**Deliverable 4**

**-Data Mining-**

**CSI 4142 - Introduction to Data Science**

**Winter 2020**

**School of Electrical Engineering and Computer Science**

**University of Ottawa**

Course Coordinator: Herna Viktor

Group 24

Salomé Dionne-Laforest 300035614

Jonathan Popowick-Bastien 8432984

Zelalem Weldemichael

7669132

Submission Date: 04/03/20

Task 2 : Cluster Analysis

In this model we were interested in which groups of neighborhoods had similarity in terms of type of crime frequency.

**Cluster analysis:**

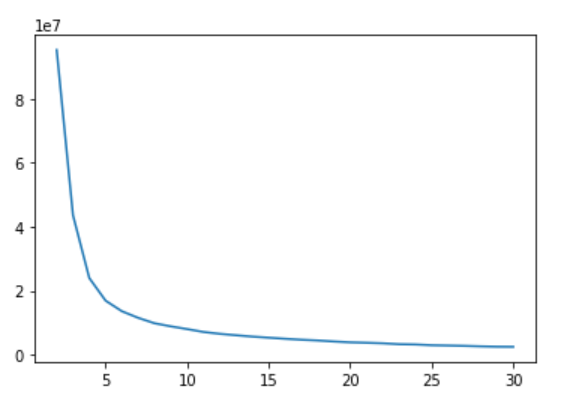
The clustering task was done using python’s Scikit-Learn library. We applied two algorithms of cluster analysis partitioning-based and model-based algorithms. We used KMeans algorithm for partitioning and Gaussian Mixture model algorithm for model-based clustering. Both had high accuracy but there was a complete difference in the way the data was clustered.

**Background:**  Before we started building any of the models, the following steps were done:

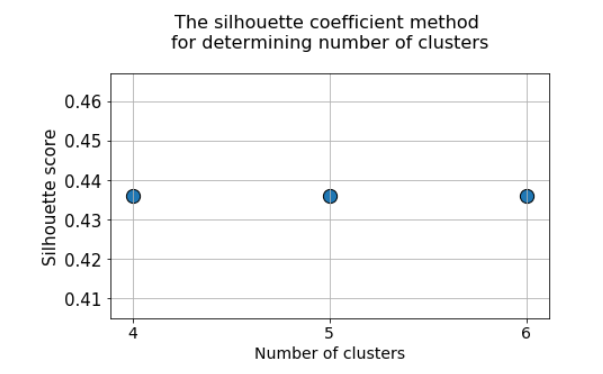
· We changed the categorical nature of the data to integer using label encoder, reason for not using one-hot was the data become to0 sparse due to the nature of the data (too many neighbourhoods and crime types)

· We then used a combination of the Elbow Rule and silhouette rule to determine the number of clusters needed.

· We ran the kMeans ranging between 2 to 31clusters we see the elbow ranging around 5 so we used silhouette to confirm that number



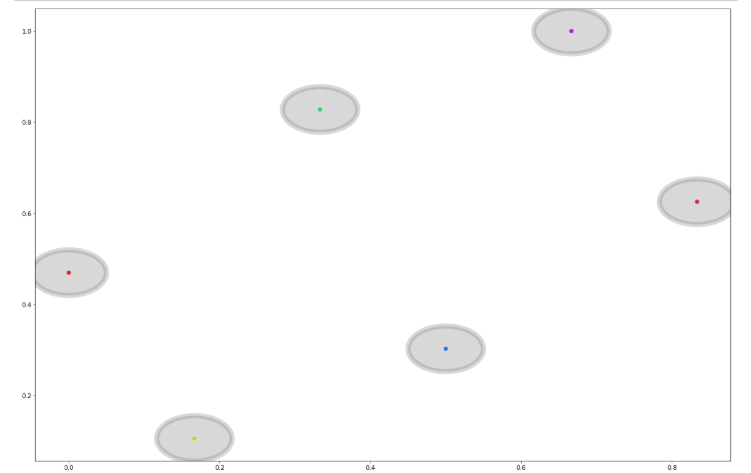
· We then used silhouette score for the suspect elbow rule result 4,5,6 and they all had the same score as shown in the graph below. Hence, we used 6 as the number of clusters for our models.



