Exam Mod3_4 Take_Home Portion

Honor Statement:
I, ____Akhila Annireddy_____, promise to complete this exam on my own, without help from other people or AI tools. I understand what it means to do my own work, and I'm committed to being honest throughout the exam. I'm also aware that all answers will be checked using GPTZero to detect AI-generated content, and if any part of my submission is flagged or found to copied from AI tools (ChatGPT, Gemini, Perplexity, ClaudeAI, etc), I understand that I may receive zero points for that section and won't question professor and TAs.

Part 1: (70 points)

Find (or create/fashion) and use a labeled dataset (with 3 label categories), 5 - 20 dimensions of data, and 30 - 100 rows of data. Have balanced data. Do not change these requirements.

Name the column in your dataset that is your label, "LABEL".

Do not normalize the data.

Your data must be quantitative only.

Your file **must be .csv**.

Dataset Description

Feature Description

Area (A) – Total surface area of the kernel's projection.

Perimeter (P) – Length of the kernel's outer boundary.

Compactness (C) – Shape compactness; higher values indicate rounder shapes.

Length – Longest dimension of the kernel.

Width – Width measured perpendicular to the length.

Asymmetry – Degree of shape irregularity or imbalance.

Groove Length – Length of the central groove along the kernel.

Class label

1 → Kama – Wheat variety with moderately compact and symmetric kernels.

2 → Rosa – Wheat variety with wider and less compact kernels.

3 → Canadian – Wheat variety with longer grooves and higher asymmetry.

Reference link to dataset : https://archive.ics.uci.edu/dataset/236/seeds

Use Python as needed. Place/paste/or type all answers to all questions into this Word Document. Please keep your answers WELL ORGANIZED so that it is easy to see what you are answering and what your answers are. Use your dataset to answer the following.

Points: 15

1) Perform Gaussian, Categorical, and Multinomial Naïve Bayes on your data. For each, Train the model - Test the model - include the resulting prediction probabilities - and include the confusion matrices. You will need to reformat your data for different Naive Bayes models. Illustrate and **clearly show** what your data looks like for each Naive Model you use and include a few sentences under the screen image of the training data to explain the format needed.

Paste, illustrate, discuss, and explain what you did here so that the reader can follow your steps. **DO NOT place any code here or anywhere else in this document. You will submit your code separately.**

Things to Include (at least – you can exceed these requirements)

Naïve Bayes Model 1: Gaussian

Screen image of formatted Training Data (first 5 rows)

Training Data (Gaussian):

	area	perimeter	compactness	length	width	asymmetry	groove_length
9	12.26	13.60	0.8333	5.408	2.833	4.756	5.360
15	11.02	13.00	0.8189	5.325	2.701	6.735	5.163
76	20.16	17.03	0.8735	6.513	3.773	1.910	6.185
22	12.19	13.36	0.8579	5.240	2.909	4.857	5.158
49	14.99	14.56	0.8883	5.570	3.377	2.958	5.175

The training dataset is split 80% from the actual dataset. Already the training dataset has numerical and continuous values. Checked for missing and null values. Also, no scaling is applied as Naive Bayes handles raw data well.

Screen image of Training labels (first 5 rows)

Training Labels:

9	0
15	0
76	2
22	0
49	1

This contains the labels corresponding to the training data which is also 80% from the dataset. Which has three classes naming class label 0,1,2 each representing a wheat type which are encoded on original dataset.

Screen image of formatted Testing Data (first 5 rows)

Testing Data (Gaussian):

	area	perimeter	compactness	length	width	asymmetry	groove_length
71	17.12	15.55	0.8892	5.850	3.566	2.858	5.746
10	11.55	13.10	0.8455	5.167	2.845	6.715	4.956
44	12.36	13.19	0.8923	5.076	3.042	3.220	4.605
39	13.89	14.02	0.8880	5.439	3.199	3.986	4.738
74	15.38	14.90	0.8706	5.884	3.268	4.462	5.795

The testing dataset is split 20% from the actual dataset. This data is unseen by the model used in testing the performance of the model. This also didn't require any additional cleaning as this is already numerical and checking for missing and null values.

Screen image of Testing Labels (first 5 rows)

Testing Labels:

71	2
10	0
44	1
39	1
74	2

This is a testing label which is 20% corresponds to the test data. The labels are encoded 0,1,2 representing wheat types.

Explanation of format requirements with respect to model choice (3 -5 sentences).
Gaussian Naive Bayes works with numerical data which is continuous. It doesn't require any scaling as it can perform well with raw data too. It cannot handle missing and null values. So these should be handled before training the model. Categorical data should be encoded.

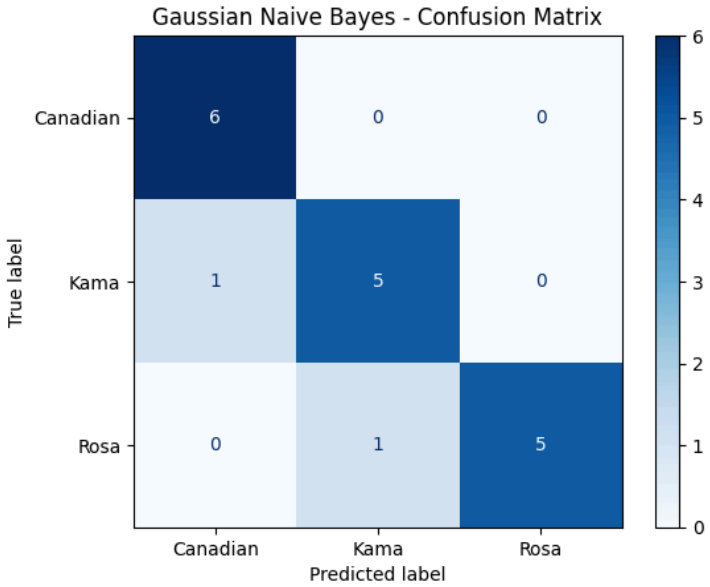
Prediction Probabilities

Prediction Probabilities (first 10 rows):

	Prob_Class_Canadian	Prob_Class_Kama	Prob_Class_Rosa
0	2.371330e-29	1.693809e-04	9.998306e-01
1	1.000000e+00	1.953322e-12	8.492559e-26
2	8.538029e-01	1.461971e-01	3.798217e-17
3	2.870518e-05	9.999713e-01	1.091805e-12
4	9.781740e-15	9.278424e-01	7.215765e-02
5	1.578700e-12	9.999996e-01	3.532665e-07
6	1.000000e+00	1.163711e-08	1.730012e-23
7	2.005459e-65	8.913675e-20	1.000000e+00
8	3.196377e-24	1.503113e-04	9.998497e-01
9	1.000000e+00	1.976089e-09	1.020962e-24

The probabilities represent the model's confidence in assigning the class label to test data. Each record represents the row in test data and has three columns pointing to each class. The values represent how confident the test data belongs to the class. Higher the confidence values represent that as the class label for the record.

Confusion Matrix (in a color and pretty format such as ConfusionMatrixDisplay)



The confusion matrix displays that all the predictions are correct for class 0, 1 from class 1 and 1 from class 2 are wrongly predicted.This says that model performed well with few errors.

Classification Report:

```
Accuracy: 0.8888888888888888

Classification Report:
              precision    recall  f1-score   support

           0       0.86      1.00      0.92         6
           1       0.83      0.83      0.83         6
           2       1.00      0.83      0.91         6

    accuracy                           0.89        18
   macro avg       0.90      0.89      0.89        18
weighted avg       0.90      0.89      0.89        18
```

The model achieved 88.89% showing good performance. Precision and recall are high for all classes with class 2 achieving precision 1 and class 0 with recall 1.

Naïve Bayes Model 2: Categorical

Screen image of formatted Training Data (first 5 rows)

Training Data (Categorical – binned):

	area	perimeter	compactness	length	width	asymmetry	groove_length
0	0.0	1.0	1.0	1.0	0.0	2.0	1.0
1	0.0	0.0	0.0	1.0	0.0	3.0	1.0
2	4.0	4.0	2.0	4.0	4.0	0.0	4.0
3	0.0	0.0	2.0	1.0	0.0	2.0	1.0
4	2.0	2.0	3.0	1.0	2.0	1.0	1.0

I have applied KbinsDiscretizer() to transform the continuous data to categorical data. Which make sthe continuous values to fall in bin ranges. Which is also 80% from dataset

Screen image of labels (first 5 rows)

Training Labels:

9	0
15	0
76	2
22	0
49	1

Target labels corresponding to train data which are encoded.

Screen image of formatted Testing Data (first 5 rows)

Testing Data (Categorical – binned):

	area	perimeter	compactness	length	width	asymmetry	groove_length
0	3.0	3.0	3.0	2.0	3.0	1.0	3.0
1	0.0	0.0	1.0	0.0	0.0	3.0	0.0
2	0.0	0.0	3.0	0.0	1.0	1.0	0.0
3	1.0	1.0	3.0	1.0	2.0	2.0	0.0
4	2.0	2.0	2.0	2.0	2.0	2.0	3.0

I have applied KbinsDiscretizer() to transform the continuous data to categorical data. Which makes the continuous values to fall in bin ranges. Which is also 20% from dataset

Screen image of Testing Labels (first 5 rows)

Testing Labels:

71	2
10	0
44	1
39	1
74	2

target label corresponding to the test data which are encoded.

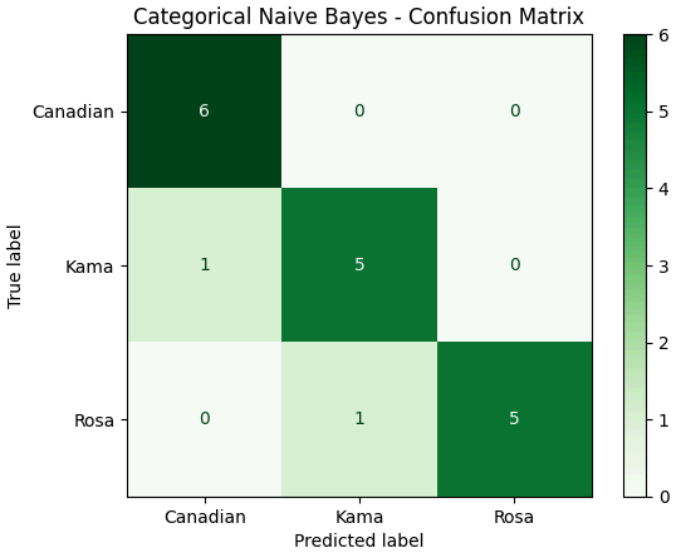
Explanation of format requirements with respect to model choice (3 -5 sentences).

Categorical naive bayes work for categorical or discrete data. It cannot handle negative values. Continuous numerical features should be converted into discrete bins. Categorical values should be encoded. Each number needs to represent either a category or a bin. Even Categorical models also cannot handle missing(null) values.

