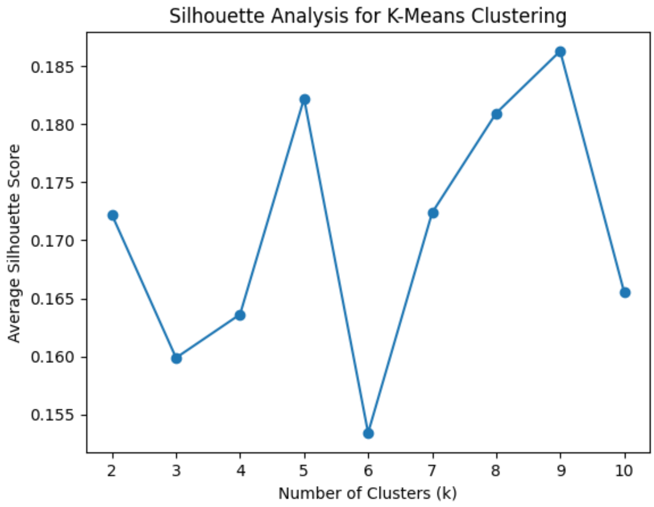
Figure (1) (matrix confusion)(entropy): **Figure (2) (matrix confusion) (GiniIndex):**



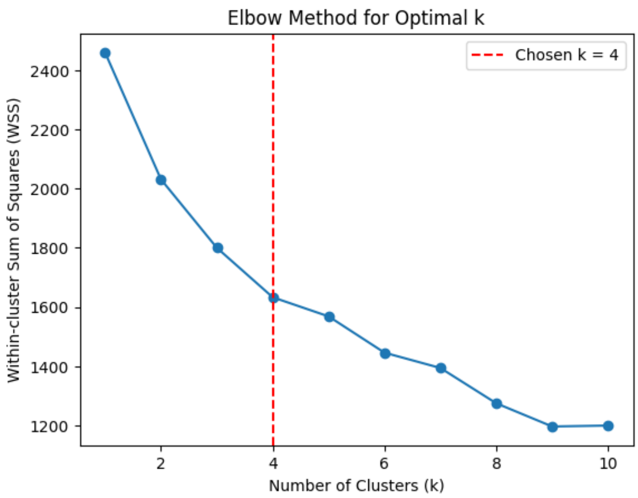

```
confusion matrix :
 [[27 13]
 [16 18]]
Accuracy: 0.6081081081081081
Error Rate: 0.3918918918918919
Sensitivity: 0.5294117647058824
Specificity: 0.675
Precision: 0.5806451612903226
```

- Clustering

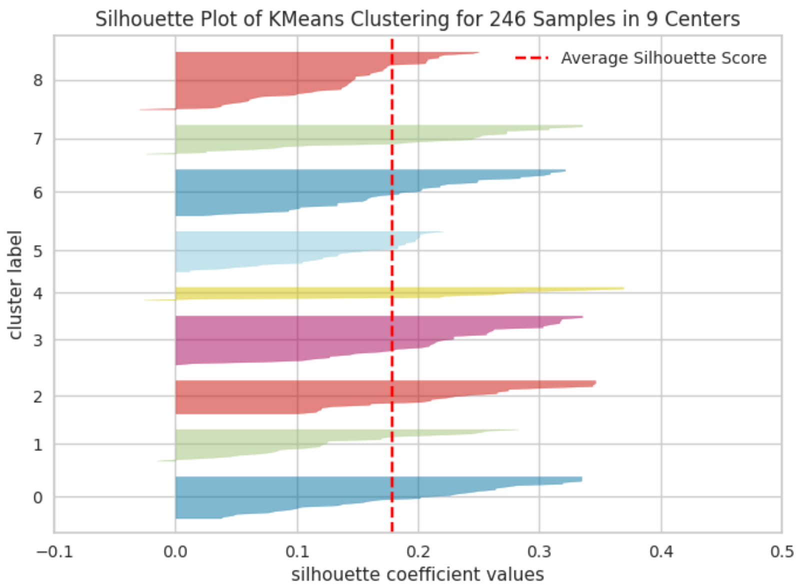
- Silhouette method (Silhouette Analysis):



