Applying Classification, Association, and Clustering Algorithms Into Two Datasets

Mariana Botelho Pereira[1]

[1]Institute of Computing – Federal Fluminense University (UFF)
Av. Gal. Milton Tavares de Souza, s/n – 24210-346 – Niterói – RJ – Brazil

marianabotelho@id.uff.br

Abstract. *This report aims to present results from analysing two datasets using classification, association, and clustering algorithms. Each algorithm was analysed with different parameters to evaluate the changes on the final scores. Several graphs were ploted for ease of visualsiation as well as tables with the accurate values.*

1. Datasets

To develop this project, two datasets were used. In this section, we'll be describing both of them.

1.1. Early Stage Diabetes Risk Prediction Dataset

This dataset consists of 16 features and 520 non-null instances. Each instance consists of questions asked to patients at the Sylhet Diabetes Hospital in Sylhet, Bangladesh. One feature presents the age of the patient, one presents the gender, one presents the diagnostics (with or without diabetes) and the other features present symptoms reported by the patients. The full research can be found at [Islam et al. 2020].

- Age: 16-90;
- Gender: Male/Female;
- Polyuria: Yes/No;
- Polydipsia: Yes/No;
- Sudden Weight Loss: Yes/No;
- Weakness: Yes/No;
- Polyphagia: Yes/No;
- Genital Thrush: Yes/No;
- Visual Blurring: Yes/No;
- Itching: Yes/No;
- Irritability: Yes/No;
- Delayed Healing: Yes/No;
- Partial Paresis: Yes/No;
- Muscle Stiffness: Yes/No;
- Alopecia: Yes/No;
- Obesity: Yes/No;
- Class: Positive/Negative.

The average age of the participants was 48 years-old and the gender distribution was 328 male and 192 female (63%/37%). The other features can be found at Table 1.

The dataset is available at
https://archive.ics.uci.edu/ml/datasets/Early+stage+diabetes+risk+prediction+dataset.

Table 1. Diabetes Dataset Quantity per Feature

	Yes	No	Distribution (Yes%/No%)
Polyuria	258	262	(49.6%/50.4%)
Polydipsia	233	287	(44.8%/55.2%)
Sudden Weight Loss	217	303	(41.7%/58.3%)
Weakness	305	215	(58.7%/41.3%)
Polyphagia	237	283	(45.6%/54.4%)
Genital Thrush	116	404	(22.3%/77.7%)
Visual Blurring	233	287	(44.8%/55.2%)
Itching	253	267	(48.7%/51.3%)
Irritability	126	394	(24.2%/75.8%)
Delayed Healing	239	281	(46.0%/54.0%)
Partial Paresis	224	296	(43.1%/56.9%)
Muscle Stiffness	195	325	(37.5%/62.5%)
Alopecia	179	341	(34.4%/65.6%)
Obesity	88	432	(16.9%/83.1%)
Class	320	200	(61.5%/38.5%)

1.2. Seoul Bike Sharing Demand

This dataset consists of 14 features and 8760 non-null instances. Each instance has weather data and the amount of bikes rented at any given hour for the year of 2019 in the city of Seoul, South Korea. It also presents which season, if it was a holiday and if it was a functioning day for each instance.

- Date: year-month-day;
- Rented Bike Count: Number of bikes rented at each hour;
- Hour: Hour of the day;
- Temperature: Temperature in Celsius;
- Humidity: Humidity in %;
- Wind Speed: Wind speed in m/s;
- Visibility: Visibility in 10m;
- Dew Point Temperature: Dew Point Temperature in Celsius;
- Solar Radiation: Solar Radiation in MJ/m^2;
- Rainfall: Rainfall in mm;
- Snowfall: Snowfall in cm;
- Seasons: Winter/Spring/Summer/Autumn;
- Holiday: Holiday/No holiday;
- Functioning Day - Non Functioning Day/Functioning Day;

At Table 2 we can visualise the average, minimal and maximal values for some features.

The dataset is available at
https://archive.ics.uci.edu/ml/datasets/Seoul+Bike+Sharing+Demand

Table 2. Bikes Datase Average, Minimal, and Maximal Features Values

	Mean	Min	Max
Rented_Bike_Count	704.7	0	3556
Temperature(°C)	12.9	-17.8	39.4
Humidity(%)	58.2	0	98
Wind_Speed(m/s)	1.7	0	7.4
Visibility(10m)	1436.8	27	2000
Dew_Point_Temperature(°C)	4.1	-30.6	27.2
Solar_Radiation(MJ/m2)	0.6	0	3.5
Rainfall(mm)	0.1	0	35
Snowfall(cm)	0.1	0	8.8

2. Classification

2.1. Cleaning and Preprocessing the Data

For the classification task, the Diabetes dataset was used with the SciKit Learn library [Pedregosa et al. 2011]. On a first analysis, we checked that the dataset doesn't have any null values, so no treatment was needed. Since the data is mostly comprised of yes or no answers, we binarized the data using the Ordinal Encoder. To run the algorithms, the data was split into the Class column and the rest, and with this, the *cross_validate* and *cross_val_predict* methods were used for the prediction. These methods guarantee a more homogeneous result.