Prediction Probabilities

```
Prediction Probabilities (first 10 rows):
   Prob_Class_Canadian  Prob_Class_Kama  Prob_Class_Rosa
0             0.000001         0.001107     9.988921e-01
1             0.999991         0.000009     4.499112e-07
2             0.929411         0.070250     3.387066e-04
3             0.075551         0.924383     6.666222e-05
4             0.000189         0.954842     4.496902e-02
5             0.000002         0.999409     5.883493e-04
6             0.998563         0.001415     2.211245e-05
7             0.000002         0.000047     9.999504e-01
8             0.000369         0.240017     7.596139e-01
9             0.999914         0.000083     2.604985e-06
```

Confusion Matrix (in a color and pretty format such as ConfusionMatrixDisplay)



The confusion matrix displays that all the predictions are correct for class 0, 1 from class 1 and 1 from class 2 are wrongly predicted.This says that model performed well with few errors.

Classification Report:

```
Accuracy: 0.8888888888888888

Classification Report:
              precision    recall  f1-score   support

           0       0.86      1.00      0.92         6
           1       0.83      0.83      0.83         6
           2       1.00      0.83      0.91         6

    accuracy                           0.89        18
   macro avg       0.90      0.89      0.89        18
weighted avg       0.90      0.89      0.89        18
```

The model achieved 88.89% showing good performance. Precision and recall are high for all classes with class 2 achieving precision 1 and class 0 with recall 1.

Naïve Bayes Model 3: Multinomial

Screen image of formatted Training Data (first 5 rows)

```
Training Data (Multinomial - count based):
     area  perimeter  compactness  length  width  asymmetry  groove_length
9      12         13            0       5      2          4              5
15     11         13            0       5      2          6              5
76     20         17            0       6      3          1              6
22     12         13            0       5      2          4              5
49     14         14            0       5      3          2              5
```

The original features are continuous. For Multinomial naive bayes, the data should be in count based inputs. So used .astype(int) to convert them to whole integers and there are no negative values and missing values

Screen image of labels (first 5 rows)

```
Training Labels:
 9     0
15     0
76     2
22     0
49     1
Name: LABEL, dtype: int64
```

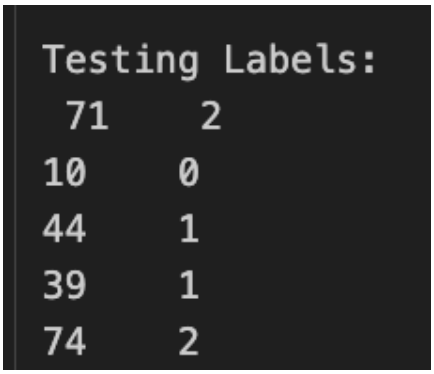
The class labels were encoded as integers to represent the three wheat types: Kama, Rosa, and Canadian. These numerical labels are used during training and model evaluation to match the classifier's expected format.

Screen image of formatted Testing Data (first 5 rows)

```
Testing Data (multinomial - Count bases):
     area  perimeter  compactness  length  width  asymmetry  groove_length
71     17         15            0       5      3          2              5
10     11         13            0       5      2          6              4
44     12         13            0       5      3          3              4
39     13         14            0       5      3          3              4
74     15         14            0       5      3          4              5
```

Same as training data, which is 20% of the original dataset which is unknown to model used to test the model performance.

Screen image of Testing Labels (first 5 rows)

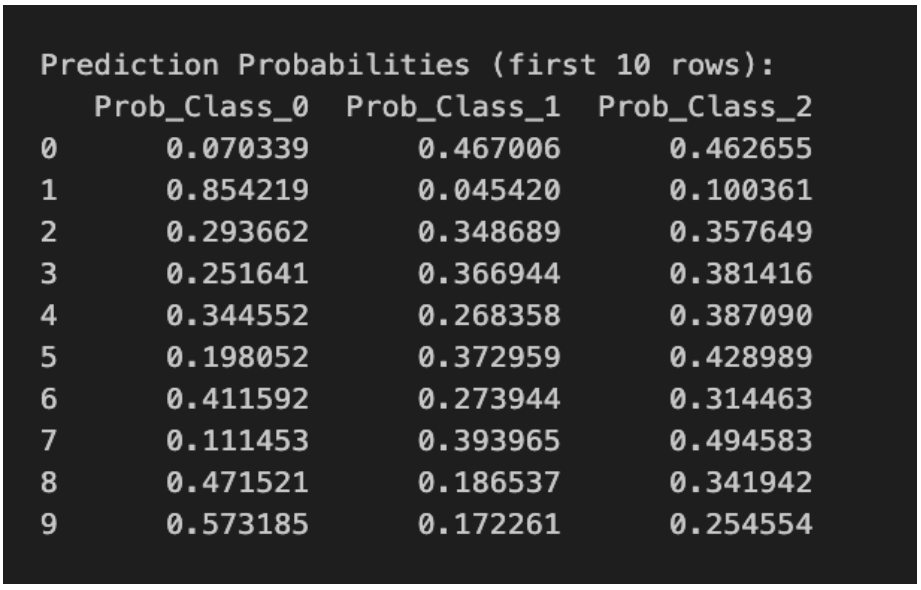


```
Testing Labels:
 71    2
10    0
44    1
39    1
74    2
```

Explanation of format requirements with respect to model choice (3 -5 sentences).

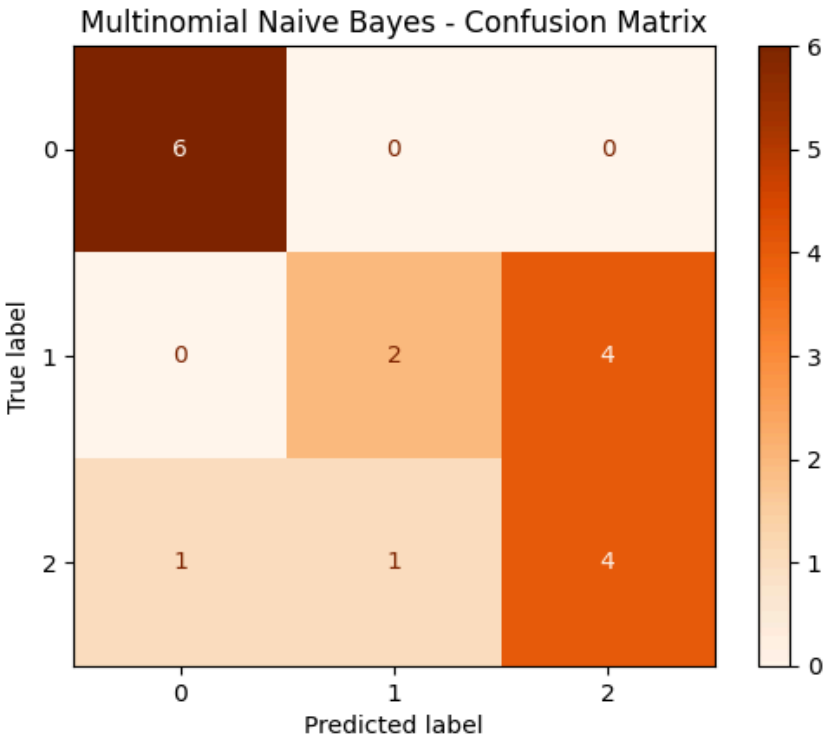
Multinomial Naive Bayes require input features to be count or frequency-based, meaning they must be non-negative whole numbers. It does not support continuous or decimal values directly; if present, such values should be rounded or converted to integers. Additionally, this model cannot handle missing values, so any such entries must be addressed during preprocessing.

Prediction Probabilities



```
Prediction Probabilities (first 10 rows):
   Prob_Class_0  Prob_Class_1  Prob_Class_2
0      0.070339      0.467006      0.462655
1      0.854219      0.045420      0.100361
2      0.293662      0.348689      0.357649
3      0.251641      0.366944      0.381416
4      0.344552      0.268358      0.387090
5      0.198052      0.372959      0.428989
6      0.411592      0.273944      0.314463
7      0.111453      0.393965      0.494583
8      0.471521      0.186537      0.341942
9      0.573185      0.172261      0.254554
```

Confusion Matrix (in a color and pretty format such as ConfusionMatrixDisplay)



The confusion matrix visually summarizes the prediction results, showing that the model struggles with classifying classes 1 and 2 accurately. It highlights misclassifications, particularly between similar classes.

Classification Report

```
Accuracy: 0.6666666666666666

Classification Report:
              precision    recall  f1-score   support

           0       0.86      1.00      0.92         6
           1       0.67      0.33      0.44         6
           2       0.50      0.67      0.57         6

    accuracy                           0.67        18
   macro avg       0.67      0.67      0.65        18
weighted avg       0.67      0.67      0.65        18
```

The classification report shows an overall accuracy of 66.67%, with class 0 performing the best. Precision, recall, and F1-score values vary across classes, indicating imbalance in model performance.

Points: 10

2) Perform Decision Tree Classification to model your data. Include at least:

Actual Testing Labels

Actual Label
2
0
1
1
2
1
0
2
2
0
0
0
1
2
2
1
0
1

Predicted Testing Labels (as predicted by the model)

Predicted Label
2
0
1
1
2
1
0
2
2
0
0
0
0
2
2
1
0
1

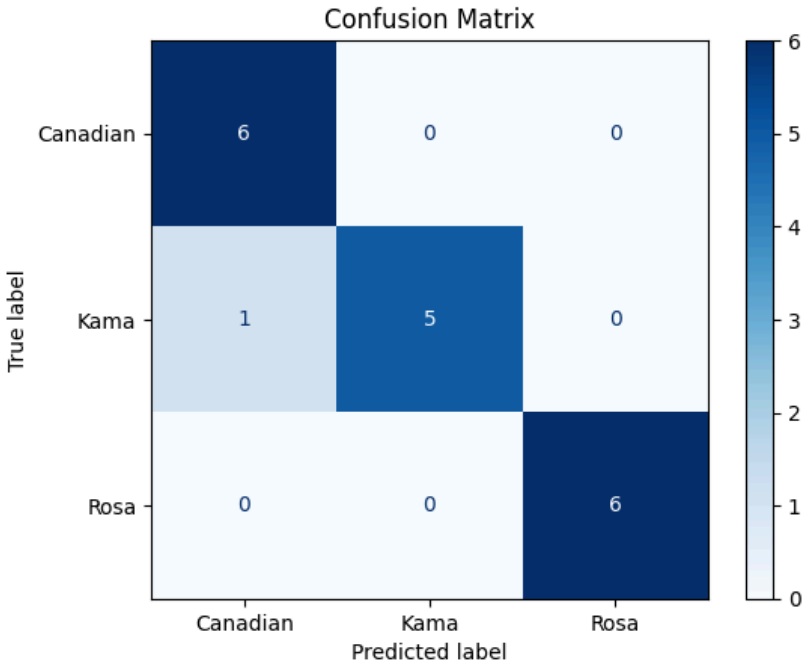
Comparison

Comparison of Actual vs Predicted Labels:

	Actual Label	Predicted Label
0	2	2
1	0	0
2	1	1
3	1	1
4	2	2
5	1	1
6	0	0
7	2	2
8	2	2
9	0	0
10	0	0
11	0	0
12	1	0
13	2	2
14	2	2
15	1	1
16	0	0
17	1	1

Record 12, is wrongly predicted

Confusion Matrix:



From the confusion matrix, Class 1,3 has perfect model classification but class 2 has few errors.

