CS 6350- BIG DATA MANAGEMENT AND ANALYTICS

TWO SIGMA CONNECT RENTAL LISTING INQUIRES

(https://www.kaggle.com/c/two-sigma-connect-rental-listing-inquiries)

Project Report

Spring 2017

May 1, 2017

GOWTHAMI KURUKURI (sxk150832)

SAI CHARAN VENNAMANANI (sxv157130)

MANOJ KUMAR ELLANTI (mxe150630)

Aim:

In this project, we aim at making a complete analysis of the Two Sigma Connect dataset to predict how popular an apartment rental listing is, based on the features like text description, photos, number of bedrooms, price, etc.

TWO SIGMA CONNECT DATASET

The Two Sigma Connect dataset is taken from an active Kaggle Competition and the link of which is given below:

https://www.kaggle.com/c/two-sigma-connect-rental-listing-inquiries/data

Number of attributes = 15
Number of instances = 49352(train.json), 74659 (test.json)

The attributes are the following:

1. Bathrooms - number of bathrooms
2. Bedrooms - number of bathrooms
3. building_id
4. created
5. description
6. display_address
7. features - a list of features about this apartment
8. latitude
9. listing_id
10. longitude
11. manager_id
12. photos - a list of photo links. You are welcome to download the pictures yourselves from renthop's site, but they are the same as imgs.zip.
13. price - in USD
14. street_address
15. **interest_level** (Target variable) - It has 3 categories: 'high', 'medium', 'low'

Programming Languages used

- Python
- Scala

Tools Used

- Databricks
- Pycharm

Techniques Used

- Decision Tree
- Random Forest

Experimental Methodology

The following procedures are implemented as part of this project:

1. Dataset preprocessing:
   - This step involves dealing with outliers in the dataset.
   - Selecting the attributes required to classify the given dataset using histograms, plots.
   - Flattening the attributes from the complex attributes to simple attributes.

     For ex: extracting of year day month from created Date and analyzing them separately.
2. The data preprocessing part is done in python and data set with the filtered features are selected.
3. The training dataset filtered with all the above procedures is then uploaded to databricks, for the databricks to work on it.
4. The classifiers are written in Scala, to analyze the uploaded dataset.

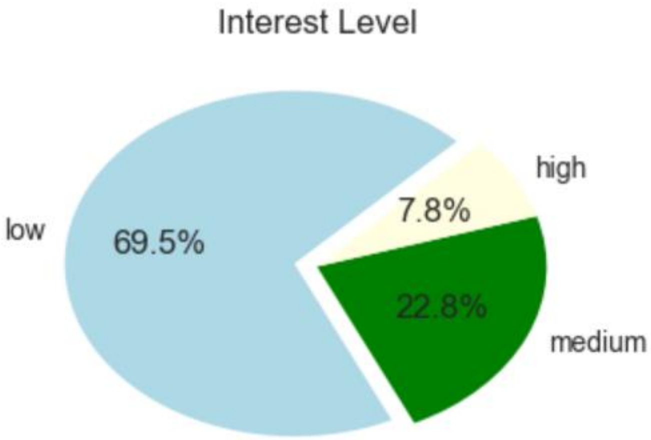
Packages Used:

Classifier	**Package**
Decision Tree	ml.Classification.DecisionTreeClassificationModel
Random Forest	ml.Classification.RandomForestCLassificationModel

Exploratory Data Analysis:

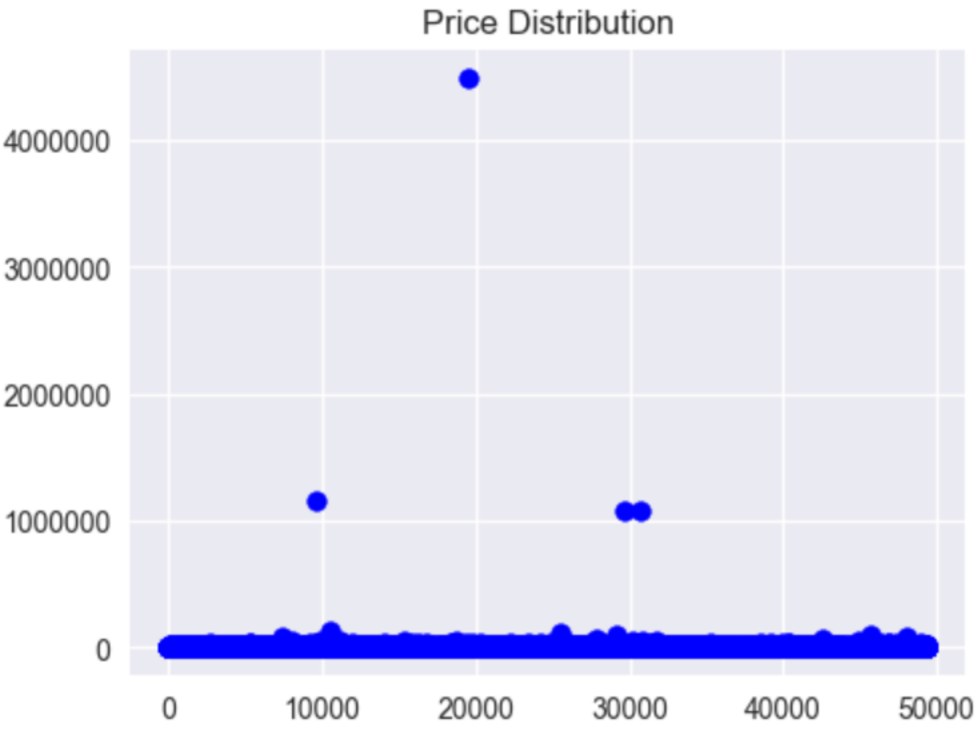
a) Analyzing the interest levels:

We found that the High interest level is present in a very small set of populations. Low interest level is present in the majority.

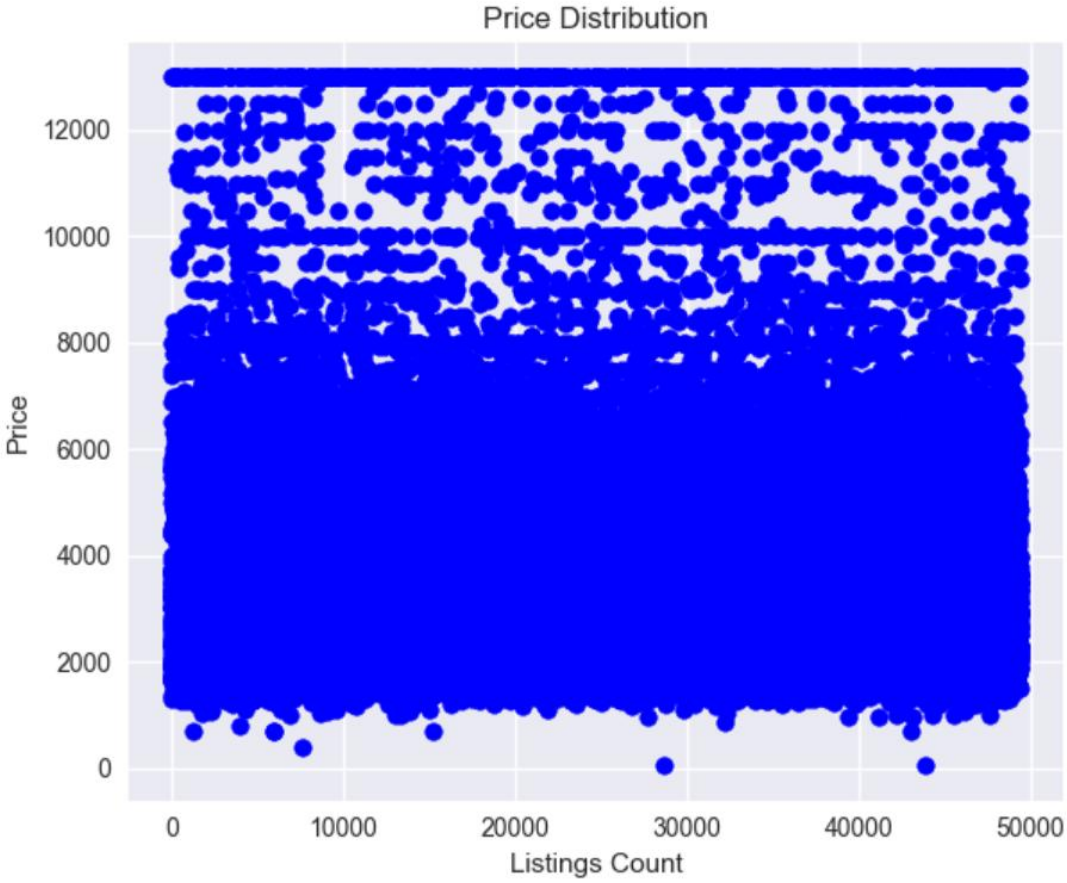


b) **Analyzing the Price Distribution**:

From the graph, we can deduce that there are some outliers in Price.



