MIS772 – Predictive Analytics
T2 2024

Assignment 1 – Individual
Student name: Huy Hung Pham
Student number: s224212292

Executive summary (1 page)

Executive Problem Statement:

DAX-Air's objective is to improve service quality and customer experience by better understanding traveler satisfaction. This can be achieved by analyzing traveler reviews of various aspects of the flight experience.

Executive Solution Statement:

We have developed two predictive models: a Decision Tree and a k-Nearest Neighbors (k-NN) classifier. Both models were evaluated using cross-validation to ensure robust performance metrics. We also used an ensemble technique with majority voting, which improved accuracy to 91.84% and Kappa to 0.83 (see Figure 11).

Recommendations:

1. Implement Voting Ensemble Model: Utilize this model to predict traveler satisfaction, leveraging the strengths of both the Decision Tree and k-NN models.

2. Monitor and Refine Models: Continuously update models with new data to maintain accuracy.

Anticipated Benefits:

- Improved Customer Satisfaction: Accurately identify and address traveler concerns to enhance service quality.

- Informed Decision-Making: Use insights from the models for data-driven improvements to the travel experience.

Data exploration, pattern discovery, and preparation (2 pages)

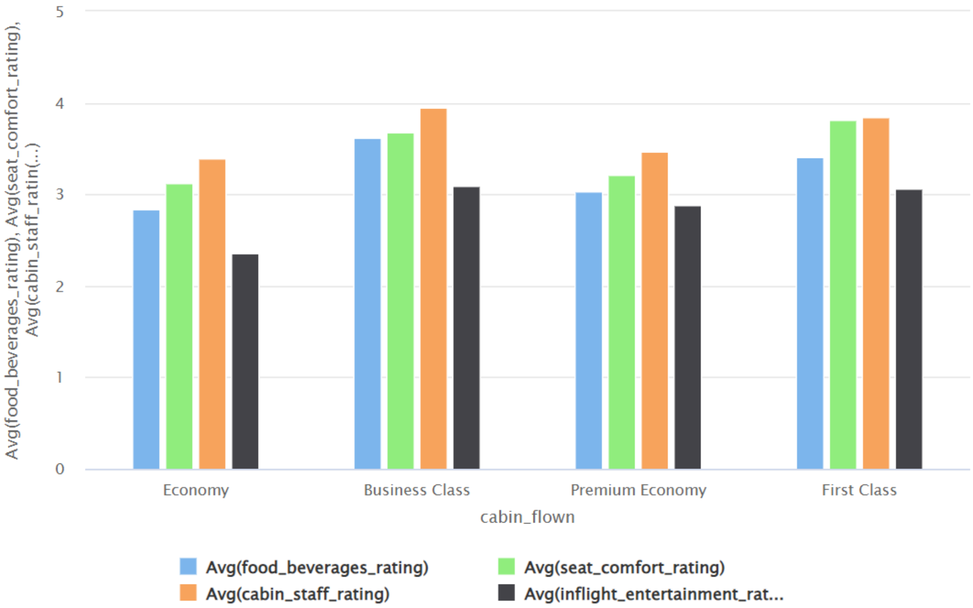
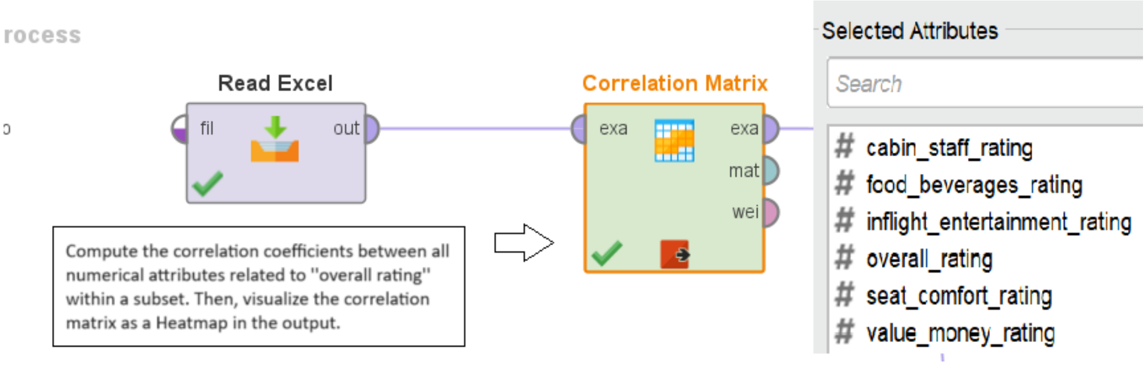