Model Accuracy:

```
Model Accuracy: 0.9444444444444444
```

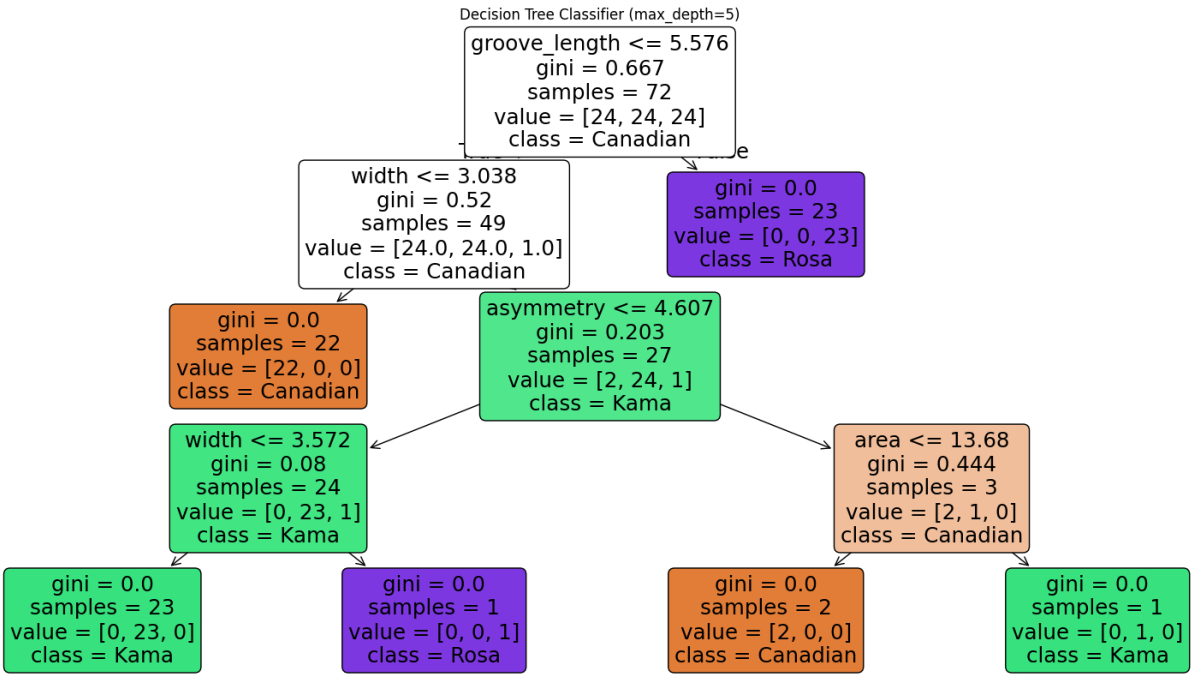
```
Classification Report:
              precision    recall  f1-score   support

           0       0.86      1.00      0.92         6
           1       1.00      0.83      0.91         6
           2       1.00      1.00      1.00         6

    accuracy                           0.94        18
   macro avg       0.95      0.94      0.94        18
weighted avg       0.95      0.94      0.94        18
```

The model achieved 94.44% accuracy with good precision. for class 1, class 2. Over all model has performed well maintaining a proper balance

The Decision Tree Image (and please limit the tree to be at least 4 levels and no more than 6 levels.)

In 3 – 5 sentences, explain the tree. Include ideas and notes that you feel are important.

The tree first checks the length of the groove to form an initial split. Then the seeds with a longer groove length are classified as Rosa straight away, and the rest are further divided on the basis of properties like width and asymmetry. These splits separate the seeds as Canadian and Kama with good accuracy. The tree is open,with depth 5 making it easily interpretable from the tree visual.



Create a new test vector and type it here. Explain what you get when you apply your Decision Tree to this Test Vector. Illustrate your results in a smart way.

Sample data : 17.2, 15.6, 0.88, 6.1, 3.6, 4.5, 5.9
area, perimeter, compactness, length, width, asymmetry, and groove length

Tested the model on a new seed sample The decision tree has categorized this sample under class 2. From the sample data, grovve_length is 5.9 which is greater than 5.576 so according to the tree we can say that it gets classified as rosa. and that's the output given by model too.

Points: 15

3) Perform SVM modeling on your data. Use three different models so that you use at least one polynomial kernel with degree 2 or 3 (your choice), one rbf kernel, and one other (your choice). For each, include at least the following:

SVM 1: Kernel: Polynomial Cost 1.0 Degree 3

Screen image of formatted Training Data (first 5 rows)

Formatted Training Data:

	area	perimeter	compactness	length	width	asymmetry	groove_length
9	12.26	13.60	0.8333	5.408	2.833	4.756	5.360
15	11.02	13.00	0.8189	5.325	2.701	6.735	5.163
76	20.16	17.03	0.8735	6.513	3.773	1.910	6.185
22	12.19	13.36	0.8579	5.240	2.909	4.857	5.158
49	14.99	14.56	0.8883	5.570	3.377	2.958	5.175

The training data is 80% from the original dataset. No scaling is applied on the dataset as per the instruction.No transformations are done except checking for missing and null values.

Screen image of labels (first 5 rows)

```
Training Labels:
9     0
15    0
76    2
22    0
49    1
3     0
```

This is training data label which corresponds as target variable to training data is 80% from the raw dataset. The labels are encoded as 0,1,2 corresponds to each wheat type.

Screen image of formatted Testing Data (first 5 rows)

```
Formatted Testing Data:
     area  perimeter  compactness  length  width  asymmetry  groove_length
71  17.12      15.55       0.8892   5.850  3.566      2.858          5.746
10  11.55      13.10       0.8455   5.167  2.845      6.715          4.956
44  12.36      13.19       0.8923   5.076  3.042      3.220          4.605
39  13.89      14.02       0.8880   5.439  3.199      3.986          4.738
74  15.38      14.90       0.8706   5.884  3.268      4.462          5.795
```

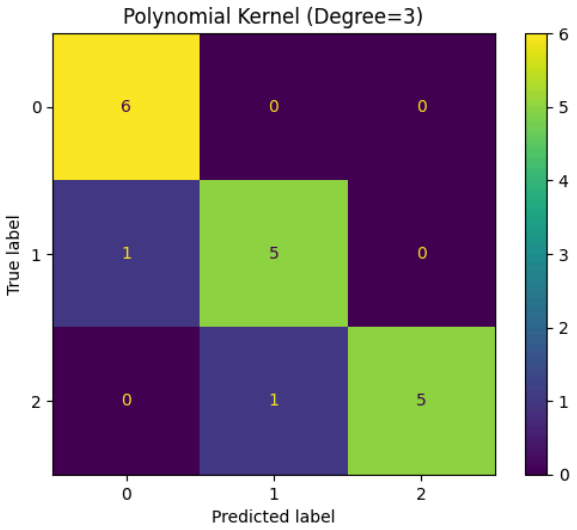
This is the testing data which 20% of the original data and no transformation are done. Checked for missing and null values. This is unknown to model and used to validate model performance.

Screen image of Testing Labels (first 5 rows)

```
Testing Labels:
71    2
10    0
44    1
39    1
74    2
57    1
```

This is the testing data labels which 20% of the original data and no transformation are done. Checked for missing and null values. This is unknown to model and used to compare from the output given by model for testing data.

Confusion Matrix (in a color and pretty format such as ConfusionMatrixDisplay)



From the confusion matrix, Class 0 has all the predictions correct while class 1 and class 2 had few errors.

Classification Report :

```
Accuracy: 0.8888888888888888
Classification Report:
              precision    recall  f1-score   support

           0       0.86      1.00      0.92         6
           1       0.83      0.83      0.83         6
           2       1.00      0.83      0.91         6

    accuracy                           0.89        18
   macro avg       0.90      0.89      0.89        18
weighted avg       0.90      0.89      0.89        18
```

The classification model achieved an overall accuracy of approximately 88.9%, performing consistently across all three classes. Precision and recall scores were strong, particularly for class 0 with perfect recall (1.00) and for class 2 with perfect precision (1.00). The macro and weighted average F1-scores of 0.89 indicate balanced performance across the dataset despite the small sample size.

SVM 2: Kernel: rbf Cost 1.0

Screen image of formatted Training Data (first 5 rows)

```
Formatted Training Data:
     area  perimeter  compactness  length  width  asymmetry  groove_length
9   12.26      13.60       0.8333   5.408  2.833      4.756          5.360
15  11.02      13.00       0.8189   5.325  2.701      6.735          5.163
76  20.16      17.03       0.8735   6.513  3.773      1.910          6.185
22  12.19      13.36       0.8579   5.240  2.909      4.857          5.158
49  14.99      14.56       0.8883   5.570  3.377      2.958          5.175
```

The training data is 80% from the original dataset. No scaling is applied on the dataset as per the instruction.No transformations are done except checking for missing and null values.

Screen image of labels (first 5 rows)

```
Training Labels:
9     0
15    0
76    2
22    0
49    1
3     0
```

This is training data label which corresponds as target variable to training data is 80% from the raw dataset. The labels are encoded as 0,1,2 corresponds to each wheat type.

Screen image of formatted Testing Data (first 5 rows)

```
Formatted Testing Data:
     area  perimeter  compactness  length  width  asymmetry  groove_length
71  17.12      15.55       0.8892   5.850  3.566      2.858          5.746
10  11.55      13.10       0.8455   5.167  2.845      6.715          4.956
44  12.36      13.19       0.8923   5.076  3.042      3.220          4.605
39  13.89      14.02       0.8880   5.439  3.199      3.986          4.738
74  15.38      14.90       0.8706   5.884  3.268      4.462          5.795
```

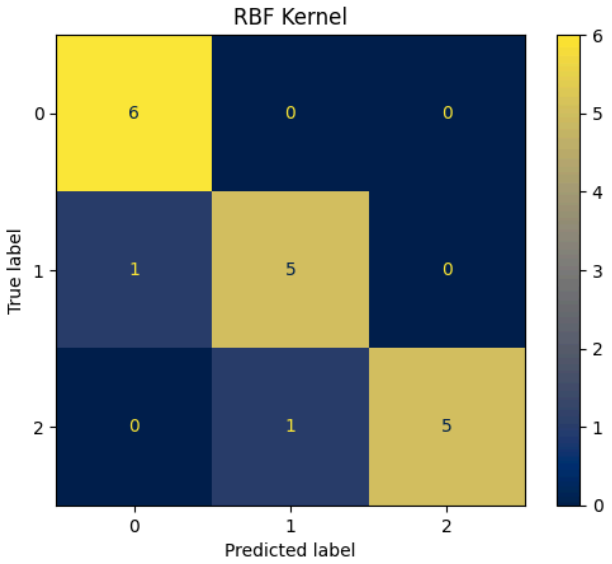
This is the testing data which 20% of the original data and no transformation are done. Checked for missing and null values. This is unknown to model and used to validate model performance.