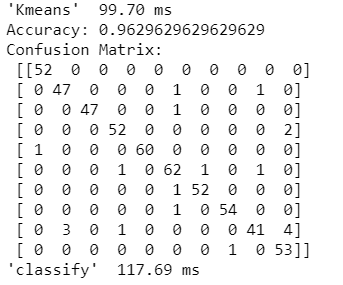
**1.Partition Based:**

**KMeans:**

Cluster Grouping

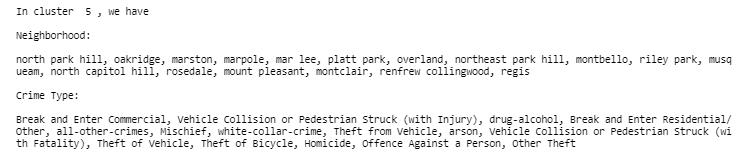


**Model Performance**

****

From the above figure we can see the model we built is pretty fast and has a high accuracy of 96.3%. We also observe that the clusters have decent distance between them. This indicates our model performed very well. When we dig deeper to see how it was grouped and analyse the neighbourhood and crimes in each cluster this was the output:



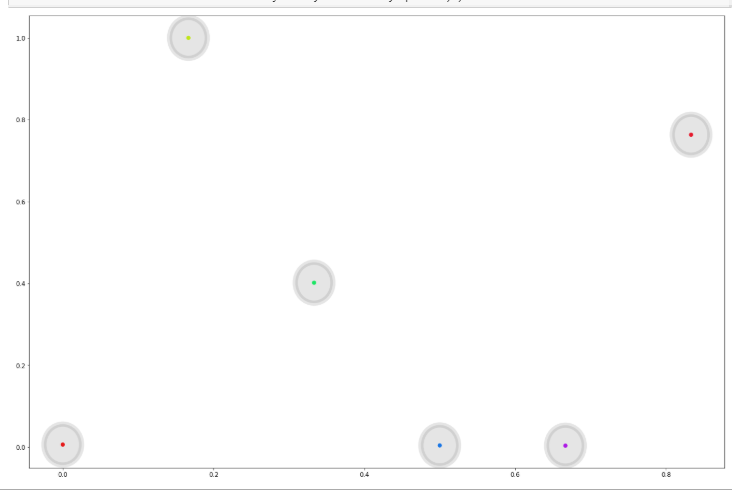


The image above is a snapshot of how the clusters were grouped, we see a sensible grouping of the neighbourhoods, but each crime type is included in all the clusters. This was unexpected as some crimes did not happen in certain neighbourhoods.

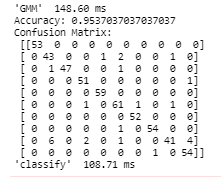
**2.Model Based:**

**Gaussian Mixture Model**

**Cluster grouping:**

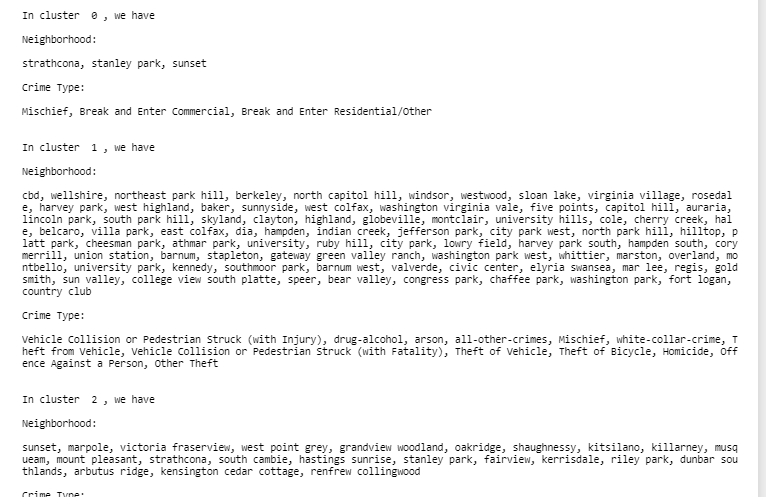
****

**Model Performance:**

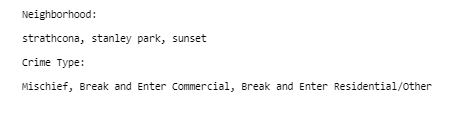
****

From the model performance we can see our model is fast and has a high accuracy of 95.37 %. We also observe the clusters formed have a good distance between them. As we did with the partition when we look at the grouping of the clusters.

Here below is a snapshot of the printout in the clusters



This was quite interesting as at first glance we see that there are a lot of neighbourhoods in cluster one and some clusters include only one neighbourhood. A further look shows us, all the neighbourhoods from Denver are all in one cluster and the rest of the clusters are Vancouver neighbourhoods grouped in different clusters. Another interesting look is we can note that unlike the partition model, the model based does not include all the crimes in all the clusters. It performed better in finding relationship between neighbourhoods in terms crime type frequencies of as it can be highlighted by result below



From the cluster above we can see that “Strathcona, Stanley park, Sunset” have a similar frequency of the following crimes “Break and Enter residential/Other”, “Break and Enter Commercial” and “Mischief”

