FOUNDATIONS OF DATA SCIENCE
PROJECT REPORT

Team Members

Aashish Kumar - ERP 23108
Muzammil Imran - ERP 22919
Syed Fazeel Junaid - ERP 23100

Dataset

Chosen from a Live Kaggle Competition
Link: https://www.kaggle.com/competitions/spaceship-titanic

Problem Description

In the future, a spaceship called the Titanic is on its first mission to transport passengers from our solar system to three new planets. Unfortunately, the spaceship collided with a spacetime anomaly concealed within a dust cloud and as a result, about 50% of the passengers have been transported to an alternate dimension. The task is to use records from the ship and predict which passengers were transported.

Business Understanding

The problem can be defined as a data mining issue as the number of records available are huge (around 13000 passengers). Our motivation to work on this
Since data for all passengers is available, the collection of data does not need to be carried out again and so valuable time can be saved. Time is essential, as any passenger transported to an alternate dimension needs to be saved as soon as possible before they are stuck forever.
The success criteria are to correctly identify as many passengers as possible who have been transported and bring them back.

Challenges Faced

- A large percentage of records available had at least one missing value. Data for those missing attributes had to be predicted which will always have deviation from real data.
- The data for some attributes was in numerical form while for others it was in categorical form. This led to error in some of our models.
- Some models were computationally time consuming as the dataset was quite large

Data Description

Our data had around 13000 records of which about 2/3 (i.e ~8700 records) were training data while the rest were records where we had to predict whether that respective passenger was transported.
The dataset had 14 attributes, one of which is the class attribute i.e Transported which had values of True of False
The other attributes are as follows

- PassengerId – Unique for each passenger. People in same groups were usually family members
- HomePlanet – One of three planets: Europa, Earth, Mars
- CryoSleep – Those in cyrosleep were restricted to cabin only
- Cabin – Cabin of stay for each passenger
- Destination – One of three planets where passengers were being originally transported to
- Age – A number
- VIP – Paid member for special service
- RoomService – Amount billed during voyage
- FoodCourt – Amount billed during voyage
- ShoppingMall – Amount billed during voyage
- Spa – Amount billed during voyage
- VRDeck – Amount billed during voyage
- Name – Only first and last name

The test data, as required, did not have the transported attribute which had to predicted
The sample file instructed that only PassengerId and Transported attributes should be in present in our submissions

Data Pre-processing

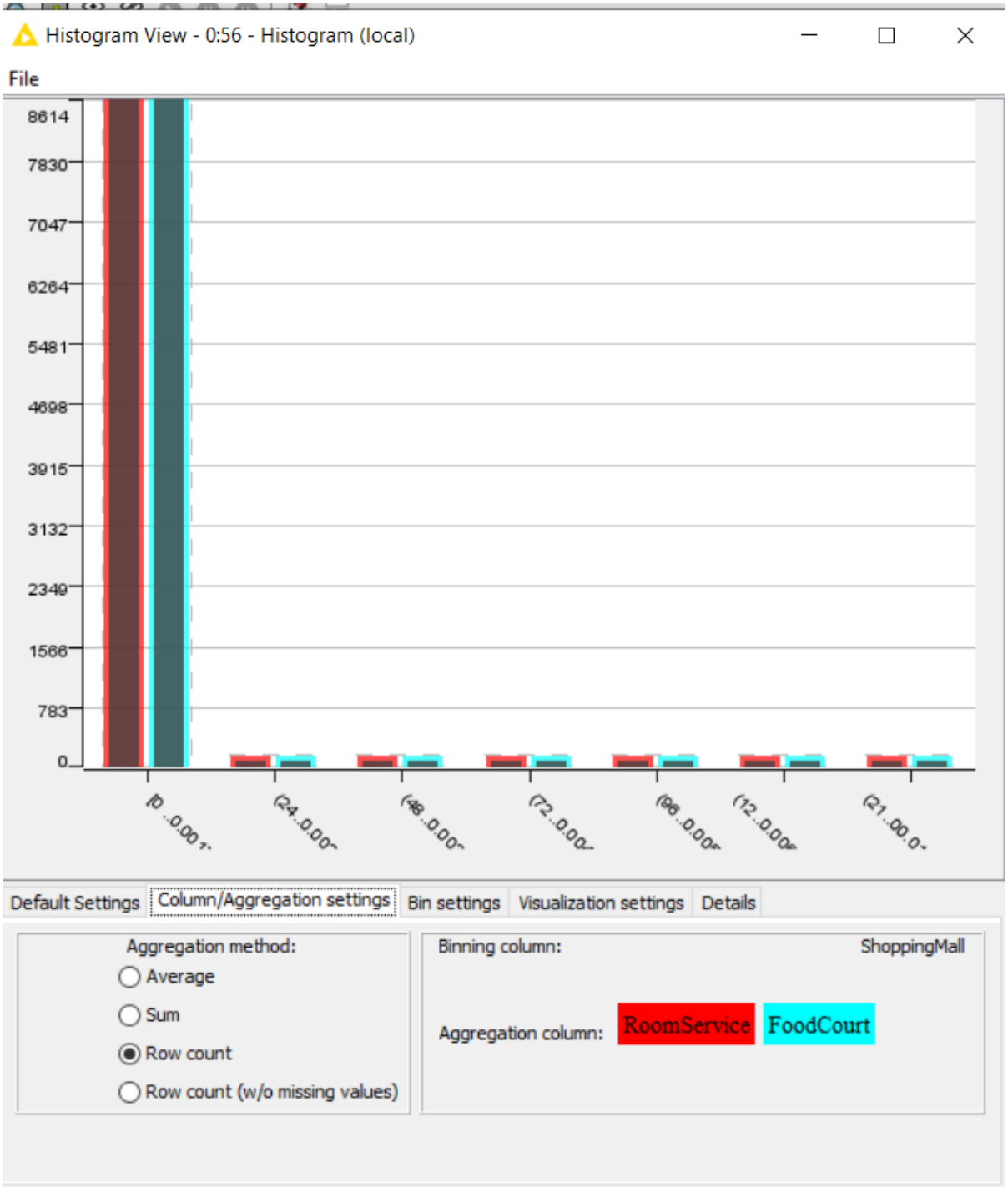
To counter the problem of missing data for some attributes for some records, we used a Missing Value node for all of our models. In the configuration, the missing string values were replaced by the Most Frequent Value of that attribute. For missing numerical values, the mean of all the values present for that attribute was used as placeholder.
For some of our models such K Nearest Neighbour, we also used Category to Number Node in some of our attempts to convert all our data to numerical form so that our data has consistency in its form.
Likewise, for some of our attempts in all our different models, we used Normalizer node to normalize data where extremely large values and extremely small values coexisted (such as in the amount of bill calculated for RoomService, ShoppingMall etc.) Different forms of Normalization such as Min-Max Normalization and Decimal Scaling were used to see which were most suited to our data. A Denormalizer node was subsequently added as well in the workflow to finalize our data with original values.