Screen image of Testing Labels (first 5 rows)

```
Testing Labels:
71    2
10    0
44    1
39    1
74    2
57    1
```

This is the testing data labels which 20% of the original data and no transformation are done. Checked for missing and null values. This is unknown to model and used to compare from the output given by model for testing data.

Confusion Matrix (in a color and pretty format such as ConfusionMatrixDisplay)



The confusion matrix for the RBF Kernel shows that the model correctly classified all instances of class 0 and most of classes 1 and 2, with only one misclassification each. This indicates strong overall performance with minimal confusion between classes.

Classification Report:

```
Accuracy: 0.8888888888888888
Classification Report:
              precision    recall  f1-score   support

           0       0.86      1.00      0.92         6
           1       0.83      0.83      0.83         6
           2       1.00      0.83      0.91         6

    accuracy                           0.89        18
   macro avg       0.90      0.89      0.89        18
weighted avg       0.90      0.89      0.89        18
```

The classification report shows that the model achieved a high overall accuracy of 88.9%, with balanced performance across all three classes. Notably, class 0 had perfect recall, and class 2 had perfect precision, reflecting the model's strong ability to correctly identify those specific categories.

SVM 3: Kernel: Linear Cost 1.0 Degree (if poly) not poly

Screen image of formatted Training Data (first 5 rows)

Formatted Training Data:

	area	perimeter	compactness	length	width	asymmetry	groove_length
9	12.26	13.60	0.8333	5.408	2.833	4.756	5.360
15	11.02	13.00	0.8189	5.325	2.701	6.735	5.163
76	20.16	17.03	0.8735	6.513	3.773	1.910	6.185
22	12.19	13.36	0.8579	5.240	2.909	4.857	5.158
49	14.99	14.56	0.8883	5.570	3.377	2.958	5.175

The training data is 80% from the original dataset. No scaling is applied on the dataset as per the instruction.No transformations are done except checking for missing and null values.

Screen image of labels (first 5 rows)

Training Labels:

9	0
15	0
76	2
22	0
49	1
3	0

This is training data label which corresponds as target variable to training data is 80% from the raw dataset. The labels are encoded as 0,1,2 corresponds to each wheat type.

Screen image of formatted Testing Data (first 5 rows)

Formatted Testing Data:

	area	perimeter	compactness	length	width	asymmetry	groove_length
71	17.12	15.55	0.8892	5.850	3.566	2.858	5.746
10	11.55	13.10	0.8455	5.167	2.845	6.715	4.956
44	12.36	13.19	0.8923	5.076	3.042	3.220	4.605
39	13.89	14.02	0.8880	5.439	3.199	3.986	4.738
74	15.38	14.90	0.8706	5.884	3.268	4.462	5.795

This is the testing data which 20% of the original data and no transformation are done. Checked for missing and null values. This is unknown to model and used to validate model performance.

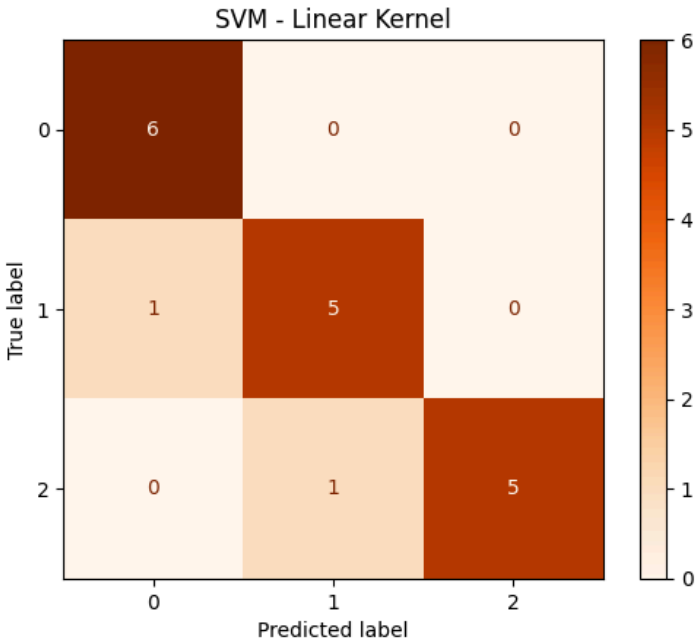
Screen image of Testing Labels (first 5 rows)

Testing Labels:

71	2
10	0
44	1
39	1
74	2
57	1

This is the testing data labels which 20% of the original data and no transformation are done. Checked for missing and null values. This is unknown to model and used to compare from the output given by model for testing data.

Confusion Matrix (in a color and pretty format such as ConfusionMatrixDisplay)



The confusion matrix for the SVM with a linear kernel shows strong performance, with only one misclassification each in classes 1 and 2. Class 0 was predicted perfectly, indicating the model's high accuracy in distinguishing that class.

Classification Report :

```
Accuracy: 0.8888888888888888
Classification Report:
              precision    recall  f1-score   support

           0       0.86      1.00      0.92         6
           1       0.83      0.83      0.83         6
           2       1.00      0.83      0.91         6

    accuracy                           0.89        18
   macro avg       0.90      0.89      0.89        18
weighted avg       0.90      0.89      0.89        18
```

This classification report indicates an overall accuracy of 88.9%, with balanced performance across all three classes. Class 2 achieved perfect precision, while class 0 had perfect recall, demonstrating the model’s strong ability to identify those categories accurately.

Next, explain which SVM did the best.

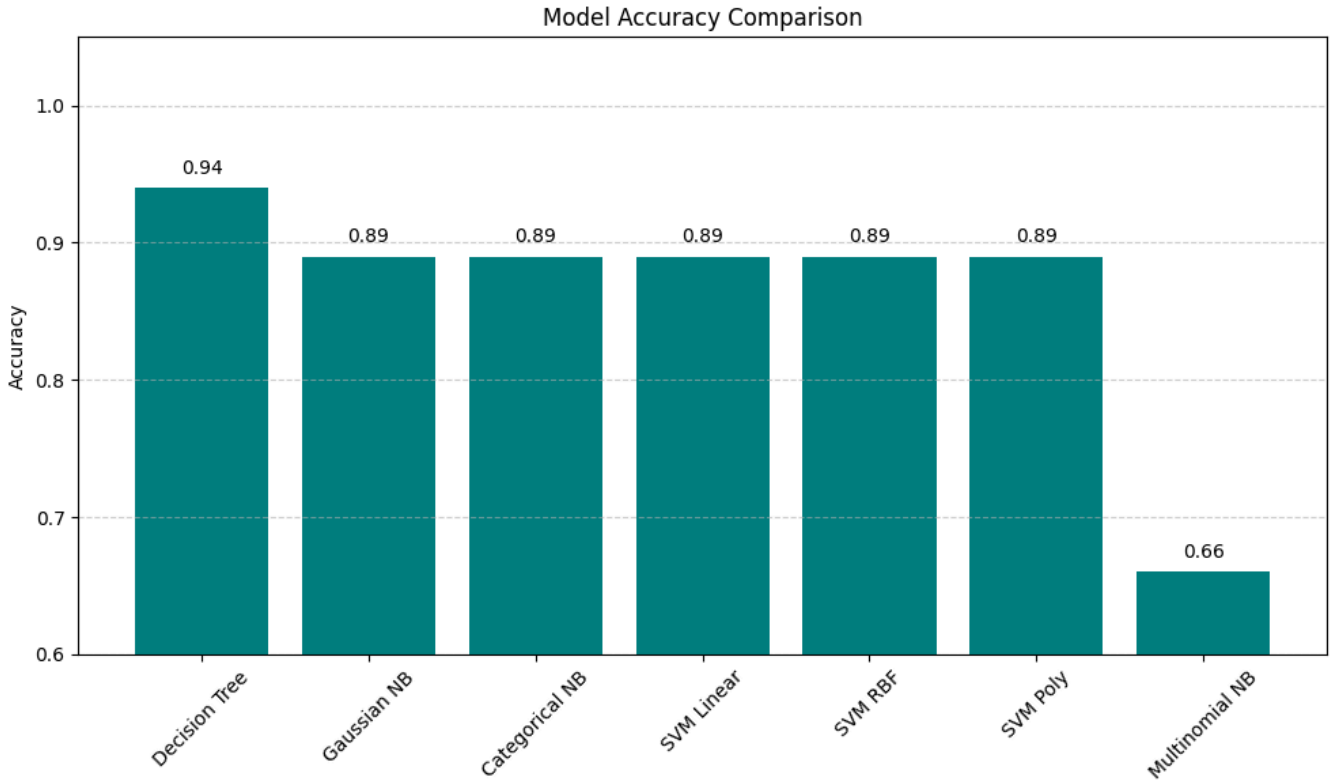
All the three SVM models gave the same accuracy of 89%. All the classification result, confusion matrices are identical. Both the linear model and complex model like polynomial and RBF are working the same which means the data is linear separable and also the dataset size is too small. Even this can affect the same performance. Data scaling can help in improving the performance of non linear models but not much on linear models. So, that would change the performance of models.

Points: 5

4) Model Comparison

You have now run many models of your dataset. Create a smart method to compare the models and then discuss and visualize which might have worked best and why this might be.

From the graph, we can see that decision tree has performed best with accuracy of 94%. While rest models like Gaussian NB, Categorical NB, and all SVM types gave an accuracy of 89%. Multinomial Naive Bayes gave the lowest accuracy of 66% which might be due to the data restriction like using only count based data. Decison tree has the capability to handle Mutliclass classification and it's easy to understand the model decision making from the decision tree.



Points: 25

5) SVM Problem Solving

For this question, you will reference the SVM slides specifically here:

https://gatesboltonanalytics.com/?page_id=304

While you will need to understand all the slides, please then go to slide 100. There you will see “Part 2 Solving for Lagrange Multipliers in SVMs”

From there, go to the next slide (101).

There, you will see that I have created three data points in 2D space called p1, p2, and p3. Then, for the next many slides – through slide 107 – SVM math is applied **by hand** to calculate the Lagrange multipliers (lambdas), the value of **w**, and the value of b. You can also see that a final linear model is created and illustrated. It’s a line that separates the black squares from the blue circle.

Your goal is to create 3 – 4 of your own points (3 is easier) and then replicate all the steps from slides 101 – 107 using your data. Show all your work, illustrate all steps (like the slides do), and show the final results.

YOU need to be the one to do this. Do this by hand – type in your work – etc. DO NOT use AI or anything that might look like you copied or borrowed your answer. I am not looking for your work to be fancy. I am looking for it to be yours.

