Data Science 2019/2020

Group 47

João Daniel
86445
joao.daniel.silva@tecnico.ulisboa.pt

João Barata
86450
jbarata1998@gmail.com

Bruno Dias
86392
bruno152251281@gmail.com

1. INTRODUCTION

The data used in this project was retrieved from 2 different studies. With respect to the first study, it was used a dataset that refers to the classification of patients with Parkinson's disease, by using speech signal processing algorithms to extract clinically useful information for its assessment. With regard to the second study, the dataset used refers to the prediction of the forest covertype on 4 different wilderness areas in the Roosevelt National Forest of northern Colorado, using cartographic variables obtained from US Geological Survey and USFS data.

Our goal will be to explore the data gathered during the collection, use different learning models and finally aim to improve the data mining process.

2. STATISTICAL DESCRIPTION

2.1 First dataset

The data used was gathered from an experimental group consisting of 188 patients with Parkinson's disease with age ranging from 33 to 87, the control group consisted of 64 healthy individuals. The dataset contains 756 instances in total and 754 attributes, by analyzing the attributes we see that from the 754, all of them are numeric, in specific 5 integers and 749 float types. On the same note, regarding the majority class and the minority, we observed a proportion of 1 to 3, therefore concluding the dataset is imbalanced. The task associated is a binary classification one, since the class attribute has 2 possible values, 0 or 1. There are no missing values in this dataset, so it wasn't necessary to do any kind of imputation.

Due to the high number of attributes, it was opted to select the 10 most "important" features, the ones that matter the most to the target variable, using the **SelectKBest** function. It was decided to use the **histogram_with_distributions** function from the labs and by the analysis of the figures, we did not verify a general distribution amongst the attributes.

The fact that the data was collected from an examination that conducted 3 different repetitions caused the data for each patient to be quite similar, in order to remove the dependency between the rows, it was decided to calculate the mean between these values, decreasing the number of rows per patient to 1 row.

2.2 Second dataset

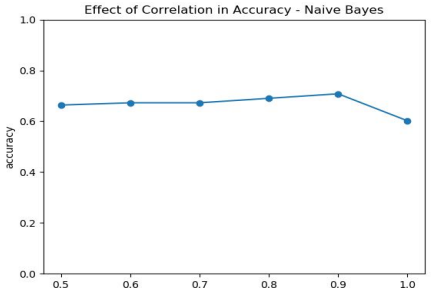
The second dataset contains 581012 observations and 54 attributes, divided into 2 different types, numeric and categorical. More specifically, the categorical ones are represented by 10 quantitative variables and the numeric ones by binary variables, representing the wilderness areas and binary soil types. Taking into account the **covtype.info** file we analyzed the class distribution of the dataset and calculating the proportion between the majority class *Lodgepole Pine* and the minority class *Cottonwood/Willow* we obtained a 1:103 proportion, therefore concluding that the dataset is unbalanced. The task associated is a multiclass classification problem due to the nature of the different kinds of possible cover types. There are no missing values, so no imputation was needed.

A similar approach to the one taken in the first dataset was implemented regarding the distribution amongst the attributes and we did not verify a common distribution, neither binomial, normal or uniform.

3. PRE-PROCESSING TECHNIQUES

3.1 First dataset

Given the high amount of features and the fact that they are subdivided into group types, it was studied if by removing highly correlated variables the classifiers scores would improve.



Testing correlation thresholds from 0.5 to 1, every correlation value surpasses the performance if this process was not done (threshold = 1). This can be explained as classifying with a large amount of features leads to overfitting since the classifier tries so many parameters to fit the data.

By analyzing the performance of the Naive Bayes and KNN algorithms we tried to conclude which sampling was the best.

OverSample	Smote	UnderSample
accuracy: 0.60	accuracy: 0.62	accuracy: 0.54
f1-score: 0.57	f1-score: 0.59	f1-score: 0.51

3.1.1 Naive Bayes performance scores

OverSample	Smote	UnderSample
accuracy: 0.93	accuracy: 0.87	accuracy: 0.79
f1-score: 0.93	f1-score: 0.86	f1-score: 0.79