Figure 1: Comparison of Ratings by Cabin Class

The chart compares ratings for different cabin classes. First and Business Class have the highest ratings for all categories. Premium Economy also scores well, especially in seat comfort and food and beverages. Economy class has the lowest ratings, particularly in inflight entertainment and cabin staff.



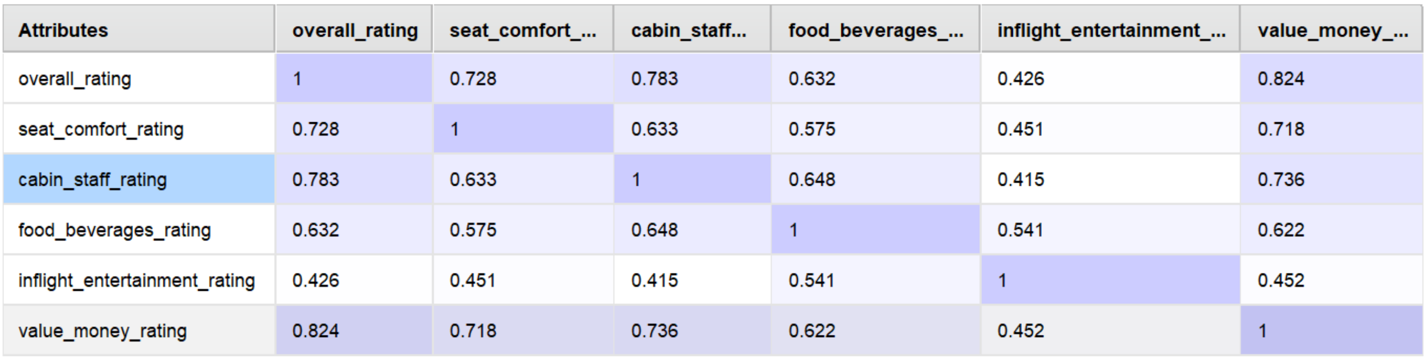
Compute the correlation coefficients between all numerical attributes related to "overall rating" within a subset. Then, visualize the correlation matrix as a Heatmap in the output.

Selected Attributes

Search

- # cabin_staff_rating
- # food_beverages_rating
- # inflight_entertainment_rating
- # overall_rating
- # seat_comfort_rating
- # value_money_rating

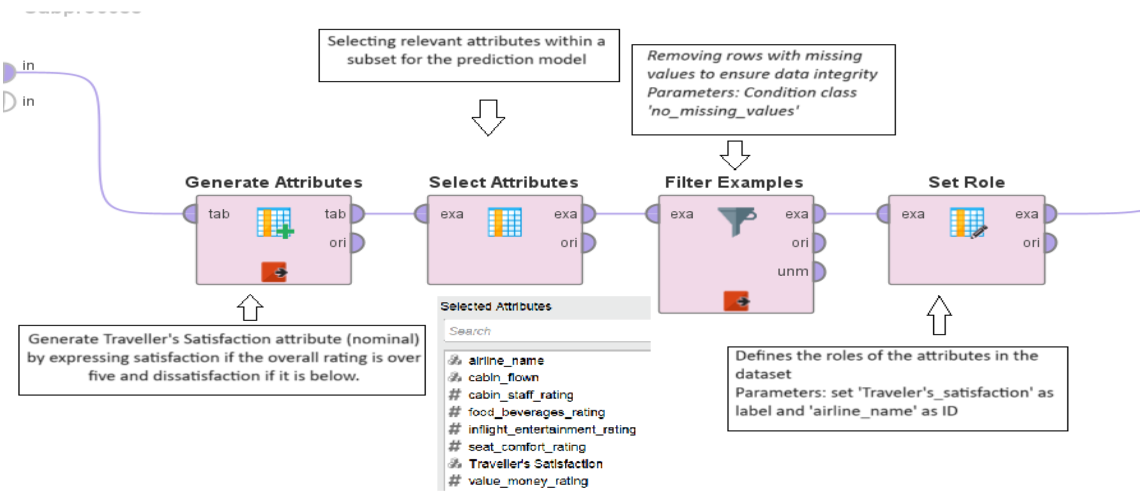
Figure 2: Attribute Correlation Process in RapidMiner