Question - 5 : svm problem solving using slides

let's consider three support vectors

$P_1(1,0)$ $y_1 = -1$

$P_2(0,1)$ $y_2 = -1$

$P_3(1,1)$ $y_3 = +1$

→ Recalling the basics of lagrangian multiplier function

$$L = \frac{1}{2} w^T w - \sum_i \lambda_i (y_i (w^T x_i + b) - 1)$$

→ Taking the derivative wrt w and equating to zero

$$\frac{dL}{dw} = \frac{d}{dw} \left(\frac{1}{2} w^T w - \sum_i \lambda_i (y_i (w^T x_i + b) - 1) \right)$$

$$\frac{d}{dw} \left(\frac{1}{2} w^T w - \sum_i \lambda_i y_i w^T x_i - \sum_i \lambda_i y_i b - 1 \right) = 0$$

$$\frac{d}{dw} \left(\frac{1}{2} w^T w - \sum_i \lambda_i y_i w^T x_i \right) = 0$$

$$\frac{d}{dw} \left(\frac{1}{2} w^T w \right) - \frac{d}{dw} \sum_i \lambda_i y_i w^T x_i = 0$$

$$\frac{d}{dw} \left(\frac{1}{2} w^2 \right) - \sum_i \lambda_i y_i \frac{d}{dw} (w^T x_i) = 0$$

$$w - \sum_i \lambda_i y_i x_i = 0$$

$$\boxed{w = \sum_i \lambda_i y_i x_i} \quad \text{—— (1)}$$

→ Taking derivative wrt b and equating to zero

$$\frac{dL}{db} = \frac{d}{db} \left(\frac{1}{2} w^T w - \sum_i \lambda_i (y_i (w^T x_i + b) - 1) \right)$$

$$\frac{d}{db} \sum_i \lambda_i b y_i = 0$$

$$\boxed{\sum_i \lambda_i y_i = 0} \quad \text{—— (2)}$$

→ Differentiating wrt λ and equating to zero

$$\frac{dL}{d\lambda} = \frac{d}{d\lambda}\left(\frac{1}{2}w^Tw - \sum_i \lambda_i y_i w^Tx_i - \sum_i \lambda_i b y_i + \sum_i \lambda_i\right)$$

$$\frac{d}{d\lambda}\left(-\sum_i \lambda_i y_i w^Tx_i - \sum_i \lambda_i b y_i + \sum_i \lambda_i\right) = 0$$

$$-\sum_i y_i w^Tx_i - \sum_i b y_i + 1 = 0$$

$$\boxed{\sum_i (y_i(w^Tx_i + b) + 1) = 0} \quad \text{——— (3)}$$

plugging in w and b values in terms of λ to get a dual form of lagrangian function

$$\boxed{L = \sum_i \lambda_i - \frac{1}{2}\sum_i\sum_j \lambda_i \lambda_j y_i y_j (x_i^T x_j)} \quad \text{— (4)}$$

So, using equation 2

$$0 = \sum_i \lambda_i y_i$$

$$\lambda_1 y_1 + \lambda_2 y_2 + \lambda_3 y_3 = 0$$

Substituting the considered support vector values

$$\lambda_1(-1) + \lambda_2(-1) + \lambda_3(1) = 0$$

$$\boxed{\lambda_3 = \lambda_1 + \lambda_2} \quad \text{——— (5)}$$

$$L = \sum_i \lambda_i - \frac{1}{2}\sum_i\sum_j \lambda_i \lambda_j y_i y_j x_i^T x_j$$

Expanding the above:

$$L = \lambda_1 + \lambda_2 + \lambda_3 - \frac{1}{2}[\lambda_1\lambda_1 y_1 y_1 x_1^T x_1 + \lambda_1\lambda_2 y_1 y_2 x_1^T x_2 + \lambda_1\lambda_3 y_1 y_3 x_1^T x_3 +$$
$$\lambda_2\lambda_1 y_2 y_1 x_2^T x_1 + \lambda_2\lambda_2 y_2 y_2 x_2^T x_2 + \lambda_2\lambda_3 y_2 y_3 x_2^T x_3 +$$
$$\lambda_3\lambda_1 y_3 y_1 x_3^T x_1 + \lambda_3\lambda_2 y_3 y_2 x_3^T x_2 + \lambda_3\lambda_3 y_3 y_3 x_3^T x_3]$$

$P_1(1,0)$ $\quad x_1 \cdot x_1 = 1^2+0^2 = 1$ $\quad x_1 \cdot x_2 = 1.0+0.1 = 0$

$P_2(0,1)$ $\quad x_2 \cdot x_2 = 0^2+1^2 = 1$ $\quad x_1 \cdot x_3 = 1.1+0.1 = 1$

$P_3(1,1)$ $\quad x_3 \cdot x_3 = 1^2+1^2 = 2$ $\quad x_2 \cdot x_3 = 0.1+1.1 = 1$

$$L = \lambda_1+\lambda_2+\lambda_3 - \frac{1}{2}[\lambda_1\lambda_1(-1)(-1)(1) + \lambda_1\lambda_2(-1)(-1)(0) + \lambda_1\lambda_3(-1)(1)(1) + \lambda_2\lambda_1(-1)(-1)(0) + \lambda_2\lambda_2(-1)(-1)(1) + \lambda_2\lambda_3(-1)(1)(1) + \lambda_3\lambda_1(1)(-1)(1) + \lambda_3\lambda_2(1)(-1)(1) + \lambda_3\lambda_3(1)(1)(2)]$$

$$L = \lambda_1+\lambda_2+\lambda_3 - \frac{1}{2}[\lambda_1^2 + 0 + (-\lambda_1\lambda_3) + 0 + \lambda_2^2 - \lambda_2\lambda_3 - \lambda_3\lambda_1 - \lambda_2\lambda_3 + 2\lambda_3^2]$$

$$- \frac{1}{2}[\lambda_1^2+\lambda_2^2+2\lambda_3^2 - 2\lambda_3\lambda_1 - 2\lambda_2\lambda_3]$$

$$L = 2\lambda_1+2\lambda_2 - \frac{1}{2}[\lambda_1^2+\lambda_2^2+2(\lambda_1+\lambda_2)^2 - 2(\lambda_1+\lambda_2)\lambda_1 - 2\lambda_2(\lambda_1+\lambda_2)]$$

$$L = 2\lambda_1+2\lambda_2 - \frac{1}{2}[\lambda_1^2+\lambda_2^2+2(\lambda_1^2+\lambda_2^2+2\lambda_1\lambda_2) - 2\lambda_1^2 - 2\lambda_1\lambda_2 - 2\lambda_2\lambda_1 - 2\lambda_2^2]$$

$$L = 2\lambda_1+2\lambda_2 - \frac{1}{2}[\lambda_1^2+\lambda_2^2+2\lambda_1^2+2\lambda_2^2+4\lambda_1\lambda_2 - 2\lambda_1^2 - 2\lambda_1\lambda_2 - 2\lambda_1\lambda_2 - 2\lambda_2^2]$$

$$\boxed{L = 2\lambda_1+2\lambda_2 - \frac{1}{2}[\lambda_1^2+\lambda_2^2]} \quad \text{—— (6)}$$

Here we have to solve for $\lambda_1, \lambda_2, \lambda_3$

$$\frac{dL}{d\lambda_1} = \frac{d}{d\lambda_1}\left(2\lambda_1+2\lambda_2 - \frac{1}{2}\lambda_1^2 - \frac{1}{2}\lambda_2^2\right]$$

$$= 2+0 - \frac{1}{2}(2\lambda_1) - 0$$

$$\boxed{0 = 2-\lambda_1} \quad \text{—— (7)}$$

$$\frac{dL}{d\lambda_2} = \frac{d}{d\lambda_2}\left(2\lambda_1+2\lambda_2 - \frac{1}{2}\lambda_1^2 - \frac{1}{2}\lambda_2^2\right]$$

$$\boxed{0 = 2-\lambda_2} \quad \text{—— (8)}$$

$\lambda_1 = 2$

$\lambda_2 = 2$

$\lambda_3 = \lambda_1 + \lambda_2$

$= 2+2$

$= 4.$

$w = \sum_i \lambda_i y_i x_i$

$= \lambda_1 y_1 x_1 + \lambda_2 y_2 x_2 + \lambda_3 y_3 x_3$

$= 2 \cdot (-1)(1,0) + 2(-1)(0,1) + 4(1)(1,1)$

$= (-2,0) + (0,-2) + (4,4)$

$\boxed{w = (2,2).}$

b/ $y_i(w^T x_i + b) - 1 = 0$

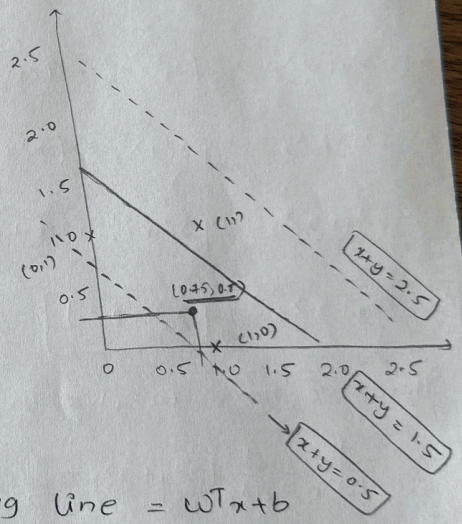
using support vector $p_1(1,0)$ $y_1 = -1$

$(-1)((2,0).(1,0) + b) = 1$

$-(2.1 + 2.0 + b) = 1$

$-2 - b = 1$

$\boxed{b = -3}$



final answers derived.

$\lambda_1 = 2$

$\lambda_2 = 2$

$\lambda_3 = 4$

$w = (2,2)$

$b = -3$

Our seperating line $= w^T x + b$

$2x_1 + 2x_2 - 3$

$2x_1 + 2x_2 = 3$

$\boxed{x_1 + x_2 = \frac{3}{2}}$

consider points $(0.75, 0.25) \rightarrow x_1 + x_2 = 0.75 + 0.25$

$= 1$

$1 < 1.5$

So, class -1

Part 2: Case Studies (30 points)

These case studies are open-ended questions and design showcase a real-world interview question which can be asked in interviews. The goal is to evaluate **your critical thinking, creativity, and problem-solving skills** as a future Machine Learning Engineer or Data Scientist.

You are expected to write the solution on your own using your own reasoning and knowledge. **If any form of AI is used for these questions you will get a direct zero (-40). In an interview it's your creativity and problem solving, not an AI model that earns the offer. Treat these case studies the same way.** We're not looking for perfection.

Your not-so-friendly TA, Pranit here,
Just so we are clear these cases studies are not copy-paste solution exercises. These cases studies are designed to test how you think because that's exactly what real interviews do.
And yes, I've already run these case studies through every AI tool on the internet with every prompt you can imagine. So, if your answers look similar like the ones I've seen before… I'll know. And just so we're crystal clear: you'll get a zero for the entire section

Case Study 1: The HoloEdu Experiment in Metaverse (15 points)

Scenario:

The year is 2050. A virtual reality learning platform called HoloEdu has revolutionized education by using VR headset and AI to make an immersive classroom in Metaverse. A major issue has occurred recently, many students are dropping the courses. Your team has been hired to build a model that predicts which students are at risk of dropping the course, so HoloEdu can provide personalized support early from preventing students from dropping a specific course.