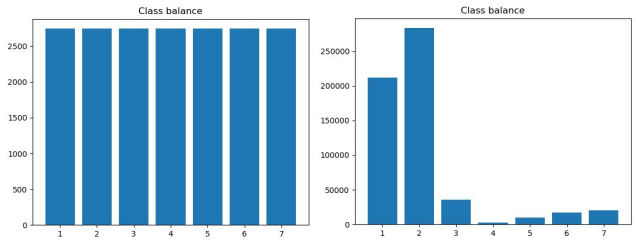
3.1.2 KNN performance scores

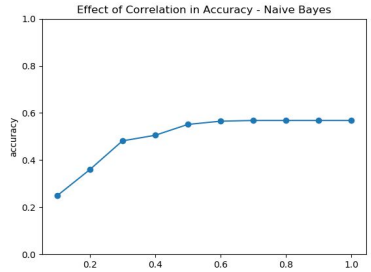
From this it can be concluded that UnderSample is the type of sampling with the worst performance, SMOTE and OverSample have rather similar scores in Naive Bayes. On the same note, looking at both scores, even though it is not a considerable difference, Oversample stands out as the best. Taking this into consideration, we use the latter in the rest of the classifiers.

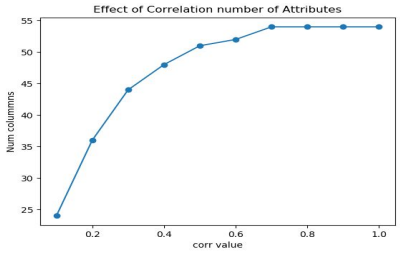
In clustering, the data was standardized by removing the mean and scaling to unit variance. In this dataset, it was studied the impact that sampling had on clustering.

3.2 Second dataset

The second dataset is very large with 500 000+ rows like we mentioned in the statistical description section. In order to reduce the original size mentioned above, it was opted to undersample, so the number of rows dropped to 2747 for each class, summing up to a total of 19229 rows, much lower than the original. We used correlation to reduce the number of attributes, but after analyzing the results we concluded that was not beneficial unless we used a very low value of minimum correlation between variables. On the same note, using the low correlation value on Naive Bayes proved to have a really low accuracy score. Therefore, it was decided not to use correlation in this dataset.







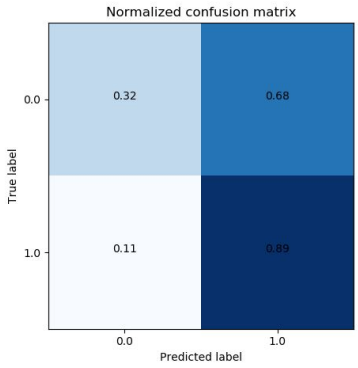
Regarding clustering and pattern mining and considering again the high number of rows, if it were to be used all the original data the algorithm would not finish in useful time, therefore, it was decided to randomly sample for each class 1/100 of their rows maintaining the original proportion between the classes.

4. CLASSIFICATION

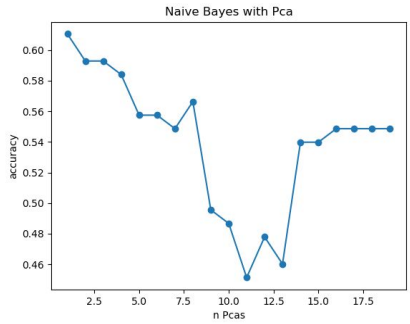
4.1 Naive Bayes

4.1.1 PD dataset

Using Naive Bayes we tried to understand which sampling method was the best for the dataset. Due to the nature of our data being mostly of the float type, the Gaussian Naive Bayes is the most appropriate. PCA analysis was also taken into consideration.



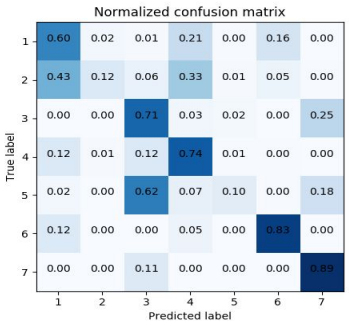
Fig(4.1.1.1) OverSample confusion matrix



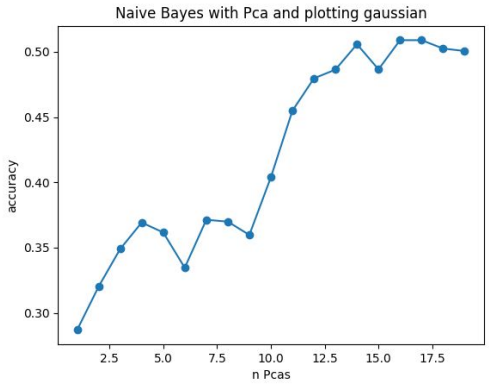
4.1.2 CT dataset

Due to the nature of this dataset variables, Gaussian Naive Bayes was the approach taken in this analysis. This classifier was performed on an undersampled dataset and it was opted to test with PCA as well.