2.2. Estimators

- Linear SVC: The first method chosen was the Linear Support Vector Classification. This method is recommended by the [SkLearn] library. In this method, we used two verions, the default one, without changing any paramethers, and a second version changing the random state to 42 (default is 0).
- Perceptron: The second method is the Perceptron, which is the simplest neural network, with only one neuron. Two versions of this method were used, the default one, and a version withe the elasticnet penalty in place (default is none).
- Logistic Regression: The third method is a Logistic Regression. According to [Swaminathan], this method is the most common for binary classification. In this method, we used the default version as well as a version with the value of C changed to 1.5 (default is 1.0).
- Random Forest Classifier: The last method is the Random Forest Classifier. In this method, we used the default version with 50 trees and another version with 200 trees.

2.3. Results

As seen from Table 3, the Random Forest Classifier had a slightly better result with the Logistic Regression close as a second candidate. In the future, other algorithms could be used to evaluate better the data, as well as a Stacking Classifier. We plotted some bar plots to better visualise the diference, and they can be seen in Figures 1 and 2. We can also visualise the Confusion matrices for each algorithms in Figure 3, Figure 4, Figure 5, and Figure 6.

Table 3. Results for each model

Model	Accuracy	Precision	Recall	F1
LinearSVC Std	0.884615	0.883914	0.943750	0.910210
LinearSVC RS42	0.867308	0.908011	0.887500	0.890781
Perceptron Std	0.807692	0.838625	0.906250	0.859223
Perceptron Penalty	0.594231	0.711648	0.471875	0.505129
Log. Regression Std	0.926923	0.951011	0.931250	0.940094
Log. Regression C	0.926923	0.951011	0.931250	0.940094
Rand. Forest Class. Std	0.971154	0.976103	0.978125	0.976829
Rand. Forest Class. NE200	0.975000	0.976103	0.984375	0.980054