Tools/Software

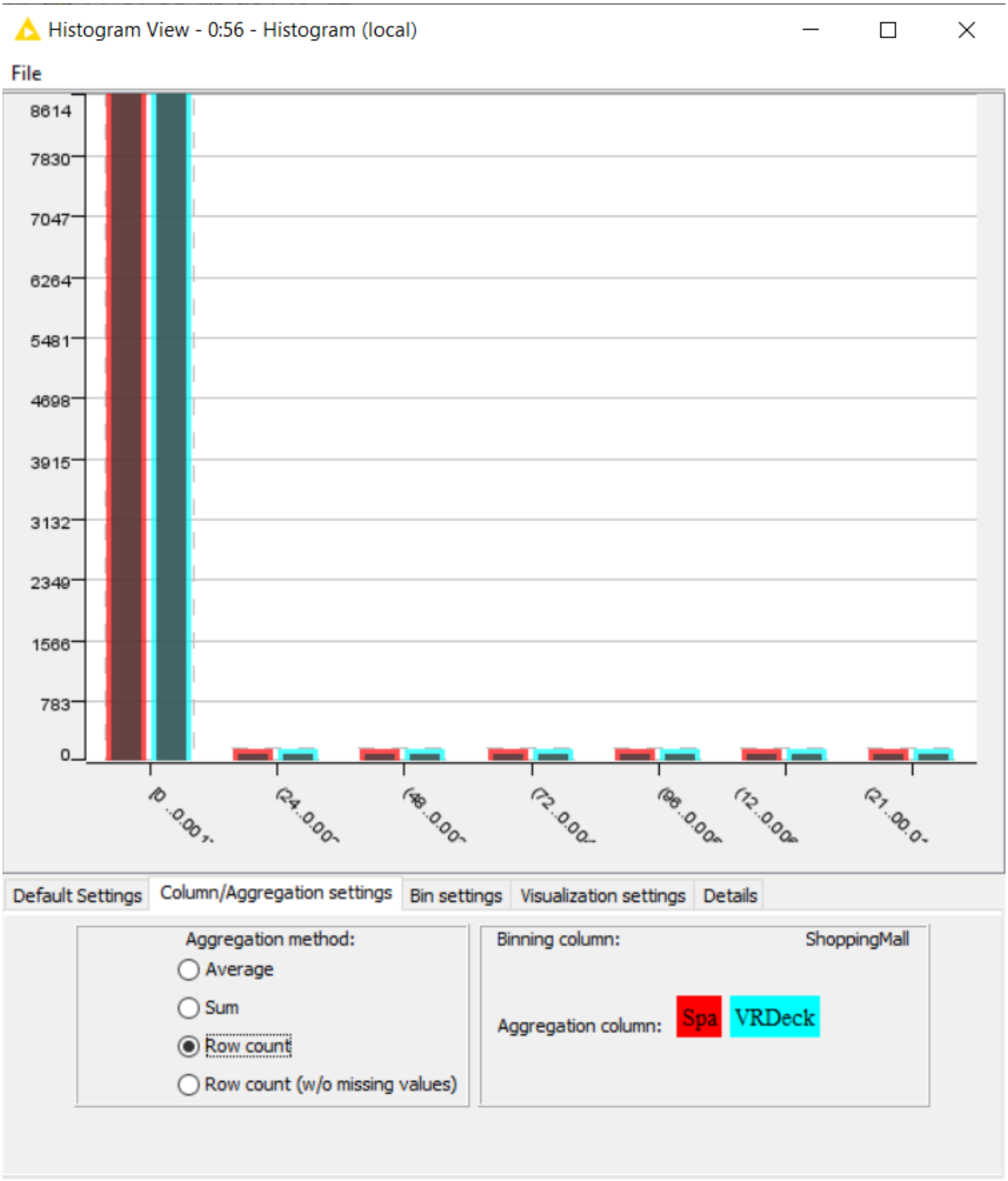
All modelling, visualization and work is done on latest version of KNIME

Data Visualization

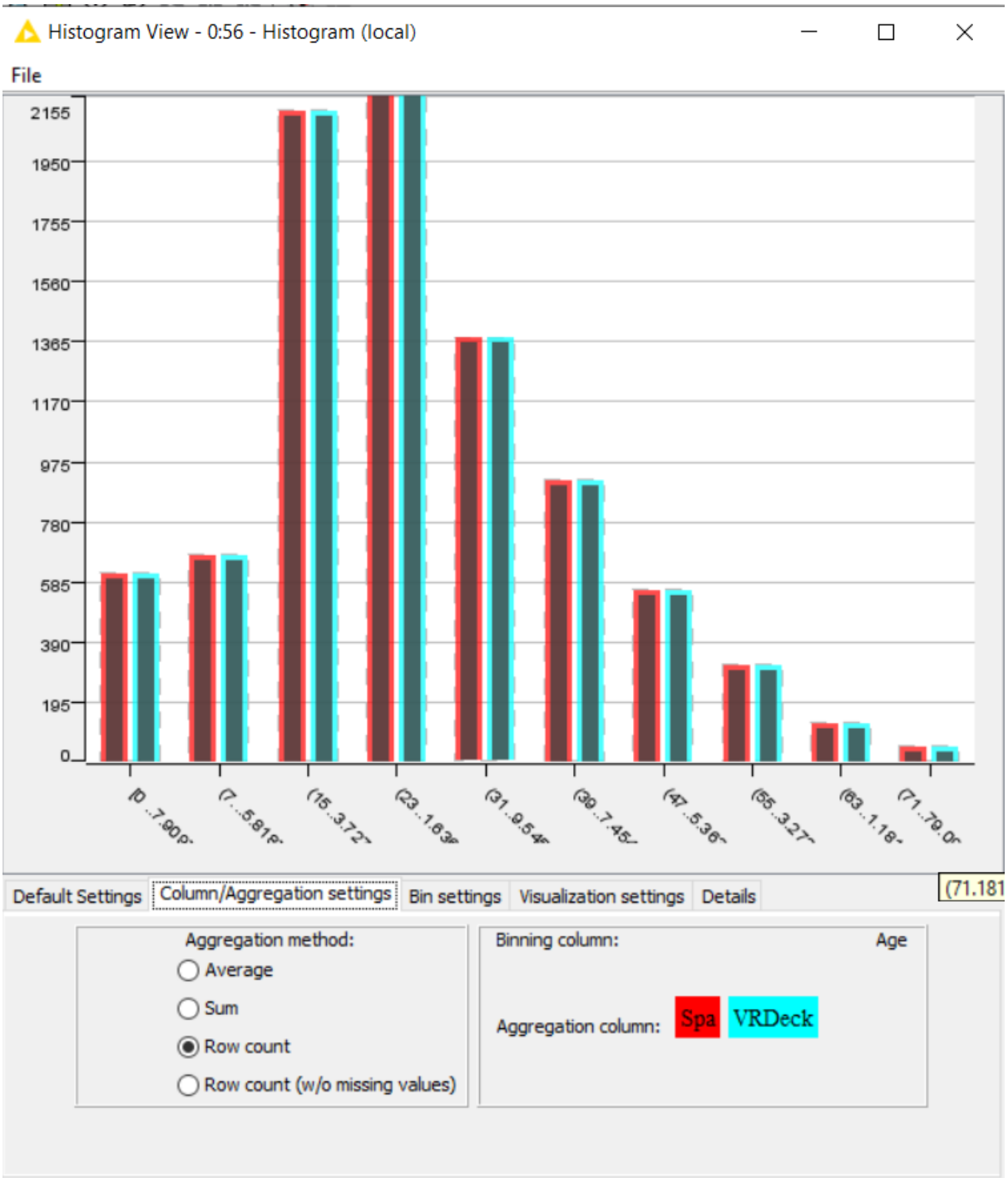
Using scatter plots and histograms, we were able to find a few trends between the different attributes and have a better insight to our dataset.



