MIS772 – Predictive Analytics

T1 2022

Assignment 1 – Individual
Student name: Mirna Arivalagan
Student number: 220142881

Executive summary (1 page)

Executive problem statement:

The purpose of this report is to gain insights on Airbnb properties in Denmark that were listed between November 2016 to October 2019. The main objective of this report is to understand if there are differences in tourists' perceptions and properties listed in areas classified as Sealand and not Sealand. Insights from this report can be used to better market properties in Denmark to the correct segments of travellers, as there may be different demographics of travellers staying in properties classified as Sealand and Not Sealand. The following attributes from the sample data set provided has been examined: reviews, overall satisfaction, type of accommodation, price per night and the number of occupants the properties can accommodate.

From the analysis using the sample data provided, there were far more properties listed in Sealand compared to the rest of Denmark (Sealand: 14, 798 properties, Not Sealand: 9070). Upon removing any missing data, I have found that there are some differences between the properties located in Sealand and not in Sealand, on average, a property in Sealand receives about 14 reviews compared to 8 reviews in properties not located in Sealand. This means that there are more travellers staying in properties in Sealand.

The neighbourhood with the highest number of average reviews is Soro which is in Sealand with an average of 27 reviews and the neighbourhood with the highest average review in Not Sealand is Billund with 26 reviews for each property.

Properties in Sealand are also far more expensive costing on average $120 a night compared to $89 a night for properties out of Sealand. The most expensive neighbourhood in Sealand is Dragor costing on average $147 a night compared to Nordfyns out of Sealand costing only on average $112 a night.

In terms of overall satisfaction, properties in Sealand has an overall average of 3.3 compared to 28 for properties not in Sealand. It is important to note that there is a strong relationship with date listing and the overall satisfaction, where properties that have been listed for a longer period is seen to have higher satisfaction ratings. In the sample analysed, that were much more older listings located in Sealand compared to the rest of Denmark, thus it makes sense to see this group having much higher overall satisfaction scores.

One of the key differences is that there are fewer properties located in Sealand that could cater to large groups of more than 10 people compared to properties located not in Sealand (Sealand: 114, Not Sealand: 148). This can be seen as properties classified as Sealand incorporates the capital city, thus most properties are expected to be more compact, whereas properties classified as Not Sealand covers rural areas where there is more space.

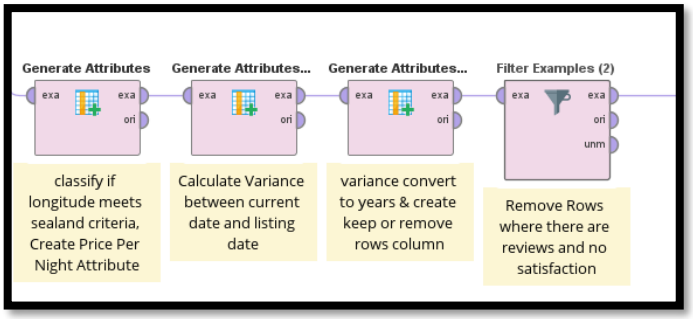
Using the sample dataset provided, I was able to design 2 models that can be used by the business to predict if a property is likely to be popular or not. The attributes that were meaningful to be included in the model and that has an impact on the popularity of a property is the minimum number of nights of stay, neighbourhood, price per night, the number of reviews and room type.

The best performing model designed was a random forest model, this model was able to accurately predict popular properties 81.46% of the time, or a weighted accuracy of 62.3%. This model was also able to correctly classify 87.09% of the popular properties.

The second-best model that I am putting forward for the business to deploy is an ensemble model that is made from 3 very different classification algorithms to ensure the predictions are robust, the model is designed to vote against the predictions, of these 3 algorithms (KNN, Random Forest & Naïve Bayes) and obtaining a vote amongst them. The prediction with the most votes will be the final prediction. By using this method, we are leveraging off the strengths of each individual models within the voting ensemble. This model had an could accurate predict 80% of the time, or a weighted accuracy of 62.30% for a more conservative approach. This model was able to correctly classify 81.75% of the popular properties.