#Removing the outliers in price and replotting the graph.



c) Analyzing photos:

Since processing photos is a complex task we have removed the photos and instead included a feature called number of photos (photoCount).

d) Analyzing Words in Description:

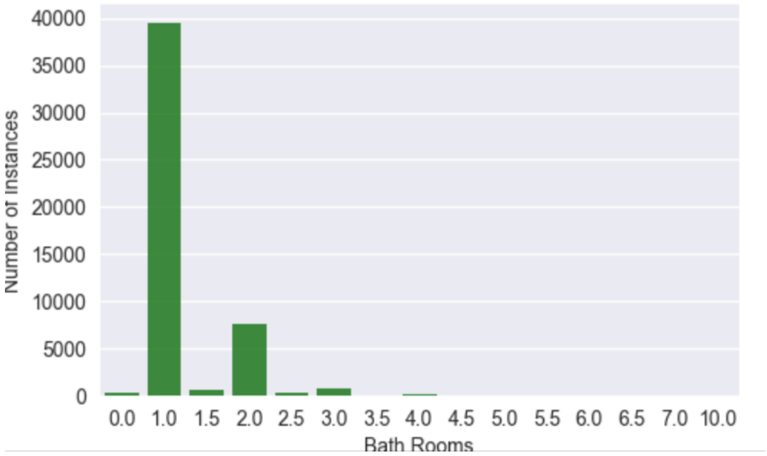
We modified the description field to only count and give number of words in the description, and later process on only word count. This reduced the storage greatly on cluster and it is far easier to handle the data on cluster.

e) **Analyzing Features:**

Similar to description processing, the features are modified to number of features.

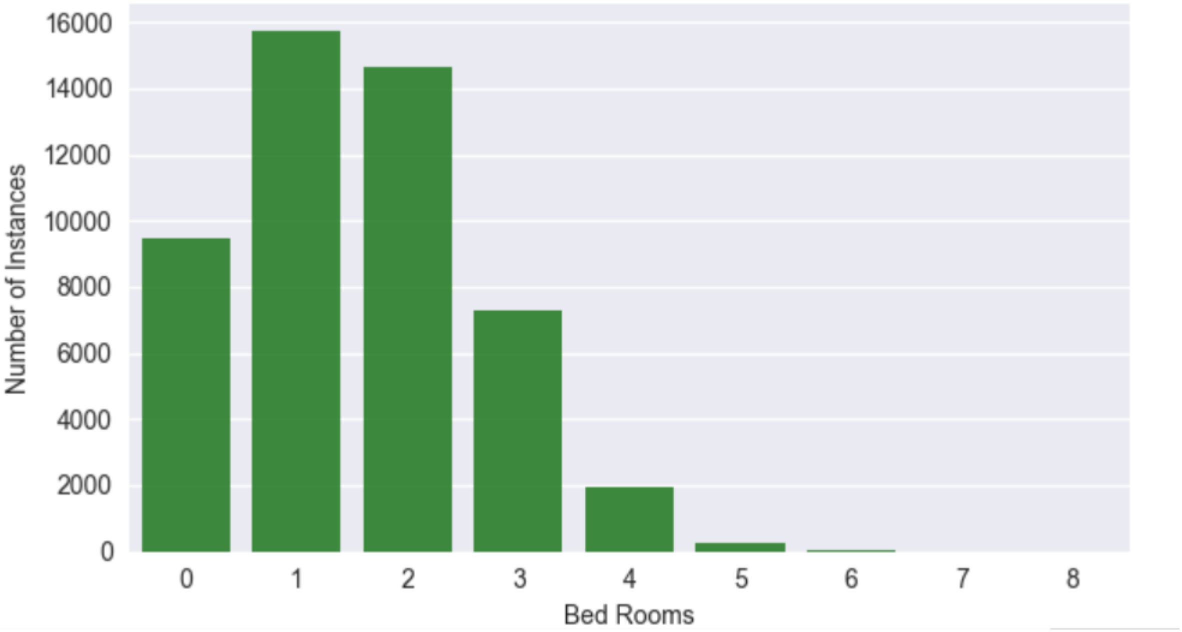
f) Analysing Bathrooms:

Analysis: Most of the listings have one bathroom.