You have access to data from previous courses:

- VR Session attendance (% of classes attended)
- Engagement score (based on eye movement and hand gestures)
- Frequency of interaction with virtual TA
- Total time spent in the office hours weekly
- Previous course completed
- Grades from previous courses
- Regions and Time zones associated with it
- Outcome: Completed, Dropped, Ongoing

Your Task:

1. Design a supervised ML pipeline to predict the course outcome
    a. Which algorithm(s) would you try first? Why?

    This comes under a classification problem with label Outcome with classes Completed, Dropped or Ongoing. I'll first go with Decision Tree. The decision tree is easy to implement and interpretable.We can use tree visualization to understand how decision making is happening and the decision tree is good enough to handle multiclass class and handles both numerical and categorical data. One issue with this is it might overfit. So next I'll choose to go with Random Forest which is a group of decision trees which reduce the issue of over fitting.

b. In the beginning would you treat this as a multiclass problem or binary problem

In the first go, I will choose to go with a multiclass problem. Each class has its own importance. Like if the student is in an ongoing class, we need a certain type of guidance to make sure he or she doesn't drop class and complete the course. But if the requirement is to target only ones which are at risk of dropping then I would take this as a binary problem and group completed and ongoing to one class and dropped to another class.This decision making is based on the requirements that we get and priority at that moment.

c. Any concerns about class imbalance? If yes, how would you resolve them?

Class imbalance impacts a lot on model performance. If there is a majority class then the model might favour this and ignore the minority one. There are many methods to handle class imbalance.Like under sampling, over sampling (SMOTE) or using inbuilt function in decision tree like class_weight='balanced'

2. Three ways HoloEdu can use your model in real-time scenarios, considering all the ethical concerns.

   Model can predict the students who are at risk of dropping students. Using this data, We can make more customized options to students which can help them to continue in course. We can provide more TA assistance and monitor the grades and provide more frequent updates and suggestions. Students should be given all the details on how the student data and performance data will be used in making the platform more customized.

3. How would you evaluate your model's performance and its usefulness in the Metaverse context? I will use metrics like accuracy, precision, recall, and F1-score to evaluate model performance on classifying the outcome. This evaluates the model performance technically, but in the ideal case we have considered the real time outcomes and reviews. User feedback and how is the student dropout rate.

Case Study 2: Clustering Emotions from Audio Footprints: VibeTune (15 points)

Scenario:

VibeTune is a mental health startup and designing headphones that tracks a user's emotions based on the music they listen to and the environment they're in currently.

Goal of VibeTune: Build an ML model that detects emotional state by clustering user based on behavioral and ambient sound data

You have access to:

- Average tempo and mood rating of daily music (0-10 scale)
- Average background noise level (in dB)
- Time of day most music is played
- Top genre + number of songs played in that genre
- Number of daily songs skipped
- Mobile application interaction frequency
- Ambient sound exposure from device sensors

Your Task:

1. Create a clustering-based ML solution to uncover user emotional segments (for example: energetic, calm, anxious, depressing)

a. Which algorithm would you use?
I will choose K - means clustering. It is very efficient to cluster the user groups based on emotional segments. It's easy to implement and fast which can make VbeTunes more robust and scalable

b. Would you apply any dimensionality reduction technique? Why or why not?
Dimensional reduction helps in reducing the dataset size without losing the relations which can help k-means to improve its efficiency. Out of all reduction techniques, I'll choose PCA as this keeps only the combinations of features that explain most of the differences between users. This helps in visualizing the data in 2D or 3D.

c. What would you use to select the most important features?
With dimensionality reduction using PCA, It gives us which features are contributing to the principal components. I'll prefer to use these as the reduced data is used in model training.

2. How can VibeTune interpret clusters to improve the emotional wellness support?
Based on the clustering the users into emotional states like active, low mood etc. Vibetunes can create customized playlists for users and provide notifications accordingly. This can help users balance their emotional mood and this helps users feel more supported and enjoy the time while using the application.
3. How would you evaluate and validate cluster quality with no labels available?
For K-means clustering , will use the Silhouette Score method to evaluate the cluster formation. Also, I would visually plot to see how well the clusters are formed. This is the technical evaluation. But as this is a real time application, we should see how users are responding to the customized options playlists and user activity on the application.

Part 3: Extra Credit (+15 points each)

PCA & Kmeans:

Sheet link:https://docs.google.com/spreadsheets/d/13vFXC-hm1WS7gIhdFDplVO6rgIRlxSTRP5UzZDV5Tvc/edit?usp=sharing

1. Is this dataset a record dataset? Yes, this is a record dataset.
    a. If so, list an observation and a variable name based on this dataset. If not, please describe the type of this dataset. From the dataset, considering the last record patient id : 15. The corresponding record represents the patient data like age, height, weight etc. The dataset has 15 records with 9 variables (Patient ID, Gender, Age, Height(cm), Weight(kg), Exercise, Frequency, Hypertension, Hypoglycemia, Disease History)
2. Prepare this data for PCA and Clustering and do not normalize yet.
    (a) Please give the complete name of the columns where PCA can be performed.
    PCA works only with numerical data. From the dataset, numerical columns are Age, Height, Weight, Hypertension and Hypoglycemia.
    (b) Explain why unselected columns are not suitable for PCA processing.
    The unselected columns are not suitable for PCA processing because PCA only supports numerical values. Gender column is a categorical column. Patient ID is just an identifier that won't play much role in model performance. Disease History, Exercise frequency are frequency columns not numerical data.
    (c) Include an image of your cleaned data.

Cleaned Data for PCA (No Normalization)

Age	Height(cm)	Weight(kg)	Hypertension	Hypoglycemia
35.0	175.0	70.0	130.0	5.2
52.0	162.0	68.0	145.0	6.8
28.0	180.0	85.0	125.0	5.5
45.0	170.0	90.0	138.0	6.2
67.0	158.0	75.0	160.0	7.1
39.0	165.0	62.0	118.0	4.9
32.0	177.0	65.0	120.0	4.8
50.0	160.0	70.0	142.0	6.5
29.0	178.0	78.0	126.0	5.1
41.0	172.0	83.0	140.0	6.0
60.0	157.0	68.0	155.0	7.0
34.0	176.0	72.0	128.0	5.4
45.0	164.0	65.0	135.0	6.3
38.0	179.0	80.0	133.0	5.7
55.0	161.0	69.0	148.0	6.6

3. Normalize your cleaned data using Sklearn StandardScaler. Include an image of the scaled data.

Normalized Data for PCA using StandardScaler

Age	Height(cm)	Weight(kg)	Hypertension	Hypoglycemia
-0.7448926478481193	0.7579649676522985	-0.420331351709198	-0.5204879673076104	-0.9926553698546854
0.7746883537620436	-0.8662456773169136	-0.6725301627347171	0.7387571148882234	1.1536265109122035
-1.3706024720405392	1.382661369563534	1.4711597309821958	-0.9402363280395549	-0.5902275172108938
0.1489785295696236	0.13326856574106308	2.1016567585459938	0.151109409863501	0.34877080562462043
2.1154951198888658	-1.366002798845902	0.2101656758545999	1.9980021970840571	1.556054363555995
-0.38734417688102213	-0.4914278361701723	-1.4291265958112747	-1.5278840330642773	-1.3950832224984768
-1.0130540010734421	1.0078435284167926	-1.0508283792729958	-1.3599846887714995	-1.529225840046408
0.595914118278495	-1.1161242380814078	-0.420331351709198	0.4869080984490567	0.7511986582684119
-1.281215354298765	1.1327828087990397	0.5884638923928787	-0.856286655893166	-1.1267979874026166
-0.20856994139747354	0.38314712650555727	1.2189609199566764	0.31900875415627883	0.08048557052875902
1.4897852956962379	-1.490942079228149	-0.6725301627347171	1.5782538363521126	1.4219117460080648
-0.8342797655898936	0.8829042480345456	-0.16813254068367883	-0.6883873116003881	-0.724370134758824
0.1489785295696236	-0.6163671165524194	-1.0508283792729958	-0.10073960657566575	0.4829134231725505
-0.4767312946227964	1.2577220891812868	0.8406627034183978	-0.2686389508684436	-0.32194228211503245
1.0428497069873666	-0.9911849576991607	-0.5464307572219576	0.9906061313273902	0.885341275816342

4. Perform PCA on the normalized data using n_components=3.
   (a) Include a screen image of the PCA-reduced data.

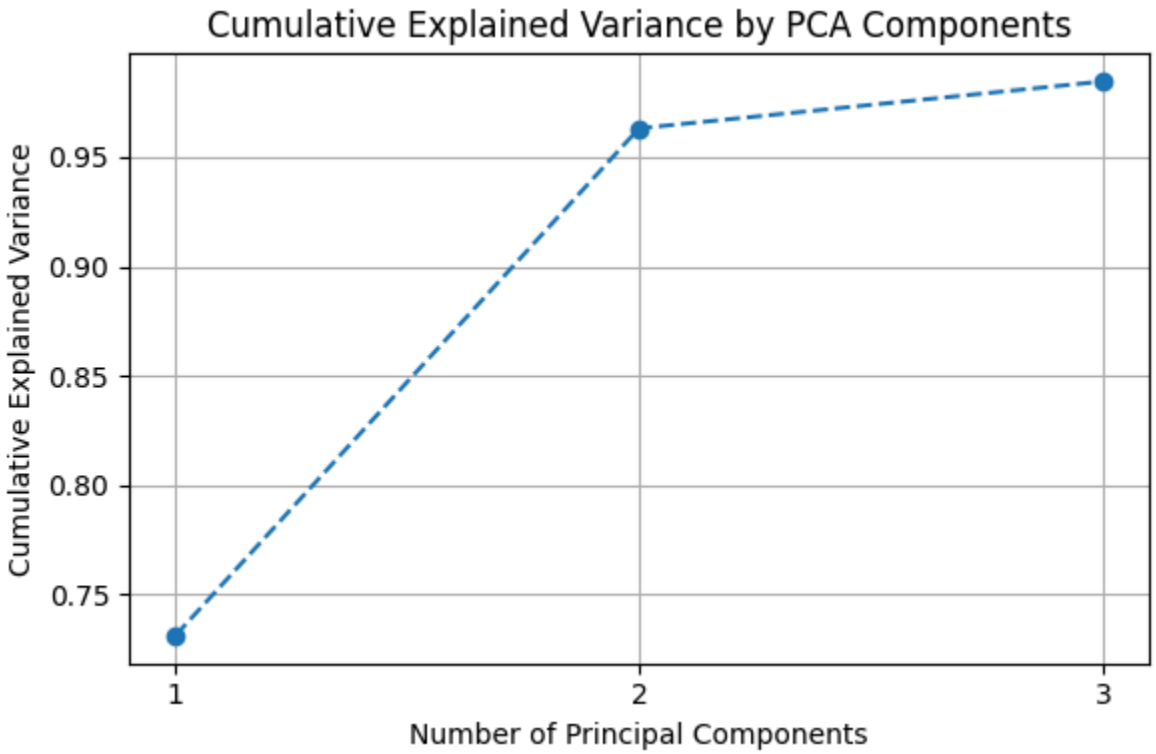
PCA-Reduced Data (3 Components)

PC1	PC2	PC3
-1.45557	-0.51285	0.3312
1.82601	-0.41064	0.18092
-2.28649	1.35556	-0.09699
0.03846	2.04327	-0.64001
3.4861	0.63487	0.08233
-1.25723	-2.11294	-0.78218
-2.32819	-1.33912	0.256
1.50398	-0.40326	-0.29047
-2.25173	0.39161	0.0296
-0.22414	1.30403	-0.02652
3.04347	-0.32139	0.1624
-1.53808	-0.23221	0.28299
0.67576	-1.03505	0.0081
-1.23451	0.97007	0.39951
2.00217	-0.33195	0.10313

(b) Include the ordered eigenvalues, its eigenvectors and explained variance ratio by each component.