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

Upon loading the data, there were missing values in the description column, which was dropped alongside host and user id as these attributes do not provide any value. Upon analysing the dataset, I noticed properties where there were more than 0 reviews but had an overall satisfaction score of 0, I treated these properties as data with missing values as to complete a review, a satisfaction rating needs to be populated, so I removed these properties. To do so, I utilised a subprocess named Data Manipulation 1 – in this subprocess, I classified properties as Sealand and Not Sealand using a logical if statement and generated a new attribute, I also utilised a combination of generate attribute and an if statement to create values as keep and remove to identify rows that needs to be removed. I then filtered out rows that needed to be removed. Within this subprocess, I also utilised a few date functions to generate an attribute that provides the variance of the current date and listing date specified in years. There was a total of 11,990 properties located in Sealand and 7194 located in the rest of Denmark (Not Sealand).



To understand if there are differences in the properties across in Sealand and Not Sealand, I analysed the relationship across the reviews, overall satisfaction, price per night and accommodates.

The average price per night for properties located in Sealand was far more expensive than those not in Sealand. (Sealand: $120.96, Not Sealand: $88.96), There were far more properties prices between $50 - $200 per night in Sealand compared to the rest of Denmark.

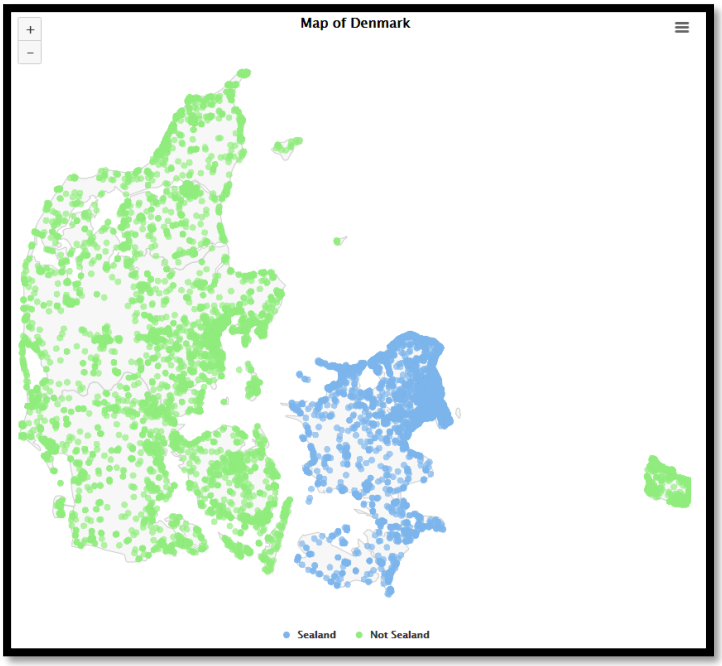
Overall, properties located in Sealand has a higher overall satisfaction score compared to the rest of Denmark (Sealand: 3.30, Not Sealand: 2.81).

The top 5 neighbourhoods with the highest overall satisfaction in Sealand can be seen in the following table:

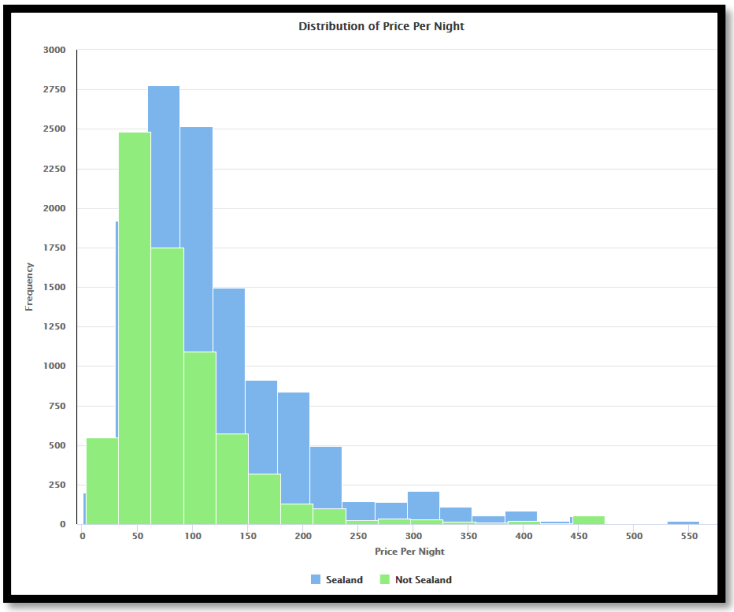
Row No.	neighborhood	average(overall_satisfaction)
1	Frederiksberg	3.751
2	København	3.687
3	Stevns	3.318
4	Hvidovre	3.162
5	Vordingborg	3.143