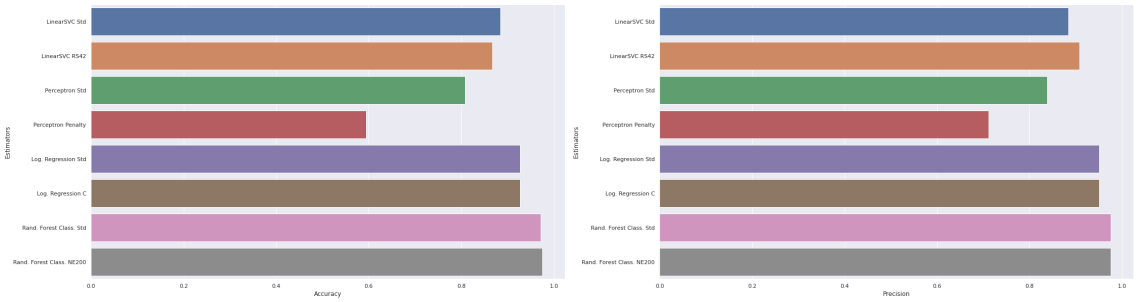

Figure 1. Accuracy and Precision Scores

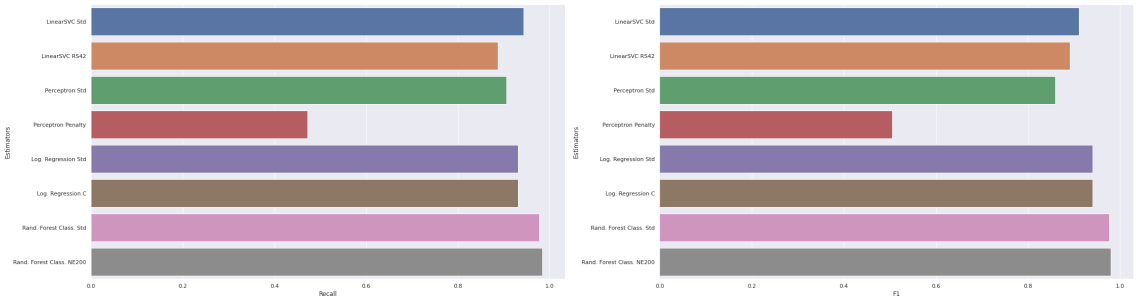

Figure 2. Recall and F1 Scores

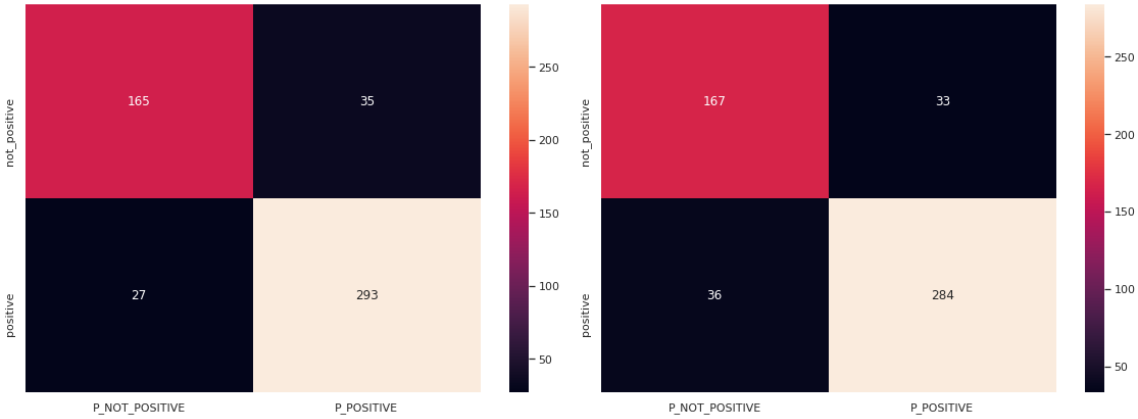

Figure 3. LinearSVC Standard and LinearSVC with RS=42 Confusion Matrices

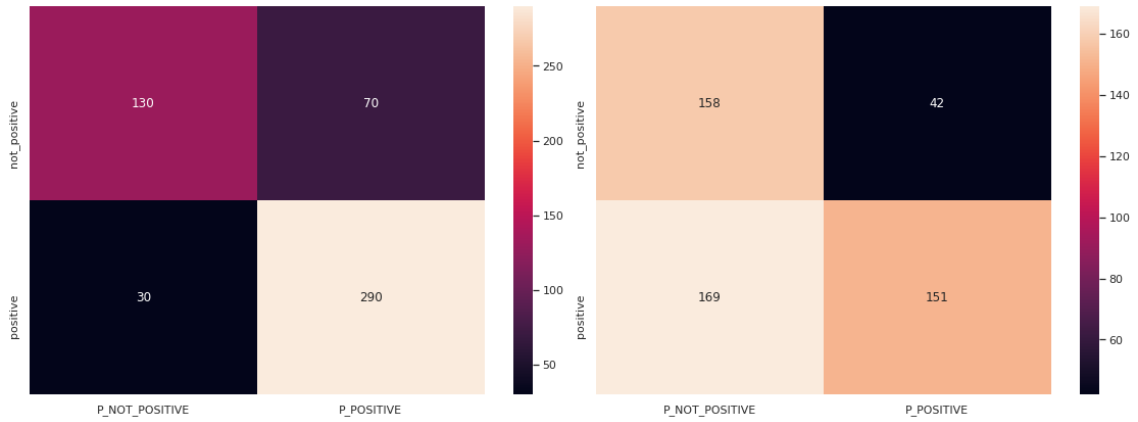

Figure 4. Standard Perceptron and Perceptron with Penalty Confusion Matrices

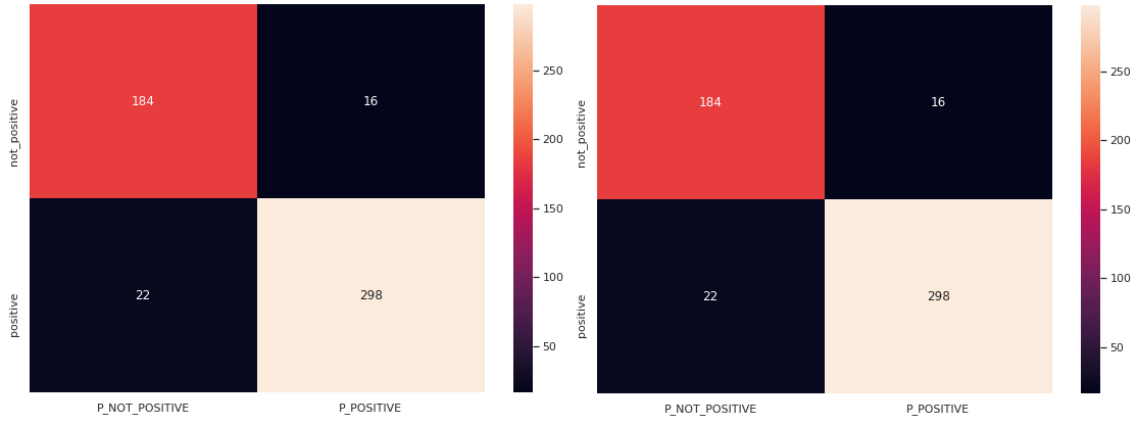

Figure 5. Standard Logistic Regression and Logistic Regression with C=1.5 Confusion Matrices