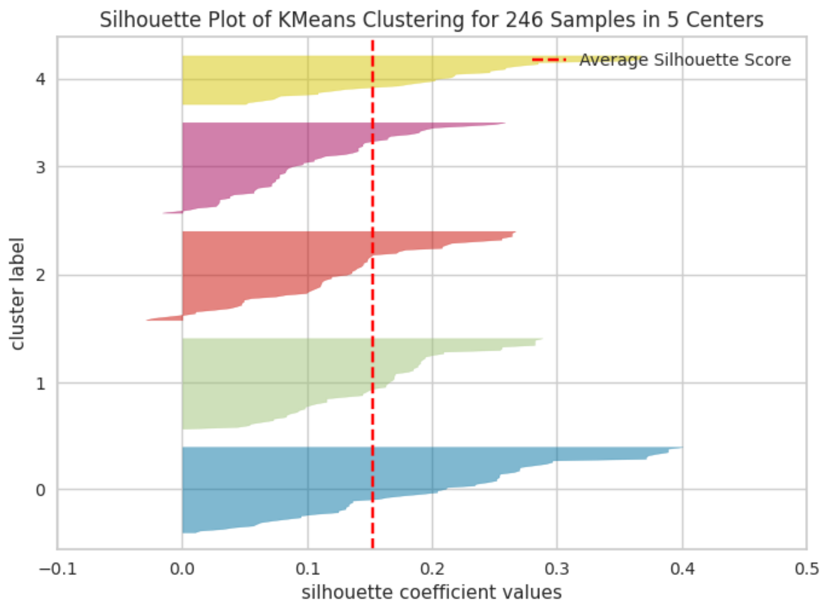
- Elbow method



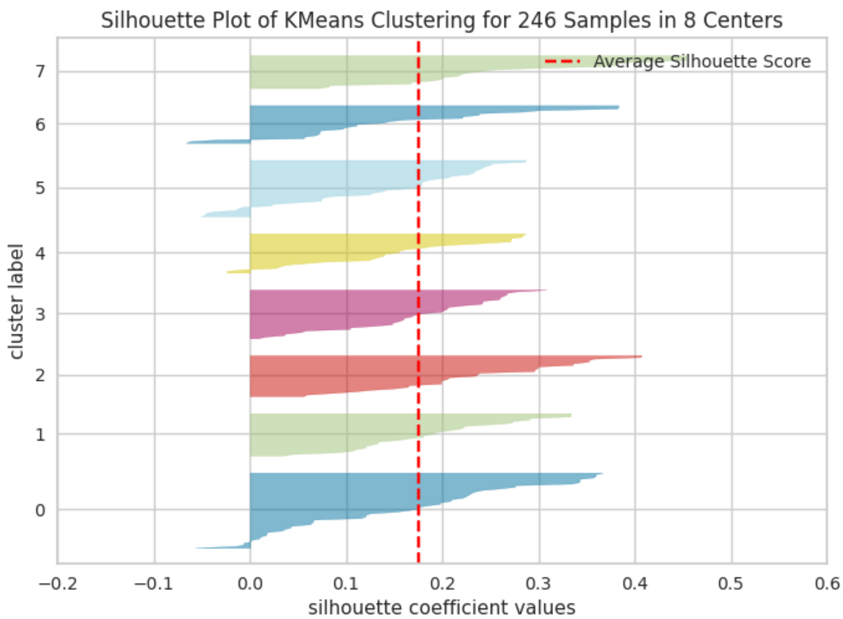
- **Clustering [K=9]:**



- **Clustering [K=5]:**



- **Clustering [K=8]:**



<table>
<tr><th>Mining task</th><th>Comparison Criteria</th></tr>
<tr><td>Classification</td><td>
Results - Gini
<table>
<tr><th>Metric</th><th>70 %training set 30% testing set</th><th>60 %training set 40% testing set</th><th>80 %training set 20% testing set</th><th>90% training set 10% testing set</th></tr>
<tr><td>Accuracy</td><td>0.60</td><td>0.6621</td><td>0.62</td><td>0.608</td></tr>
</table>
Results - Entropy
<table>
<tr><th>Metric</th><th>70 %training set 30% testing set</th><th>60 %training set 40% testing set</th><th>80 %training set 20% testing set</th><th>90% Training set 10% Testing set</th></tr>
<tr><td>Accuracy</td><td>0.64</td><td>0.626</td><td>0.62</td><td>0.6</td></tr>
</table>
</td></tr>
<tr><td>Clustering</td><td>
-
<table>
<tr><th></th><th>K=9</th><th>K=5</th><th>K=8</th></tr>
<tr><td>Average Silhouette width</td><td>0.1784279148509518</td><td>0.1519666106128939</td><td>0.1749348910251044</td></tr>
<tr><td>total within-cluster sum of square</td><td>1202.1795760464634</td><td>1550.551138764653</td><td>1250.7346543811886</td></tr>
</table>
</td></tr>
</table>