Whereas the top 5 neighbourhoods not in Sealand with the highest overall satisfaction score can be seen in the following table:

Row No.	neighborhood	average(overall_satisfaction)
1	Ærø	3.824
2	Billund	3.795
3	Esbjerg	3.482
4	Haderslev	3.477
5	Struer	3.438



Map of Denmark – Sealand and Not Sealand



Distribution of price per night – Sealand and Not Sealand

From both the tables above, the neighbourhood with the highest overall satisfaction is AERo that is found in Not Sealand with an overall satisfaction of 3.824. When analysed the overall satisfaction for all neighbourhoods together, 3 out of the top 5 neighbourhoods were found in Not Sealand.

Across both categories, the most expensive property was valued at $589 a night, and on the lower quartile, properties in Sealand was priced at $70 or less a night and rest of Denmark was priced $50 a night or less.

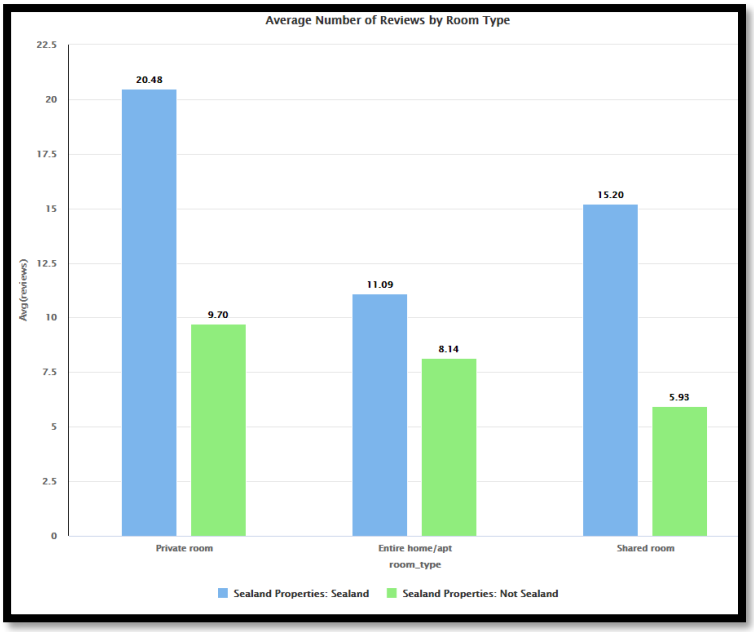
One of the key differences is with larger properties, there are far more properties that can accommodate groups of 8 or more travellers in the rest of Denmark compared to Sealand.

Properties in Sealand had a much higher average number of reviews for each property than that in the rest of Denmark (Sealand: 13.25, Not Sealand: 8.55). When analysed by breaking down into room types, there are some significant differences here. Private rooms in Sealand had the highest average number of reviews with 20.48, compared to 9.70 for Not Sealand, shared rooms had the next highest average with 15.20 for Sealand and 5.93 Not Sealand, interestingly, the difference for the entire home/apt type properties, the variance between Sealand and Not Sealand is not that huge like the other 2 room types, where properties in Sealand had an average of 11.0, and Not Sealand's average is 8.14.

What this finding means is that travellers to Sealand maybe more business type travellers or couples preferring to stay in rooms, and travellers that are staying in properties not in Sealand, are more likely to be groups of family and friends, or corporate retreats staying in larger properties.

Next, I filtered out properties located in Sealand to begin the EDA stage prior to building the model. I analysed the relationship between the property listing date and overall satisfaction, what can be observed is that the average floats around 4.7- 4.8 and then drops right down from early 2019. (Image 5).