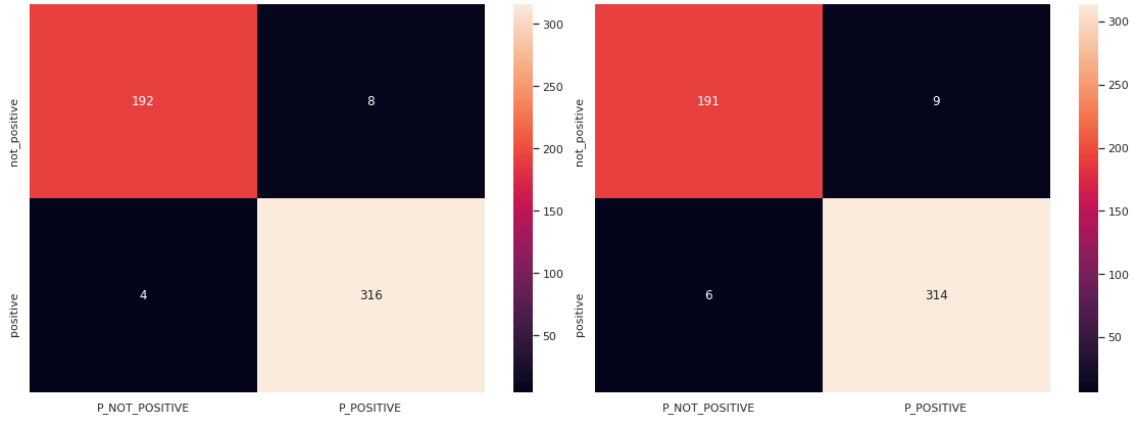

Figure 6. Standard Random Forest and Random Forest With 200 Trees Confusion Matrices

2.4. Auto-ML

An Auto Machine Learning library was used to evaluate the dataset. The library chosen was the Auto SkLearn and can be found at https://pypi.org/project/automl/. The *AutoSklearnClassifier* method was used with two parameters: *time_left_for_this_task* and *per_run_time_limit*. We chose 60 seconds for each task and 600 seconds for all the tasks. The method gave the following results:
Dataset name: c4308004-bdd1-11eb-80aa-00155d986d27
Metric: accuracy
Best validation score: 1.000000
Number of target algorithm runs: 184
Number of successful target algorithm runs: 178
Number of crashed target algorithm runs: 0
Number of target algorithms that exceeded the time limit: 6
Number of target algorithms that exceeded the memory limit: 0
Accuracy: 0.981
Unfortunately, this library doesn't return which algorithm had the best result.

3. Association

3.1. Cleaning and Preprocessing the Data

For the association task, the Diabetes Dataset was also used. In this case for better interpretation, the data was changed from binary to textual. The text is compromised of "<name of the symptom>" if True or "No <name of the symptom>" if False.

3.2. Results

To run the apriori algorithm, the Efficient Apriori library was used. To run the method, we chose the minimal support value to be 0.5 and the minimal confidence to be 0.6. After running the algorithm, we had 26 associations, from which 20 had one value on each side, 3 had one value on the left and two on the right, and 3 had two values on the left and one on the right. The lift values varied from 1.002 to 1.088. The associations are presented as follow:

- One to one associations:
  - {No Obesity} -> {Male} (conf: 0.632, supp: 0.525, lift: 1.002)
  - {Male} -> {No Obesity} (conf: 0.832, supp: 0.525, lift: 1.002)
  - {No Obesity} -> {No Alopecia} (conf: 0.662, supp: 0.550, lift: 1.010)
  - {No Alopecia} ->{No Obesity} (conf: 0.839, supp: 0.550, lift: 1.010)
  - {No Obesity} -> {No Genital Thrush} (conf: 0.787, supp: 0.654, lift: 1.013)
  - {No Genital Thrush} -> {No Obesity} (conf: 0.842, supp: 0.654, lift: 1.013)
  - {No Irritability} -> {No Alopecia} (conf: 0.668, supp: 0.506, lift: 1.018)
  - {No Alopecia} -> {No Irritability} (conf: 0.771, supp: 0.506, lift: 1.018
  - {No Obesity} -> {No Irritability} (conf: 0.782, supp: 0.650, lift: 1.033)
  - {No Irritability} -> {No Obesity} (conf: 0.858, supp: 0.650, lift: 1.033)