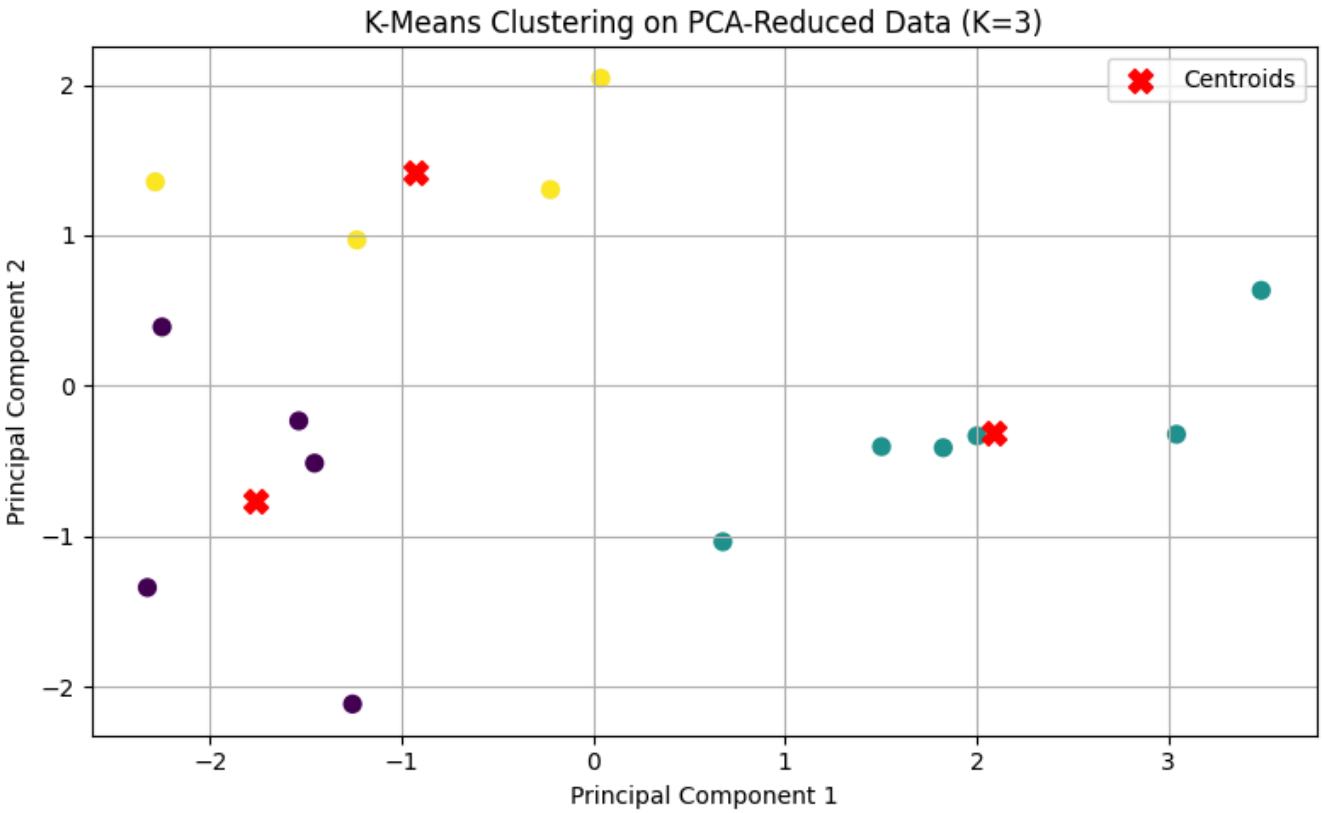
```
Eigenvalues:
[3.91690747 1.24293796 0.11487789]

Eigenvectors (Principal Components):
PC1: [ 0.51573539 -0.48318994 -0.10547508  0.49260208  0.49675214]
PC2: [-0.01341545  0.27524702  0.90050012  0.25099213  0.22396781]
PC3: [-0.08238733  0.71840424 -0.41108772  0.52693669  0.17450583]

Explained Variance Ratio:
[0.73115606 0.23201509 0.02144387]
```

(c) Include a plot to show the Cumulative Explained Variance by Principal Components.



5. Perform k-means on the PCA-reduced data. K=3. What are the centroids? Include the result of a 2D visualization.

```
Centroids (in PC1, PC2, PC3 space):
[[-1.76615954 -0.76110116  0.02352121]
 [ 2.08958062 -0.31123496  0.04106766]
 [-0.9266715   1.41822889 -0.09100301]]
```



6. Based on the 2D PCA plot of the clustered data, summarize the results in a way that a non-technical audience can understand. Use the eigenvectors to explain feature contributions, assign possible labels to clusters, and provide a brief conclusion on what distinguishes them. *(Hint: Consider how the eigenvectors relate to the original columns used for analysis to interpret the meaning of each cluster.)*

   From the 2D PCA plot, three distinct clusters are formed using features like age, height, weight, blood pressure, and blood sugar. The first principal component is strongly contributed by age, hypertension, and hypoglycemia, likely representing older individuals with higher health risks. The second component is contributed by height and weight, indicating individuals with a heavier body. The third cluster, positioned between the others, represents individuals with more balanced values across all features—neither too young nor too old, and with average health indicators. These clusters help identify meaningful health patterns for more personalized care.

ARM:

Link:https://drive.google.com/file/d/1ullL6vMqhT0QAJu-Bwic5oXaba-rzctD/view?usp=sharing
Use the attached dataset to perform ARM

Required Parameters
Support = 0.01
Confidence = 0.01
minlen = 2 (minimum rule length)

Create a Word Document or PDF with the following

1. Print and paste a screenshot of the top 10 rules for support

```
========== TOP 10 RULES BY SUPPORT ==========
     lhs                                                          rhs                                                          support confidence  coverage     lift count
[1]  {Eligibility unknown as battery range has not been researched} => {Battery Electric Vehicle (BEV)}                         0.4634635  1.0000000 0.4634635 1.2677665   463
[2]  {Battery Electric Vehicle (BEV)}                             => {Eligibility unknown as battery range has not been researched} 0.4634635  0.5875635 0.7887888 1.2677665   463
[3]  {Clean Alternative Fuel Vehicle Eligible}                    => {Battery Electric Vehicle (BEV)}                         0.3253253  0.7629108 0.4264264 0.9671928   325
[4]  {Battery Electric Vehicle (BEV)}                             => {Clean Alternative Fuel Vehicle Eligible}                0.3253253  0.4124365 0.7887888 0.9671928   325
[5]  {TESLA-MODEL Y}                                              => {Battery Electric Vehicle (BEV)}                         0.1831832  1.0000000 0.1831832 1.2677665   183
[6]  {Battery Electric Vehicle (BEV)}                             => {TESLA-MODEL Y}                                          0.1831832  0.2322335 0.7887888 1.2677665   183
[7]  {TESLA-MODEL 3}                                              => {Battery Electric Vehicle (BEV)}                         0.1741742  1.0000000 0.1741742 1.2677665   174
[8]  {Battery Electric Vehicle (BEV)}                             => {TESLA-MODEL 3}                                          0.1741742  0.2208122 0.7887888 1.2677665   174
[9]  {TESLA-MODEL Y}                                              => {Eligibility unknown as battery range has not been researched} 0.1681682  0.9180328 0.1831832 1.9808094   168
[10] {Eligibility unknown as battery range has not been researched} => {TESLA-MODEL Y}                                         0.1681682  0.3628510 0.4634635 1.9808094   168
```

2. Print and paste a screenshot of the top 10 rules for confidence

```
========== TOP 10 RULES BY CONFIDENCE ==========
     lhs                     rhs                                                          support confidence   coverage     lift count
[1]  {BMW-I3}             => {Clean Alternative Fuel Vehicle Eligible}                    0.01001001          1 0.01001001 2.345070    10
[2]  {FORD-MUSTANG MACH-E} => {Eligibility unknown as battery range has not been researched} 0.01201201          1 0.01201201 2.157667    12
[3]  {FORD-MUSTANG MACH-E} => {Battery Electric Vehicle (BEV)}                             0.01201201          1 0.01201201 1.267766    12
[4]  {VOLVO-XC90}          => {Plug-in Hybrid Electric Vehicle (PHEV)}                     0.01201201          1 0.01201201 4.734597    12
[5]  {TOYOTA-PRIUS PRIME}  => {Plug-in Hybrid Electric Vehicle (PHEV)}                     0.01301301          1 0.01301301 4.734597    13
[6]  {HYUNDAI-IONIQ 5}     => {Eligibility unknown as battery range has not been researched} 0.01501502          1 0.01501502 2.157667    15
[7]  {HYUNDAI-IONIQ 5}     => {Battery Electric Vehicle (BEV)}                             0.01501502          1 0.01501502 1.267766    15
[8]  {TOYOTA-RAV4 PRIME}   => {Plug-in Hybrid Electric Vehicle (PHEV)}                     0.01601602          1 0.01601602 4.734597    16
[9]  {TOYOTA-RAV4 PRIME}   => {Clean Alternative Fuel Vehicle Eligible}                    0.01601602          1 0.01601602 2.345070    16
[10] {KIA-EV6}             => {Eligibility unknown as battery range has not been researched} 0.01701702          1 0.01701702 2.157667    17
```