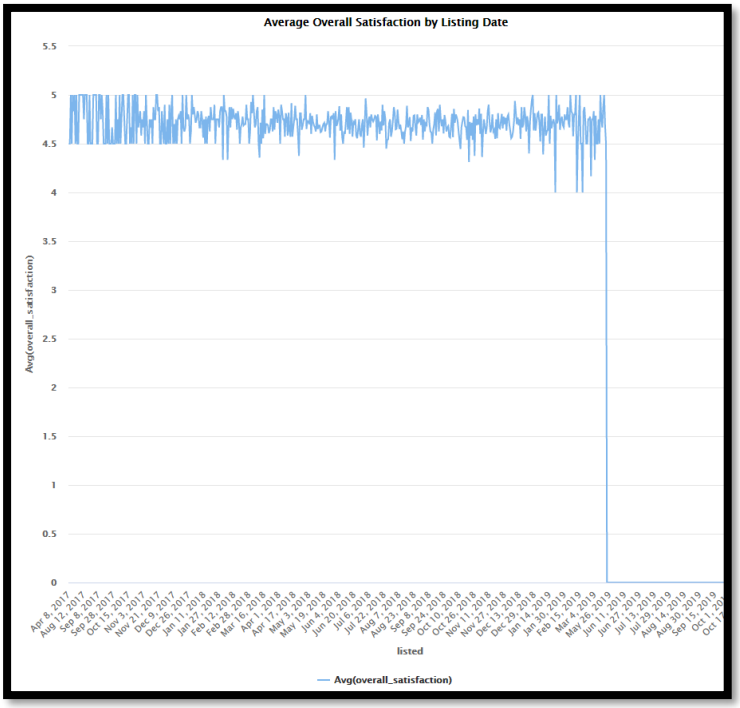
When I look at this relationship using the years since listing, properties that have been listed for more than 4 years is seeing an average satisfaction of 4.71, and properties less than 4 years is seeing an average satisfaction of 1.82. So, I have used these parameters as a threshold using a logical IF statement to classify popular or not popular. There were a total of 2898 popular properties and 4296 not popular.



Average number of reviews by room type – Sealand and Not Sealand

Row No.	neighborhood	average(overall_satisfaction)
1	Ærø	3.824
2	Billund	3.795
3	Frederiksberg	3.751
4	København	3.687
5	Esbjerg	3.482

Neighbourhoods with the highest average overall satisfaction



Relationship of overall satisfaction by listing date

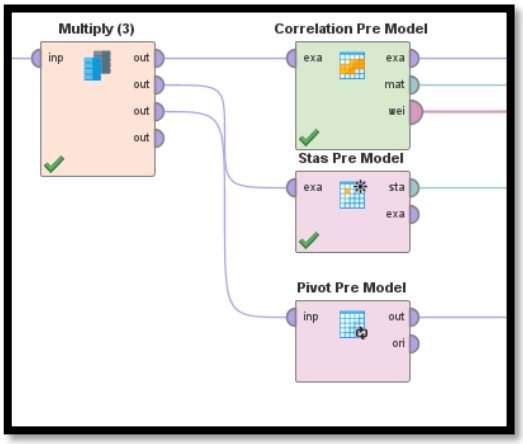
Predictive modelling (2 pages)

To model out the prediction of if a property will be popular or not popular with tourists travelling to the rest of Denmark (Not Sealand), KNN and Random Forests algorithms were utilised. Justification for choosing KNN is because of the way it works with finding the nearest datapoints or neighbours, so this algorithm is extremely well suited in to solve our business problem, as the purpose of this model to predict popular properties based on a combination of attributes, so once the algorithm detects a property that is likely to be popular, it can then look at the data points that are close by to identify other properties with similar attributes that may also be popular. KNN is also an easy model to implement, thus maintaining this model and fine tuning it to suit further business needs will be easy.

Random Forest is the second algorithm that was used to model out the predictions, I've chosen to use this algorithm as this algorithm is essentially an ensemble of decision trees, thus making it more robust to overfitting issues, as it uses the average of all predictions. Random Forest is also robust when it comes to dealing with missing data, as it can calculate the proximity-weighted average of the missing values and replace them.

To decide on the attributes to include in the classification model, I used another subprocess to generate a correlation matrix, data distribution using the descriptive statistic operator, and a pivot table to view the average reviews, satisfaction and price per night for properties classified as Not Sealand grouped by neighbourhood.