**Lessons Learned:**

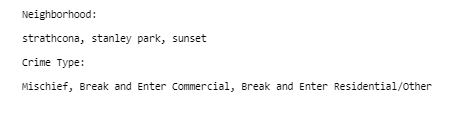
**Partition Based:**

As discussed above, even though the accuracy is high it's hard to make a conclusion based on this model as each cluster included all the crime types, even though it's clear some neighbourhoods in the cluster did not have a record of certain crimes. This could be because:

* The data is not large enough to help our model learn better or
* Hard assignment used by K-Means might be leading to mis-grouping

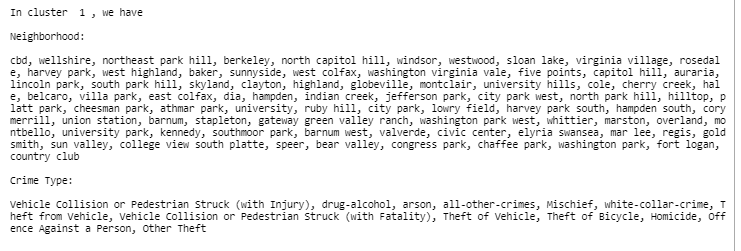
**Model Based:**

As discussed above, this model was quite interesting, and we found the results to be better than partitioning. For instance, the model seems to recognize the neighbourhoods that are from two different cities as each cluster included neighbourhoods from 1 city. Also, not all the crime types are included in every cluster.

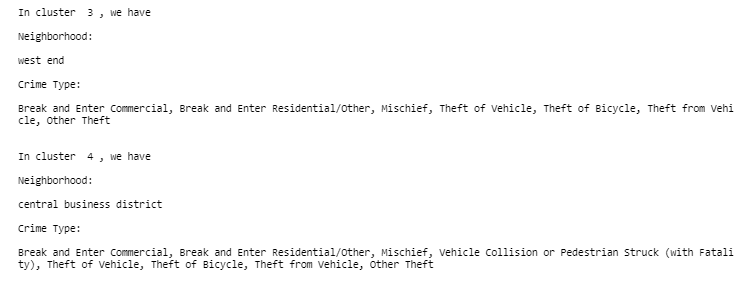


From the cluster above we can see that “Strathcona, Stanley park, ‘Sunset” have a similar frequency of the following crimes: “Break and Enter residential/Other, “Break and Enter Commercial, Mischief”.

The model also placed all the Denver neighbourhoods in one cluster, indicating all the neighbourhoods from Denver have a similar crime type frequency for all the crime types. As indicated by the result below

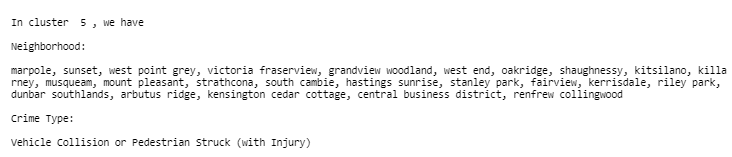


The model also had some clusters that included only one neighbourhoods as it can be seen by the image below



From the picture above we can see that “Central business district” and “west end” have a different frequency of crimes than any other neighbourhood in Vancouver.

This image below also shows the neighbourhoods that have the same frequency of “Vehicle Collision or Pedestrian Struck (with Injury)” in Vancouver



In summary from this model we learn that

* In Denver all crimes happen at similar frequency in all the neighbourhoods
* In Vancouver with the exception of few neighbourhoods the crimes happen at a similar frequency in all the neighbourhoods

**Conclusion:**

From our analysis we believe the Model-Based algorithm is better for our data, as it shows some correlation between certain neighbourhoods. This could be because unlike K-Means Gaussian-Mixture does not bias the cluster sizes to have specific structures. Having said that even in the model-based we observe some inconsistency as we did not expect all the crimes in Denver to happen at a similar frequency in all the neighbourhoods. This could be due to small data sets as we are looking only in two years of data or it could be that it needs more work in terms of feature engineering in order to find a solid relationship.

Task 1 : Classification

In this model, we want to predict what type of crime is committed depending on the predictors. To make it easier, we chose the following predictors : neighborhood, night-time or daytime, temperature, month and season.

We decided to use ensembles (bagging and boosting) and decision trees to classify the data. We used R to do it. The code is in appendix.