Attributes	overall_rating	seat_comfort_...	cabin_staff...	food_beverages_...	inflight_entertainment_...	value_money_...
overall_rating	1	0.728	0.783	0.632	0.426	0.824
seat_comfort_rating	0.728	1	0.633	0.575	0.451	0.718
cabin_staff_rating	0.783	0.633	1	0.648	0.415	0.736
food_beverages_rating	0.632	0.575	0.648	1	0.541	0.622
inflight_entertainment_rating	0.426	0.451	0.415	0.541	1	0.452
value_money_rating	0.824	0.718	0.736	0.622	0.452	1

Figure 3: Correlation Matrix of Airline Customer Satisfaction Attributes visualised by a Heat Map

I have connected the 'Correlation_Matrix' operator and found a positive correlation between 'value_for_money_rating' and 'overall_rating.' Additionally, cabin staff and seat comfort are strongly linked to overall satisfaction and perceived value for money.



Selecting relevant attributes within a subset for the prediction model

Removing rows with missing values to ensure data integrity Parameters: Condition class 'no_missing_values'

Generate Traveller's Satisfaction attribute (nominal) by expressing satisfaction if the overall rating is over five and dissatisfaction if it is below.

Selected Attributes

Search

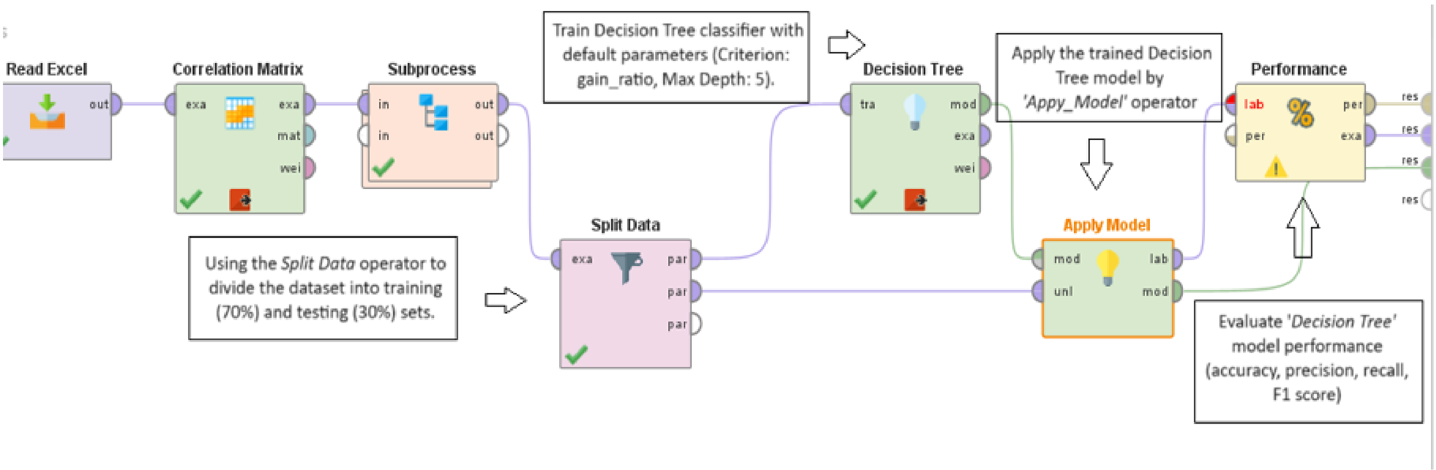
- airline_name
- cabin_flown
- # cabin_staff_rating
- # food_beverages_rating
- # inflight_entertainment_rating
- # seat_comfort_rating
- Traveller's Satisfaction
- # value_money_rating

Defines the roles of the attributes in the dataset
Parameters: set 'Traveler's_satisfaction' as label and 'airline_name' as ID

Figure 4: Data preparation process in RapidMiner 2024

In the Subprocess operator, I created a new attribute named "Traveler's Satisfaction" based on the "overall_rating". Then, I addressed missing values using the Filter Examples operator. I chose "Traveler's Satisfaction" as the label attribute for prediction and "airline_name" as the ID attribute to identify each record.