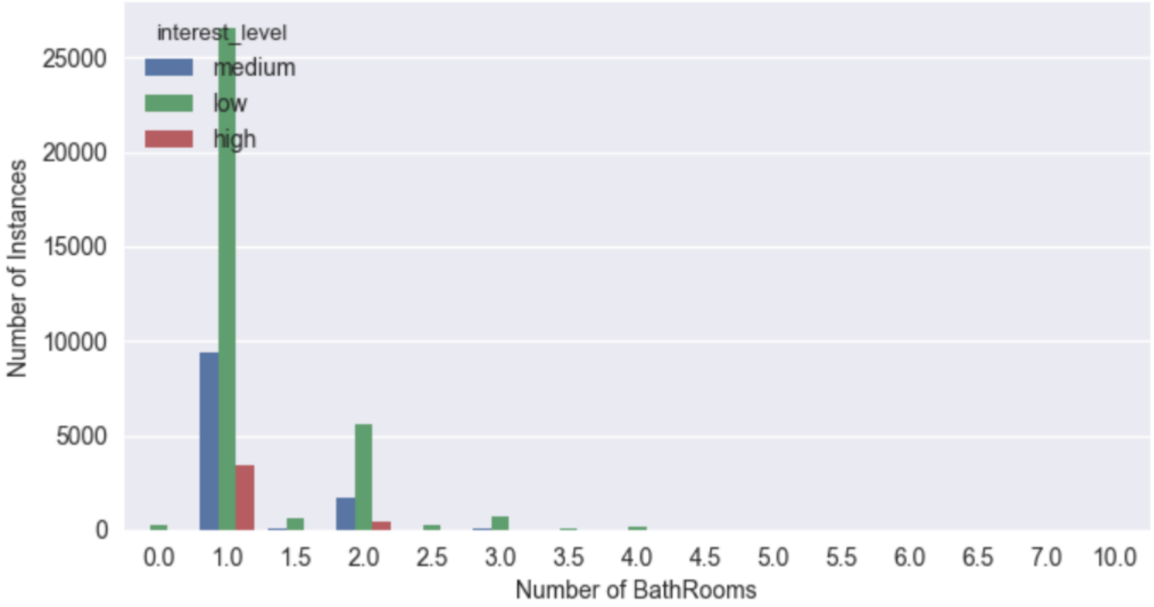
g) Analysing BedRooms:

Analysis shows most of the listings have 1 Bed Room



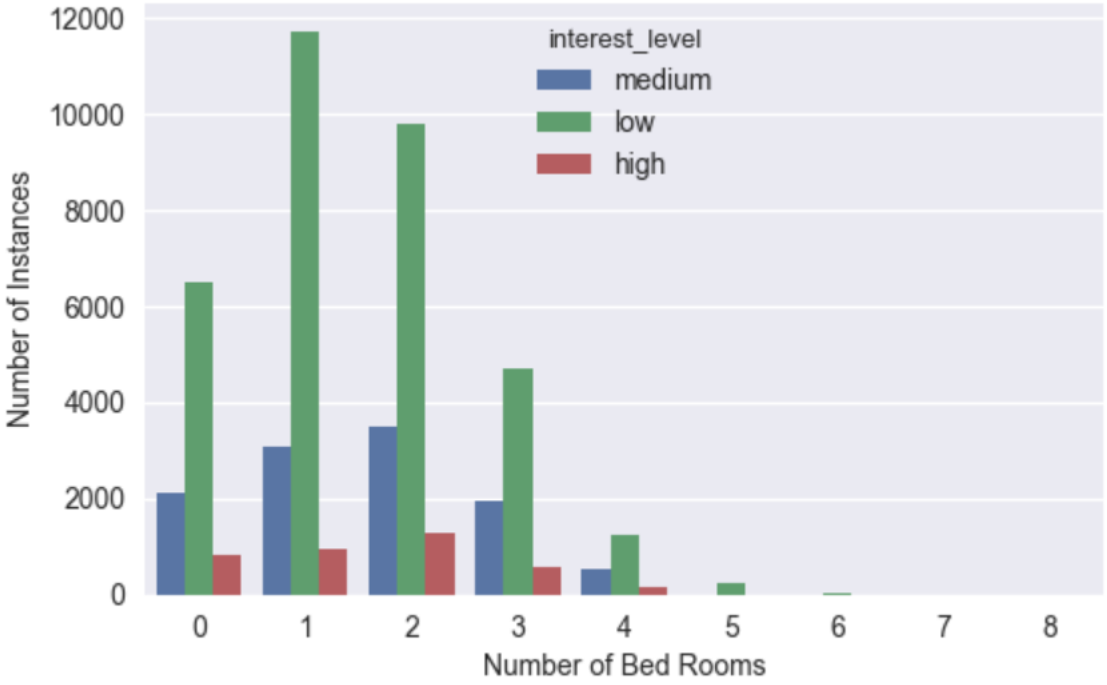
h) **Analyzing Interest Level vs BathRooms:**