I have found the numerical attributes in the dataset to be right skewed in distribution.



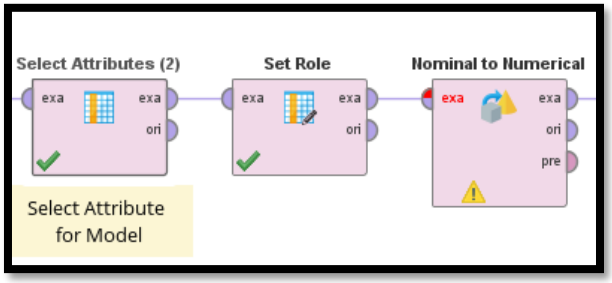
Pre modelling exploratory analysis

Attributes	reviews	overall_satisfaction	accommodates	bedrooms	minstay	Price Per Night
reviews	1	0.501	-0.038	-0.092	-0.018	-0.153
overall_satisfaction	0.501	1	-0.032	-0.060	-0.032	-0.196
accommodates	-0.038	-0.032	1	0.770	0.054	0.388
bedrooms	-0.092	-0.060	0.770	1	0.059	0.393
minstay	-0.018	-0.032	0.054	0.059	1	0.139
Price Per Night	-0.153	-0.196	0.388	0.393	0.139	1

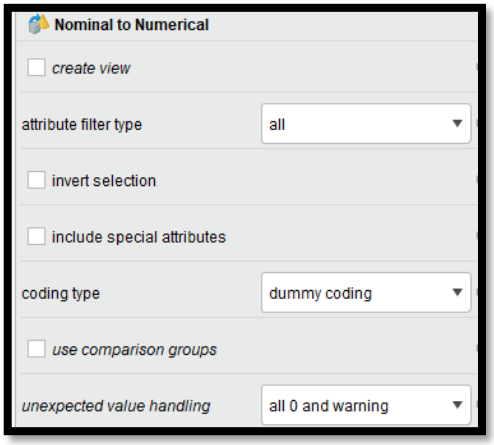
Correlation matrix of attributes

The attributes accommodates and bedrooms has a moderate to high positive correlation with each other, so when building the model, only one of these should be used to avoid any multicollinearity issues with the model.

Both models were designed with the following attributes as predictor attributes: mainstay, neighbourhood, price per night, reviews and room type. I've used the Select attributes operator and set role to set the popular/not popular attribute as the label. Once this is completed, I used the nominal to numerical operator to convert categorical data into numerical data using dummy coding.

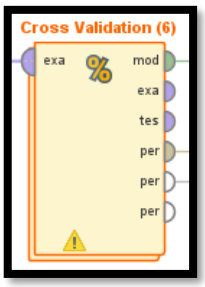


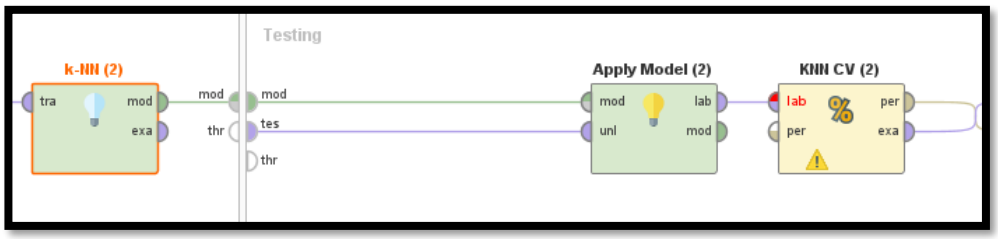
Model pre-processing – select attributes, specify label and dummy coding



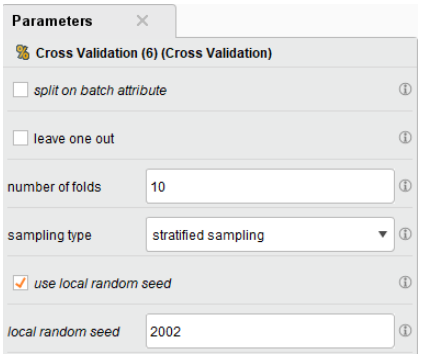
As we have a sufficient data to work with, I have opted to use cross validation across both base and optimised models as this will reduce model variance and ensure that the predictions are robust. I've set the number of folds to be 10 and