Predictive modelling (2 pages)



Train Decision Tree classifier with default parameters (Criterion: gain_ratio, Max Depth: 5).

Using the *Split Data* operator to divide the dataset into training (70%) and testing (30%) sets.

Apply the trained Decision Tree model by *'Appy_Model'* operator

Evaluate *'Decision Tree'* model performance (accuracy, precision, recall, F1 score)

Figure 5: Decision Tree model for predicting traveller's satisfaction

I used the Split Data operator to divide the dataset into a 70% training set and a 30% testing set for Holdout validation. I set the random seed to 2001 for consistency. Additionally, I limited the maximum depth to 5 for the Decision Tree operator to avoid overtraining. The trained model was tested using the Apply Model operator and the Performance (Classification) operator provided performance metrics in the Result view.

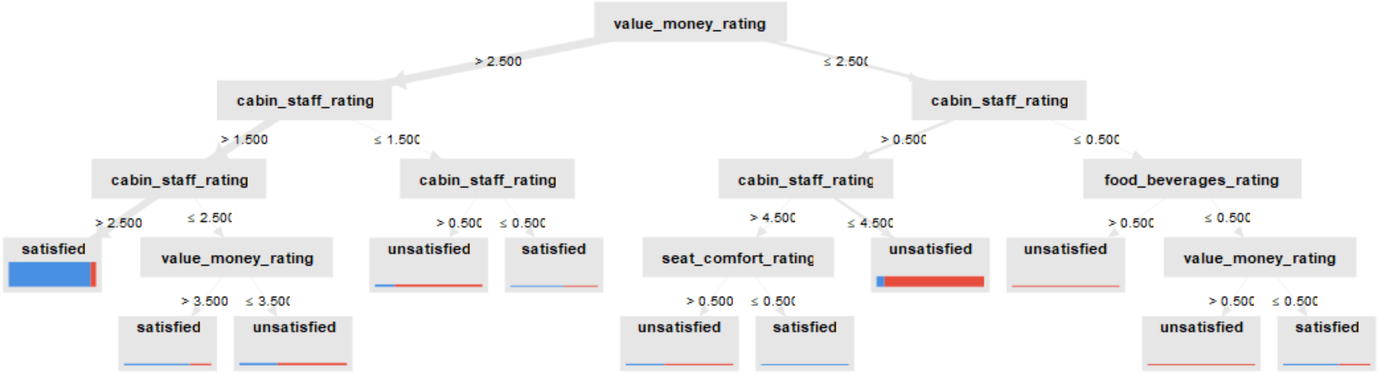
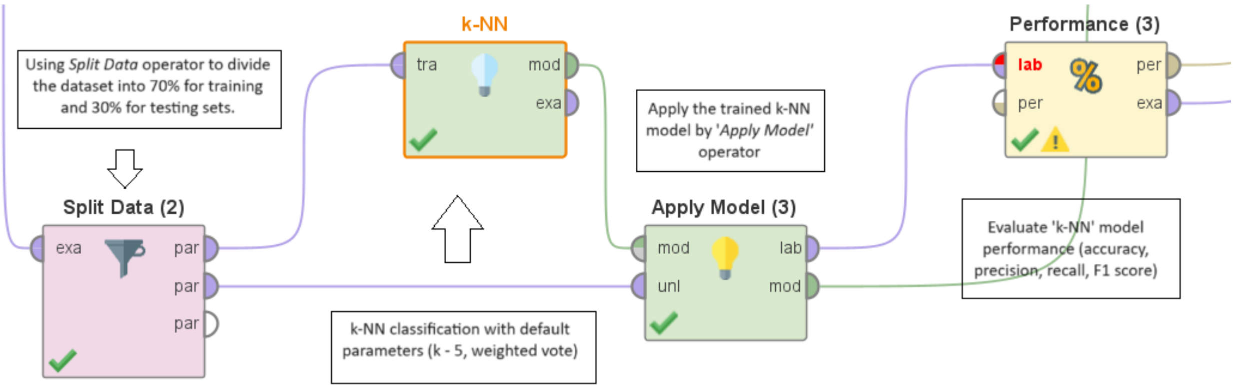