Listings with one bathroom has the lowest interest level.



i) Interest Level vs Bed Rooms:

Listings with one bedroom has the lowest interest level.



RANDOM FOREST CLASSIFIER:

Here, we used Random Forest Classifier to complete the analysis of the Two Sigma Connect dataset to predict how popular an apartment rental listing is based on features like text description, photos, number of bedrooms, price, etc.

The number of trees parameter is varied and the logloss is calculated for various values. We found that the optimum log loss occurs at 1000.

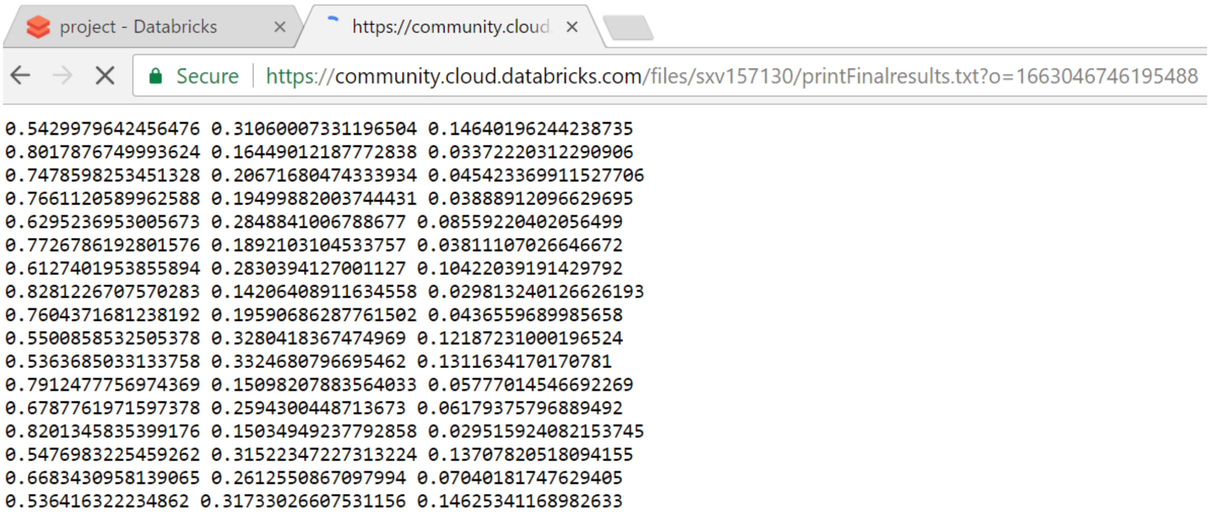
Using random forest since, we can prevent overfitting since the smaller sub trees are generated by creating random subsets of features and then combining the sub trees. Also, since the split at node is no longer the best split, the bias of the forest can be more than decision tree. But, due to averaging the overall bias decreases.

Also, since the process is parallelized the slowness disadvantage is reduced.

Log Loss:

Classifier	Number of Trees	LogLoss
Random Forest	1000	0.3806262495083649
Random Forest	1500	0.3809414151030321
Random Forest	2000	0.381042575432639

The probabilities of interest levels are written into a file as below:



project - Databricks
https://community.cloud
Secure
https://community.cloud.databricks.com/files/sxv157130/printFinalresults.txt?o=1663046746195488

```
0.542997964245647б 0.31060007331196504 0.14640196244238735
0.8017876749993624 0.16449012187772838 0.03372220312290906
0.747859825345132‌8 0.20671680474333934 0.045423369911527706
0.7661120589962588 0.19499882003744431 0.0388891209662969‌5
0.6295236953005673 0.284884100678867‌7 0.08559220402056499
0.7726786192801576 0.1892103104533757 0.03811107026646672
0.6127401953855894 0.2830394127001127 0.10422039191429792
0.8281226707570283 0.14206408911634558 0.029813240126626193
0.7604371681238192 0.19590686287761502 0.0436559689985658
0.5500858532505378 0.3280418367474969 0.12187231000196524
0.5363685033133758 0.3324680796695462 0.1311634170170781
0.7912477756974369 0.15098207883564033 0.05777014546692269
0.6787761971597378 0.2594300448713673 0.06179375796889492
0.8201345835399176 0.15034949237792858 0.029515924082153745
0.5476983225459262 0.31522347227313224 0.13707820518094155
0.6683430958139065 0.2612550867097994 0.07040181747629405
0.536416322234862 0.31733026607531156 0.14625341168982633
```

