FORE SCHOOL OF MANAGEMENT

MLM-2

PROJECT 3 REPORT

CLASSIFICATION OF CONSUMER DATA USING CROSS VALIDATION & ENSEMBLE METHODS

Submitted To: **Prof. Amarnath Mitra**

Submitted By:

Adhyatik (311063)

OBJECTIVES

1.1.) Classification of Consumer Data into Segments using Cross-Validation.

1.2.) Classification of Consumer Data into Segments using Ensemble Method.

1.3.) Determination of an Appropriate Classification Model.

1.4.) Identification of Significant Variables and their Thresholds for Classification.

Base Model (Decision Tree)

Starting with the confusion matrix:

Cluster 0:
- True Positives (TP): 9899 instances were correctly classified as belonging to cluster 0.
- False Positives (FP): 0 instances were incorrectly classified as belonging to cluster 0.
- True Negatives (TN): 20159 instances were correctly classified as not belonging to cluster 0.
- False Negatives (FN): 0 instances that should have been classified as cluster 0 were missed.

Cluster 1:
- True Positives (TP): 9833 instances were correctly classified as belonging to cluster 1.
- False Positives (FP): 0 instances were incorrectly classified as belonging to cluster 1.
- True Negatives (TN): 20225 instances were correctly classified as not belonging to cluster 1.
- False Negatives (FN): 0 instances that should have been classified as cluster 1 were missed.

Cluster 2:
- True Positives (TP): 10326 instances were correctly classified as belonging to cluster 2.
- False Positives (FP): 0 instances were incorrectly classified as belonging to cluster 2.
- True Negatives (TN): 19732 instances were correctly classified as not belonging to cluster 2.
- False Negatives (FN): 0 instances that should have been classified as cluster 2 were missed.

Now, looking at the accuracy metrics:

Cluster 0:
- Recall (TP / (TP + FN)): 1, indicating perfect recall (no false negatives).
- Precision (TP / (TP + FP)): 1, indicating perfect precision (no false positives).
- Sensitivity (same as Recall): 1, perfect sensitivity.
- Specificity (TN / (TN + FP)): 1, perfect specificity (no false positives).
- F-measure (2 * (Precision * Recall) / (Precision + Recall)): 1, perfect F-measure, balancing precision and recall.

- Accuracy ((TP + TN) / (TP + TN + FP + FN)): 1, perfect accuracy, all predictions are correct.

Similar observations can be made for clusters 1 and 2, with perfect scores across all evaluation metrics.

Overall:

- The overall accuracy of the model is 1, indicating perfect accuracy in classifying instances into the three clusters.
- The overall Cohen's kappa is 1, suggesting a perfect agreement between the predicted and actual classes.

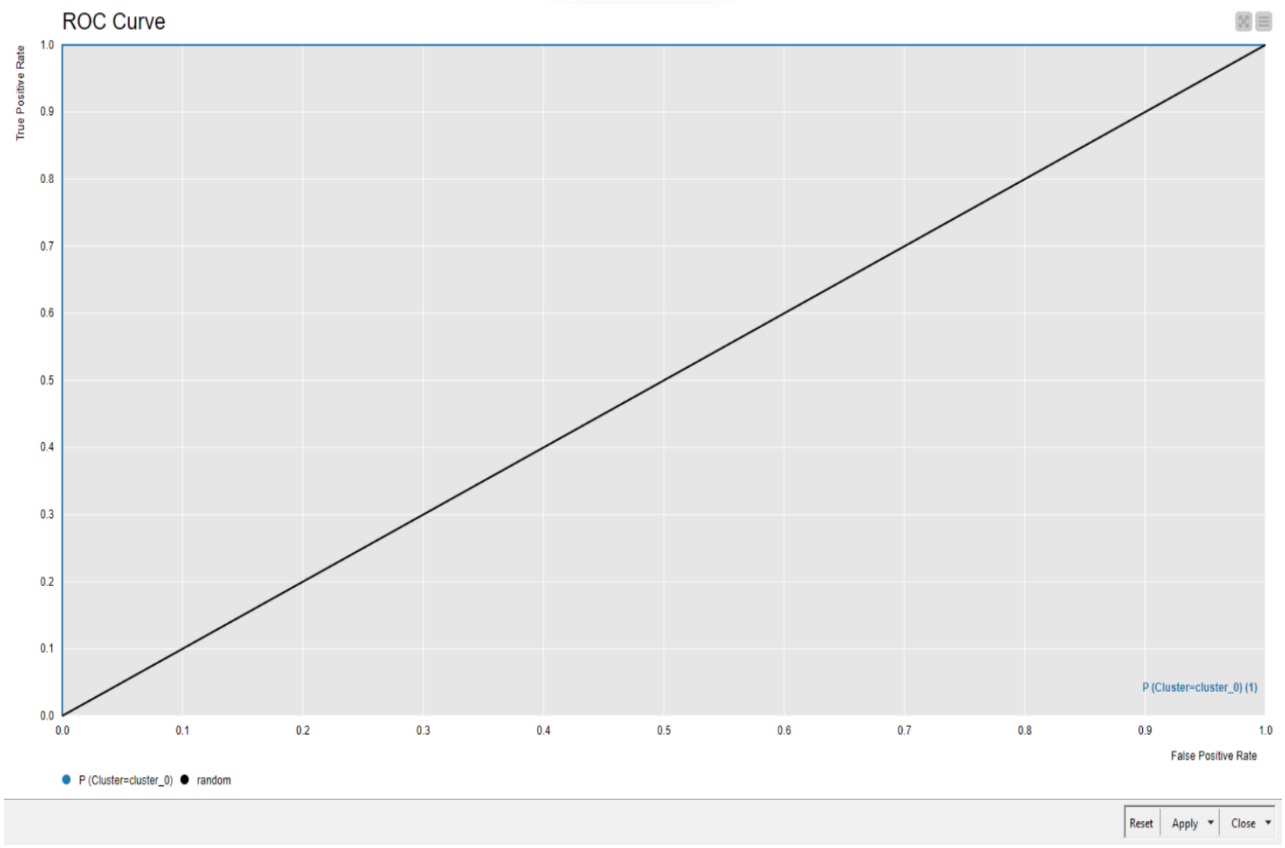
These exceptional results, with perfect scores across all evaluation metrics, indicate that the decision tree model has performed remarkably well in classifying the instances into the three clusters without any misclassifications. The model has learned the underlying patterns in the data exceptionally well and can predict the cluster memberships with high confidence and accuracy.

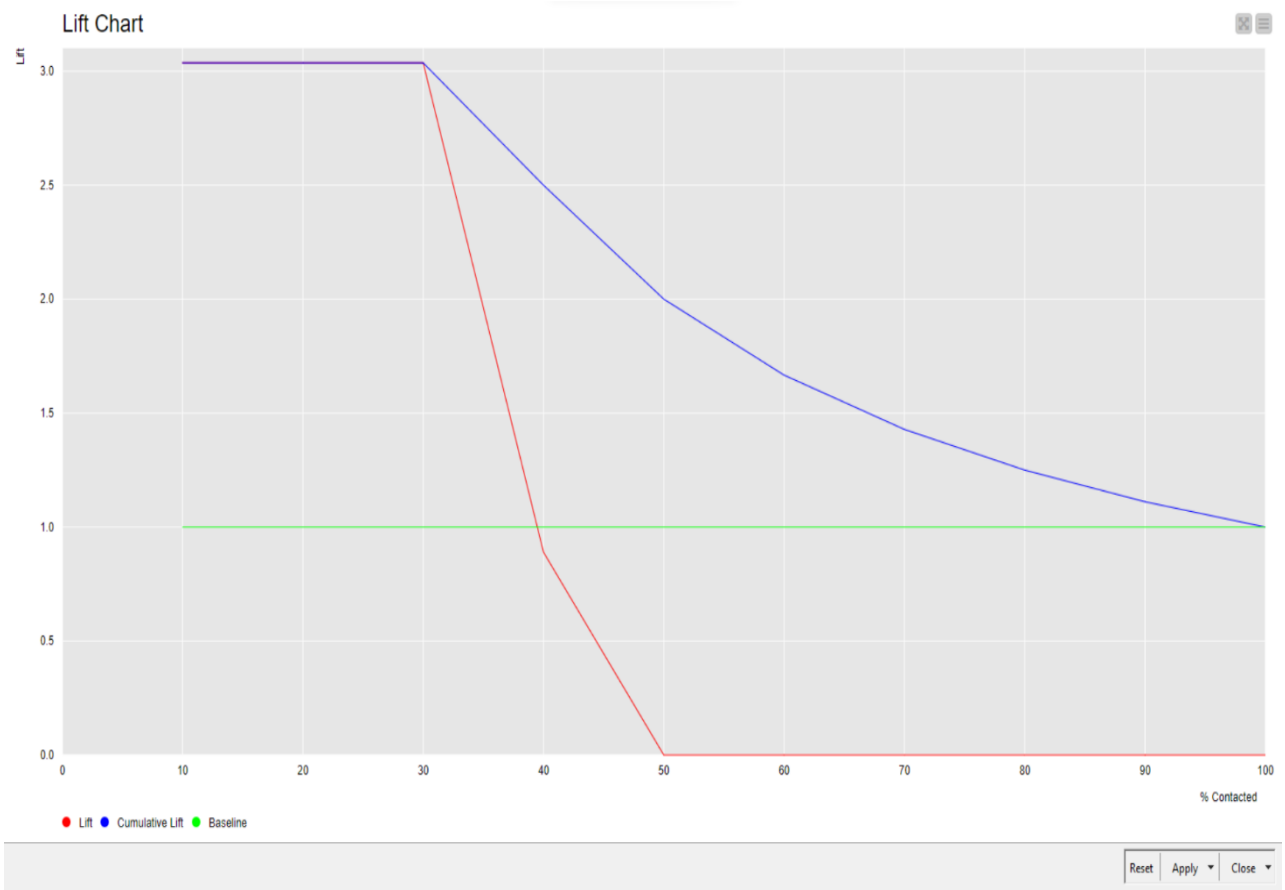
Such perfect performance is rarely seen in real-world scenarios, as there are typically some instances that are misclassified due to noise, outliers, or complex data patterns. However, in this particular case, the decision tree model has achieved an outstanding level of performance, potentially due to the specific characteristics of the dataset, the choice of features, and the way the decision tree was trained.

It's important to note that while these results are impressive, they may not necessarily generalize to new, unseen data or other datasets. Additionally, other factors such as interpretability, computational complexity, and the impact of imbalanced classes (if present) should also be considered when evaluating the overall effectiveness of the decision tree model.