  - {No Irritability} -> {No Genital Thrush} (conf: 0.815, supp: 0.617, lift: 1.049)
  - {No Genital Thrush} -> {No Irritability} (conf: 0.795, supp: 0.617, lift: 1.049)
  - {No Obesity} -> {No Muscle Stiffness} (conf: 0.660, supp: 0.548, lift: 1.056)
  - {No Muscle Stiffness} -> {No Obesity} (conf: 0.877, supp: 0.548, lift: 1.056)
  - {No Sudden Weight Loss} -> {No Obesity} (conf: 0.884, supp: 0.515, lift: 1.065)
  - {No Obesity} -> {No Sudden Weight Loss} (conf: 0.620, supp: 0.515, lift: 1.065)
  - {No Genital Thrush} -> {No Alopecia} (conf: 0.708, supp: 0.550, lift: 1.080)
  - {No Alopecia} -> {No Genital Thrush} (conf: 0.839, supp: 0.550, lift: 1.080)
  - {No Muscle Stiffness} -> {No Irritability} (conf: 0.825, supp: 0.515, lift: 1.088)
  - {No Irritability} -> {No Muscle Stiffness} (conf: 0.680, supp: 0.515, lift: 1.088)
- One to two associations:
  - {No Obesity} -> {No Genital Thrush, No Irritability} (conf: 0.639, supp: 0.531, lift: 1.035)
  - {No Genital Thrush} -> {No Irritability, No Obesity} (conf: 0.683, supp: 0.531, lift: 1.051)
  - {No Irritability} -> {No Genital Thrush, No Obesity} (conf: 0.701, supp: 0.531, lift: 1.071)
- Two to one associations:
  - {No Genital Thrush, No Irritability} -> {No Obesity} (conf: 0.860, supp: 0.531, lift: 1.035)
  - {No Irritability, No Obesity} -> {No Genital Thrush} (conf: 0.817, supp: 0.531, lift: 1.051)
  - {No Genital Thrush, No Obesity} -> {No Irritability} (conf: 0.812, supp: 0.531, lift: 1.071)

Lowering the support value to 0.4 increased the number of associations to 145 (89 one to one, 17 one to two, and 39 two to one), with a lift between 0.909 and 1.535.

4. Clustering

4.1. Cleaning and Preprocessing the Data

For the clustering task, we used the Seoul Bikes dataset. This dataset was created for a regression task, but we thought it would be interesting to try the clustering methods. On a first thought, given that the majority of the data is weather related, we imagined that the methods would return clusters close to the seasons.
On a first analysis, the Date, Hour, Seasons, Holiday and Functioning Day columns were removed. These columns represent the time of the year for each line, and they might've

acted as classes. We ended up with 9 columns, 8 representing weather data, and 1 representing the amount of bicycles rented.
We then plotted the data into histograms, presented at Figure 7. We can see that throughout the year, there is high visibility, low solar radiation, and little to no rain and snow.
We also analysed if there was correlations between the data. We used three methods, Pearson, Kendall, and Spearman, all available within the Pandas library. This can be seen in Figure 8. Little to no correlation was seen, except for a high correlation between Temperature and Dew Point Temperature, but we decided to keep both columns.