K=1 results KNN



Fig(4.1.2.1) OverSample confusion matrix

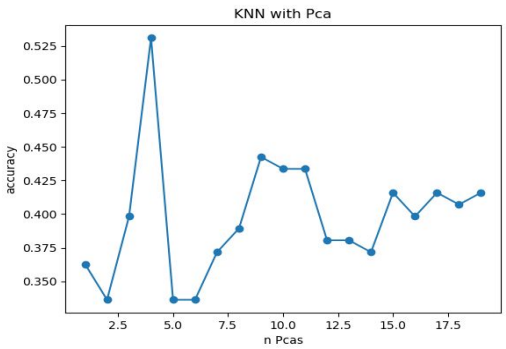


4.2 Instance-based learning KNN

4.2.1 PD dataset

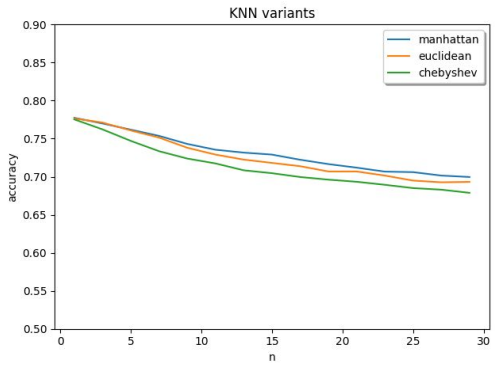
The KNN algorithm requires the data to be normalized, so it was done using StandardScaler. As previously mentioned the KNN had a very high performance with OverSampling returning values such as k= 1 and distance = **manhattan**.

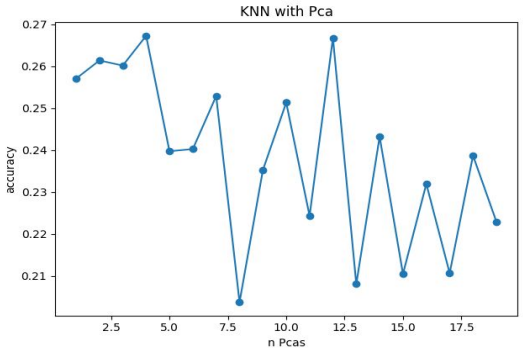
It was also tested the use of PCA with KNN.



4.2.2 CT dataset

Regarding this dataset, we took the same approach and started by normalizing the data using StandardScaler. No correlation was used and the dataset was undersampled. The results revealed that the classifier performed the best with **manhattan** distance and a k = 1.





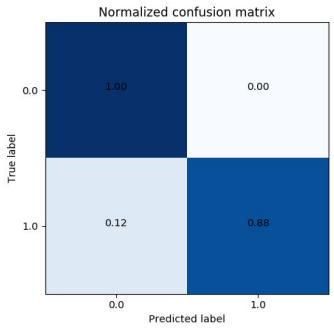
Similarly to the *PD dataset,* PCA with KNN was also analyzed.

4.3 Decision Trees

4.3.1 PD dataset

When it comes to decision trees, firstly, we decided to use the 2 different criteria **gini** and **entropy** that choose the best variable to split the dataset. Secondly, an overfitting control technique was also implemented **min_sample_leaf** with varying values. For pruning, different maximum depth values were used to study the impact on accuracy scores. Lastly, the best values for each of the above were calculated and the accuracy score was 0.94, with the **min_sample_leaf = 0.0075**, **gini** as the criteria and 5 as the **maximum depth**. We

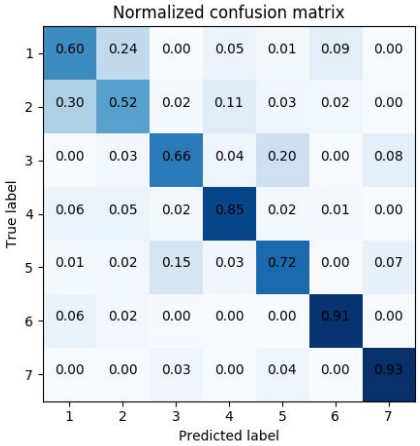
concluded that even though the difference was not substantial it did behave better for lower max_depth values.



This classifier as represented by the confusion matrix above performed very well on predicting instances from the 2 classes.