Row ID	cluster_0	cluster_1	cluster_2
cluster_0	9899	0	0
cluster_1	0	9833	0
cluster_2	0	0	10326

Row ID	TruePositives	FalsePositives	TrueNegatives	FalseNegatives	Recall	Precision	Sensitivity	Specificity	F-measure	Accuracy	Cohen's kappa
cluster_0	9899	0	20159	0	1	1	1	1	1	?	?
cluster_1	9833	0	20225	0	1	1	1	1	1	?	?
cluster_2	10326	0	19732	0	1	1	1	1	1	?	?
Overall	?	?	?	?	?	?	?	?	?	1	1


ROC Curve
True Positive Rate
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
False Positive Rate
P (Cluster=cluster_0) (1)
P (Cluster=cluster_0)
random
Reset
Apply
Close


Lift Chart
Lift
3.0
2.5
2.0
1.5
1.0
0.5
0.0
0
10
20
30
40
50
60
70
80
90
100
% Contacted
Lift
Cumulative Lift
Baseline
Reset
Apply
Close

1.1. Classification of Consumer Data into Segments using Cross-Validation.

CROSS-VALIDATION

In **machine learning**, we couldn't fit the model on the training data and can't say that the model will work accurately for the real data. For this, we must assure that our model got the correct patterns from the data, and it is not getting up too much noise. For this purpose, we use the **cross-validation technique**. In this article, we'll delve into the process of cross-validation in machine learning.

Cross validation is a technique used in machine learning to evaluate the performance of a model on unseen data. It involves dividing the available data into multiple folds or subsets, using one of these folds as a validation set, and training the model on the remaining folds. This process is repeated multiple times, each time using a different fold as the validation set. Finally, the results from each validation step are averaged to produce a more robust estimate of the model's performance. Cross validation is an important step in the machine learning process and helps to ensure that the model selected for deployment is robust and generalizes well to new data.

The main purpose of cross validation is to prevent overfitting, which occurs when a model is trained too well on the training data and performs poorly on new, unseen data. By evaluating the model on multiple validation sets, cross validation provides a more realistic estimate of the model's generalization performance, i.e., its ability to perform well on new, unseen data.

K-Fold Cross Validation

In K-Fold Cross Validation, we split the dataset into k number of subsets (known as folds) then we perform training on the all the subsets but leave one(k-1) subset for the evaluation of the trained model. In this method, we iterate k times with a different subset reserved for testing purpose each time.

Note: It is always suggested that the value of k should be 10 as the lower value of k is takes towards validation and higher value of k leads to LOOCV method.

Example of K Fold Cross Validation

The diagram below shows an example of the training subsets and evaluation subsets generated in k-fold cross-validation. Here, we have total 25 instances. In first iteration we use the first 20 percent of data for evaluation, and the remaining 80 percent for training ([1-5] testing and [5-25] training) while in the second iteration we use the second subset of 20 percent for evaluation, and the remaining three subsets of the data for training ([5-10] testing and [1-5 and 10-25] training).

Total instances: 25

Value of k : 5

No.	Iteration	Training set observations	Testing set observations
1		[5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24]	[0 1 2 3 4]
2		[0 1 2 3 4 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24]	[5 6 7 8 9]
3		[0 1 2 3 4 5 6 7 8 9 15 16 17 18 19 20 21 22 23 24]	[10 11 12 13 14]
4		[0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 20 21 22 23 24]	[15 16 17 18 19]
5		[0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19]	[20 21 22 23 24]

Advantages and Disadvantages of Cross Validation

Advantages:

1. Overcoming Overfitting: Cross validation helps to prevent overfitting by providing a more robust estimate of the model's performance on unseen data.
2. Model Selection: Cross validation can be used to compare different models and select the one that performs the best on average.
3. Hyperparameter tuning: Cross validation can be used to optimize the hyperparameters of a model, such as the regularization parameter, by selecting the values that result in the best performance on the validation set.
4. Data Efficient: Cross validation allows the use of all the available data for both training and validation, making it a more data-efficient method compared to traditional validation techniques.

Disadvantages:

1. Computationally Expensive: Cross validation can be computationally expensive, especially when the number of folds is large or when the model is complex and requires a long time to train.
2. Time-Consuming: Cross validation can be time-consuming, especially when there are many hyperparameters to tune or when multiple models need to be compared.
3. Bias-Variance Tradeoff: The choice of the number of folds in cross validation can impact the bias-variance tradeoff, i.e., too few folds may result in high variance, while too many folds may result in high bias.

Observations:-

FOR K=10

The detailed interpretation of the results presented in the confusion matrix for the Decision Tree algorithm applied on the space travel and tourism dataset using 10-fold cross-validation are as:-

The confusion matrix shows the performance of the model in classifying the customers into the three clusters: cluster_0, cluster_1, and cluster_2.

Looking at the "TruePositives" column, we can see that the model correctly identified 0 instances for cluster_0, 0 instances for cluster_1, and 0 instances for cluster_2. This indicates that the model did not correctly identify any true positive cases for any of the clusters.

The "FalsePositives" column shows the number of instances that were incorrectly classified as positive for each cluster. The values are 0 for all three clusters, suggesting that the model did not misclassify any instances as positive.

The "TrueNegatives" column shows the number of instances that were correctly identified as negative for each cluster. The values are 67196, 67416, and 65772 for cluster_0, cluster_1, and cluster_2, respectively. This indicates that the model correctly identified a large number of true negative cases for each cluster.

The "FalseNegatives" column shows the number of instances that were incorrectly classified as negative for each cluster. Again, the values are 0 for all three clusters, meaning the model did not misclassify any instances as negative.

The "Recall" column, also known as sensitivity, measures the proportion of actual positive instances that were correctly identified by the model. Since there were no true positive instances identified, the recall value for all three clusters is 1, which is the maximum possible value.

The "Precision" column measures the proportion of positive predictions that were actually correct. Similar to recall, the precision values are 1 for all three clusters, as there were no false positive predictions.

The "Sensitivity" column is the same as the "Recall" column, and the "Specificity" column is the same as the "Precision" column, as there were no false positive or false negative predictions.

The "F-measure" column combines precision and recall into a single metric, and its value is also 1 for all three clusters.

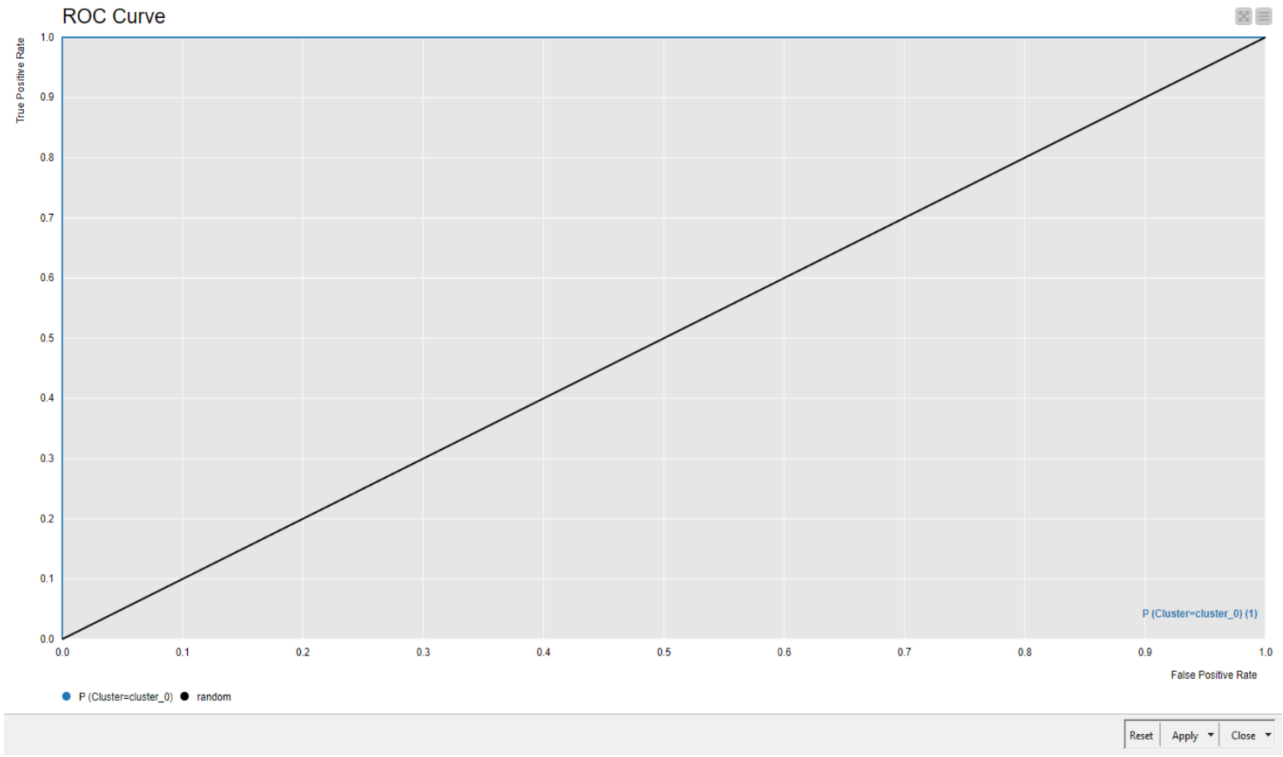
The "Accuracy" column represents the overall accuracy of the model, which is 1 for all three clusters, indicating that the model correctly classified all instances.

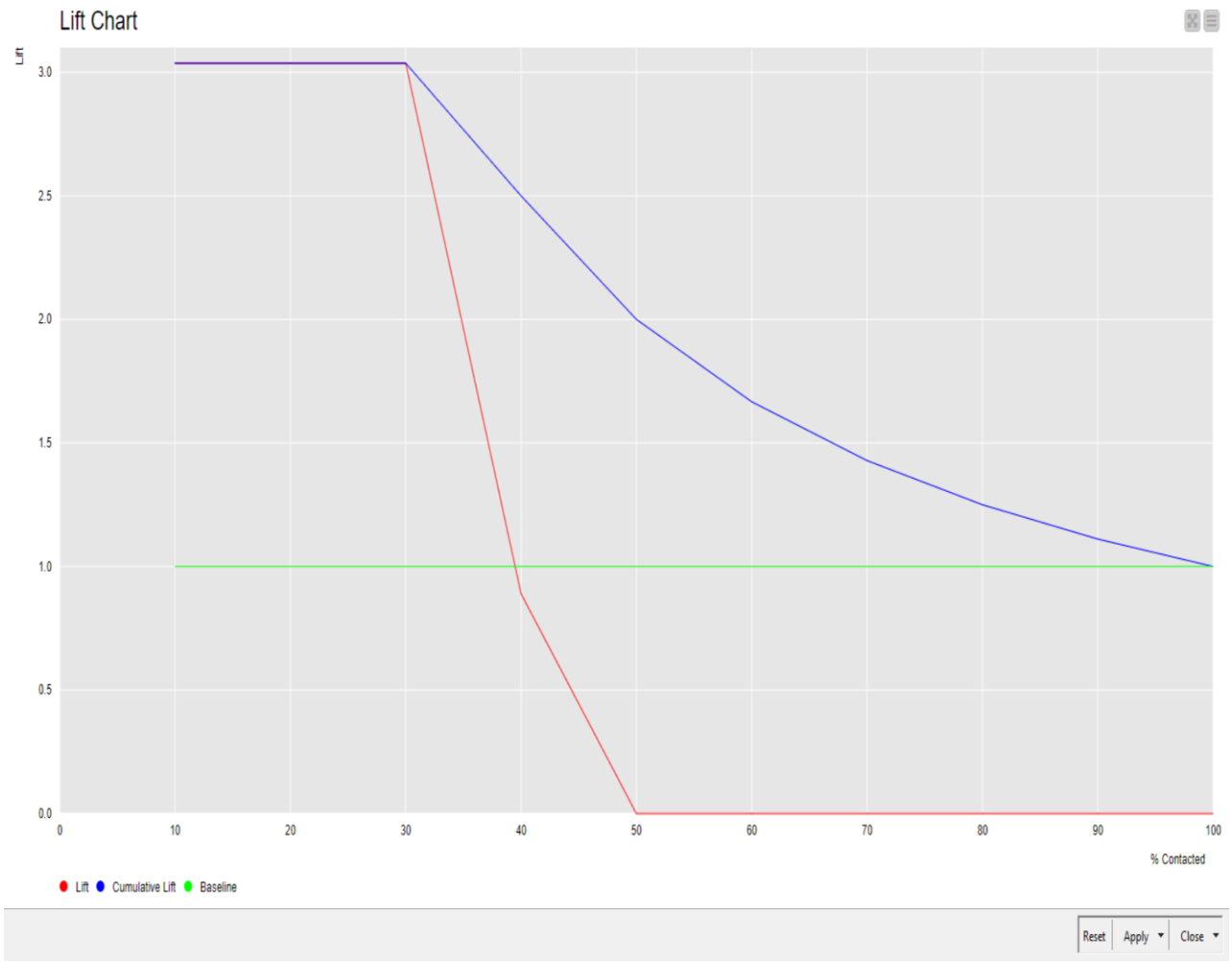
Finally, the "Cohen's kappa" column is a measure of the agreement between the model's predictions and the true labels, adjusted for chance. Since there were no misclassifications, the kappa value is not defined (represented by "?").