Figure 6: The Decision Tree model of Traveller's satisfaction

I used the Decision Tree model to analyze the data. I adjusted the parameters to prevent overfitting by setting the "max_depth" at 5 instead of 10. In the tree diagram, the "value_money_rating" attribute is the most significant factor in determining traveler satisfaction. The "cabin_staff_rating" attribute also plays a crucial role. The "seat_comfort_rating" attribute becomes important when the cabin staff and value for money ratings are low.



Using *Split Data* operator to divide the dataset into 70% for training and 30% for testing sets.

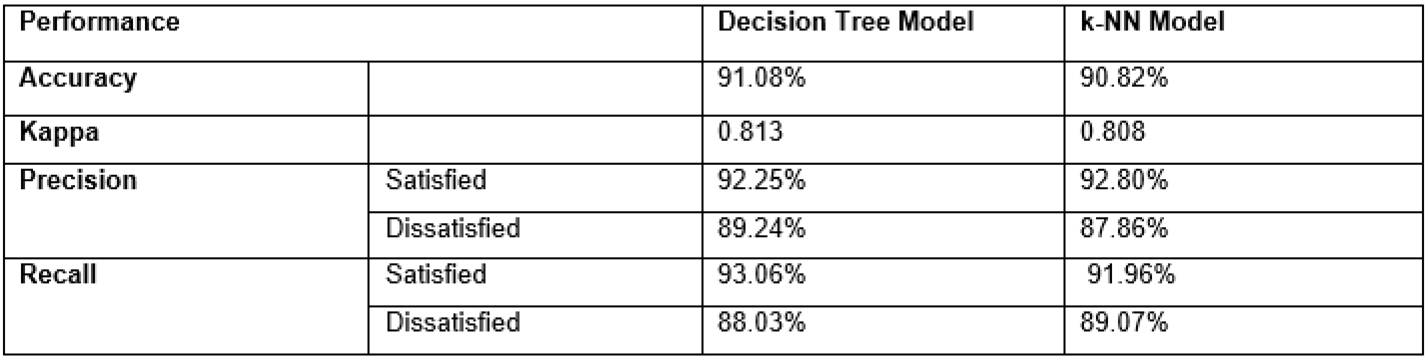
Apply the trained k-NN model by '*Apply Model*' operator

Evaluate 'k-NN' model performance (accuracy, precision, recall, F1 score)

k-NN classification with default parameters (k - 5, weighted vote)

Figure 7: The k-NN classifier model

The k-NN model has classified instances into two classes: "satisfied" and "dissatisfied" based on their 6 features (dimensions) on 20968 examples. The summary indicates that the model uses 5 neighbors for classification, where the neighbors' votes are weighted, likely based on inverse distance, giving closer neighbors more influence.