7. Findings

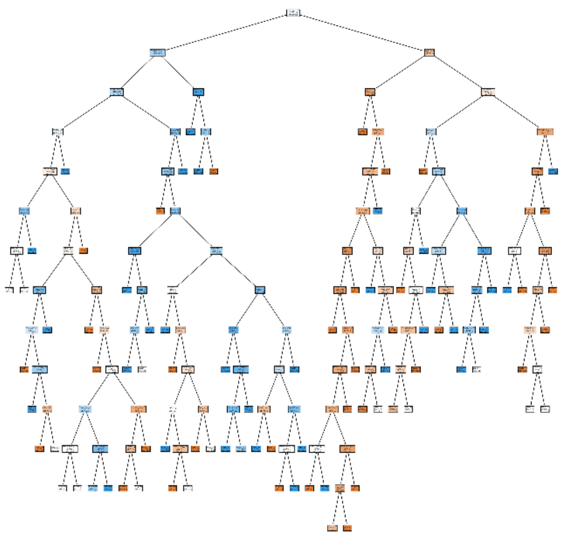
We have examinated the adaptability of students in online learning environments. With the surge in online education, notably accelerated by the COVID-19 pandemic, our project seeks to investigate students' data to identify the key factors impacting their adaptability.

By uncovering insights, to ensure our analysis of student flexibility is accurate, we used different methods to prepare the data well. We made plots like box plots and histograms to see the data clearly and decide what to do next. We got rid of any missing or unusual data that could mess up our results. Also, we changed some data to make it easier to work with and give each part of the data the same importance. These steps helped us understand and improve student flexibility in online learning.

As a result, we utilized data mining techniques, focusing on classification and clustering. For classification, we employed the decision tree method to build our model. We experimented with four different sizes of training and testing data to find the optimal setup for constructing and evaluating our model. Here are our findings:

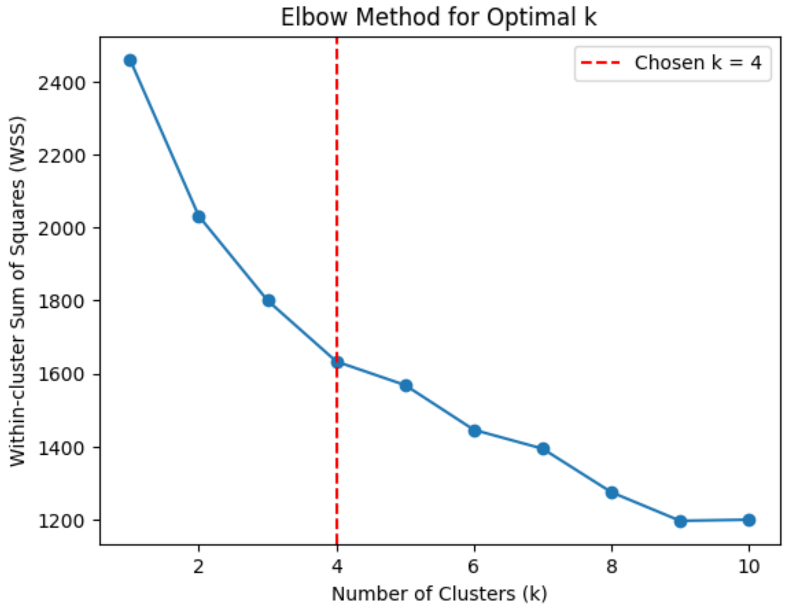
- **70% Training, 30% Testing: Accuracy = 0.64**
- **60% Training, 40% Testing: Accuracy = 0.626**
- **80% Training, 20% Testing: Accuracy = 0.62**
- **90% Training, 10% Testing: Accuracy = 0.6**

The model achieving the highest accuracy, trained on 70% of the data and tested on 30%, stood out with an accuracy score of 0.64. This outcome suggests that this particular training-testing split yielded the most successful performance among the evaluated scenarios.
The rationale behind its effectiveness likely stems from the equilibrium between the training and testing set sizes. By allocating 70% of the data for training, the model had ample information to discern complex patterns and nuances within the dataset. Meanwhile, the 30% reserved for testing facilitated rigorous evaluation without the risk of overfitting.
While other factors such as data quality and feature selection could have influenced the results, the 70-30 split emerged as the optimal configuration for maximizing accuracy in this context.



- **Root Node: The root node (the top node) represents the entire dataset. This node splits the data based on the attribute that provides the highest information gain, which is calculated using entropy.**
- **Internal Nodes: Each internal node represents a decision point where the data is further split based on other attributes. The choice of attribute at each internal node is again determined by the attribute that provides the highest information gain.**
- **Branches: The branches represent the outcome of a decision at an internal node, leading to another internal node or a leaf node.**
- **Leaf Nodes: The leaf nodes (the nodes at the end of the branches) represent the final predictions of the model. In your case, these would be the predicted levels of "student flexibility".**
- **Path: A path from the root node to a leaf node represents a rule. For example, if a path from the root to a leaf node passes through the decisions A=True, B=False, and C=True, then the rule is "If A is True, B is False, and C is True, then predict the class label at the leaf node".**

For Clustering, we used K-means algorithm with 3 different K to find the optimal number



of clusters, we calculated the average silhouette width for each K, and we concluded the following results:

- For K =9:

Average Silhouette Width: 0.1784
Within-Cluster Sum of Squares (WSS): 1202.18
Interpretation: With K =9, the clusters exhibit a relatively high average silhouette width, indicating well-defined clusters. Additionally, the WSS is relatively low, suggesting that the clusters are compact and tightly packed around their centroids.