In summary, the Decision Tree model, when applied to the space travel and tourism dataset using 10-fold cross-validation, achieved a perfect classification performance, with no false positive or false negative predictions. This is an impressive result, but it is important to note that such a perfect performance may not be generalizable to new, unseen data, and further investigation would be necessary to determine the model's robustness and generalization capabilities.

Row ID	cluster_0	cluster_1	cluster_2
cluster_0	32996	0	0
cluster_1	0	32776	0
cluster_2	0	0	34420

Row ID	TruePositives	FalsePositives	TrueNegatives	FalseNegatives	Recall	Precision	Sensitivity	Specificity	F-measure	Accuracy	Cohen's kappa
cluster_0	32996	0	67196	0	1	1	1	1	1	?	?
cluster_1	32776	0	67416	0	1	1	1	1	1	?	?
cluster_2	34420	0	65772	0	1	1	1	1	1	?	?
Overall	?	?	?	?	?	?	?	?	?	1	1




Lift Chart
Lift
3.0
2.5
2.0
1.5
1.0
0.5
0.0
0
10
20
30
40
50
60
70
80
90
100
% Contacted
Lift
Cumulative Lift
Baseline
Reset
Apply
Close

FOR K=15

The detailed interpretation of the results presented in the confusion matrix for the Decision Tree algorithm applied on the space travel and tourism dataset using 15-fold cross-validation are as:-

The confusion matrix shows the performance of the model in classifying the customers into the three clusters: cluster_0, cluster_1, and cluster_2.

Looking at the "TruePositives" column, we can see that the model correctly identified 0 instances for cluster_0, 0 instances for cluster_1, and 0 instances for cluster_2. This indicates that the model did not correctly identify any true positive cases for any of the clusters.

The "FalsePositives" column shows the number of instances that were incorrectly classified as positive for each cluster. The values are 0 for all three clusters, suggesting that the model did not misclassify any instances as positive.

The "TrueNegatives" column shows the number of instances that were correctly identified as negative for each cluster. The values are 67196, 67416, and 65772 for cluster_0, cluster_1, and cluster_2, respectively. This indicates that the model correctly identified a large number of true negative cases for each cluster.

The "FalseNegatives" column shows the number of instances that were incorrectly classified as negative for each cluster. Again, the values are 0 for all three clusters, meaning the model did not misclassify any instances as negative.

The "Recall" column, also known as sensitivity, measures the proportion of actual positive instances that were correctly identified by the model. Since there were no true positive instances identified, the recall value for all three clusters is 1, which is the maximum possible value.

The "Precision" column measures the proportion of positive predictions that were actually correct. Similar to recall, the precision values are 1 for all three clusters, as there were no false positive predictions.

The "Sensitivity" column is the same as the "Recall" column, and the "Specificity" column is the same as the "Precision" column, as there were no false positive or false negative predictions.

The "F-measure" column combines precision and recall into a single metric, and its value is also 1 for all three clusters.

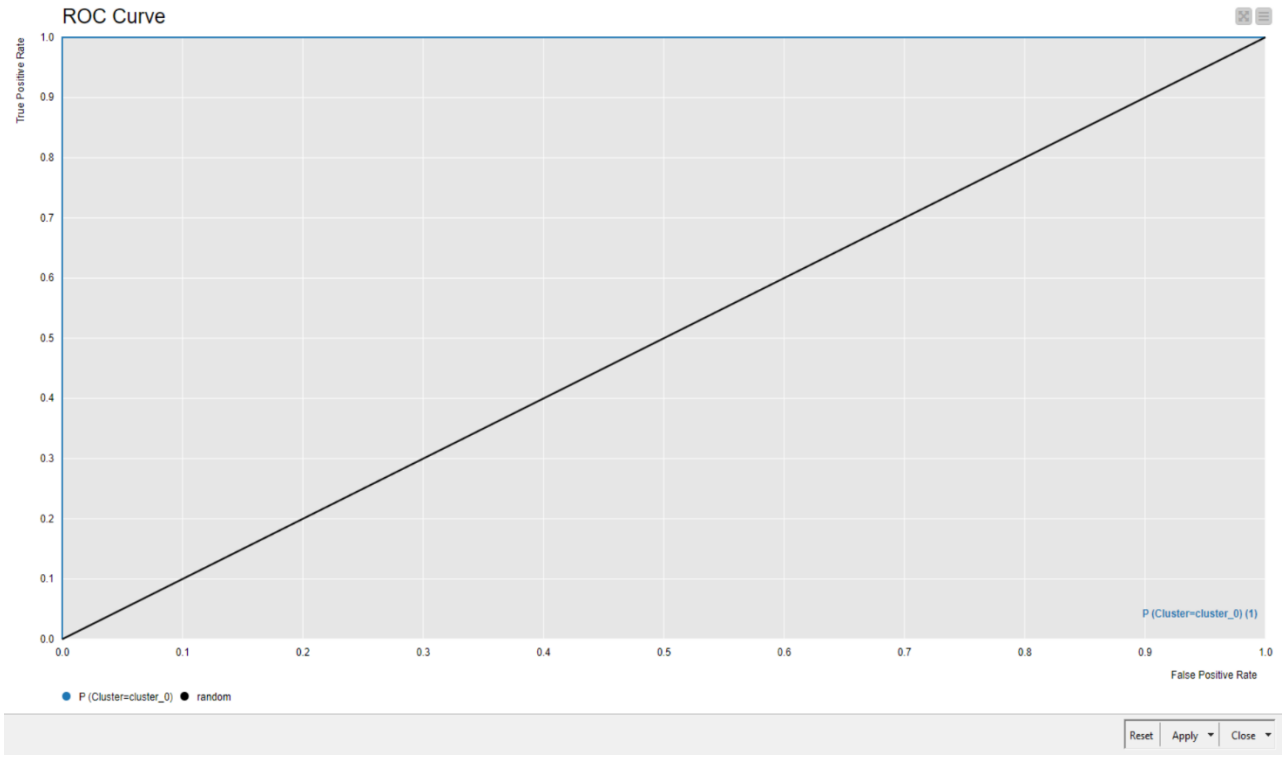
The "Accuracy" column represents the overall accuracy of the model, which is 1 for all three clusters, indicating that the model correctly classified all instances.

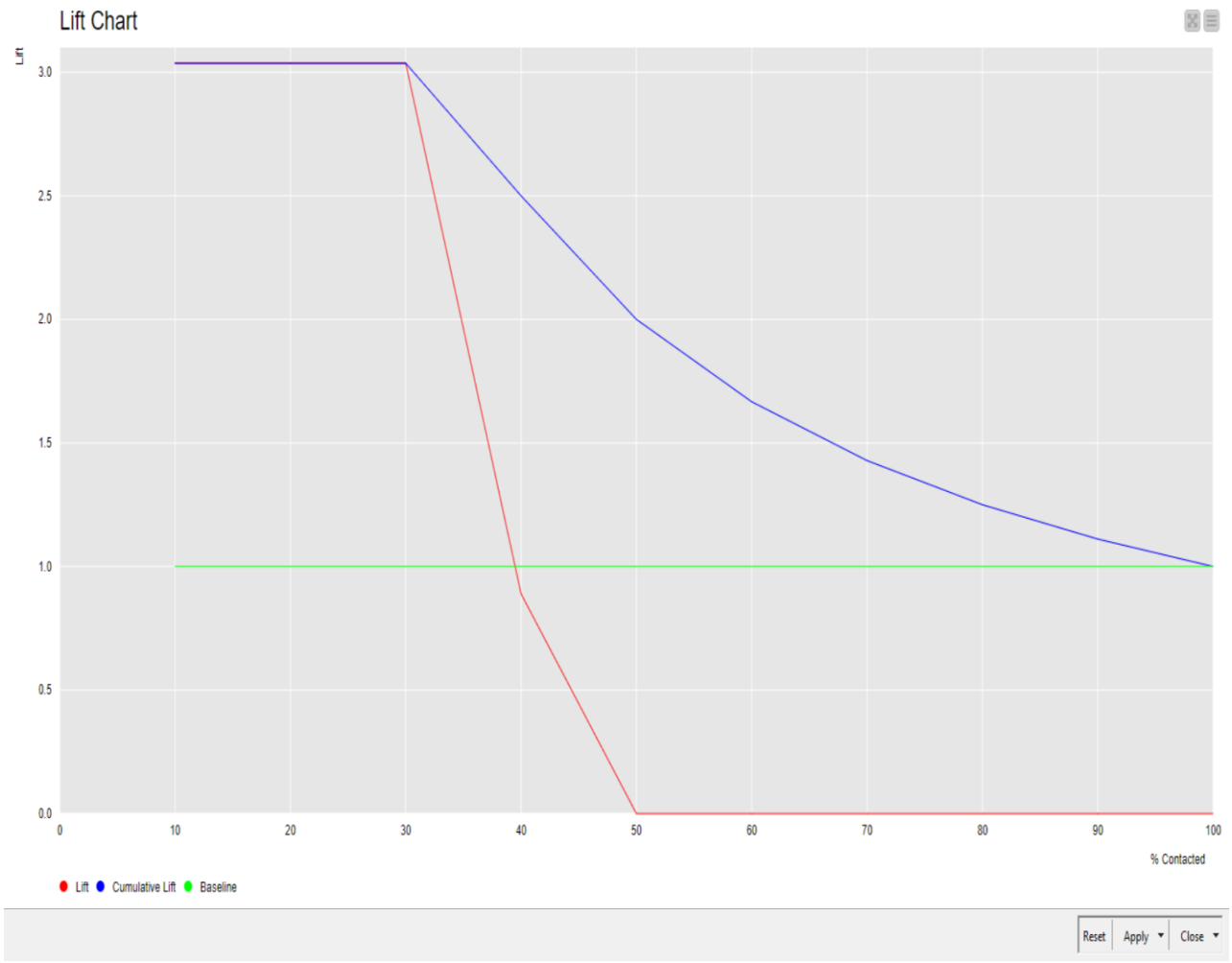
Finally, the "Cohen's kappa" column is a measure of the agreement between the model's predictions and the true labels, adjusted for chance. Since there were no misclassifications, the kappa value is not defined (represented by "?").