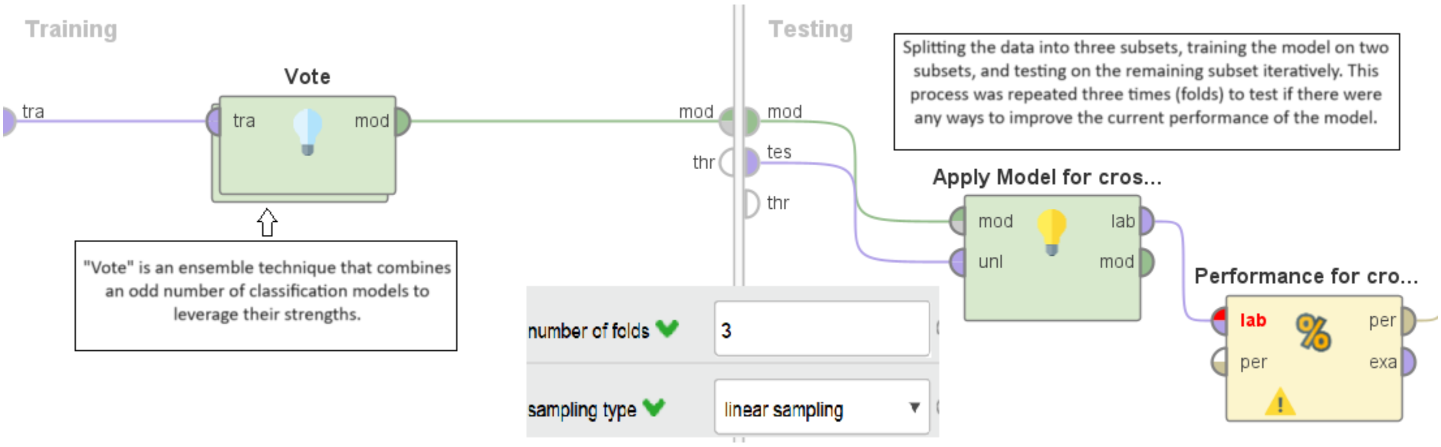


Performance		Decision Tree Model	k-NN Model
Accuracy		91.08%	90.82%
Kappa		0.813	0.808
Precision	Satisfied	92.25%	92.80%
	Dissatisfied	89.24%	87.86%
Recall	Satisfied	93.06%	91.96%
	Dissatisfied	88.03%	89.07%

Figure 8: The comparation of performance of two classifier models.

After comparing both classifier models, the Decision Tree model had slightly higher accuracy (91.08% vs. 90.82%) and a slightly better Kappa statistic (0.813 vs. 0.808), indicating better overall agreement between predicted and actual labels. We chose the Decision Tree for the next step because of its slightly better performance, balanced precision and recall for both classes, and its visualization for data-driven decision-making.

Model evaluation and improvement (2 pages).



Splitting the data into three subsets, training the model on two subsets, and testing on the remaining subset iteratively. This process was repeated three times (folds) to test if there were any ways to improve the current performance of the model.

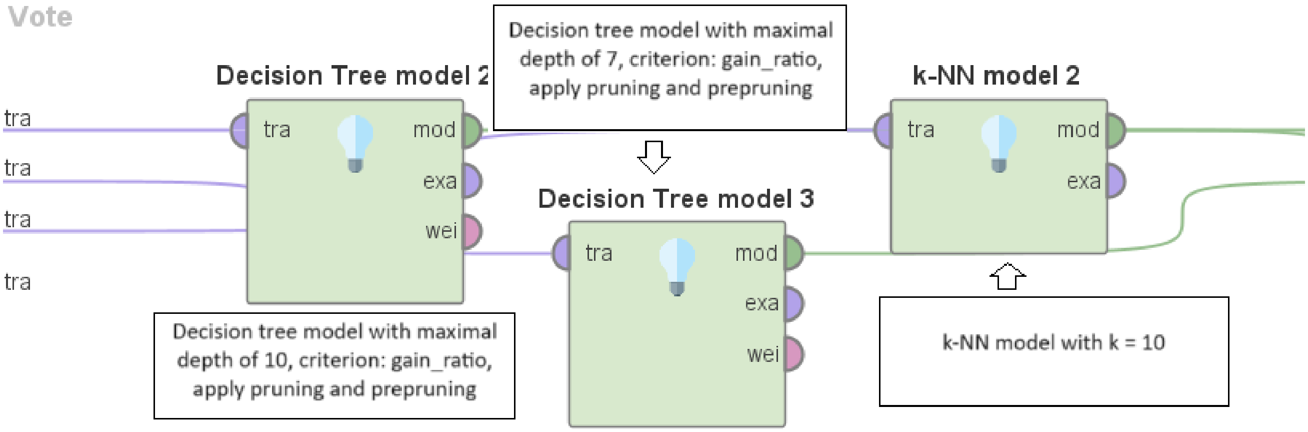
"Vote" is an ensemble technique that combines an odd number of classification models to leverage their strengths.

number of folds: 3

sampling type: linear sampling

Figure 9: k-fold cross-validation process in RM

I have used a Voting ensemble technique in a 3-fold cross-validation framework to boost model performance by leveraging multiple classification models and enhancing prediction accuracy.