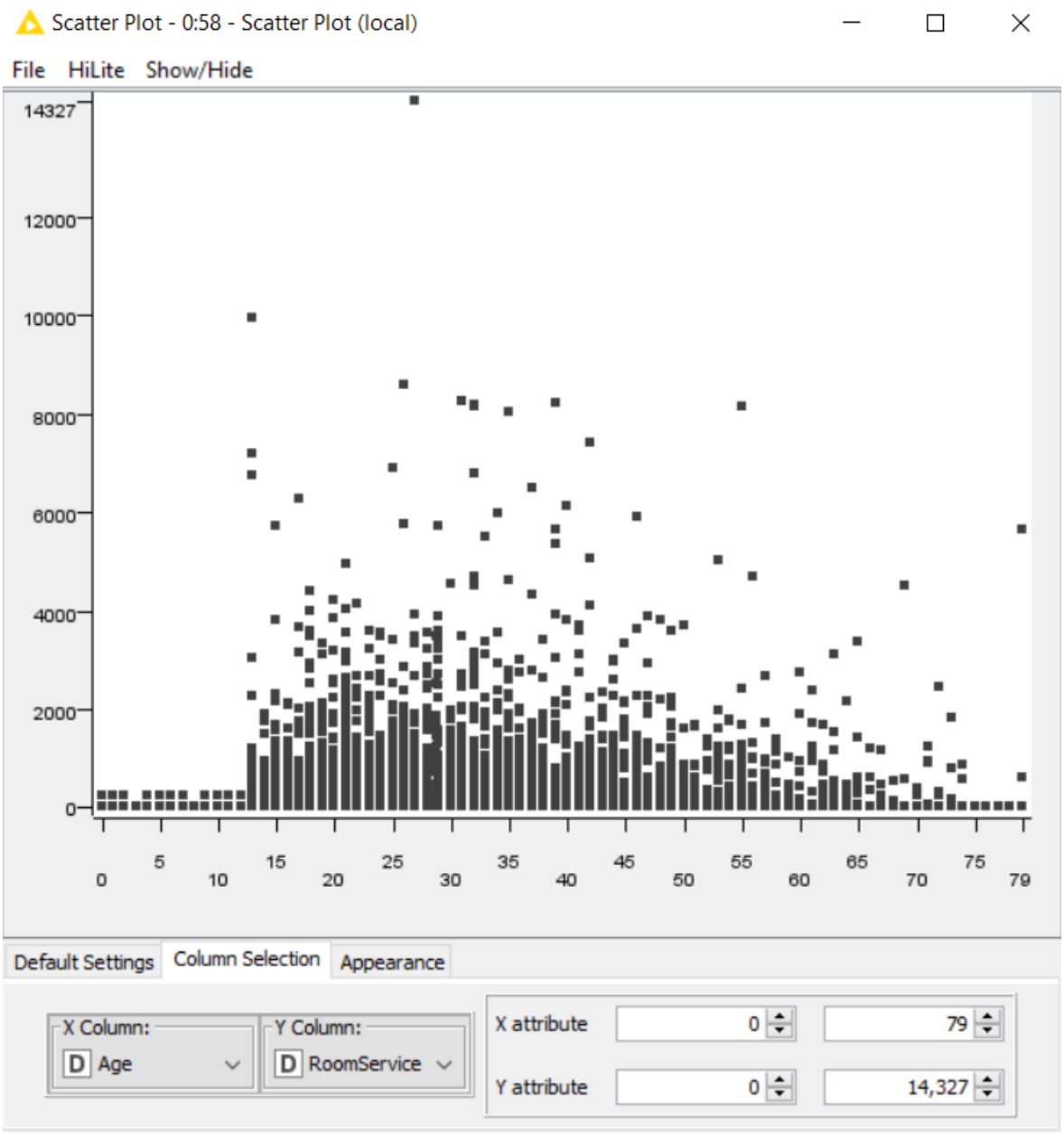
Here the bar chart shows that the people who do not spend much in the shopping mall end up spending in room service and food court.



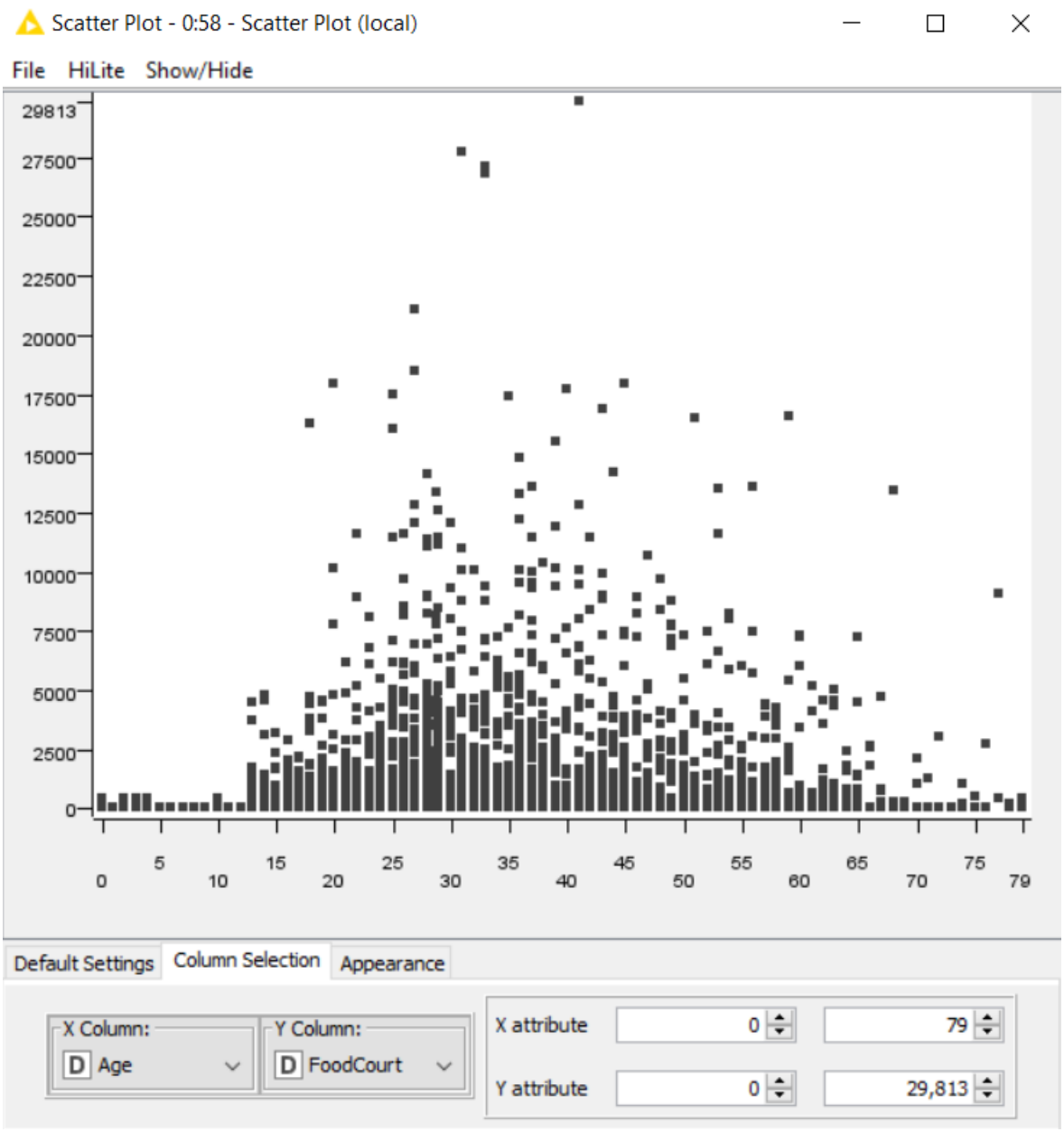
This shows that majority of the people that did not spend much in the shopping mall end up going in Spa and VR Deck.