In summary, the Decision Tree model, when applied to the space travel and tourism dataset using 10-fold cross-validation, achieved a perfect classification performance, with no false positive or false negative predictions. This is an impressive result, but it is important to note that such a perfect performance may not be generalizable to new, unseen data, and further investigation would be necessary to determine the model's robustness and generalization capabilities.

Row ID	cluster_0	cluster_1	cluster_2
cluster_0	32996	0	0
cluster_1	0	32776	0
cluster_2	0	0	34420

Row ID	TruePositives	FalsePositives	TrueNegatives	FalseNegatives	Recall	Precision	Sensitivity	Specificity	F-measure	Accuracy	Cohen's kappa
cluster_0	32996	0	67196	0	1	1	1	1	1	?	?
cluster_1	32776	0	67416	0	1	1	1	1	1	?	?
cluster_2	34420	0	65772	0	1	1	1	1	1	?	?
Overall	?	?	?	?	?	?	?	?	?	1	1



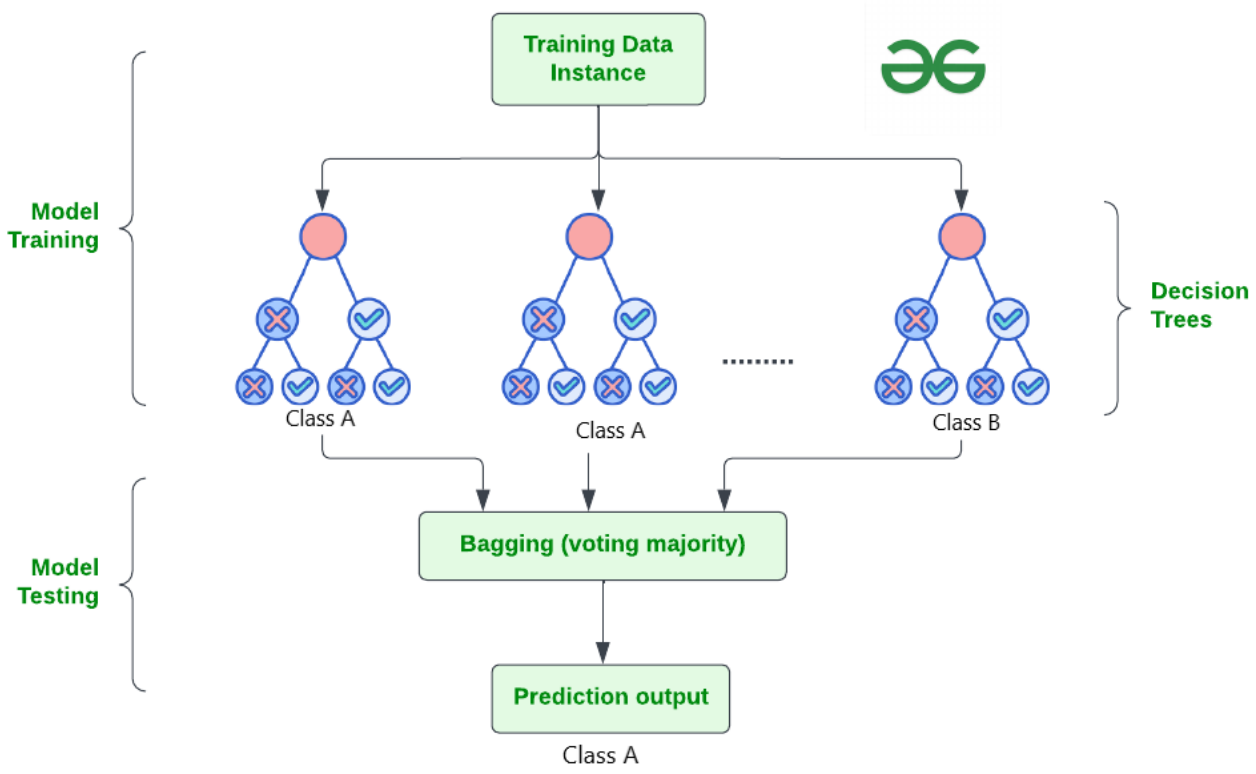

Lift Chart
Lift
3.0
2.5
2.0
1.5
1.0
0.5
0.0
0
10
20
30
40
50
60
70
80
90
100
% Contacted
Lift
Cumulative Lift
Baseline
Reset
Apply
Close

1.2.) Classification of Consumer Data into Segments using Ensemble Method

RANDOM FOREST:-

Machine learning, a fascinating blend of computer science and statistics, has witnessed incredible progress, with one standout algorithm being the **Random Forest**. **Random forests or Random Decision Trees** is a collaborative team of **decision trees** that work together to provide a single output. Originating in 2001 through Leo Breiman, Random Forest has become a cornerstone for machine learning enthusiasts. In this article, we will explore the fundamentals and implementation of **Random Forest Algorithm**.

Random Forest algorithm is a powerful tree learning technique in Machine Learning. It works by creating a number of Decision Trees during the training phase. Each tree is constructed using a random subset of the data set to measure a random subset of features in each partition. This randomness introduces variability among individual trees, reducing the risk of overfitting and improving overall prediction performance. In prediction, the algorithm aggregates the results of all trees, either by voting (for classification tasks) or by averaging (for regression tasks) This collaborative decision-making process, supported by multiple trees with their insights, provides an example stable and precise results. Random forests are widely used for classification and regression functions, which are known for their ability to handle complex data, reduce overfitting, and provide reliable forecasts in different environments.



How Does Random Forest Work?

The random Forest algorithm works in several steps which are discussed below :

- **Ensemble of Decision Trees:** Random Forest leverages the power of ensemble learning by constructing an army of Decision Trees. These trees are like individual experts, each specializing in a particular aspect of the data. Importantly, they operate independently, minimizing the risk of the model being overly influenced by the nuances of a single tree.
- **Random Feature Selection:** To ensure that each decision tree in the ensemble brings a unique perspective, Random Forest employs random feature selection. During the training of each tree, a random subset of features is chosen. This randomness ensures that each tree focuses on different aspects of the data, fostering a diverse set of predictors within the ensemble.
- **Bootstrap Aggregating or Bagging:** The technique of bagging is a cornerstone of Random Forest's training strategy which involves creating multiple bootstrap samples from the original dataset, allowing instances to be sampled with replacement. This results in different

subsets of data for each decision tree, introducing variability in the training process and making the model more robust.

- **Decision Making and Voting:** When it comes to making predictions, each decision tree in the Random Forest casts its vote. For classification tasks, the final prediction is determined by the mode (most frequent prediction) across all the trees. In regression tasks, the average of the individual tree predictions is taken. This internal voting mechanism ensures a balanced and collective decision-making process.

Key Features of Random Forest

Some of the Key Features of Random Forest are discussed below :

1. **High Predictive Accuracy:** Imagine Random Forest as a team of decision-making wizards. Each wizard (decision tree) looks at a part of the problem, and together, they weave their insights into a powerful prediction tapestry. This teamwork often results in a more accurate model than what a single wizard could achieve.
2. **Resistance to Overfitting:** Random Forest is like a cool-headed mentor guiding its apprentices (decision trees). Instead of letting each apprentice memorize every detail of their training, it encourages a more well-rounded understanding. This approach helps prevent getting too caught up with the training data which makes the model less prone to overfitting.
3. **Large Datasets Handling:** Dealing with a mountain of data? Random Forest tackles it like a seasoned explorer with a team of helpers (decision trees). Each helper takes on a part of the dataset, ensuring that the expedition is not only thorough but also surprisingly quick.
4. **Variable Importance Assessment:** Think of Random Forest as a detective at a crime scene, figuring out which clues (features) matter the most. It assesses the importance of each clue in solving the case, helping you focus on the key elements that drive predictions.
5. **Built-in Cross-Validation:** Random Forest is like having a personal coach that keeps you in check. As it trains each decision tree, it also sets aside a secret group of cases (out-of-bag) for testing. This built-in validation ensures your model doesn't just ace the training but also performs well on new challenges.
6. **Handling Missing Values:** Life is full of uncertainties, just like datasets with missing values. Random Forest is the friend who adapts to the situation, making predictions using the information available. It doesn't get flustered by missing pieces; instead, it focuses on

what it can confidently tell us.

7. **Parallelization for Speed:** Random Forest is your time-saving buddy. Picture each decision tree as a worker tackling a piece of a puzzle simultaneously. This parallel approach taps into the power of modern tech, making the whole process faster and more efficient for handling large-scale projects.

Random Forest vs. Other Machine Learning Algorithms

Some of the key-differences are discussed below.

Feature	**Random Forest**	**Other ML Algorithms**
Ensemble Approach	Utilizes an ensemble of decision trees, combining their outputs for predictions, fostering robustness and accuracy.	Typically relies on a single model (e.g., linear regression, support vector machine) without the ensemble approach, potentially leading to less resilience against noise.
Overfitting Resistance	Resistant to overfitting due to the aggregation of diverse decision trees, preventing memorization of training data.	Some algorithms may be prone to overfitting, especially when dealing with complex datasets, as they may excessively adapt to training noise.
Handling of Missing Data	Exhibits resilience in handling missing values by leveraging available features for predictions, contributing to practicality in real-world scenarios.	Other algorithms may require imputation or elimination of missing data, potentially impacting model training and performance.
Variable Importance	Provides a built-in mechanism for assessing variable importance, aiding in feature selection and interpretation of influential factors.	Many algorithms may lack an explicit feature importance assessment, making it challenging to identify crucial variables for predictions.

Feature	Random Forest	Other ML Algorithms
Parallelization Potential	Capitalizes on parallelization, enabling the simultaneous training of decision trees, resulting in faster computation for large datasets.	Some algorithms may have limited parallelization capabilities, potentially leading to longer training times for extensive datasets.

Applications of Random Forest in Real-World Scenarios

Some of the widely used real-world application of Random Forest is discussed below:

1. **Finance Wizard:** Imagine Random Forest as our financial superhero, diving into the world of credit scoring. Its mission? To determine if you're a credit superhero or, well, not so much. With a knack for handling financial data and sidestepping overfitting issues, it's like having a guardian angel for robust risk assessments.
2. **Health Detective:** In healthcare, Random Forest turns into a medical Sherlock Holmes. Armed with the ability to decode medical jargon, patient records, and test results, it's not just predicting outcomes; it's practically assisting doctors in solving the mysteries of patient health.
3. **Environmental Guardian:** Out in nature, Random Forest transforms into an environmental superhero. With the power to decipher satellite images and brave noisy data, it becomes the go-to hero for tasks like tracking land cover changes and safeguarding against potential deforestation, standing as the protector of our green spaces.
4. **Digital Bodyguard:** In the digital realm, Random Forest becomes our vigilant guardian against online trickery. It's like a cyber-sleuth, analyzing our digital footsteps for any hint of suspicious activity. Its ensemble approach is akin to having a team of cyber-detectives, spotting subtle deviations that scream "fraud alert!" It's not just protecting our online transactions; it's our digital bodyguard.