4.3.2 CT dataset

In this dataset, we applied this classifier on the undersampled dataset. We repeated the process implemented on the first dataset, by drawing a graph with the different parameters, **criteria**, **min_samples_leaf**, **max_depth,** and their impact on accuracy. After the analysis of the figures, we got results starting from **criteria** = gini, **min_samples_leaf** =0.001, **max_depth** = 50. This dataset didn't behave better for the lowest max_depth values. Lastly, the accuracy score for the best values calculated for each parameter was 0.74.



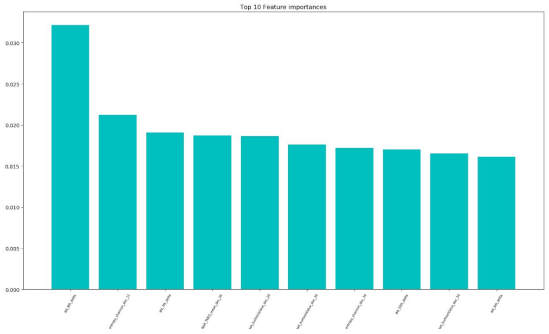
4.4 Random Forests

The best performance of **random forests** was achieved by experimenting on different parameters which are the number of trees in the forest, the maximum depth of the tree and the number of features to consider when looking for the best split.

4.4.1 PD dataset

This classifier had an accuracy and an f1-score of 0.98. with number of estimators = 100, max depth = 5 and feature = sqrt.

As expected of this type of classifier. its performance was very high.



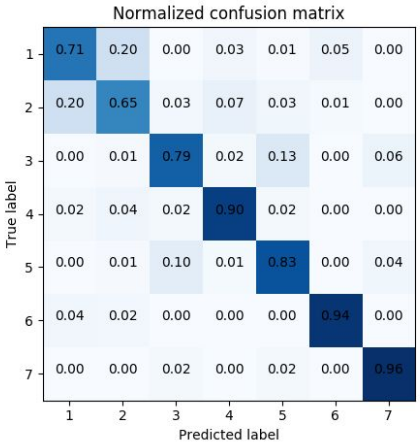
This graph represents the 10 most important features for splitting the data.

The group which variables are present the most in this top 10 is the **wavelet features** group with 4.

4.4.2 CT dataset

This classifier had an accuracy and an f1-score of 0.83. with number of estimators = 200, max depth = 25 and feature = sqrt.

The most important variables are 'Vertical Distance To Hydrology', 'Hillshade 3pm', 'Slope', 'Horizontal Distance To Hydrology', 'Horizontal Distance To Roadways', 'Elevation', 'Comanche Peak Wilderness AreaCache la Poudre Wilderness Area', 'Hillshade Noon', 'Hillshade 9am', 'Aspect'.



This performance was high, although not as near as in the previous dataset, in which the classification was almost perfect.

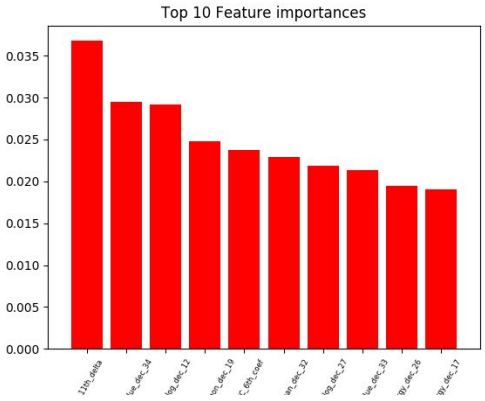
4.5 XGBoost

The best performance of **XGboost** was achieved by experimenting with different parameters which are the number of trees in the forest, the maximum depth of the tree and the number of features to consider.

4.5.1 PD dataset

This classifier had an accuracy and an f1-score of 0.98. with number of estimators = 50, max depth = 5 and feature = log2.

As expected of this type of classifier its performance was very high.

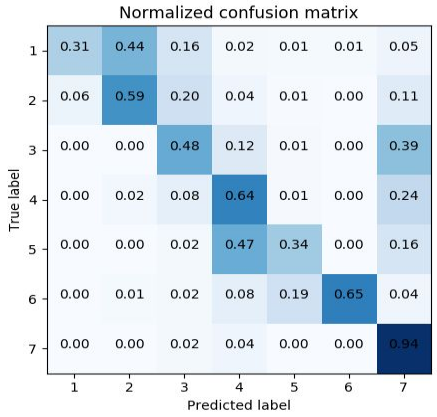


This graph represents the 10 most important features for splitting the data. The most important variables to split belong to the **Wavelet** group and the remainder are from the **TQWT**.