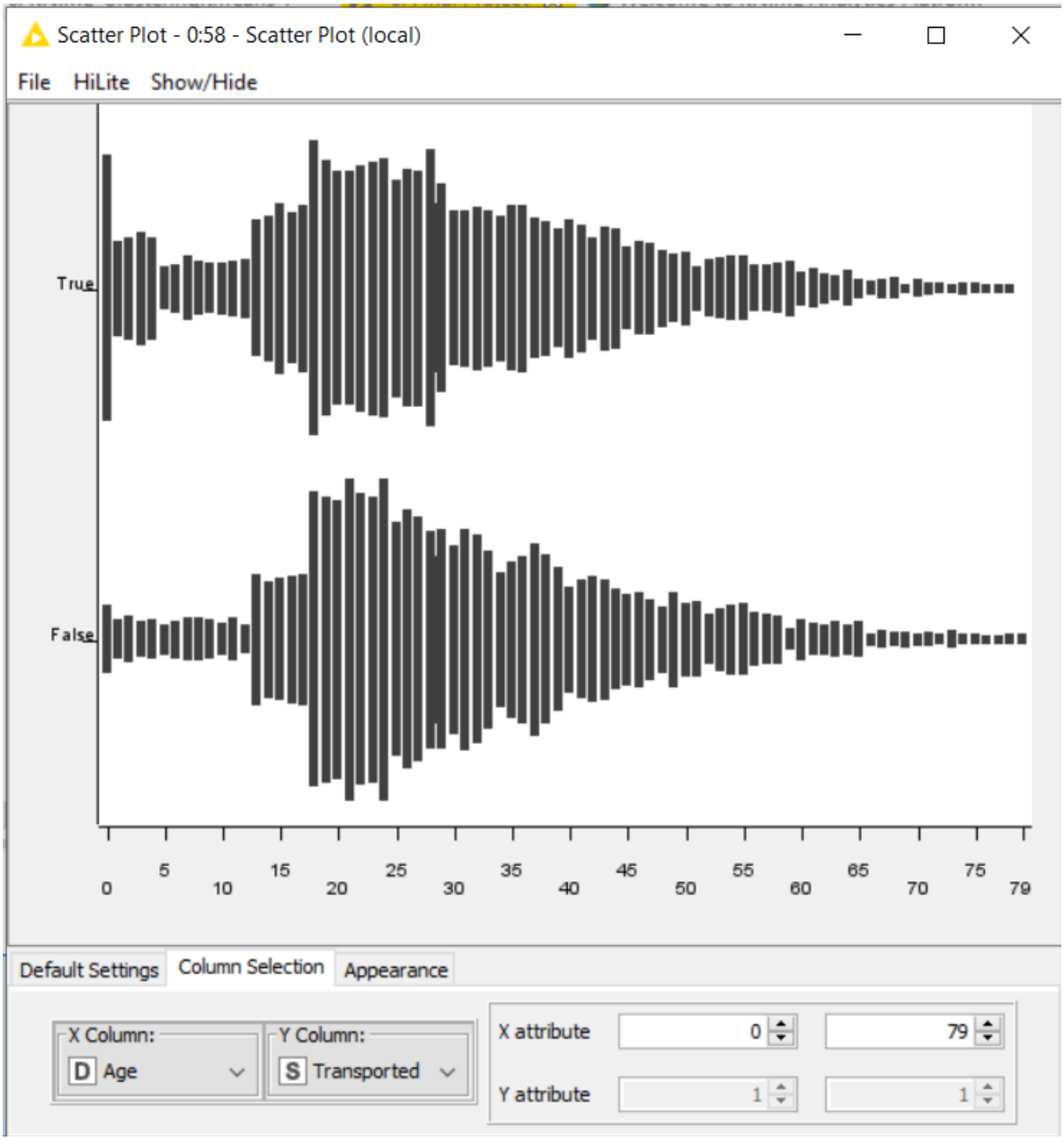
Majority of the people in the age bracket of 15 to 31 are to found spending on Spa and VRDeck.



Children do not spend much on room services and it makes sense.



Majority of the people in the prime age of 15 to 60 are spending the most money in the food court.