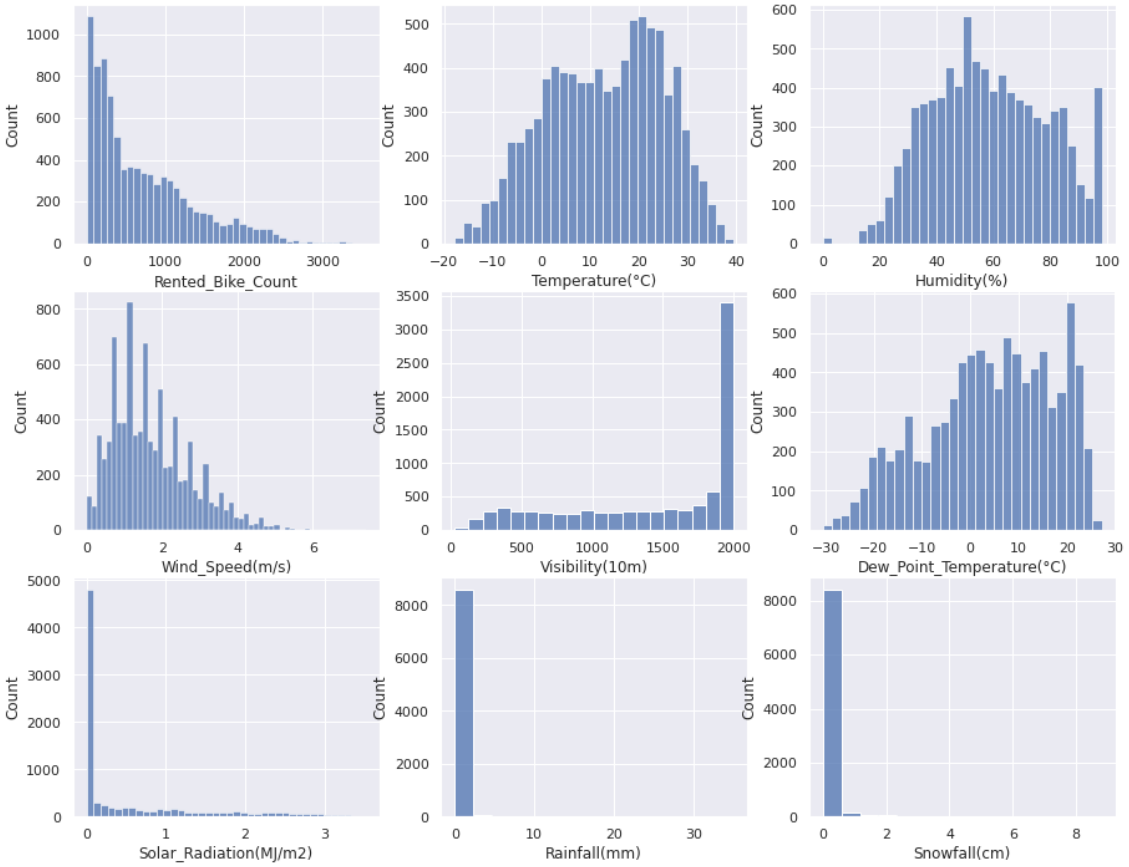

Figure 7. Initial Analysis with Histograms

Given that the data had different scales for each column, a MinMax Scaler was used to normalise them.

4.2. K-Means

For the K-Means algorithm, we have to determine the number of clusters. To help with that, we used the Elbow Method, which generated the graphic in Figure 9. We can see that the slope of the curve starts to stabilise after the number 4, which will be the number of clusters used.

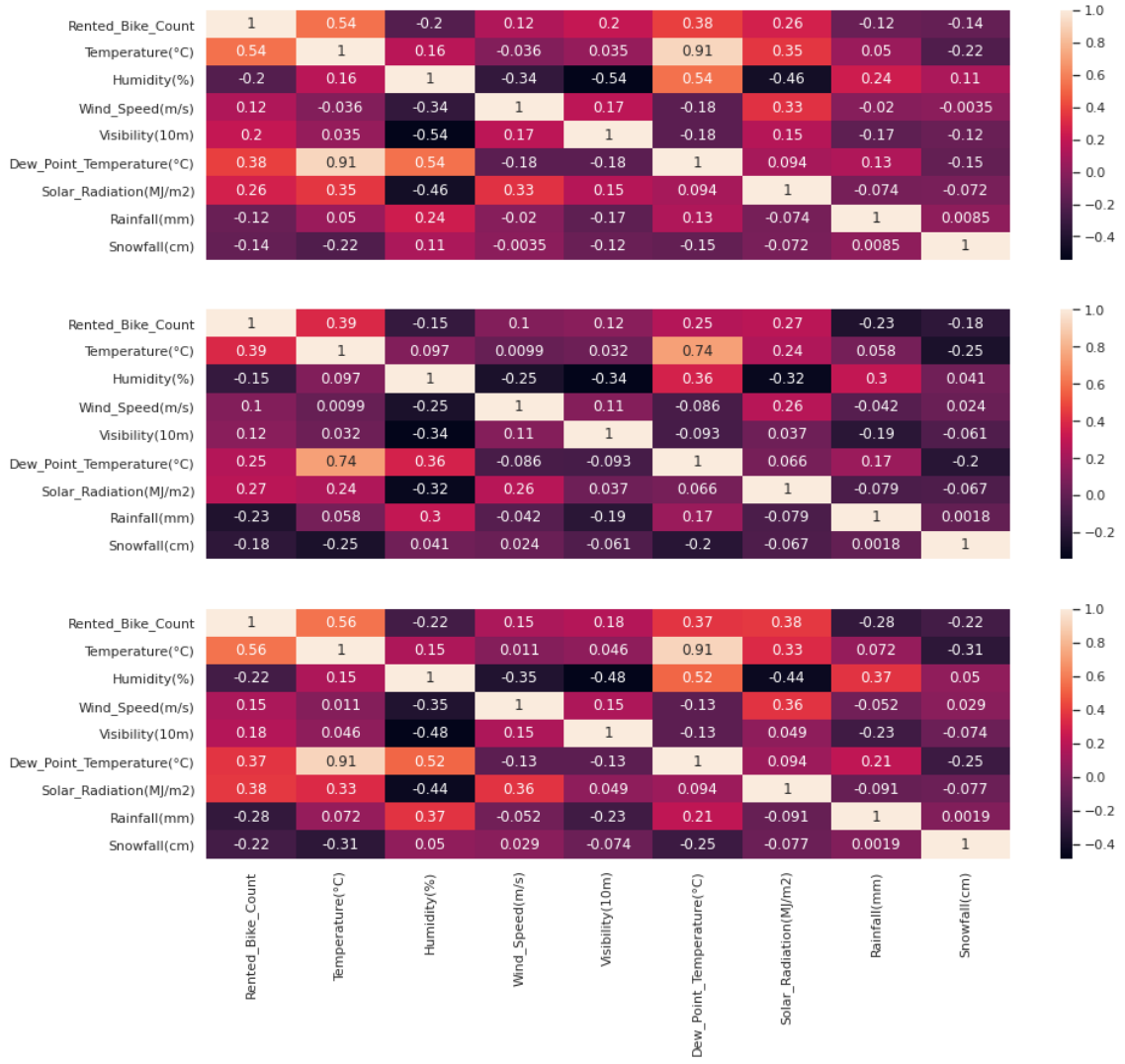

Figure 8. Correlations Between Columns

Figure 9. The Elbow Method

Given that the data is multidimensional, we cannot plot a scatterplot with the clusters, but the centroid values are presented as follow:

- Cluster 0: [3.16699463e-01, 7.19016088e-01, 4.24143492e-01, 3.10189399e-01, 8.09207987e-01, 6.85085250e-01, 6.12990535e-01, 3.70096225e-06, 1.90819582e-17, 3.00000000e+00]
- Cluster 1: [1.23499177e-01, 5.07629317e-01, 7.85822178e-01, 1.88809997e-01, 2.98048667e-01, 6.52232953e-01, 4.99140449e-02, 1.30657062e-02, 1.86005028e-02, 1.00000000e+00]
- Cluster 2: [9.16257461e-02, 2.88631364e-01, 4.24990968e-01, 2.66221737e-01, 8.96434365e-01, 3.01609790e-01, 9.22721599e-02, 4.15018481e-05, 1.31587092e-02, -4.21884749e-15]
- Cluster 3: [2.90307462e-01, 6.67644537e-01, 6.58389090e-01, 2.00516615e-01, 9.01440302e-01, 7.55494926e-01, 5.63408023e-02, 1.95400369e-03, -6.24500451e-17, 2.00000000e+00]

Table 4 presents the mean value for each column in each cluster.

Table 4. Mean value for each cluster

Cluster	0	1	2	3
Dew_Point_Temperature(°C)	8.997927	7.099065	-13.166954	13.067607
Humidity(%)	41.566062	77.010573	41.649115	64.522131
Rainfall(mm)	0.000130	0.457300	0.001453	0.068390
Rented_Bike_Count	1126.183290	439.163074	325.821153	1032.333333
Snowfall(cm)	0.000000	0.163684	0.115797	0.000000
Solar_Radiation(MJ/m2)	2.157727	0.175697	0.324798	0.198320
Temperature(°C)	23.327720	11.236397	-1.290286	20.389268
Visibility(10m)	1623.567358	615.050020	1795.665002	1805.541716
Wind_Speed(m/s)	2.295402	1.397194	1.970041	1.483823

Given that the dataset had 8760 values, we expected to have roughly 2190 per cluster. Figure 10 presents the quantity of values per cluster.

Figure 10. K-Means Clustering

Another possible representation of the data is using a Pair Plot. The Pair Plot uses a scatter plot for each pair of features and represents the clusters using colours. The upper triangle and the lower triangle represents the same data and the diagonal plots are distribution plots, showing the marginal distribution of the data. The generated graph can be seen in Figure 11.