4.5.2 CT dataset

This classifier had an accuracy and an f1-score of 0.56. with number of estimators = 100, max depth = 25 and feature = sqrt.

The most important variables are 'Vertical Distance To Hydrology', 'Hillshade 3pm', 'Slope', '39', 'Elevation', 'Horizontal Distance To Hydrology', 'Hillshade Noon', 'Horizontal Distance To Roadways', 'Hillshade 9am', '38'.



As shown by this matrix, the performance of the algorithm is not that accurate.

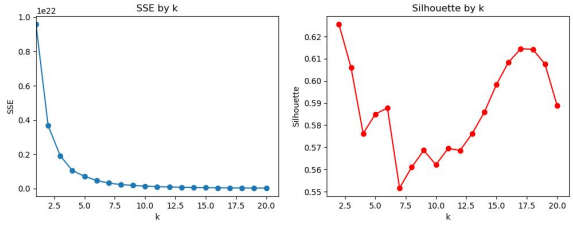
5. UNSUPERVISED

5.1 Clustering

For both datasets, we applied two clustering techniques: **k-means** and **hierarchical agglomerative**. In k-means we start by applying the algorithm varying the value of k and plotting the corresponding sum of squared errors (inertia) and silhouette. We then retrieve the best k by analyzing the elbow in the inertia curve and then record a contingency matrix. Finally, we use PCA to plot the clustering method.

5.1.1 PD dataset

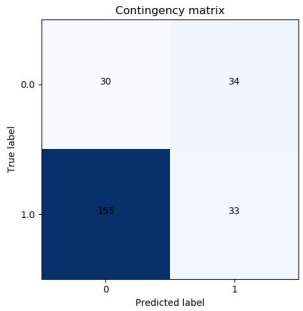
5.1.1.1 K-Means



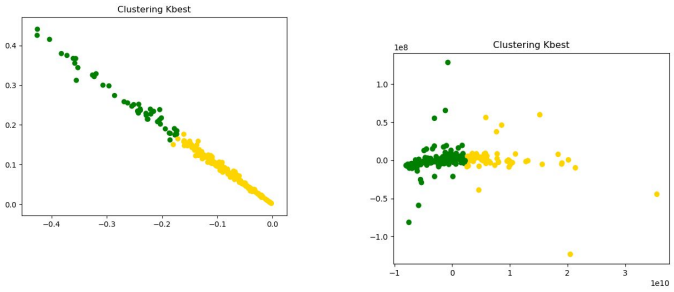
The "elbow" can be seen starting at k=2 and this is where the silhouette is the highest. Some silhouette stats for k-means with k=2:

Silhouettes	**Rand Index**
[all] = 0.625	[all] = 0.206
[SelectKBest=2] = 0.662	[SelectKBest=2] = 0.244
[PCA 2 components] =0.625	[PCA 2 components] = 0.206

Purity[all]: 0.75

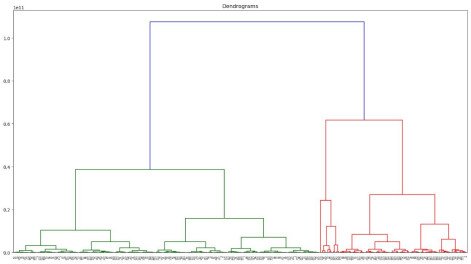


Fig(5.1.1.1.1) Contingency matrix from k-means, k=2



Fig(5.1.1.1.2)Visualizations with SelectKbest and PCA components= 2.

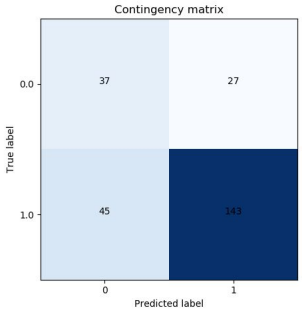
5.1.1.2 Hierarchical



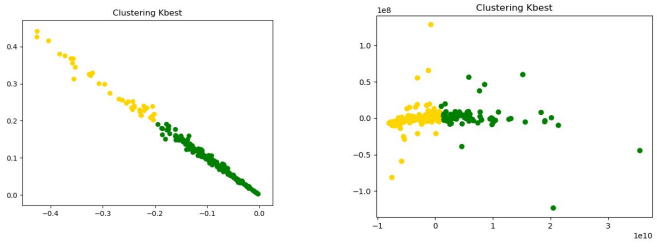
Plotting the dendrogram it is clear that the best number of clusters is 2.