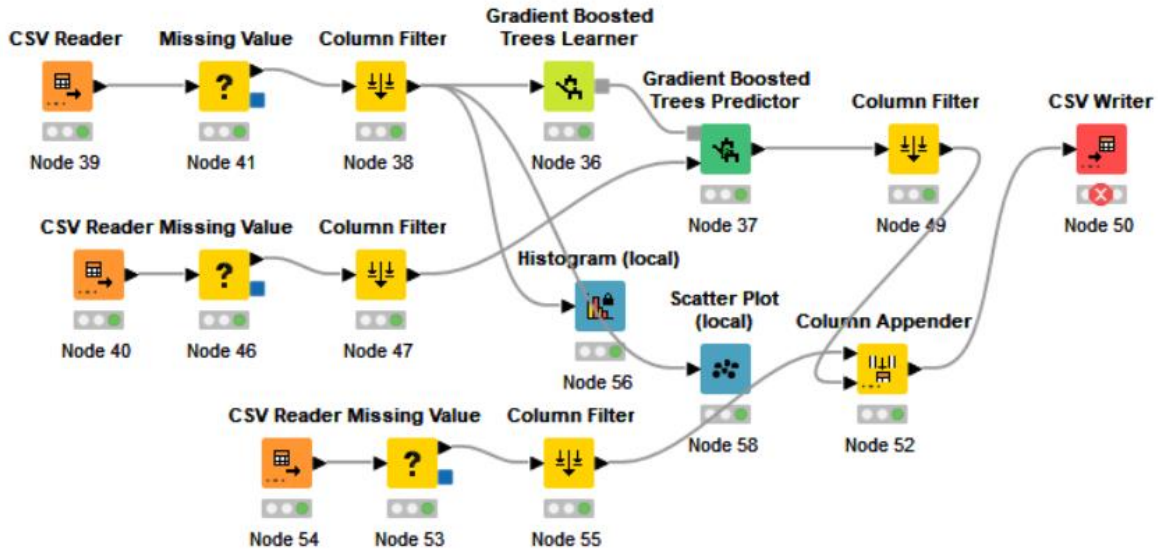
Age does not play a significant role in differentiating if the passenger was transported. However, an important fact to note is that a lot of passengers with age 0(infants) have been transported.

Modelling

The following models were tried

1. Gradient Boosted
2. Naïve Bayes
3. K Nearest Neighbour
4. Decision Tree
5. Random Forest
6. Tree Ensemble
7. Logistic Regression

Gradient Boosting



The Workflow model that yielded the highest score (above screenshot)

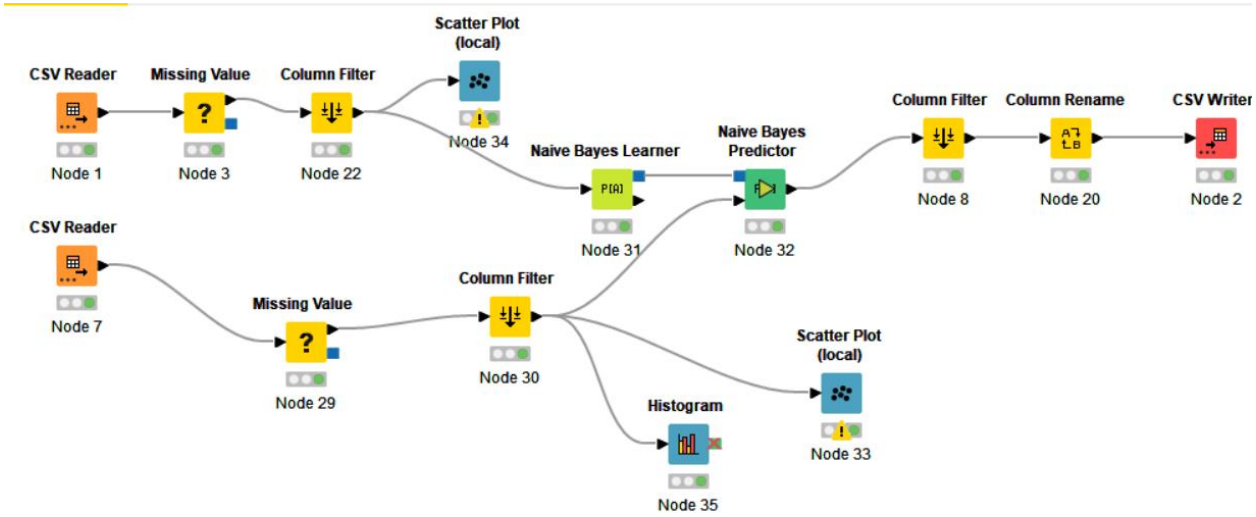
Using the CSV reader node, we import the train and test data to Knime. I used the missing value node to replace the missing values in the train and test data. The string value was set to be replaced by the most frequent value while the missing numerical values were replaced by the mean. Using the Column filter node, I removed the passenger ID and the name columns as they are clearly irrelevant in the learning and predicting process.

We start with a random starting point of 0.01 learning rate, 1000 models and limit tree depth to 5. We will use different permutation to optimize the result and increase the score, while also analyzing trends. Using the gradient boosted node set at 0.01 learning rate, 1000 models and tree depth 5, the score was 0.79144. Changing the learning rate to 0.001 while the other variables remained constant increased the score to 0.79191 while changing it to 0.002 decreased the score to 0.79167. Next, we set the learning rate to 0.1 resulting in a score of **0.77881.** Hence, we conclude that the ideal learning rate is 0.001.

Since we know the ideal learning rate is 0.001, we change the depth to no limit while models are 1000 which results in 0.73041. Changing the depth to 3, 7, 10 and 20 results in **0.76782,** 0.79191, 0.78700

and 0.73906 respectively. The highest score up till now is 0.79191. Using normalizer reduced the score to 0.79120. Removing Cabin column did not change the score of 0.79191 therefore cabin column is also of no significance and can be considered irrelevant. Removing destination and home planet along with cabin name and ID also reduced the score to 0.78770.
Now we move on to applying different permutations to increase the score by adjusting all values simultaneously. We lower the models to 500, with learning rate 0.001 and depth limited to 5 that results in 0.78442. In the next attempt we change the learning rate to 0.01 which increases the score to 0.79682. We learn that 0.001 is not necessarily the idea learning rate. We lower the depth to 3 and the score increases to **0.79775**. Next, we lower the models to 300 that results in another increase to **0.79798. Lowering the models to 200, lowers the score to 0.79307**.
Therefore, we conclude that the highest possible score attained through gradient boosted tree leaner is 0.79798 with optimum learning rate 0.01, models 300 and depth 3.

Naïve Bayes



The workflow model that gave the highest score (above screenshot)