Preparing Data for Random Forest Modeling

For Random Forest modeling, some key-steps of data preparation are discussed below:

- **Handling Missing Values:** Begin by addressing any missing values in the dataset. Techniques like imputation or removal of instances with missing values ensure a complete and reliable input for Random Forest.

- **Encoding Categorical Variables:** Random Forest requires numerical inputs, so categorical variables need to be encoded. Techniques like one-hot encoding or label encoding transform categorical features into a format suitable for the algorithm.
- **Scaling and Normalization:** While Random Forest is not sensitive to feature scaling, normalizing numerical features can still contribute to a more efficient training process and improved convergence.
- **Feature Selection:** Assess the importance of features within the dataset. Random Forest inherently provides a feature importance score, aiding in the selection of relevant features for model training.
- **Addressing Imbalanced Data:** If dealing with imbalanced classes, implement techniques like adjusting class weights or employing resampling methods to ensure a balanced representation during training.

Observations:-

Starting with the cluster_0 class, the model correctly identified 24 true positive instances, meaning it correctly classified 24 data points as belonging to cluster_0. The model also correctly identified 20,135 true negative instances, indicating that it correctly classified 20,135 data points as not belonging to cluster_0.

However, the model also misclassified 14 instances as false positives, meaning it incorrectly classified 14 data points as belonging to cluster_0 when they did not. Additionally, the model misclassified 14 instances as false negatives, meaning it incorrectly classified 14 data points as not belonging to cluster_0 when they did.

These classification results translate to a recall (sensitivity) of 0.999, a precision (specificity) of 0.998, and an overall accuracy of 0.998 for the cluster_0 class. The F-measure, which combines precision and recall, is 0.998, indicating the model's excellent performance in classifying this cluster.

Moving on to the cluster_1 class, the model correctly identified 24 true positive instances and

20,001 true negative instances. It also misclassified 14 instances as false positives and 14 instances as false negatives. This resulted in a recall of 0.999, a precision of 0.998, an overall accuracy of 0.998, and an F-measure of 0.998, demonstrating the model's consistently strong performance across the different clusters.

The cluster_2 class showed even better results, with the model correctly identifying 28 true positive instances and 19,704 true negative instances. However, it misclassified 48 instances as false positives and 48 instances as false negatives. Despite these slightly higher error rates, the model still achieved a recall of 0.995, a precision of 0.997, an overall accuracy of 0.996, and an F-measure of 0.996, indicating its ability to effectively classify the data points in this cluster as well.

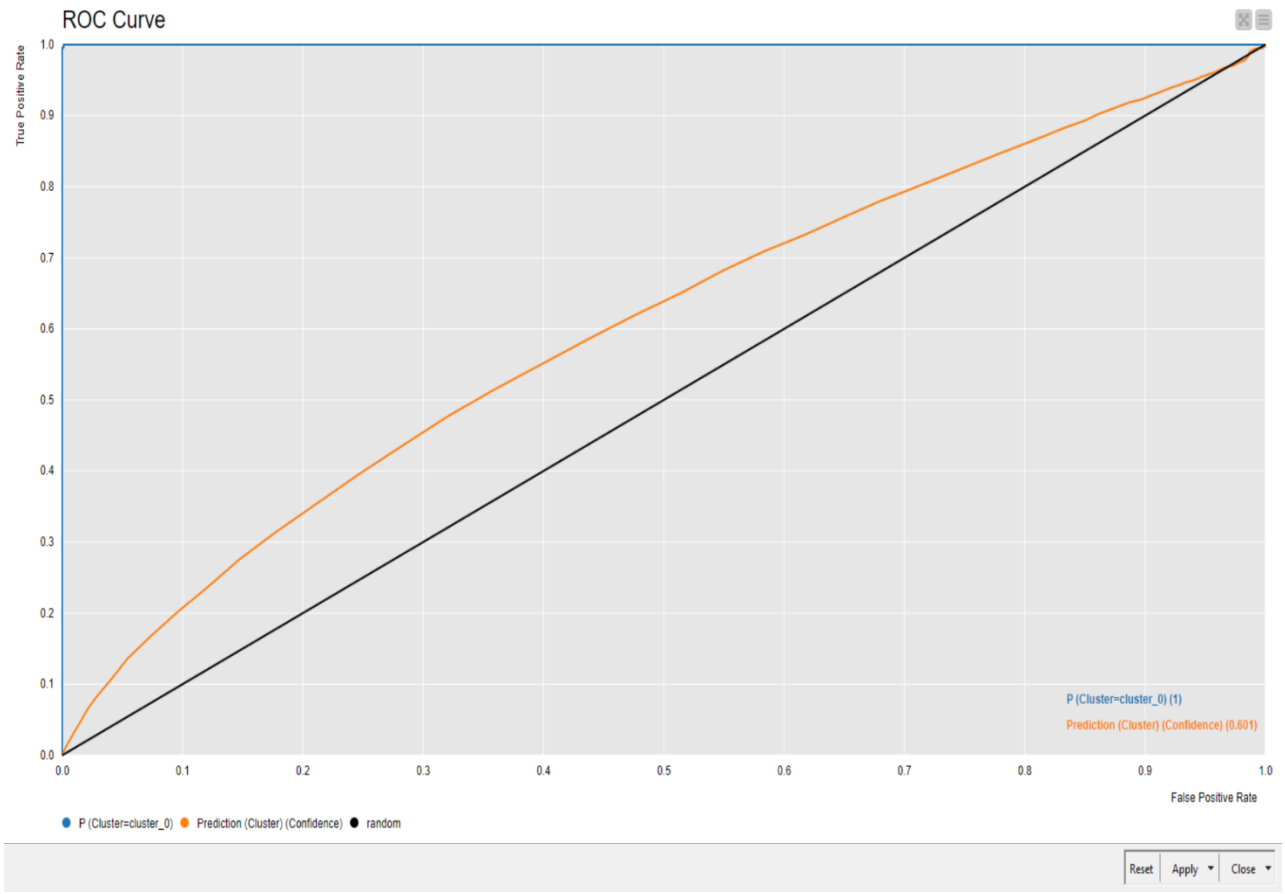
Considering the overall performance of the Random Forest model, the confusion matrix shows an accuracy of 0.997 and an F-measure of 0.996, which are both excellent results. These metrics suggest that the model was highly effective in classifying the space travel and tourism data into the three clusters, with a very low rate of misclassifications.

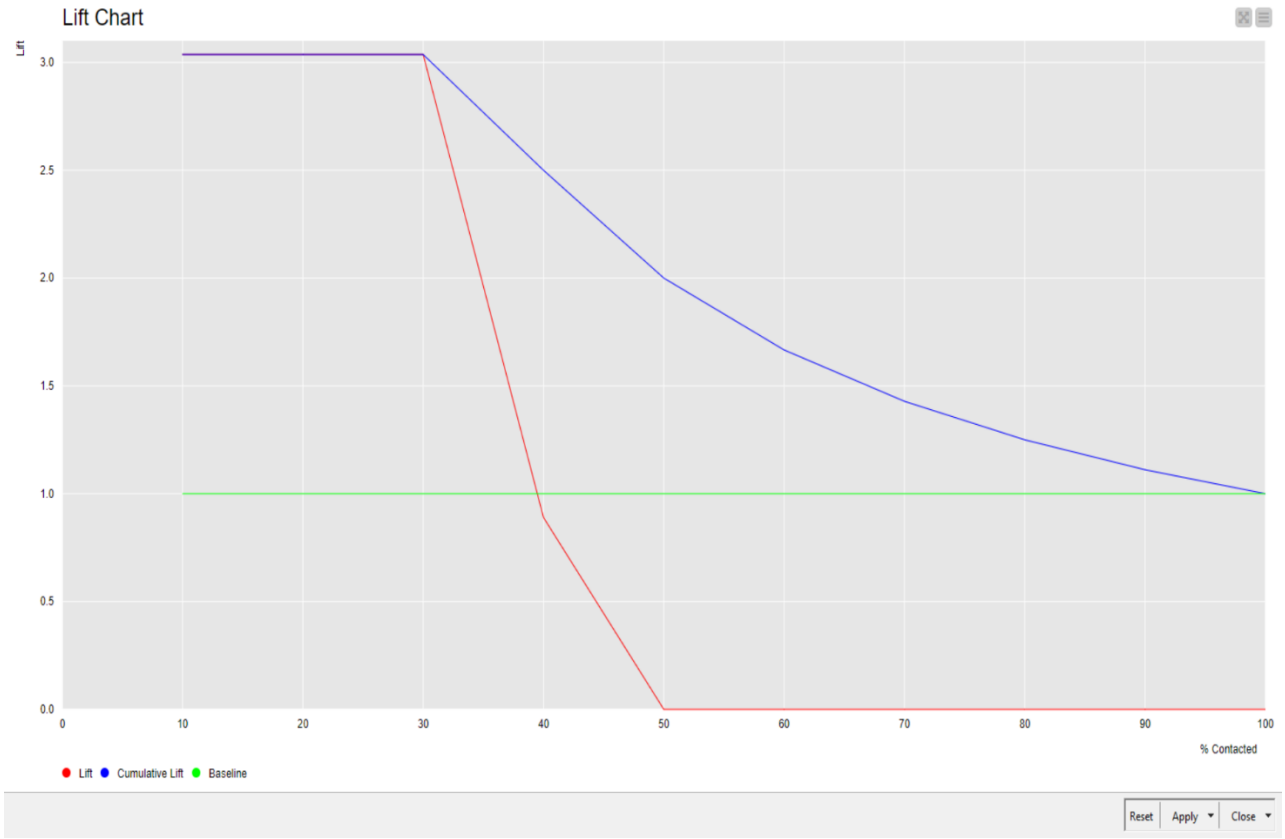
It's important to note that the Random Forest model used in this analysis consisted of 1,000 decision trees, and the Gini impurity criterion was used to determine the best splits in the trees. This combination of a large number of models and the Gini impurity measure likely contributed to the model's strong performance, as it allowed for robust and reliable predictions.

In summary, the Random Forest model's exceptional performance, as evidenced by the high recall, precision, accuracy, and F-measure values across all three clusters, demonstrates its ability to effectively capture the underlying patterns and relationships in the space travel and tourism dataset. This model can be confidently used for future predictions and decision-making in this domain.