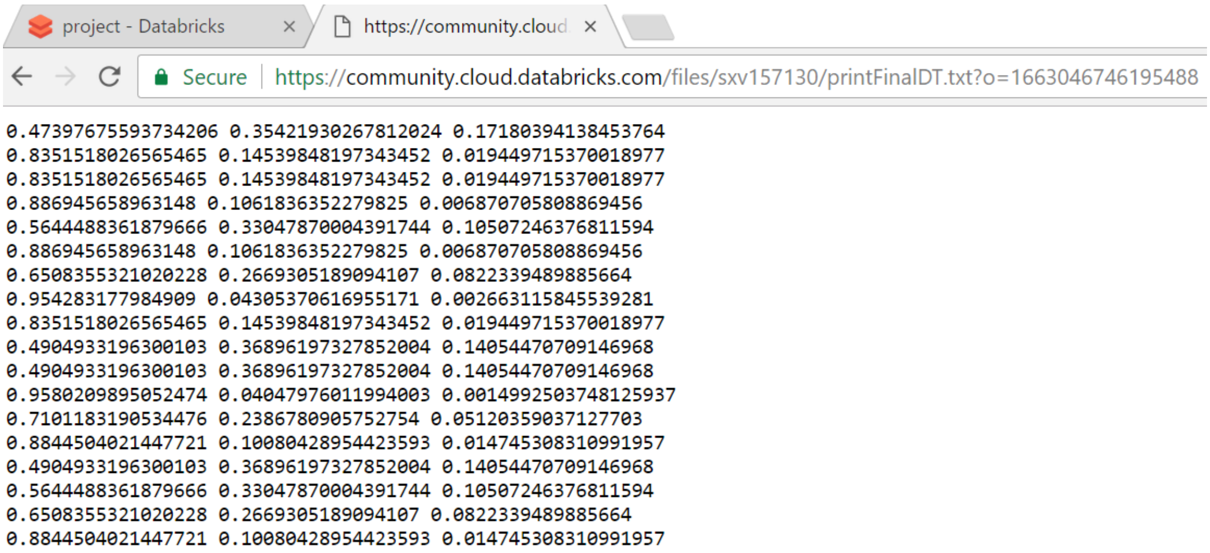
DECISION TREES:

Here, we used Decision Trees Classifier to complete the analysis of the Two Sigma Connect dataset to predict how popular an apartment rental listing is based on the features like text description, photos, number of bedrooms, price, etc.

Decision Trees are one of the most basic and simple machine learning algorithms for learning tasks of classification and Regression. They can deal with non linearities and features extractions. Random forests, Boosting works on the similar basis of Decision trees.

Here, unlike Random Forests Split chosen at each node is the best split as far as the information gain is concerned. Also, Each label is fixed at each leaf node using decision trees. Since, the processing on Decision Trees approximately scales linearly in the number of training instances, and in the number of features.

The log loss obtained using decision trees is 0.38208965979840087


project - Databricks
https://community.cloud
Secure
https://community.cloud.databricks.com/files/sxv157130/printFinalDT.txt?o=1663046746195488

```
0.47397675593734206 0.35421930267812024 0.17180394138453764
0.835151802656546 5 0.1453984819734345 2 0.019449715370018977
```

Model Evaluation

According to Kaggle competition evaluation, the results are evaluated using multiclass Logistic regression loss.

$$logloss = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{i,j}\log(p_{i,j})$$

N is number of Listings,

M is number of Class labels,

Pi,j is predicted probability,

Y i,j is 1 if observed label belongs to class j,

Y i ,j is 0 , if observed label does not belong to class j.

Conclusion

After evaluating on various valid features, we found that Random Forest Classifier and Decision Tree classifiers give the same average log loss. Since Decision tree is not as consistent as Random Forest and small change in data leads to change in log loss.

Hence we consider Random Forest as the best option to predict the interest level of rental listing.