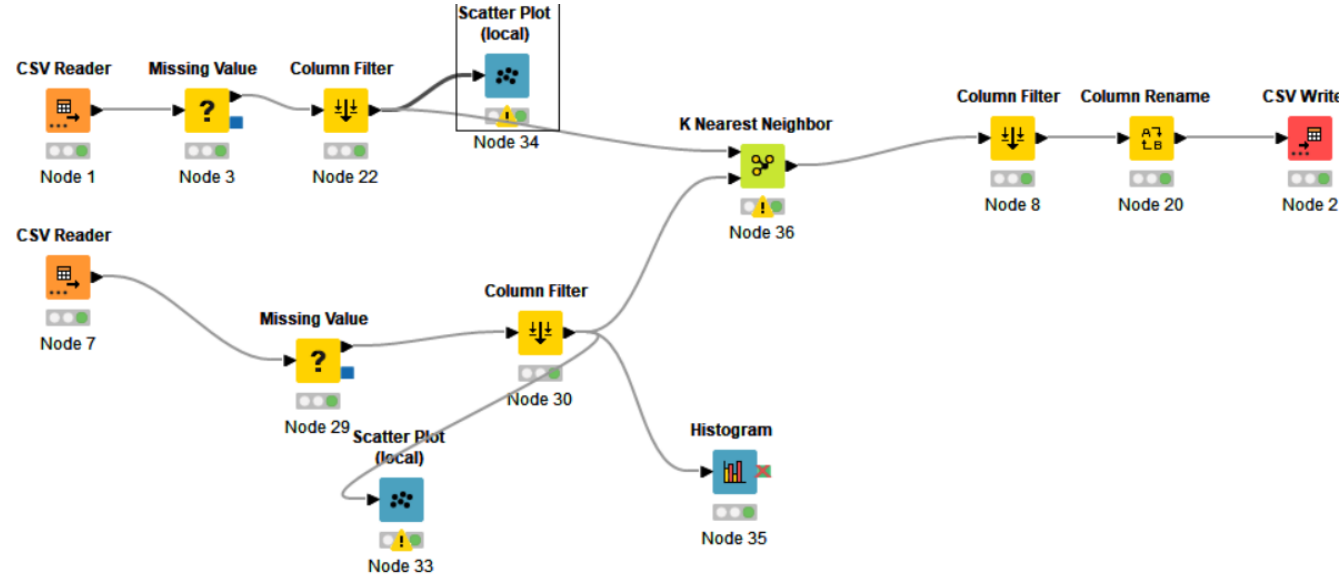
Using the CSV reader node, import the train and test data. The missing value node is used to substitute values for the missing values in the train and test data. The string value was set to be replaced by the most frequent value while the missing numerical values were replaced by the mean. Using the Column filter node, the passenger ID and the name columns are removed due to being irrelevant in our modeling.
For the naïve bayes learner node, the box to ignore missing values is checked. All the other configurations are kept at default. These are: default probability=0.0001, minimum standard deviation=0.0001, threshold standard deviation=0.0 and max number of unique nominal values=20. The predictor node was left completely on default settings. The column filter node at the end was used to only allow for the prediction column and passenger ID columns to be displayed (as instructed

in the submission file). The column rename node was to change the column name as instructed (i.e Transported and PassengerId). The submission scored 0.56651 on Kaggle.
Tried different computations next to achieve a higher score. In the naïve bayes learner node, unticked the box of ignoring missing values. The score remained fixed at 0.56651. Expected as we had already tackled missing values in the dataset with the node.
Next the default probability values (DP) were changed, while all other variables were kept constant. At 0.001 DP, the score was 0.70072. Then the default probability was kept at several values such as 0.0015, 0.002, 0.0025, 0.0030. From these 4 values the highest score was achieved at DP=0.002 which gave the score of 0.71942. Further testing at DP=0.0019 and DP=0.0021 resulted in scores getting lower at each side which meant that for best result, DP had to be set at 0.002.
With this DP set, we changed the minimum standard deviation (MSD). Took a range of values from the initial 0.0001 to 0.001, 0.002, 0.005, 0.01, 0.05, 0.1, 0.5, 1. At all these values, the score for submission did not change, which meant that MSD had no effect on our data set. Hence, we kept the default value of MSD=0.0001 as final. The same was repeated for threshold standard deviation (TSD). For a range of values, the scores did not change so we kept the initial value of TSD=0.0 as final. The same was repeated maximum number of unique nominal values (UNV). When set at UNV=10, the score jumped to 0.71966. However, when checked for further values such as UNV=20,50,5,100,1000 there were no further changes and the score remained constant.

For our next step, we introduced normalizers and denormalizer. The normalizer node was placed before the learner, initially only for the training data. All 3 options were tried (i.e., min max, z score and decimal scaling), The scores dropped for all 3, with the best faring being the min max normalization which gave a score of 0.60042. Normalizer was used for the test data as well now as logically both data should be normalized together. The score improved to 0.70212.
The last stage was just trying to make changes in column filters by selecting different combinations through column filters and using different missing value settings. In one of this, the missing values node was set as Mean for numerical value and fix value for string (the fixed value was kept 'UNKNOWN'). HomePlanet and Cabin were also excluded along with Name and PassengerId. This gave our highest score of 0.73322. Despite other tries, the score did not increase from this.

In conclusion, Naïve bayes wasn't the most efficient with the best score being 0.73322.

K Nearest Neighbour

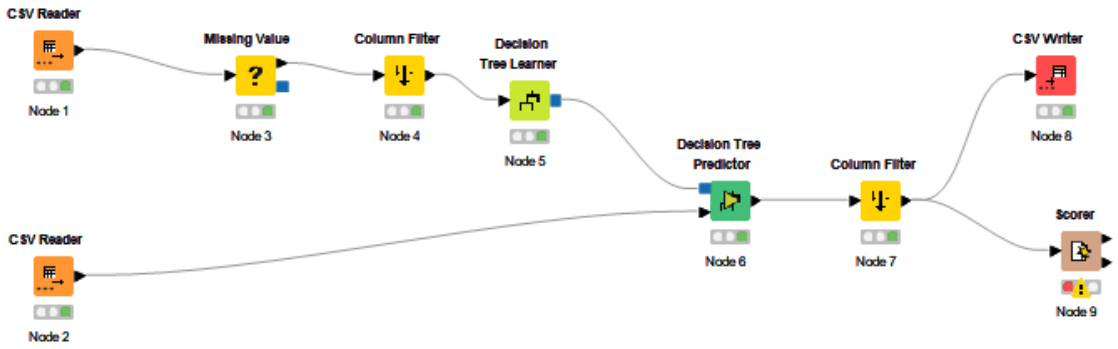


The Workflow model that yielded the highest score (above screenshot)

Using the CSV reader node, import the train and test data. The missing value node is used to substitute values for the missing values in the train and test data. The string value was set to be replaced by the most frequent value while the missing numerical values were replaced by the mean. Using the Column filter node, the passenger ID and the name columns are removed due to being irrelevant in our modeling.
The initial K Nearest Neighbour (KNN) node was kept at default settings i.e unchecked box of weighing by distance and unchecked on output class probability (this is not of use in our case so will remain this throughout). The column filter node at the end was used to only allow for the prediction column and passenger ID columns to be displayed (as instructed in the submission file). The column rename node was to change the column name as instructed (i.e Transported and PassengerId).
Now, for different values of k, different scores were observed. Going from k=1 to k=9, it was observed that scores on Kaggle rose and fell alternatively with the score rising for an odd value of k and falling for an even value. For k=1, Score=0.76502: k=2, score=0.73252: k=3, score=0.78325 and so on till k=9, score=0.79214. The overall trend was a rising score which meant that k should be even higher but now only odd values of k were chosen.
So, k=11, k=13 and k=15 was used which gave 0.79261, 0.79284, 0.79261 as scores respectively. Seeing scores fall again meant that **k=13** was ideal
Now 'weigh neighbours by distance' box was checked as this is said to eliminate the influence of large boundary values. For k=13, the score became **0.79611.** To confirm, scores on k=11 and k=15 were below this, so this remained the highest score till now.
A few combinations were tried in column filter. Excluding cabin as well brought no change. The rest led to a decrease in observed scores. Different configurations for missing values such as fixing values and using median or mode on numerical values lowered the score.
Next normalizer and denormalizer nodes were used. Initally only training data normalized. First min max was chosen Initially the age column was not normalized as it did not have huge deviations as compared to other numerical attributes. The score fell to 0.76665. Age was then included as well but that led to a

dramatic fall in score, so it was excluded again. Z score normalization only gave 0.58101, Decimal scaling gave 0.75543 as score.
Next a normalizer for test data as well. This led to score of 0.61538 and despite few changes the scores did not pick up so was discarded.
Next tried turning all categorical attributes to numerical. Used category to number node. Tried on both test data and training data. Scores fell to 0.35726. Tried again with a few attributes filtered and for different values of K in the KNN node but scores hovered between 0.2700 and 0.4000 and did not rise. Eventually had to drop those nodes, concluding that turning all attributes to numerical did not suit our dataset.
In conclusion, the highest score for KNN model gave us a respectable score of 0.79611.