3. Print and paste a screenshot of the top 10 rules for lift

```
========== TOP 10 RULES BY LIFT ==========
     lhs                                    rhs                                          support confidence   coverage     lift count
[1]  {Clean Alternative Fuel Vehicle Eligible,
      Plug-in Hybrid Electric Vehicle (PHEV)} => {TOYOTA-RAV4 PRIME}                    0.01601602  0.1584158 0.10110110 9.891089    16
[2]  {Clean Alternative Fuel Vehicle Eligible,
      Plug-in Hybrid Electric Vehicle (PHEV)} => {CHEVROLET-VOLT}                       0.02302302  0.2277228 0.10110110 9.891089    23
[3]  {JEEP-WRANGLER}                         => {Not eligible due to low battery range} 0.02402402  1.0000000 0.02402402 9.081818    24
[4]  {JEEP-WRANGLER,
      Plug-in Hybrid Electric Vehicle (PHEV)} => {Not eligible due to low battery range} 0.02402402  1.0000000 0.02402402 9.081818    24
[5]  {Not eligible due to low battery range} => {JEEP-WRANGLER}                        0.02402402  0.2181818 0.11011011 9.081818    24
[6]  {Not eligible due to low battery range,
      Plug-in Hybrid Electric Vehicle (PHEV)} => {JEEP-WRANGLER}                        0.02402402  0.2181818 0.11011011 9.081818    24
[7]  {Clean Alternative Fuel Vehicle Eligible,
      Plug-in Hybrid Electric Vehicle (PHEV)} => {BMW-X5}                               0.01601602  0.1584158 0.10110110 7.912871    16
[8]  {Plug-in Hybrid Electric Vehicle (PHEV),
      Seattle}                               => {Not eligible due to low battery range} 0.01601602  0.5517241 0.02902903 5.010658    16
[9]  {Plug-in Hybrid Electric Vehicle (PHEV)} => {Not eligible due to low battery range} 0.11011011  0.5213270 0.21121121 4.734597   110
[10] {VOLVO-XC90}                            => {Plug-in Hybrid Electric Vehicle (PHEV)} 0.01201201  1.0000000 0.01201201 4.734597    12
```

4. Keep the same values mentioned above and set default values for lhs and rhs “NISSAN-LEAF”; sort the rules and paste the top 5 rules with the highest lift values

```
========== TOP 5 RULES WITH 'NISSAN-LEAF' BY LIFT ==========
    lhs                                    rhs                                          support confidence   coverage     lift count
[1] {Battery Electric Vehicle (BEV),
     Clean Alternative Fuel Vehicle Eligible,
     Seattle}                           => {NISSAN-LEAF}                             0.01301301  0.2765957 0.04704705 2.971174    13
[2] {Battery Electric Vehicle (BEV),
     Clean Alternative Fuel Vehicle Eligible} => {NISSAN-LEAF}                       0.07307307  0.2246154 0.32532533 2.412804    73
[3] {Clean Alternative Fuel Vehicle Eligible,
     Seattle}                           => {NISSAN-LEAF}                             0.01301301  0.2166667 0.06006006 2.327419    13
[4] {Clean Alternative Fuel Vehicle Eligible} => {NISSAN-LEAF}                       0.07307307  0.1713615 0.42642643 1.840754    73
[5] {NISSAN-LEAF}                       => {Clean Alternative Fuel Vehicle Eligible} 0.07307307  0.7849462 0.09309309 1.840754    73
```

1. One of the rules generated will be {Clean Alternative Fuel Vehicle Eligible} => {NISSAN-LEAF}. What is the support, confidence, lift, and count?

```
    lhs                                          rhs              support confidence   coverage     lift count
[1] {Clean Alternative Fuel Vehicle Eligible}  => {NISSAN-LEAF} 0.07307307  0.1713615 0.42642643 1.840754    73
[2] {Clean Alternative Fuel Vehicle Eligible,
     Seattle}                                  => {NISSAN-LEAF} 0.01301301  0.2166667 0.06006006 2.327419    13
[3] {Battery Electric Vehicle (BEV),
     Clean Alternative Fuel Vehicle Eligible}  => {NISSAN-LEAF} 0.07307307  0.2246154 0.32532533 2.412804    73
[4] {Battery Electric Vehicle (BEV),
     Clean Alternative Fuel Vehicle Eligible,
     Seattle}                                  => {NISSAN-LEAF} 0.01301301  0.2765957 0.04704705 2.971174    13
```

Support - 0.073
Confidence - 0.1713615
Lift - 1.840754
Count - 73

5. Keep the same values mentioned above and set default values for rhs and lhs “Eligibility unknown as battery range has not been researched”; sort the rules and paste the top 5 rules with the highest lift values

```
    lhs                                                          rhs                support confidence   coverage     lift count
[1] {Bothell,
     Eligibility unknown as battery range has not been researched}  => {TESLA-MODEL Y} 0.01001001  0.5263158 0.01901902 2.873167    10
[2] {Battery Electric Vehicle (BEV),
     Bothell,
     Eligibility unknown as battery range has not been researched}  => {TESLA-MODEL Y} 0.01001001  0.5263158 0.01901902 2.873167    10
[3] {Bellevue,
     Eligibility unknown as battery range has not been researched}  => {TESLA-MODEL Y} 0.03603604  0.5070423 0.07107107 2.767952    36
[4] {Battery Electric Vehicle (BEV),
     Bellevue,
     Eligibility unknown as battery range has not been researched}  => {TESLA-MODEL Y} 0.03603604  0.5070423 0.07107107 2.767952    36
[5] {Eligibility unknown as battery range has not been researched,
     Kent}                                                       => {TESLA-MODEL Y} 0.01101101  0.5000000 0.02202202 2.729508    11
```

1. One of the rules generated {Eligibility unknown as battery range has not been researched} => {Battery Electric Vehicle (BEV)}. What you can infer from this rule is that there is a positive or negative correlation. Explain in 2-3 sentences.

```
   lhs                                                                  rhs
support confidence   coverage     lift count
[1]  {Eligibility unknown as battery range has not been researched}  => {Battery Electric
Vehicle (BEV)} 0.46346346          1 0.46346346 1.267766   463
[2]  {Eligibility unknown as battery range has not been researched,
      FORD-MUSTANG MACH-E}                                           => {Battery Electric
Vehicle (BEV)} 0.01201201          1 0.01201201 1.267766    12
[3]  {Eligibility unknown as battery range has not been researched,
      HYUNDAI-IONIQ 5}                                               => {Battery Electric
Vehicle (BEV)} 0.01501502          1 0.01501502 1.267766    15
[4]  {Auburn,
      Eligibility unknown as battery range has not been researched}  => {Battery Electric
Vehicle (BEV)} 0.01101101          1 0.01101101 1.267766    11
[5]  {Eligibility unknown as battery range has not been researched,
      KIA-EV6}                                                       => {Battery Electric
Vehicle (BEV)} 0.01701702          1 0.01701702 1.267766    17
[6]  {Eligibility unknown as battery range has not been researched,
      VOLKSWAGEN-ID.4}                                               => {Battery Electric
Vehicle (BEV)} 0.01801802          1 0.01801802 1.267766    18
[7]  {Eligibility unknown as battery range has not been researched,
      Sammamish}                                                     => {Battery Electric
Vehicle (BEV)} 0.01501502          1 0.01501502 1.267766    15
[8]  {Eligibility unknown as battery range has not been researched,
```

```
      Renton}                                                                 => {Battery Electric
Vehicle (BEV)} 0.01601602          1 0.01601602 1.267766    16
[9]  {Eligibility unknown as battery range has not been researched,
      Kent}                                                                   => {Battery Electric
Vehicle (BEV)} 0.02202202          1 0.02202202 1.267766    22
[10] {Eligibility unknown as battery range has not been researched,
      TESLA-MODEL X}                                                          => {Battery Electric
Vehicle (BEV)} 0.01201201          1 0.01201201 1.267766    12
[11] {Eligibility unknown as battery range has not been researched,
      Olympia}                                                                => {Battery Electric
Vehicle (BEV)} 0.01101101          1 0.01101101 1.267766    11
[12] {Bothell,
      Eligibility unknown as battery range has not been researched}           => {Battery Electric
Vehicle (BEV)} 0.01901902          1 0.01901902 1.267766    19
[13] {Eligibility unknown as battery range has not been researched,
      Redmond}                                                                => {Battery Electric
Vehicle (BEV)} 0.02402402          1 0.02402402 1.267766    24
[14] {Eligibility unknown as battery range has not been researched,
      TESLA-MODEL S}                                                          => {Battery Electric
Vehicle (BEV)} 0.01001001          1 0.01001001 1.267766    10
[15] {Eligibility unknown as battery range has not been researched,
      Vancouver}                                                              => {Battery Electric
Vehicle (BEV)} 0.03103103          1 0.03103103 1.267766    31
[16] {Eligibility unknown as battery range has not been researched,
      Kirkland}                                                               => {Battery Electric
Vehicle (BEV)} 0.05005005          1 0.05005005 1.267766    50
[17] {Eligibility unknown as battery range has not been researched,
      NISSAN-LEAF}                                                            => {Battery Electric
Vehicle (BEV)} 0.02002002          1 0.02002002 1.267766    20
[18] {Bellevue,
      Eligibility unknown as battery range has not been researched}           => {Battery Electric
Vehicle (BEV)} 0.07107107          1 0.07107107 1.267766    71
[19] {Eligibility unknown as battery range has not been researched,
      Seattle}                                                                => {Battery Electric
Vehicle (BEV)} 0.08108108          1 0.08108108 1.267766    81
[20] {Eligibility unknown as battery range has not been researched,
      TESLA-MODEL 3}                                                          => {Battery Electric
Vehicle (BEV)} 0.06606607          1 0.06606607 1.267766    66
[21] {Eligibility unknown as battery range has not been researched,
      TESLA-MODEL Y}                                                          => {Battery Electric
Vehicle (BEV)} 0.16816817          1 0.16816817 1.267766   168
[22] {Eligibility unknown as battery range has not been researched,
      Kent,
      TESLA-MODEL Y}                                                          => {Battery Electric
Vehicle (BEV)} 0.01101101          1 0.01101101 1.267766    11
[23] {Bothell,
      Eligibility unknown as battery range has not been researched,
      TESLA-MODEL Y}                                                          => {Battery Electric
Vehicle (BEV)} 0.01001001          1 0.01001001 1.267766    10
[24] {Eligibility unknown as battery range has not been researched,
      Redmond,
      TESLA-MODEL Y}                                                          => {Battery Electric
Vehicle (BEV)} 0.01001001          1 0.01001001 1.267766    10
[25] {Eligibility unknown as battery range has not been researched,
      Kirkland,
      TESLA-MODEL Y}                                                          => {Battery Electric
Vehicle (BEV)} 0.01601602          1 0.01601602 1.267766    16
[26] {Bellevue,
```

```
      Eligibility unknown as battery range has not been researched,
      TESLA-MODEL Y}                                                  => {Battery Electric
Vehicle (BEV)} 0.03603604          1 0.03603604 1.267766    36
[27] {Eligibility unknown as battery range has not been researched,
      Seattle,
      TESLA-MODEL 3}                                                  => {Battery Electric
Vehicle (BEV)} 0.01001001          1 0.01001001 1.267766    10
[28] {Eligibility unknown as battery range has not been researched,
      Seattle,
      TESLA-MODEL Y}                                                  => {Battery Electric
Vehicle (BEV)} 0.02402402          1 0.02402402 1.267766    24
```

The rule {Eligibility unknown as battery range has not been researched} => {Battery Electric Vehicle (BEV)} indicates a strong positive correlation, as evidenced by a lift of 1.2678 and a confidence of 1.0. This means that every time the eligibility is unknown, it consistently leads to a Battery Electric Vehicle in the dataset. The support of 0.4635 further shows that this pattern appears in nearly 46% of all transactions, making it a highly relevant and reliable rule.