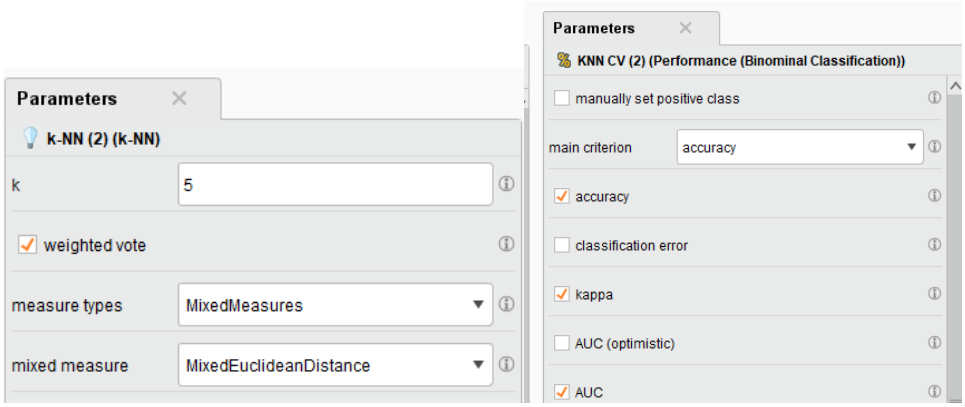
decided to use a stratified sampling method. What this will do is that it will split the dataset in 90% for training and 10% for testing, the 90% will be split into a further 10 folds for training and validation. Within the cross-validation operator, we specify the KNN model, and I've used k = 5 neighbours as my base model. I've also included the performance operator to output the model performance metrics such as Kappa and accuracy.





Cross validation parameters





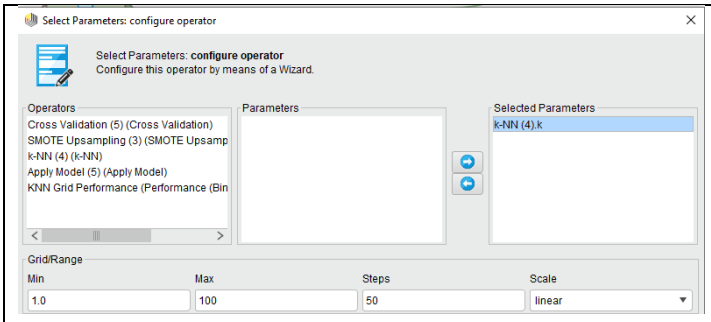
Applying model and performance output

The process above was replicated with the random forest model using the same cross validation parameters, and for the model, I have used the default parameters for my base model: number of trees = 100, criterion = gini_index & maximal depth = 10.

The results of these 2 base models can be viewed on table X – detailed discussion can be found in the following model evaluation section. The performance of both base models were moderate to high, but not powerful. Both models had behaved slightly differently to each other, as the KNN model had higher recall values for the not popular class compared to popular, and the random forest model was better at recalling the popular class, which in this instance is what needed to be predicted.

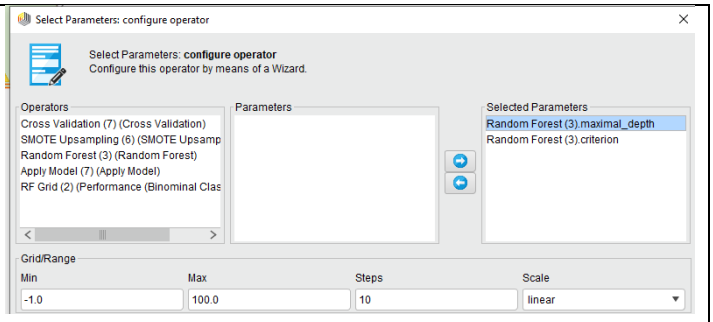
Upon running the 2 base models, I ran a couple more experiments to see if the performance of the models can be improved further. The first experiment run was using a grid search, which is a technique to run the model multiple times across a combination of different parameters.

For the KNN model, my parameters were to find the best K value ranging from 1 to 100 in 50 steps. And for Random Forest, the parameters to test was the maximal depth with values ranging from -1 to 100 in 10 steps, and criterion is gini_index or information_gain.