- For K =5:

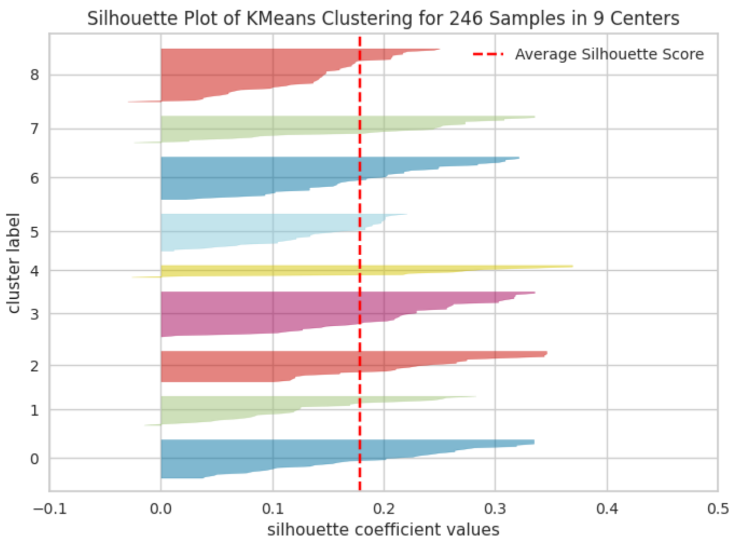
Average Silhouette Width: 0.1520
Within-Cluster Sum of Squares (WSS): 1550.55

Interpretation: With K =5, the average silhouette width is lower compared to K=9K=9, indicating less well-defined clusters. The WSS is relatively high, suggesting that the clusters are less compact compared to K=9K=9, with more spread-out data points within each cluster.

- For K =8:

Average Silhouette Width: 0.1749
Within-Cluster Sum of Squares (WSS): 1250.73
Interpretation: With K =8, the average silhouette width is higher compared to K =5 but slightly lower than K =9. The WSS is intermediate between the values for K =5 and K =9, indicating moderately compact clusters.

In summary, each value of K yields different clustering performance metrics. K =9 appears to result in the most well-defined and compact clusters based on both average silhouette width and WSS, followed by K =8 and then K =5.



in conclusion, both models have proven valuable in predicting the level of flexibility exhibited by students, thereby contributing significantly to our overarching goal of assisting individuals in adapting to online learning environments. However, given that our dataset includes a class label "student flexibility," supervised learning models, particularly classification models, are deemed more accurate and suitable for application.
Supervised learning approaches are more accurate than unsupervised learning model(clustering), as the expected output is known beforehand this way we make use of the class label attribute. we harness this existing knowledge to refine the accuracy and relevance of our predictive models, empowering students to make informed decisions about their learning strategies and adaptability in online educational settings.

8. References

- *https://www.kaggle.com/datasets/shariful07/student-flexibility-in-online-learning*

- *King Saud university - IT326 lab*
- *https://lms.ksu.edu.sa/bbcswebdav/pid-9410056-dt-content-rid-147775017_1/courses/Merged_IT326_74557_52846_11_452/Lab_week%232_Python_Introduction.pdf*
- *https://lms.ksu.edu.sa/bbcswebdav/pid-9443189-dt-content-rid-148159257_1/courses/Merged_IT326_74557_52846_11_452/Lab_week%233_Data%20Exploration%20and%20Visualization%20using%20Python.pdf*
- *https://lms.ksu.edu.sa/bbcswebdav/pid-9536829-dt-content-rid-148887652_1/courses/Merged_IT326_74557_52846_11_452/Data%20Preprocessing%20-%20Python%283%29.pdf*
- *https://lms.ksu.edu.sa/bbcswebdav/pid-9577926-dt-content-rid-150572431_1/courses/Merged_IT326_74557_52846_11_452/Lab7%20Classification-%20Python%281%29.pdf*
- *https://lms.ksu.edu.sa/bbcswebdav/pid-9595752-dt-content-rid-151204445_1/courses/Merged_IT326_74557_52846_11_452/Clustering_Python.pdf*