Decision tree model with maximal depth of 7, criterion: gain_ratio, apply pruning and prepruning

Decision tree model with maximal depth of 10, criterion: gain_ratio, apply pruning and prepruning

k-NN model with k = 10

Figure 10: Voting ensemble technique inside 3-fold Cross validation process.

In the voting ensemble, I tried different Decision Tree models with depths of 10 and 7 and a k-NN model with a larger k of 10. I need help deciding which model works best and provides a new prediction accuracy to compare with the two remaining models I have created.

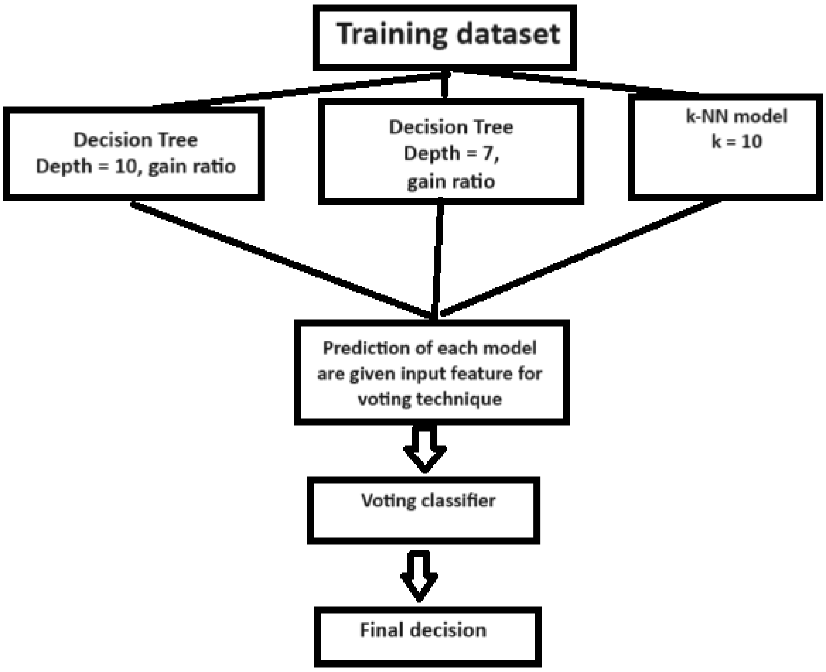
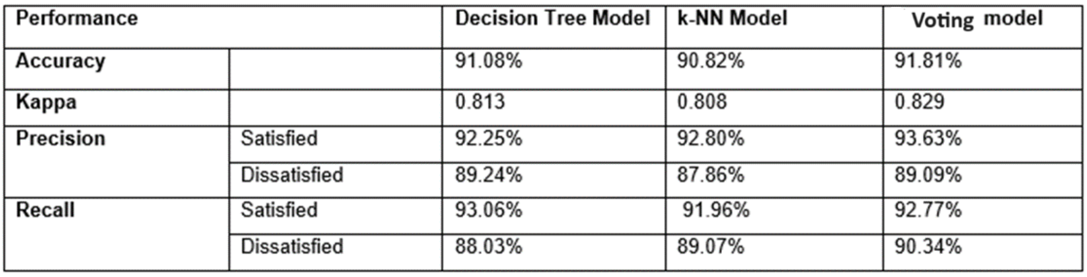


Figure 11: Structure of a Voting model

I used a Voting model with attribute-based to combine predictions from three base models in Figure 10. This improved prediction accuracy and robustness by leveraging the strengths of each base model. I compared it with the previous model to assess performance improvement.



Performance		Decision Tree Model	k-NN Model	Voting model
Accuracy		91.08%	90.82%	91.81%
Kappa		0.813	0.808	0.829
Precision	Satisfied	92.25%	92.80%	93.63%
	Dissatisfied	89.24%	87.86%	89.09%
Recall	Satisfied	93.06%	91.96%	92.77%
	Dissatisfied	88.03%	89.07%	90.34%

Figure 11: Performance Comparison of Decision Tree, k-NN, and Voting Models

After a comprehensive analysis, the Stacking Model emerges as the optimal choice for future predictions. Its impressive accuracy and balanced precision, along with consistent insights, empower better decision-making and enhance customer satisfaction.