Decision Tree

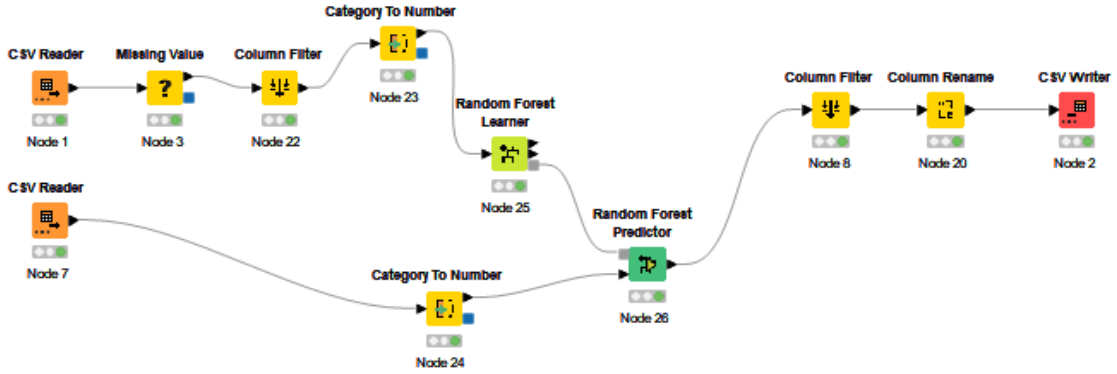


The Workflow model that yielded the highest score (above screenshot)

We started by keeping the model standard and did not make any changes except for filtering out the unnecessary columns. Then we started making changes step by step and then noted that how those changes are impacting our scores. Our first entry with the basic model had a score of 0.74959 while our second entry with score in which pruning was switched to MDL had a score a score of).77343 and in our third entry we increased the number of threads per node from 2 to 3 which increased our score to 0.77437. We tried several combinations of MDL, Pruning Method, gain ratio and Gini index, number of threads and number of records to store for view but our score revolved around the same number giving us the highest score (0.77646) when Pruning method was switched to MDL and there was no reduced error pruning. Also, we changed the number of threads and number records to store for view to 20000 while gain ratio to Gini index. Binary nominal splits were ticked. However, our lowest score was 0.74327 and it

was when quality measure was changed to gain ratio and reduced error pruning were ticked while number of threads were 11. Moreover, we tried to make several other small changes but at gain ratio the score was almost the same in every entry.

Random Forest

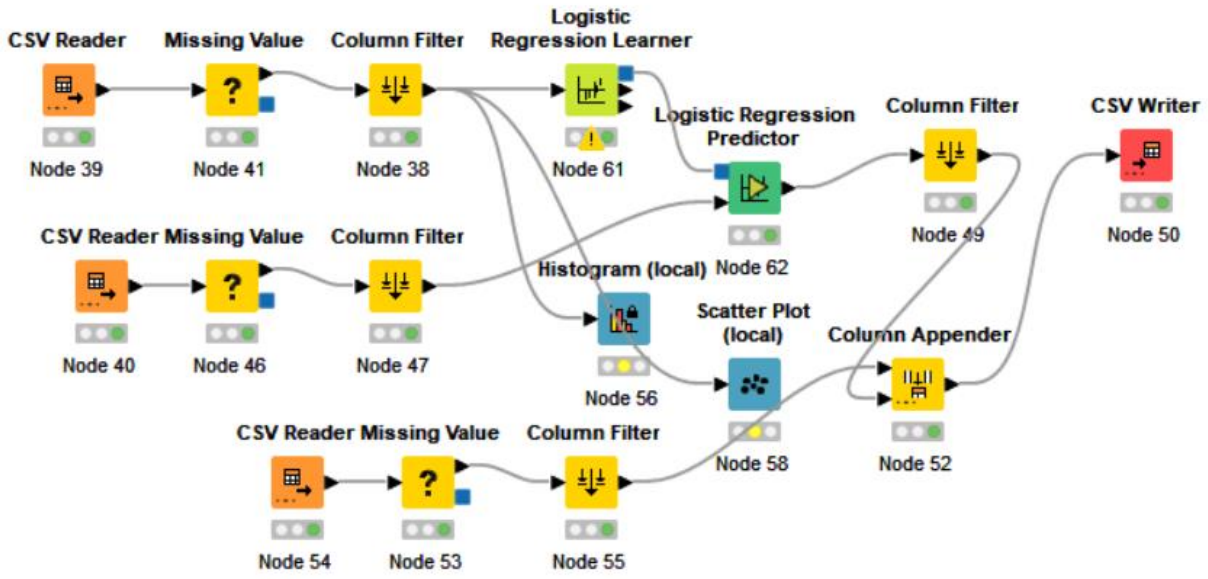


The Workflow model that yielded the highest score (above screenshot)

We made our next workflow which included the random forest learner but the scores we obtained with the help of the random forest learner were much lower than what we achieved from our previous workflow. With Random Forest Learner and Normalizer (using decimal scaling) we obtained a score of 0.39957. Our lowest score (0.28173) was with Random Forest Learner and Normalizer (using z-score) and denormalizer. Our highest score (0.40869) with the Random Forest learner was with the basic model. The changes we made were that we used Decimal-Scaling, Min-Max Normalization, and z-score in the Normalizer. We used different combinations but as mentioned earlier the scores were lower as compared to our previous model.

All things considered we also noticed a pattern that when the number of threads was increased in our first model there was an increase in our score. Moreover, unticking reduced error pruning gave a boost to our score and our score significantly increased when we changed pruning method to MDL. Further comparing the models, we can conclude that Random Forest isn't as effective for our dataset and hence we abandoned further use of this model.