KNN Grid Parameters

Best K value: 13



Random Forest Grid Parameters

Best Max Depth 9

Best Criterion: information gain

Model evaluation and improvement (2 pages)

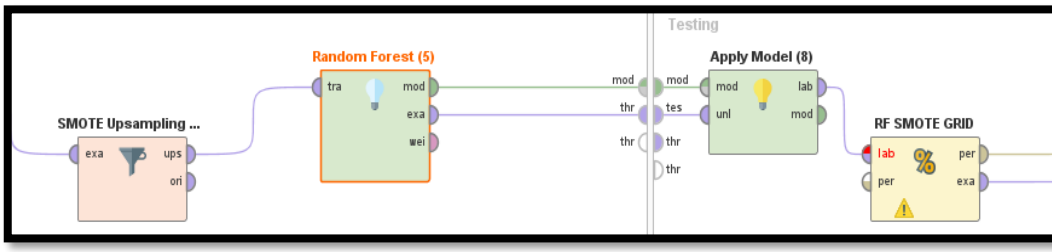
To evaluate the models, I have decided to prioritise the performance of the models using Kappa and Recall of the popular class. Using Kappa over accuracy will ensure that the weighting of the classes is factored in, and the accuracy is adjusted based on these weights. As the purpose of this model is to predict properties that are likely to be popular that Airbnb can then use with their advertising, due to the current economic and financial constraints, it's crucial that the prediction is accurate, thus preference is placed over recall over precision. Airbnb does not have the luxury of having the model predict relevant properties that may be popular. If this model is designed for Airbnb's internal recommendation system that the users are using, then precision could be prioritised over recall.

Comparing the base models named KNN CV and RF CV in table 1 against the hyperparameter tuned models named KNN Grid and RF Grid, we can observe that the grid search did make some improvements in the models. There was only a slight increase in performance for the KNN based models, the kappa value between these 2 models had increased from 0.562 to 0.582, and the recall for the popular class has increased from 74.43 to 77.57.

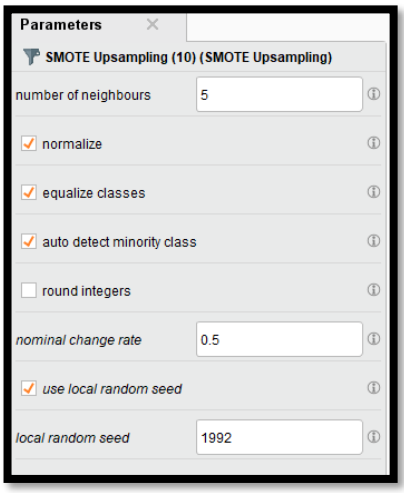
The grid search for the random forest models did optimize the random forest model much more. The kappa value increased from 0.586 to 0.623, and there was almost a 10% increase with the recall values for the popular class, increasing from 77.71 to 87.09.

Seeing that the grid search for the random forest improved the model significantly, I ran another test to see if by combining a grid search and using a SMOTE technique to balance the classes, I was able to optimize the performance of the model even further.

To design this model, I started with the optimize parameter operator, and within it, I had set a cross validation parameter, and within the cross validation I had used the Smote Up sampling operator only on the training data to balance the classes to train the model. This technique creates synthetic data points in the dataset to balance the popular and not popular classes. With the Smote operator, I chose to use 5 as the number of neighbours and set a local random seed to ensure reproducibility. To ensure that we are comparing similar models, I had used the exact same grid parameters as my RF Grid model.