Silhouettes	**Rand Index**
[all] = 0.603	[all] =0.157
[SelectKBest=2] = 0.234	[SelectKBest=2] = 0.234
[PCA 2 components] = 0.603	[PCA 2 components] = 0.156

Purity[all]: 0.746



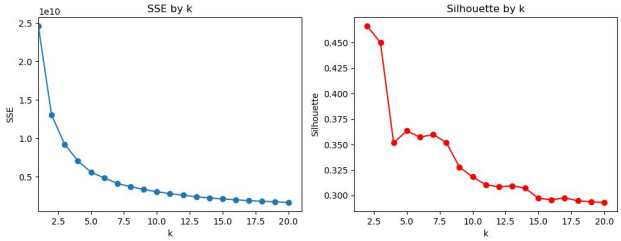
Fig(5.1.1.2.1)Contingency matrix from hierarchical agglomerative with k=2



Visualizations with SelectKbest = 2 and PCA components = 2.

5.1.2 CT dataset

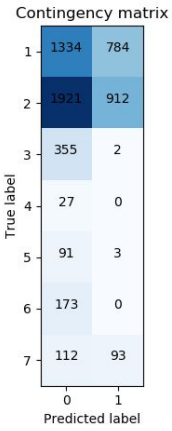
5.1.2.1 K-Means



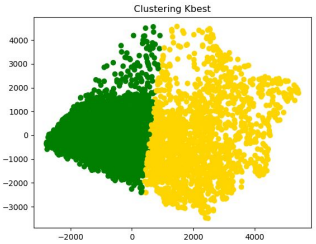
The "elbow" can be seen starting at k=2 and this is where the silhouette is the highest. Some silhouette stats for k-means with k=2:

Silhouettes	**Rand Index**
[all] = 0.466	[all] = 0
[SelectKBest=2] = 0.934	[SelectKBest=2] = 0.126
[PCA 2 components] = 0.477	[PCA 2 components] = 0

Purity[all]: 0..488

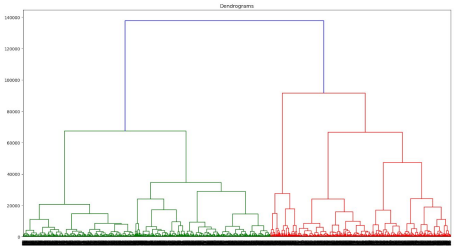


Fig(5.1.2.1.1)Contingency matrix from k-means, k=2



Fig(5.1.2.1.2)Visualization with PCA components = 2.

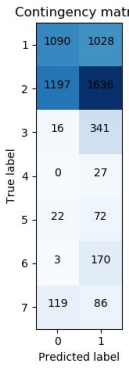
5.1.2.2 Hierarchical



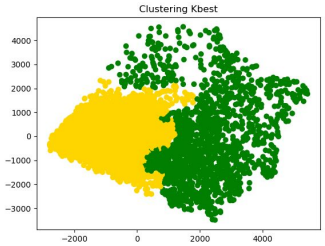
Plotting the dendrogram it is clear that the best number of clusters is 2 as well.

Silhouettes	**Rand Index**
[all] =0.365	[all] = 0
[SelectKBest=2] = 0.94	[SelectKBest=2] = 0.13
[PCA 2 components] = 0.47	[PCA 2 components] = 0

Purity: 0.49



Fig(5.1.2.2.2)Contingency matrix from hierarchical, k=2



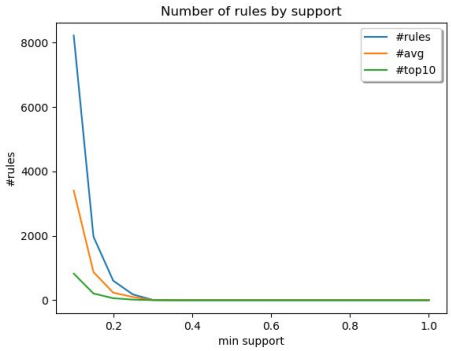
Fig(5.1.2.2.2)Visualization with PCA components = 2.

5.2 Association Rule Mining

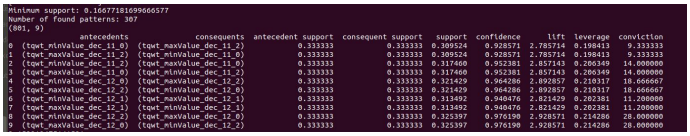
For both datasets, we selected 10 Kbest features. Then we used a number of Bins = 3, and finally filtered the rules in which the length of the patterns is higher than 2 and the minimum value of confidence is 90%.