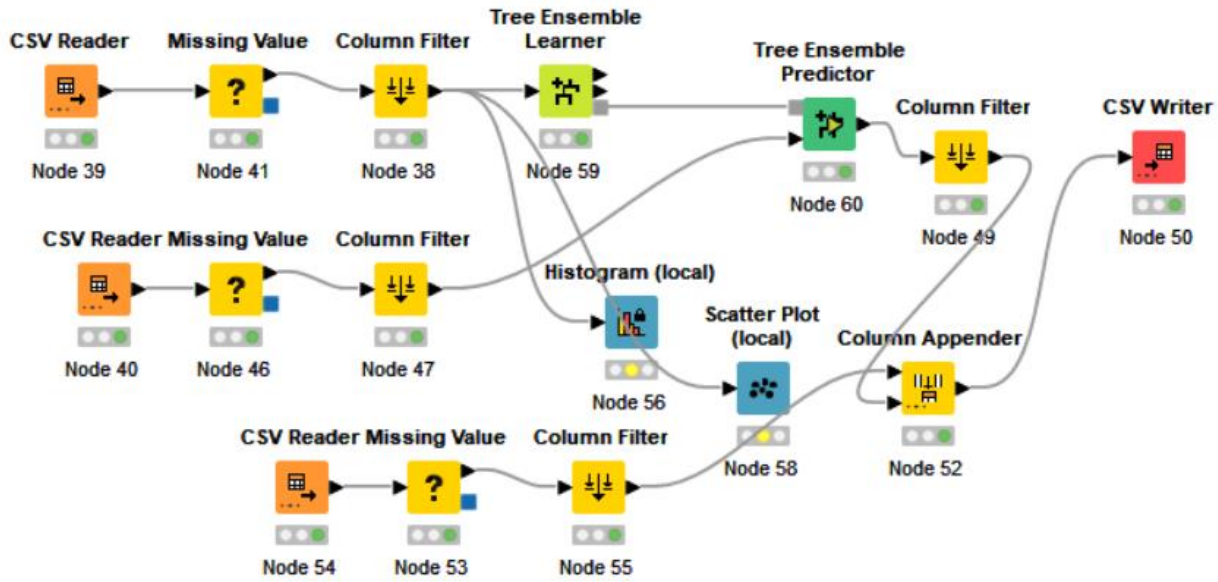
Logistic Regression



The Workflow model that yielded the highest score (above screenshot)

Logistic Regression Learner was not very effective and resulted in the score of **0.54641.**
A few changes were made in the configuration of the learner, but the scores remained low and due to lack of clear understanding, the model was not pursued further.

Tree Ensemble



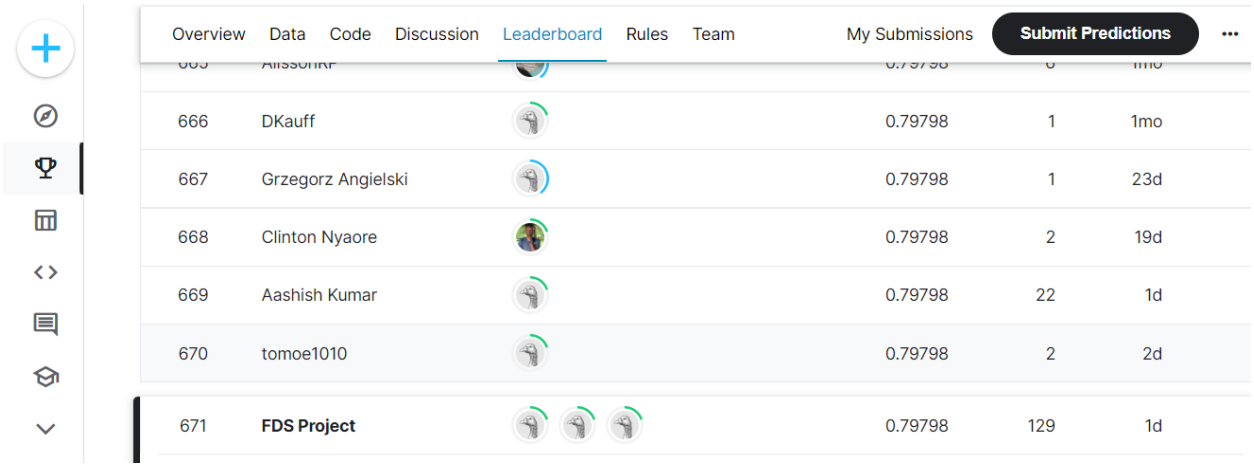
The Workflow model that yielded the highest score (above screenshot)

Tree Ensemble Model resulted in the score of 0.79635 with split criterion is information gain ratio with models 100. Enabling the hilting to 3000 did not change the score 0.79635. Changing the models to 1000 decreased the score to 0.79541, and 300 models resulted in 0.79658. Changing the split criterion to Gini index lowered the score to 0.79214. Therefore, tree ensemble as one of our highest scoring models.

Results

This is how all 7 of our models stack up

Model	Best Score
Gradient Boosted	0.79798
Naïve Bayes	0.73322
K Nearest Neighbour	0.79611
Decision Tree	0.77646
Random Forest	0.40869
Logistic Regression	0.54651
Tree Ensemble	0.79658



Evaluation

- Apart from Random Forest and Logistic Regression, all our models were quite consistent. These two fared the worst for our data set. Consequently, they will not be taken into consideration for our eventual deployment.
- Although Naïve bayes model scored a modest 0.73322 score, it did not score high enough as other models. Naïve Bayes is more suited for categorical values and multi-class prediction and since our dataset had many numerical attributes and only a single class prediction, it is not recommended to be used.

- Decision Tree gives a good score of 0.77646, but probably gets a little too complex. Decision Trees often tend to overfit and due to our data set being large and having a huge range of values, this creates very complex decision trees and are hence prone to sampling errors.
- Decision Tree Ensemble provides a higher score of 0.79658 as they tend to use multiple decision trees to produce better predictions while reducing the issues created by overfitting. So, using the ensemble is preferrable over a single decision tree.
- K Nearest Neighbour is one of the simplest data algorithms. Since we have only 2 number of classes, the choice of an odd value of K is preferred as has been done here by taking k=13. The value is neither too high to be too generalised and neither too low for noise to be highly influential. It is our second-best model and our 2nd recommendation for deployment.
- The best model is Gradient Boosted in our situation and so it is the model that we recommend for our present problem. It is suitable for our large data set and it's the most accurate in its predictions. Gradient boosting continuously builds models sequentially and helps eliminate the errors of the previous ones making it extremely reliable for our classification problem.

Future Work/Recommendation

Although our dataset was quite expansive, there were a few limitations that we faced and hope that for future purposes, the crew aboard the Titanic Spaceship take the following into consideration.

Ensure that each of the passengers' records are completely filled and no data is missing

Group data together for amenities. Since RoomService, ShoppingMall, VRDeck, Spa etc. all come under luxury or services, record them all together under one column to reduce overall data size and hence make computation faster

If a group of people are related to each other or travelling together, record their data in the same location for easier access.

Deployment

The **Gradient Boosted model** is applied to predict which of our passengers have been accidentally transported to the alternate dimension when the spaceship crashed into the space cloud anomaly. A workflow using this model is present in the zip folder as our best workflow.

Thank You !