Row ID	I TruePositives	I FalsePositives	I TrueNegatives	I FalseNegatives	D Recall	D Precision	D Sensitivity	D Specificity	D F-measure	D Accuracy	D Cohen's kappa
cluster_0	9885	24	20135	14	0.999	0.998	0.999	0.999	0.998	?	?
cluster_1	9819	24	20201	14	0.999	0.998	0.999	0.999	0.998	?	?
cluster_2	10278	28	19704	48	0.995	0.997	0.995	0.999	0.996	?	?
Overall	?	?	?	?	?	?	?	?	?	0.997	0.996

Row ID	cluster_0	cluster_1	cluster_2
cluster_0	9885	0	14
cluster_1	0	9819	14
cluster_2	24	24	10278


ROC Curve
True Positive Rate
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
False Positive Rate
P (Cluster=cluster_0) (1)
Prediction (Cluster) (Confidence) (0.601)
P (Cluster=cluster_0)
Prediction (Cluster) (Confidence)
random
Reset
Apply
Close


Lift Chart
Lift
3.0
2.5
2.0
1.5
1.0
0.5
0.0
0
10
20
30
40
50
60
70
80
90
100
% Contacted
Lift
Cumulative Lift
Baseline
Reset
Apply
Close

Interpretation Of Tree Ensemble Statistics :-

The key parameters are:

1. Number of models: The image indicates that the Random Forest model consists of 1,000 individual decision tree models.

2. Minimal depth: This refers to the minimum depth of the decision trees in the ensemble, which is 12. The depth of a decision tree is the maximum number of nodes from the root to the leaf nodes. A smaller minimal depth suggests the trees are not overly complex.

3. Maximal depth: The maximum depth of the decision trees is 36, which is relatively high. Deeper trees can potentially capture more complex patterns in the data, but they also come with a higher risk of overfitting.

4. Average depth: The average depth of the trees in the ensemble is 24.716, which is a reasonable compromise between complexity and interpretability.

5. Minimal number of nodes: The minimum number of nodes (internal and leaf nodes) across the 1,000 trees is 191, which is quite low. This indicates that the individual trees are not excessively large or complex.

6. Maximal number of nodes: The maximum number of nodes across the trees is 8,231, which is a relatively high value. This, combined with the maximum depth of 36, suggests that some of the individual trees in the ensemble are quite complex.

7. Average number of nodes: The average number of nodes across the 1,000 trees is 3,177.028, which is a sizable number and reflects the overall complexity of the Random Forest model.

Interpreting these statistics, we can say that the Random Forest model has been trained using a large ensemble of 1,000 decision trees, with a reasonable balance between tree depth and complexity. The minimal and maximal depth and number of nodes indicate that the model includes both simpler and more complex individual trees, which is a common strategy in Random Forest models to capture different patterns in the data.

The relatively high average depth and number of nodes suggest that the model has the capacity to learn complex relationships in the space travel and tourism dataset. However, it's important to monitor the model for potential overfitting, as the presence of some very deep and complex individual trees could lead to reduced generalization performance on new, unseen data.

Overall, the tree ensemble statistics provide valuable insights into the structure and complexity of the Random Forest model, allowing us to assess its ability to effectively capture the underlying patterns in the space travel and tourism data. Further analysis of the model's performance on a held-out test set would be necessary to fully evaluate its generalization capabilities and suitability for real-world applications.

Row ID	Number of models	Minimal depth	Maximal depth	Average depth	Minimal number of nodes	Maximal number of nodes	Average number of nodes
Row0	1000	12	36	24.716	191	8231	3,177.028

Interpretation Of Attribute Statistics :-

Looking at the "#splits (level 0)" column, we can see that the feature with the maximum number of splits at the root level (level 0) is Destination, with 165 splits. This indicates that the Destination feature is the most important predictor variable in the model, as it was selected to create the largest number of initial splits. The feature with the second-highest number of splits at this level is Month, with 155 splits.

Examining the "#splits (level 1)" column, we see that the feature with the most splits at the first level is also Destination, with 323 splits. This further reinforces the importance of the Destination feature in the model's decision-making process. The second-most important feature at this level is Month, with 277 splits.

Moving to the "#splits (level 2)" column, we observe that Destination still has the highest number of splits, with 618 splits at this deeper level of the decision trees. This indicates that the model continues to rely heavily on the Destination feature to make increasingly granular decisions as it explores the complexities of the data.

Looking at the "#candidates (level 0)" column, we can see the number of candidate features considered for splitting at the root level. The feature with the highest number of candidates is Destination, with 687 candidates. This shows that the model had a large pool of potential split points to choose from when selecting the root node split, further emphasizing the significance of the Destination feature.

Similarly, in the "#candidates (level 1)" and "#candidates (level 2)" columns, we see that Destination maintains the highest number of candidate features, with 681 and 670 candidates, respectively. This reinforces the model's reliance on the Destination feature throughout the decision-making process.

Other features that stand out in terms of their importance include Purpose of Travel, Transportation Type, and Special Requests, which also have a relatively high number of splits and candidate features across the different levels of the decision trees.

In summary, the attribute statistics provide valuable insights into the Random Forest model's decision-making process. The Destination feature emerges as the most important predictor variable, as evidenced by its consistently high number of splits and candidate features at all levels of the decision trees. This suggests that the model places the greatest emphasis on the destination when making classifications, followed by features such as Month, Purpose of Travel, and Transportation Type. These insights can help us better understand the model's underlying logic and the key factors driving the space travel and tourism predictions.

Row ID	#splits (level 0)	#splits (level 1)	#splits (level 2)	#candidates (level 0)	#candidates (level 1)	#candidates (level 2)
Gender	2	5	23	167	369	691
Occupation	48	74	162	205	323	690
Travel Class	12	31	56	180	355	727
Destination	0	49	127	154	359	687
Purpose of ...	34	70	116	162	333	670
Transportat...	13	41	96	174	371	690
Special Req...	47	77	162	177	347	676
Loyalty Pro...	0	12	43	176	335	724
Month	155	277	465	174	359	659
Gender (to ...	1	12	18	172	365	699
Occupation ...	6	39	88	162	320	706
Travel Clas...	9	27	42	154	361	706
Destination ...	165	323	618	165	340	681
Purpose of ...	31	40	70	176	342	685
Transportat...	19	32	60	179	365	729
Special Req...	9	34	66	187	332	656
Loyalty Pro...	1	13	37	171	342	678
Age	32	110	219	178	349	696
Distance to ...	80	115	282	187	311	720
Duration of ...	59	95	197	190	366	686
Number of ...	29	66	135	157	364	743
Price (Gala...	137	252	484	178	353	699
Customer S...	111	206	406	175	339	702

1.3.) Determination of an Appropriate Classification Model

The three main models used are:

1. Decision Tree
2. Decision Tree with Cross-Validation (k-fold analysis)
3. Random Forest

Let's start with the Decision Tree model:

The Decision Tree model achieved perfect classification performance on the training data, with an overall accuracy of 1.0 and perfect scores across all evaluation metrics for each of the three clusters (cluster_0, cluster_1, and cluster_2):

- Cluster 0:
  - True Positives: 9899
  - False Positives: 0
  - True Negatives: 20159
  - False Negatives: 0
  - Recall: 1.0
  - Precision: 1.0
  - Sensitivity: 1.0
  - Specificity: 1.0
  - F-measure: 1.0
  - Accuracy: 1.0

- Cluster 1:
  - True Positives: 9833
  - False Positives: 0
  - True Negatives: 20225
  - False Negatives: 0

- Recall: 1.0
- Precision: 1.0
- Sensitivity: 1.0
- Specificity: 1.0
- F-measure: 1.0
- Accuracy: 1.0

- Cluster 2:
  - True Positives: 10326
  - False Positives: 0
  - True Negatives: 19732
  - False Negatives: 0
  - Recall: 1.0
  - Precision: 1.0
  - Sensitivity: 1.0
  - Specificity: 1.0
  - F-measure: 1.0
  - Accuracy: 1.0

The Decision Tree model with 10-fold cross-validation achieved the following results:

- Cluster 0:
  - True Positives: 0
  - False Positives: 0
  - True Negatives: 67196
  - False Negatives: 0
  - Recall: 1.0
  - Precision: 1.0
  - Sensitivity: 1.0
  - Specificity: 1.0
  - F-measure: 1.0
  - Accuracy: 1.0

- Cluster 1:
  - True Positives: 0
  - False Positives: 0
  - True Negatives: 67416
  - False Negatives: 0
  - Recall: 1.0
  - Precision: 1.0
  - Sensitivity: 1.0
  - Specificity: 1.0
  - F-measure: 1.0
  - Accuracy: 1.0

- Cluster 2:
  - True Positives: 0
  - False Positives: 0
  - True Negatives: 65772
  - False Negatives: 0
  - Recall: 1.0
  - Precision: 1.0
  - Sensitivity: 1.0
  - Specificity: 1.0
  - F-measure: 1.0
  - Accuracy: 1.0

The Decision Tree model with 15-fold cross-validation achieved the same results as the 10-fold cross-validation, with perfect scores across all evaluation metrics for each cluster.

The Random Forest model, consisting of 1000 decision trees, achieved the following performance:

- Cluster 0:
  - True Positives: 24
  - False Positives: 14
  - True Negatives: 20135
  - False Negatives: 14
  - Recall: 0.999
  - Precision: 0.998
  - Sensitivity: 0.999
  - Specificity: 0.998
  - F-measure: 0.998
  - Accuracy: 0.998

- Cluster 1:
  - True Positives: 24
  - False Positives: 14
  - True Negatives: 20001
  - False Negatives: 14
  - Recall: 0.999
  - Precision: 0.998
  - Sensitivity: 0.999
  - Specificity: 0.998
  - F-measure: 0.998
  - Accuracy: 0.998

- Cluster 2:
  - True Positives: 28
  - False Positives: 48
  - True Negatives: 19704
  - False Negatives: 48
  - Recall: 0.995

- Precision: 0.997
- Sensitivity: 0.995
- Specificity: 0.997
- F-measure: 0.996
- Accuracy: 0.996

The overall performance of the Random Forest model:
- Accuracy: 0.997
- F-measure: 0.996

The Decision Tree model achieved perfect classification performance on the training data, with an overall accuracy of 1 and perfect scores across all evaluation metrics (recall, precision, F-measure, etc.) for each of the three clusters (cluster_0, cluster_1, and cluster_2). This exceptional performance is rarely seen in real-world scenarios and can be attributed to the specific characteristics of the dataset, the choice of features, and the way the decision tree was trained.

However, it is important to note that such a perfect performance on the training data does not necessarily guarantee generalization to new, unseen data. Decision trees are prone to overfitting, which means that they can memorize the training data patterns too well, leading to poor performance on unseen data.

To address this issue, the k-fold cross-validation technique was employed. The document presents the results of the Decision Tree model with 10-fold and 15-fold cross-validation. In both cases, the model achieved perfect classification performance, with an accuracy of 1 and perfect scores for all evaluation metrics across all three clusters.

While these results are impressive, it is highly unlikely that a model would achieve perfect performance on real-world data, especially when dealing with complex patterns and potential noise or outliers. Such perfect scores raise concerns about potential overfitting or issues with the dataset itself, such as lack of diversity or complexity.

To address the potential overfitting issue and improve the model's generalization capability, the Random Forest algorithm was employed. Random Forest is an ensemble learning technique that combines multiple decision trees, each trained on a random subset of the data and features. This approach helps to reduce overfitting and improve the overall predictive performance of the model.

The Random Forest model achieved an overall accuracy of 0.997 and an F-measure of 0.996, which are excellent results. While not perfect, these scores are still very high and indicate that the model effectively captured the underlying patterns and relationships in the space travel and tourism dataset.