5.2.1 PD DataSet

We iterated with an increasing value of support in order to obtain the total number of rules, the number of rules whose lift is better than the average and the number of rules in the top 10%, all regarding the respective support value as shown by the following figure.



Then, we fixed the minimum number of patterns to a value of 300 and we iterated with a decreasing number of support so we can get the **minsupport** in which the number of patterns found is the highest or equal to the fixed minimum value. The following figure shows the first 10 rules found.

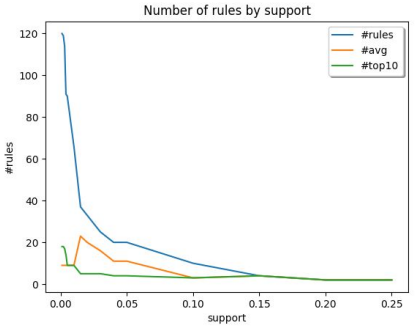


Minimum support: 0.16677181699666577
Number of found patterns: 307
(881, 9)

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
0	(tqwt_minValue_dec_11_0)	(tqwt_maxValue_dec_11_2)	0.333333	0.333333	0.309524	0.928571	2.785714	0.198413	9.333333
1	(tqwt_maxValue_dec_11_2)	(tqwt_minValue_dec_11_0)	0.333333	0.333333	0.309524	0.928571	2.785714	0.198413	9.333333
2	(tqwt_minValue_dec_11_2)	(tqwt_maxValue_dec_11_0)	0.333333	0.333333	0.317460	0.952381	2.857143	0.206349	14.000000
3	(tqwt_maxValue_dec_11_0)	(tqwt_minValue_dec_11_2)	0.333333	0.333333	0.317460	0.952381	2.857143	0.206349	14.000000
4	(tqwt_minValue_dec_12_0)	(tqwt_maxValue_dec_12_2)	0.333333	0.333333	0.321429	0.964286	2.892857	0.210317	18.666667
5	(tqwt_maxValue_dec_12_2)	(tqwt_minValue_dec_12_0)	0.333333	0.333333	0.321429	0.964286	2.892857	0.210317	18.666667
6	(tqwt_minValue_dec_12_1)	(tqwt_maxValue_dec_12_1)	0.333333	0.333333	0.313492	0.940476	2.821429	0.202381	11.200000
7	(tqwt_maxValue_dec_12_1)	(tqwt_minValue_dec_12_1)	0.333333	0.333333	0.313492	0.940476	2.821429	0.202381	11.200000
8	(tqwt_minValue_dec_12_2)	(tqwt_maxValue_dec_12_0)	0.333333	0.333333	0.325397	0.976190	2.928571	0.214286	28.000000
9	(tqwt_maxValue_dec_12_0)	(tqwt_minValue_dec_12_2)	0.333333	0.333333	0.325397	0.976190	2.928571	0.214286	28.000000

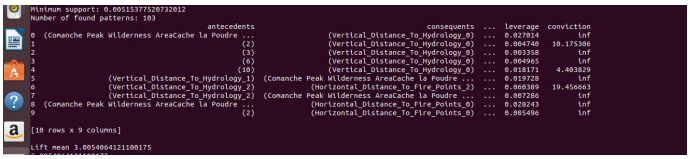
5.2.3 CT DataSet

We iterated with an increasing value of support like in the previous dataset but this time with really low support values, in order to obtain the total number of rules, the number of rules whose lift is better than the average and the number of rules in the top 10%, all regarding the respective support value as shown by the following figure



Regarding this plot, we can see that the minimum support corresponding to an adequate number of rules found is really low and the number of patterns obtained is not that high.

Then, we fixed the minimum number of patterns to a value of 100 and we iterated with a decreasing number of support so we can get the **minsupport** in which the number of patterns found is higher or equal to the fixed minimum value. The following figure shows the first 10 rules found as well as the **minsupport** and lift mean of the rules.



6. CRITICAL ANALYSIS

6.1 PD Dataset

Naive-Bayes: Even with oversampling this classifier struggled when predicting instances from class 0, having a 0.32 accuracy score only. In regard to the PCA, increasing the number of PCA's decreased the overall performance of the classifier.

Instance-based learning KNN: KNN proved to have a really high performance on this dataset. When tuning the parameters, **manhattan** distance was the most adequate for this dataset, this can be explained due to the fact that this measurement is better for high dimension datasets. We concluded that having a value of 1 for the nearest neighbor was the best number for this problem due to its binary nature, preventing ties. Finally, we decided it would be interesting testing the use of PCA with KNN, analyzing the different accuracy scores we got to the conclusion that using PCA on an already high accuracy algorithm would decrease the overall performance.