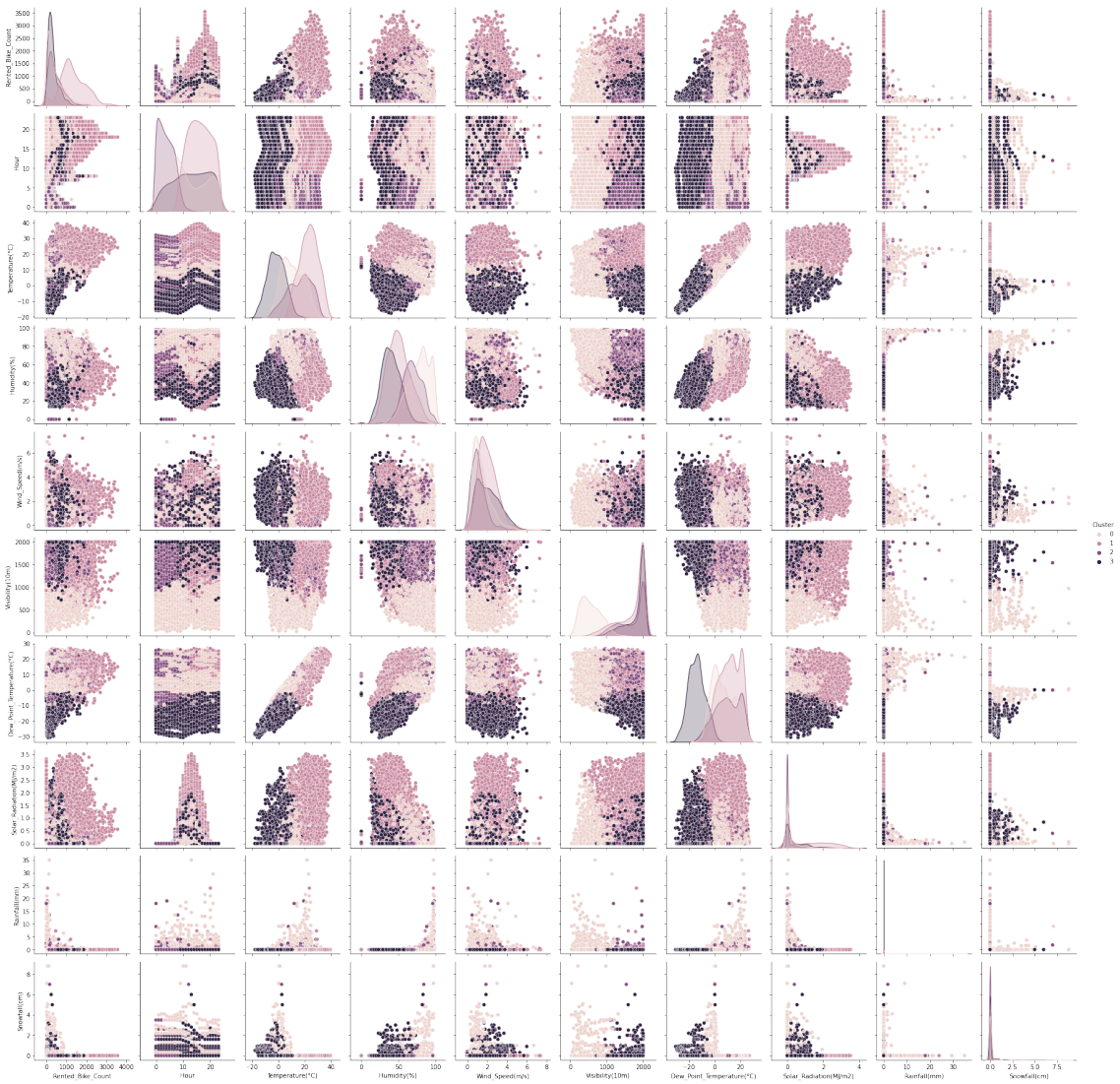

Figure 11. Pair Plots With Soul Bike Data

4.3. DBSCAN

With the DBSCAN method, we don't determine the quantity of clusters. Instead, we gave the algorithm the maximum distance between two samples and left the number of samples needed for a neighbourhood to be considered a cluster with the default value of 5. After experimenting with multiple values, we noticed that values above 1 created one single cluster with all the points and no noise. Values below 0.2, had more noise than clustered points. With values in between, we always had one cluster with more than 80% of the points and the other 20% distributed around the rest of the clusters. This was not surprising, given that the data was not meant for a clustering task.

With the Figure 12, we ended up with a cluster of 8737 points. Noise, cluster 1 and cluster 2 had 9, 7, and 6 points respectively.

Figure 12. DBSCAN with esp=0.6

5. Conclusions and Future Work

In this work, we used three methods to evaluate data. The first one, classification, we used four algorithms, LinearSVC, Perceptron, Logistic Regression, and Random Forest Classifier, to try and classify a database about early signs of diabetes. The Random Forest Classifier with 200 trees had the best results, with an accuracy of 0.975000, precision of 0.976103, recall of 0.984375, and F1 of 0.980054. The worst one was the Perceptron with elasticnet penalty, reaching an accuracy of0.594231, precision of 0.711648, recall of 0.471875, and F1 of 0.505129. In the future, running a Random Forest with more trees, might reach even better results.

For the association task, we ended up with a set of rules that mirror each other. When we had A then B, we also had B then A, with the same lift and support values but with different confidences. The rules also only presented false values, as in "No ¡symptom¿". This is interesting but medical knowledge would be needed to understand the reason, and from being a health related issue, no speculations will be made.

The clustering task presented itself as the hardest task. We had a multidimensional dataset and didn't want to get rid of any feature, especially since only two had a high correlation. This created four nine-dimensional arrays with the centroids, which precluded the creation of a scatter plot. The DBSCAM algorithm showed itself to not be able to cluster the data in a efficient way, which was not a surprise, since the data was not created with a clustering task in mind. The K-Means algorithm was able to create 4 clusters with slightly different values. In the future, finding better ways to interpret the data would be interesting, specially with more than two dimensions.

References

Islam, M. M. F., Ferdousi, R., Rahman, S., and Bushra, H. Y. (2020). Likelihood prediction of diabetes at early stage using data mining techniques. In Gupta, M., Konar, D.,

Bhattacharyya, S., and Biswas, S., editors, *Computer Vision and Machine Intelligence in Medical Image Analysis*, pages 113–125, Singapore. Springer Singapore.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

SkLearn. Choosing the right estimator.

Swaminathan, S. Logistic regression - detailed overview.