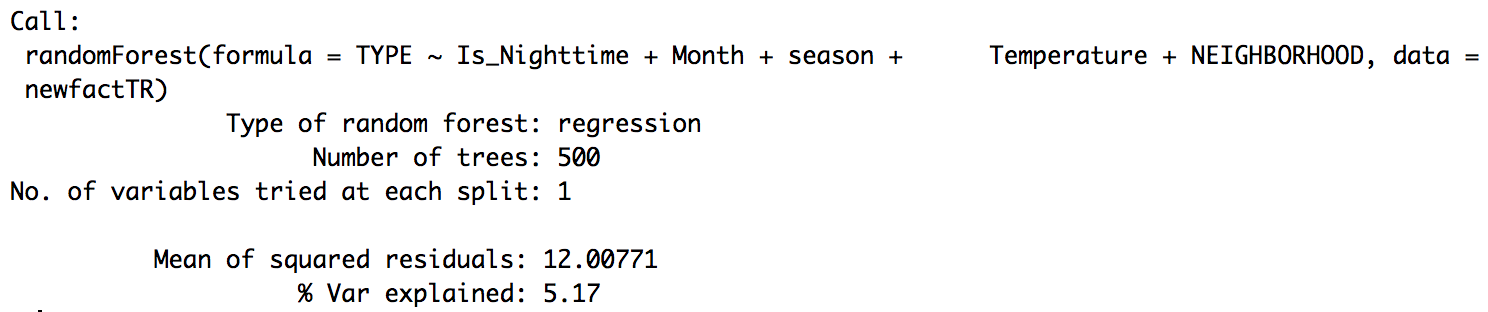
Mean Squared Error =

*Ensemble : Boosting*

|  |  |
| --- | --- |
| Shrinkage parameter : 0.001  Mean Squared Error : 11.40805 | Shrinkage parameter : 0.01  Mean Squared Error : 11.7126 |

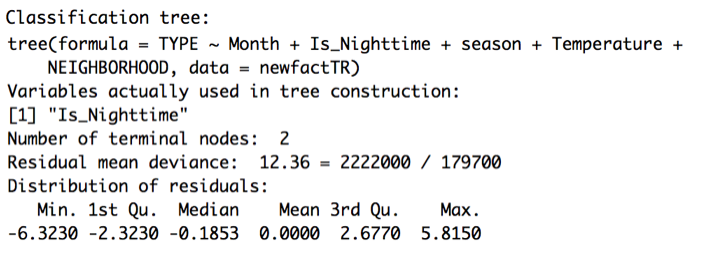
*Ensemble : Bagging*

Mean Squared Error : 11.88624



*Decision tree*

Mean Squared Error : 12.0506



**Lessons learned:**

Boosting seems to be doing better than the other two classifiers. Using a shrinkage parameter of 0.001, we get a mean squared error term of 11.41 while the mean squared error for bagging and decision tree are 11.89 and 12.05 respectively. Therefore, boosting is better at classifying the predictors in the right class.

That could be explained with the number of parameters the methods are using. We can see that the boosting method includes more parameters. For bagging and decision trees, they both seem to include only the night-time or daytime predictor. Bagging is better than the decision tree because it created its tree averaging 500 trees, but as we can see in the boosting plot, the neighbourhood is an important predictor that should be used to predict the class (as we would indeed suspect). The time of the day and the temperature are also very important for the prediction. As for the month and season, they have a smaller effect on the data, so they are less important to make predictions.

Appendix : R code for Task 1

*mergefact = merge(factTable, crimeTable,by.x = Crime\_key,by.y = Crime\_key)*

*mergefact = merge(factTable, crimeTable,by.x = 'Crime\_key',by.y = 'Crime\_key')*

*mergefact = merge(mergefact, dateTable, by.x = 'Date\_key', by.y = 'Date\_key')*

*mergefact = merge(mergefact, weatherTable, by.x = 'Weather\_key', by.y = 'Weather\_key')*

*mergefact = merge(mergefact, locationTable, by.x = 'Location\_key', by.y = 'Location\_key')*

*newfact = mergefact[,c(7,8,9,10,13,18,19,23,24,31,32)]*

*newfactTR = newfact[which(newfact$Year==2015),]*

*newfactTE = newfact[which(newfact$Year==2016),]*

Boosting :

*boost.type = gbm(TYPE~Is\_Nighttime+Month+season+Temperature+NEIGHBORHOOD,data = newfactTR, distribution = "gaussian", n.trees=5000)*

*yhat.boost=predict(boost.type,newdata = newfactTE, n.trees=5000)*

*mean((yhat.boost-newfactTE[,5])^2)*

*boost.type = gbm(TYPE~Is\_Nighttime+Month+season+Temperature+NEIGHBORHOOD,data = newfactTR, distribution = "gaussian", n.trees=5000, shrinkage = 0.01)*

*yhat.boost=predict(boost.type,newdata = newfactTE, n.trees=5000)*

*mean((yhat.boost-newfactTE[,5])^2)*

Bagging :

*bag.type = randomForest(TYPE~Is\_Nighttime+Month+season+Temperature,data = newfactTR)*

*yhat.bag=predict(bag.type,newdata = newfactTE)*

*mean((yhat.bag-newfactTE[,5])^2)*

Decision tree :

*tree.type = tree(TYPE~Is\_Nighttime+Month+season+Temperature+NEIGHBORHOOD,data = newfactTR)*

*yhat.tree=predict(tree.type,newdata = newfactTE)*

*mean((yhat.tree-newfactTE[,5])^2)*