Decision Trees: For this classifier, we obtained a really high value for the accuracy mainly due to the **max_depth** value calculated being really low, this prevented the tree to over-fit the data. The different criteria performed quite similarly with **gini** proving to have an edge by a small percentage, this proved that choosing one criteria over another won't have a significant impact on the overall performance of the model.

Random forests: This classifier had a near-perfect performance on this dataset with an overall accuracy score of 0.98. Even though the concept is similar, random forests have a high level of robustness compared to decision trees because they are basically a collection of the previous, this explains the higher accuracy achieved. Similarly to the criteria **gini** and **entropy,** when it comes to the **max_features**, between sqrt and log2 we concluded that the tree wasn't very sensitive about the value of this parameter.

XGBoost: This classifier had a near-perfect performance on this dataset with an overall accuracy score of 0.98. This value is as high as expected since it is an iterative optimization and one of the best classifiers for datasets in general.

Clustering: Firstly, the value chosen for the optimal clusters matches the number of classes in this dataset. The performance of both algorithms is similar: **silhouette** values are relatively high, but the **rand index** is low. This means that points in the same cluster are quite similar to each other and different from those in other clusters but they don't represent well the original classes. Also, the visualizations are quite similar. With the contingency matrixes, it can be concluded that the algorithms can't split different classes in different clusters.

Association Rule Mining: Even though the lift presented is in good range, the rules associated are not the best because they are from variables from within the same category, so despite these are rules that we can follow, since we selected the 10 **Kbest** variables from the all dataset, we are not sure that we have selected at least one from each category, so our rules would have a lower importance when compared to rules between variables from different categories.

6.2 CT Dataset

Naive-Bayes: This classifier was applied on an undersampled dataset, the accuracy score was 0.56 overall. Analyzing the confusion matrix, we concluded that the classifier had really low performances on predicting the 5th and 2nd class. As the number of PCA's increased the accuracy decreased.

Instance-based learning KNN: The performance achieved using this classifier was lower compared to the first dataset. Even though the dataset had much more variables, the values stood rather similar on the number of neighbors and distance measurement used, proving **manhattan** does perform better on higher dimensions problems. PCA had a negative effect on the overall accuracy score as well.

Decision Trees: For this classifier, unlike the *PD dataset,* the accuracy score was lower and the **max_depth** value was higher, we used values ranging from 5 to 50 for the different depths on the first dataset but decided to increase the values to 150 due to the increased complexity of the dataset. The obtained value showed that the peak performance was achieved on the **max_depth = 50** value, showing that the classifier could not perform well on low values (**underfitting)** but not also on high values (tendency to **overfit**). The **min_samples_leaf** is also smaller (**0.001 * n_samples)** indicating that the tree needs more leaves to perform better due to the higher amount of attributes in the data.

Random forests and XGBoost: The accuracy achieved with **randomforests** (of value 0.83) wasn't as high as in the previous dataset, but we also have to consider that an almost perfect accuracy obtained in the previous dataset was referring to only 2 classes contrasting to the 7 classes that the CT dataset contains. Regarding this, classification using random forests is actually good. On the other hand, using **XGBoost** we have obtained a low accuracy (of value 0.56) and the same thing happens for XGBoost classification but since the value is just slightly over 50%, we conclude that XGBoost is not a good classifier for this dataset.

Clustering: Even though there are 7 classes, both algorithms computed two clusters. This can be explained due to the fact that the data set is very highly imbalanced. The performance of both algorithms is similar: **silhouette** values are relatively low, and so is the **rand index**. This means that the points are poorly clustered; the clusters are near each other and the belonging points are not similar to each other. Purity reflects this as well.

With the contingency matrixes, it can be concluded that the algorithms can't split different classes in different clusters, as expected from the analysis above.

Association Rule Mining: The mean of the rules' lift obtained for this dataset has a good range (~3), but there is not much evidence for this rules since the minimum support found is really low, we do not have enough information (proof) between the antecedent and the consequent.

7.CONCLUSION

To sum up, the results previously shown were crucial to our understanding of how the various classifiers behave when being applied on different datasets. Namely, their performance is highly dependent on the format, type of classification problem and type of the variables. Preprocessing proved to be the reason for an increase in the accuracy scores throughout the data exploration.