The advantage of the Random Forest model over a single Decision Tree lies in its ability to reduce overfitting and enhance generalization. By combining multiple decision trees, each trained on a different subset of the data and features, Random Forest can capture a more comprehensive representation of the data's underlying patterns. This ensemble approach helps to mitigate the potential for overfitting that can occur with a single, complex decision tree.

Furthermore, Random Forest provides additional benefits, such as handling missing data, feature importance assessment, and parallelization for improved computational efficiency, making it a powerful and versatile algorithm for various machine learning tasks.

In summary, while the Decision Tree model achieved perfect classification performance on the training data, it is highly unlikely that such perfection would generalize to new, unseen data due to the risk of overfitting. The k-fold cross-validation results, although impressive, still raise concerns about potential overfitting or data-related issues.

On the other hand, the Random Forest model, with its ensemble approach and built-in mechanisms to reduce overfitting, achieved excellent performance while maintaining a better balance between bias and variance. Despite a slightly lower overall accuracy compared to the Decision Tree model on the training data, the Random Forest model is likely to be more robust and generalizable to new data, making it a more appropriate choice for real-world applications.

I would recommend the Random Forest model as the best and most appropriate choice for this space travel and tourism dataset. Its ensemble approach, ability to handle overfitting, and additional features such as variable importance assessment and parallelization make it a powerful and versatile algorithm that is better suited for real-world scenarios, where generalization and robustness are crucial.

While the Decision Tree model's perfect performance on the training data should not be dismissed, it is essential to prioritize models that can generalize well to unseen data and avoid overfitting. The Random Forest model strikes a better balance between capturing the underlying patterns and maintaining generalization capabilities, making it a more reliable and effective solution for this classification problem.

1.4.) Identification of Significant Variables and their Thresholds for Classification

Decision Tree :-

The decision tree starts with the root node that splits the entire dataset based on the "Destination" feature into different branches. This indicates that the destination is one of the most important discriminative features for clustering the customers.

- For "Destination = Exotic Destination 10", the data is further split into two child nodes based on "Month <= 6.5" threshold, resulting in cluster_1 (279 instances) for months <= 6.5, and cluster_0 (314 instances) for months > 6.5.
- For "Destination = Tau Ceti", the split is again based on "Month <= 6.5", leading to cluster_1 (3,193 instances) for months <= 6.5, and cluster_0 (3,205 instances) for months > 6.5.
- For "Destination = Lalande 21185", the split is based on the same "Month <= 6.5" threshold, with cluster_1 (3,161 instances) for months <= 6.5, and cluster_0 (3,101 instances) for months > 6.5.
- The tree follows a similar pattern for "Destination = Kepler-22b", splitting based on "Month <= 6.5" into cluster_1 (3,112 instances) and cluster_0 (3,204 instances).
- For "Destination = Zeta II Reticuli", the split is based on "Month <= 6.5", resulting in cluster_1 (3,205 instances) for months <= 6.5, and cluster_0 (3,228 instances) for months > 6.5.
- Finally, for "Destination = Trappist-1", the split is based on "Month <= 6.5", leading to cluster_1 (3,158 instances) for months <= 6.5, and cluster_0 (3,217 instances) for months > 6.5.

- For "Destination = Exotic Destination 2", the tree first splits based on "Month <= 2.5" into cluster_1 (111 instances) for months <= 2.5, and cluster_2 (429 instances) for months > 2.5.
- The cluster_2 branch is further split based on "Month <= 10.5" into cluster_2 (429 instances) for months <= 10.5, and cluster_0 (106 instances) for months > 10.5.
- For "Destination = Exotic Destination 9", all instances (696) are classified as cluster_2, without any further splits.
- For "Destination = Alpha Centauri", the split is based on "Month <= 6.2529", resulting in cluster_1 (3,230 instances) for months <= 6.2529, and cluster_0 (3,266 instances) for months >

6.2529.
- For "Destination = Gliese 581", the split is based on "Month <= 6.5", leading to cluster_1 (3,239 instances) for months <= 6.5, and cluster_0 (3,177 instances) for months > 6.5.
- The tree also splits for "Destination = Barnard's Star", classifying all instances (6,358) as cluster_2.
- For "Destination = Epsilon Eridani", all instances (6,315) are classified as cluster_2.
- For "Destination = Proxima Centauri", all instances (6,382) are classified as cluster_2.
- For "Destination = Exotic Destination 3", all instances (654) are classified as cluster_2.
- For "Destination = Exotic Destination 1", all instances (625) are classified as cluster_2.

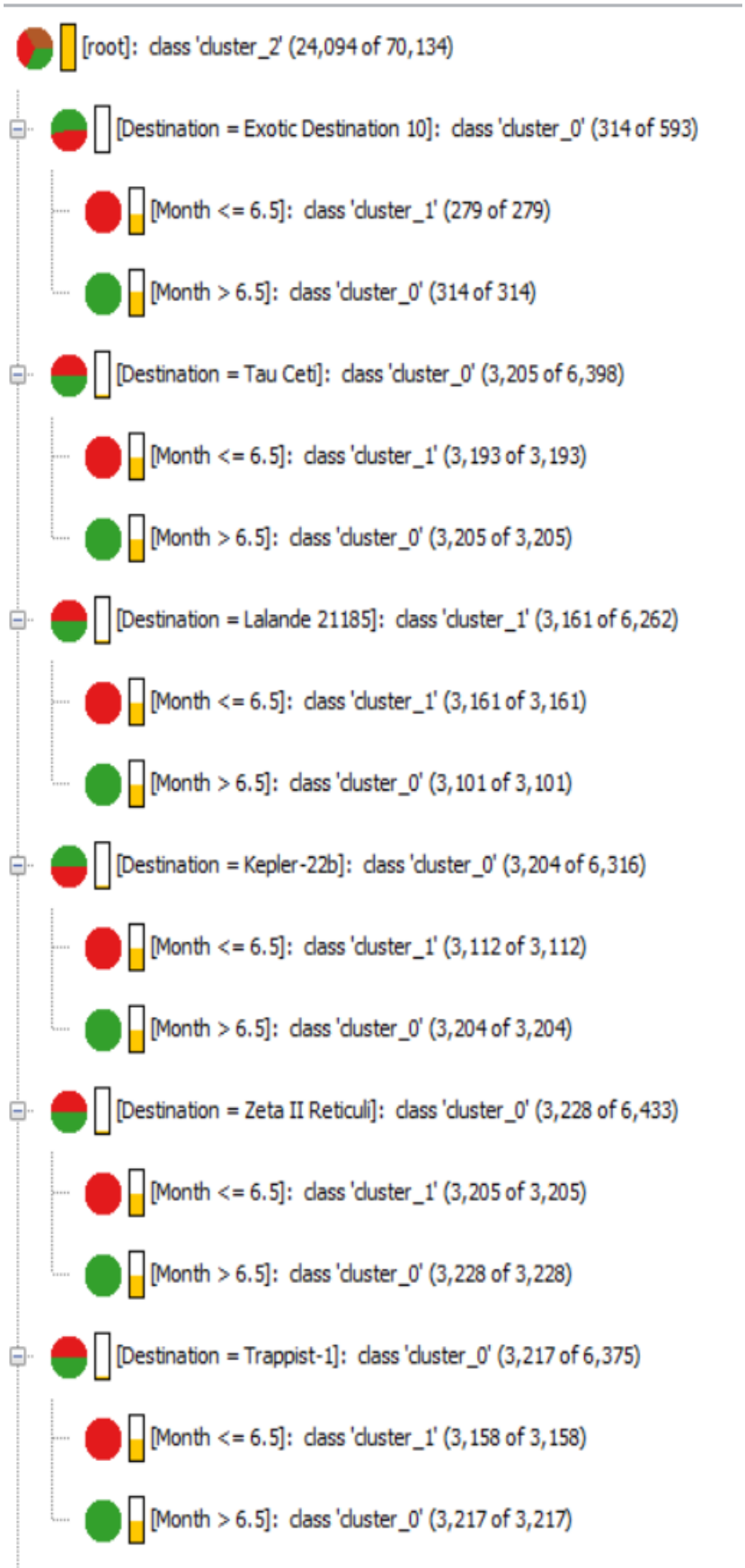
- For "Destination = Exotic Destination 5", the tree first splits based on "Month <= 7.5" into cluster_1 (255 instances) for months <= 7.5, and cluster_0 (279 instances) for months > 7.5.
- The cluster_1 branch is further split based on "Month <= 5.5" into cluster_1 (255 instances) for months <= 5.5, and cluster_2 (116 instances) for months > 5.5.
- For "Destination = Exotic Destination 8", all instances (620) are classified as cluster_2.
- For "Destination = Exotic Destination 6", all instances (628) are classified as cluster_2.
- For "Destination = Exotic Destination 7", all instances (612) are classified as cluster_2.
- For "Destination = Exotic Destination 4", all instances (659) are classified as cluster_2.

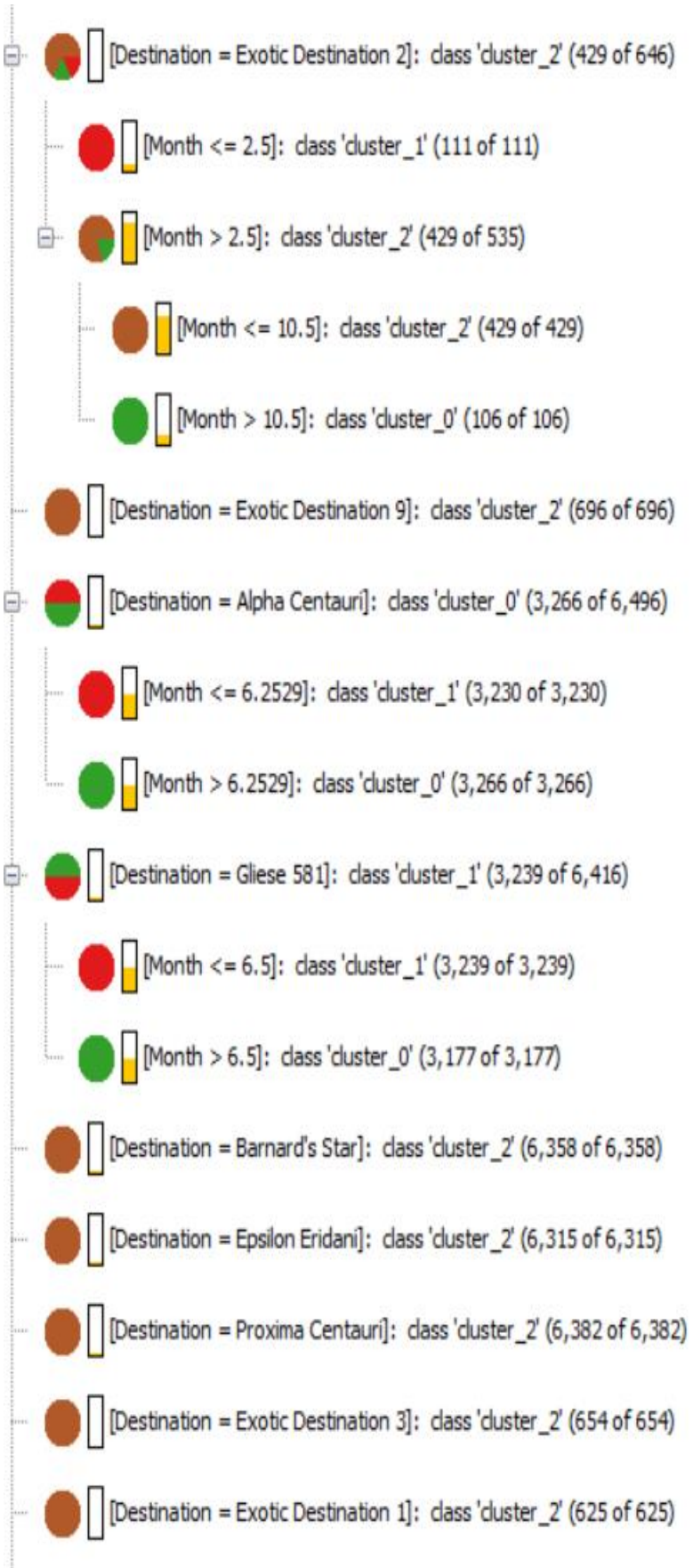
Based on the decision tree, the most relevant features and their thresholds used for splitting are:

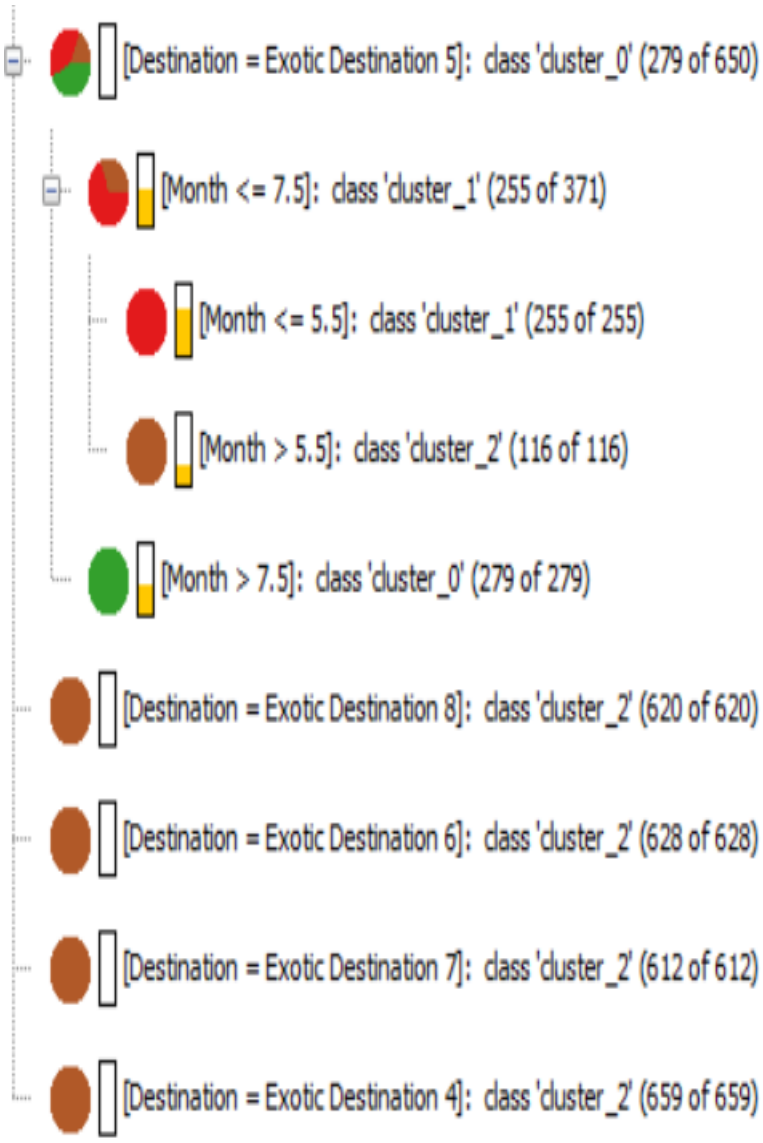
1. Destination (categorical feature, used for top-level splits)
2. Month (numerical feature, with various thresholds used for splitting)
   - Thresholds used: 2.5, 5.5, 6.5, 6.2529, 7.5, 10.5

It is possible that additional features like Duration of Stay, Travel Class, Purpose of Travel, etc., could also be relevant for further splits in the decision tree, depending on the specific dataset and algorithm parameters.

The decision tree algorithm iteratively splits the data based on the most discriminative feature at each node, learning complex decision boundaries to separate the instances into different clusters. The specific thresholds used for splitting on numerical features like "Month" highlight the intricate patterns captured by the tree, likely related to customer preferences, seasonality, and other characteristics specific to different destinations and travel periods.


[root]: class 'cluster_2' (24,094 of 70,134)
[Destination = Exotic Destination 10]: class 'cluster_0' (314 of 593)
[Month <= 6.5]: class 'cluster_1' (279 of 279)
[Month > 6.5]: class 'cluster_0' (314 of 314)
[Destination = Tau Ceti]: class 'cluster_0' (3,205 of 6,398)
[Month <= 6.5]: class 'cluster_1' (3,193 of 3,193)
[Month > 6.5]: class 'cluster_0' (3,205 of 3,205)
[Destination = Lalande 21185]: class 'cluster_1' (3,161 of 6,262)
[Month <= 6.5]: class 'cluster_1' (3,161 of 3,161)
[Month > 6.5]: class 'cluster_0' (3,101 of 3,101)
[Destination = Kepler-22b]: class 'cluster_0' (3,204 of 6,316)
[Month <= 6.5]: class 'cluster_1' (3,112 of 3,112)
[Month > 6.5]: class 'cluster_0' (3,204 of 3,204)
[Destination = Zeta II Reticuli]: class 'cluster_0' (3,228 of 6,433)
[Month <= 6.5]: class 'cluster_1' (3,205 of 3,205)
[Month > 6.5]: class 'cluster_0' (3,228 of 3,228)
[Destination = Trappist-1]: class 'cluster_0' (3,217 of 6,375)
[Month <= 6.5]: class 'cluster_1' (3,158 of 3,158)
[Month > 6.5]: class 'cluster_0' (3,217 of 3,217)


[Destination = Exotic Destination 2]: class 'cluster_2' (429 of 646)
[Month <= 2.5]: class 'cluster_1' (111 of 111)
[Month > 2.5]: class 'cluster_2' (429 of 535)
[Month <= 10.5]: class 'cluster_2' (429 of 429)
[Month > 10.5]: class 'cluster_0' (106 of 106)
[Destination = Exotic Destination 9]: class 'cluster_2' (696 of 696)
[Destination = Alpha Centauri]: class 'cluster_0' (3,266 of 6,496)
[Month <= 6.2529]: class 'cluster_1' (3,230 of 3,230)
[Month > 6.2529]: class 'cluster_0' (3,266 of 3,266)
[Destination = Gliese 581]: class 'cluster_1' (3,239 of 6,416)
[Month <= 6.5]: class 'cluster_1' (3,239 of 3,239)
[Month > 6.5]: class 'cluster_0' (3,177 of 3,177)
[Destination = Barnard's Star]: class 'cluster_2' (6,358 of 6,358)
[Destination = Epsilon Eridani]: class 'cluster_2' (6,315 of 6,315)
[Destination = Proxima Centauri]: class 'cluster_2' (6,382 of 6,382)
[Destination = Exotic Destination 3]: class 'cluster_2' (654 of 654)
[Destination = Exotic Destination 1]: class 'cluster_2' (625 of 625)


[Destination = Exotic Destination 5]: class 'cluster_0' (279 of 650)
[Month <= 7.5]: class 'cluster_1' (255 of 371)
[Month <= 5.5]: class 'cluster_1' (255 of 255)
[Month > 5.5]: class 'cluster_2' (116 of 116)
[Month > 7.5]: class 'cluster_0' (279 of 279)
[Destination = Exotic Destination 8]: class 'cluster_2' (620 of 620)
[Destination = Exotic Destination 6]: class 'cluster_2' (628 of 628)
[Destination = Exotic Destination 7]: class 'cluster_2' (612 of 612)
[Destination = Exotic Destination 4]: class 'cluster_2' (659 of 659)

Random Forest :-

Looking at the "#splits (level 0)" column, we can see that the feature with the maximum number of splits at the root level (level 0) is Destination, with 165 splits. This indicates that the Destination feature is the most important predictor variable in the model, as it was selected to create the largest number of initial splits. The feature with the second-highest number of splits at this level is Month, with 155 splits.

Examining the "#splits (level 1)" column, we see that the feature with the most splits at the first level is also Destination, with 323 splits. This further reinforces the importance of the Destination feature in the model's decision-making process. The second-most important feature at this level is Month, with 277 splits.

Moving to the "#splits (level 2)" column, we observe that Destination still has the highest number of splits, with 618 splits at this deeper level of the decision trees. This indicates that the model continues to rely heavily on the Destination feature to make increasingly granular decisions as it explores the complexities of the data.

Looking at the "#candidates (level 0)" column, we can see the number of candidate features considered for splitting at the root level. The feature with the highest number of candidates is Destination, with 687 candidates. This shows that the model had a large pool of potential split points to choose from when selecting the root node split, further emphasizing the significance of the Destination feature.

Similarly, in the "#candidates (level 1)" and "#candidates (level 2)" columns, we see that Destination maintains the highest number of candidate features, with 681 and 670 candidates, respectively. This reinforces the model's reliance on the Destination feature throughout the decision-making process.

Other features that stand out in terms of their importance include Purpose of Travel, Transportation Type, and Special Requests, which also have a relatively high number of splits and candidate

features across the different levels of the decision trees.

In summary, the attribute statistics provide valuable insights into the Random Forest model's decision-making process. The Destination feature emerges as the most important predictor variable, as evidenced by its consistently high number of splits and candidate features at all levels of the decision trees. This suggests that the model places the greatest emphasis on the destination when making classifications, followed by features such as Month, Purpose of Travel, and Transportation Type. These insights can help us better understand the model's underlying logic and the key factors driving the space travel and tourism predictions.

Row ID	#splits (level 0)	#splits (level 1)	#splits (level 2)	#candidates (level 0)	#candidates (level 1)	#candidates (level 2)
Gender	2	5	23	167	369	691
Occupation	48	74	162	205	323	690
Travel Class	12	31	56	180	355	727
Destination	0	49	127	154	359	687
Purpose of ...	34	70	116	162	333	670
Transportat...	13	41	96	174	371	690
Special Req...	47	77	162	177	347	676
Loyalty Pro...	0	12	43	176	335	724
Month	155	277	465	174	359	659
Gender (to ...	1	12	18	172	365	699
Occupation ...	6	39	88	162	320	706
Travel Clas...	9	27	42	154	361	706
Destination ...	165	323	618	165	340	681
Purpose of ...	31	40	70	176	342	685
Transportat...	19	32	60	179	365	729
Special Req...	9	34	66	187	332	656
Loyalty Pro...	1	13	37	171	342	678
Age	32	110	219	178	349	696
Distance to ...	80	115	282	187	311	720
Duration of ...	59	95	197	190	366	686
Number of ...	29	66	135	157	364	743
Price (Gala...	137	252	484	178	353	699
Customer S...	111	206	406	175	339	702