Smote Technique



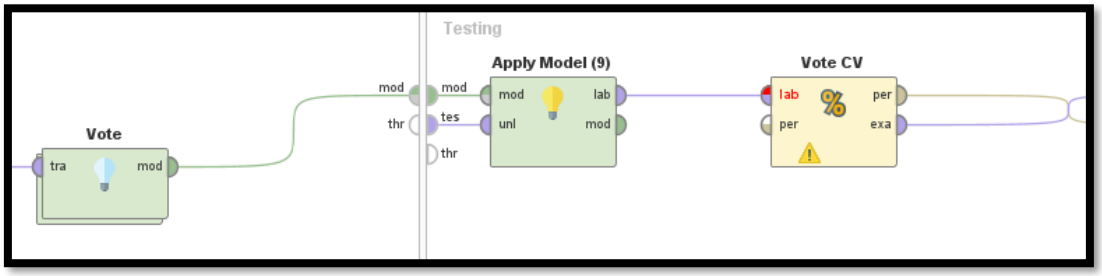
As per table 1 – The model RF Smote Grid provided mixed results when compared to the RF Grid model. The kappa score dropped from 0.627 to 0.623, however the recall of the popular class increased significantly from 87.09 to 99.31. This may appear that this is the best model so far, but we will also need to assess the recall of the not popular class to make a clearer judgment. The recall of the not popular class dropped by about 10% from 77.65 to 67.76. Thus, between the 3 different versions of the random forest model, the best performing model was the model with just the grid search.

Model	Accuracy	Kappa	Recall (Popular)	Recall Not Popular	Precision Popular	Precision Not Popular	AUC
KNN CV	79.07	0.565	74.43	82.19	73.82	82.65	0.895
KNN GRID	79.72	0.582	77.57	81.17	73.54	84.29	0.907
RF CV	80.01	0.586	77.71	81.56	73.98	84.43	0.907
RF GRID	81.46	0.627	87.09	77.65	72.45	89.92	0.916
RF SMOTE GRID	80.47	0.623	99.31	67.76	67.51	99.32	0.912
Vote CV	80.03	0.593	81.75	78.86	72.29	86.49	0.790
Vote CV Smote	80.51	0.621	96.76	69.55	68.19	96.95	0.779

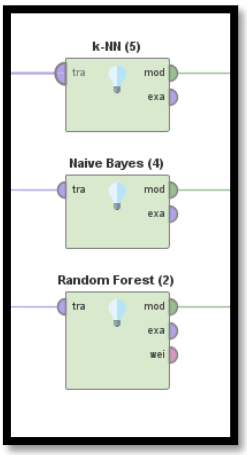
Table 1

I then ran another 2 rounds of further experimentation and testing to see if better performance compared to the grid search random forest model can be obtained. I chose to run these experiments using a voting ensemble model. Within the voting model, there was 3 models, a Knn model and random forest model, both using the parameters found to be best from the grid searched versions of these model, and finally I included a Naïve Bayes classification algorithm within the vote model. I thought by combining a distance based (KNN), tree based (random forest) and probability based (Naïve Bayes) algorithms, this provide with robust predictions.

When designing this voting model, I included cross validations with the exact same parameters as my previous models and tested with a combination of using Smote and without using Smote.



Voting Model



Results for this experiment between both Vote CV and Vote CV Smote vary, overall, the vote model with the Smote applied saw the kappa value go up from 0.593 to 0.621 and the recall of the popular class increased significantly from 81.75 to 96.76. The smote did decrease the precision of the popular class and the AUC value as can be observed in Table 2.

Based on the experiments above, one of the clear findings is that using SMOTE isn't always needed when designing models, as we can observe here, it did reduce the performance of the models with regards to precision and the AUC value.

To decide on the best 2 models that Airbnb can deploy to predict properties that are likely to be popular or not, we will assess using the kappa and the recall values of the popular class. Based on this, the RF Grid model was the most powerful model in terms of accuracy factoring in the weights of the classes, this model had the highest kappa score of 0.627 and a powerful popular class recall score of 87.09. The second model that Airbnb can deploy to use within the business is the cross validated vote ensemble with Knn, Random Forest and Naïve Bayes. This model had a kappa score of 0.593 and a popular class recall of 81.75.

There are limitations and assumptions to this modelling that need to be investigated further. Airbnb did not specify any parameters as to what constitutes a property that is popular. I built these models with some assumptions and by looking at the relationship between overall satisfaction and length of time since listing. The dataset supplied did not contain any recent data, and due to significant external factors, such as Covid and the economic downturn, the predictors in this model may not necessarily be reflective of current circumstances. Further testing on these models is required using more recent data. Travellers' needs may have also changed, price could be the single attribute that determines if a property is popular or not, as due to the economic downturn, I would assume that travellers may not be willing to spend as much as they would have pre covid.