Data and Information Quality

Politecnico di Milano

Project Report

PROJECT ID: 46

PROJECT NUMBER: 2

ASSIGNED DATASETS: cancer, house

STUDENTS: Valeria Panté (10712755), Luca Tombesi (10865393)

ASSIGNED TASK: Classification

Introduction

The assigned task is the analysis of the effect of outlier detection on different classification techniques. In particular, the original dataset was injected with different percentages of outliers (10%, 20%, 30%, 40%, 50%) and the performances were analyzed on each of them independently, before and after using outlier detection. Given that, for each assigned dataset, different techniques were used based on their diversity. This paper will analyze the steps applied to each of them separately.

Cancer Dataset

- SETUP CHOICES
    - As ML techniques for the classification task, we opted for Random Forest with 10 base classifiers and SVM. We decided to choose the first one because it is an ensemble method that allows us to improve the performance by using multiple base learners. The second one was chosen because it is effective even in high dimensional spaces.
    - Given that the assigned task was classification, the metrics chosen to evaluate the performances are Accuracy, Precision, Recall and F1-score. All of these metrics were chosen to have a complete overview of how the model performed. Furthermore we plotted the confusion matrix to see where the models were mistaking more.
    - To detect the outliers, we tried several attempts and the best performing methods proved to be Kernel Density as basic method and Isolation forest (Ensemble-based methods) as advanced one.

- PIPELINE IMPLEMENTATION

    First of all we performed data profiling to have a better understanding of the data at hand. From this step we noticed that data contained 9 numerical

features, so we decided to not perform any dimensionality reduction or feature selection given that the number of features was not too big, in fact any attempt of doing that reduced the performance of the outliers detectors. For the same reason we didn't standardize the data, also because the range of data was quite small (all the features values were between 0 and 10).

After these initial steps, we performed the initial classification with the techniques mentioned above. Then, we moved to outlier detection. Using the Kernel Density method, we needed to choose adequate hyperparameters. First, for the bandwidth we found out that the best value was 2,1. Secondly, we chose as a threshold to consider a data point as an outlier the 2% quantile. As an output, we had the following number of outliers:

- 50% injection: 324 (47,4%)
- 40% injection: 298 (43,6%)
- 30% injection: 277 (40,6%)
- 20% injection: 178 (26,1%)
- 10% injection: 66 (9,6%)

For the second technique used, Isolation Forest, we needed to tune the contamination parameters, which was set to the percentage of injection. The resulting outliers were the following:

- 50% injection: 341 (49,9%)
- 40% injection: 273 (39,9%)
- 30% injection: 205 (30,0%)
- 20% injection: 137 (20,1%)
- 10% injection: 69 (10,1%)

As it can be noticed from the previous data, the advanced technique performed better than the basic one.

Finally, we removed the outliers and we fitted the models again.

- RESULTS
  - The results obtained by the models on the original injected data are the following:

Injection	Model	Accuracy	Precision	Recall	F1-score
50%	Random Forest	0.8978	0.8617	0.886	0.8725
	SVM	0.6934	0.4661	0.4882	0.4514
40%	Random Forest	0.927	0.927	0.9059	0.9153

	SVM	0.6788	0.8382	0.5111	0.4252
30%	Random Forest	0.9051	0.8876	0.8669	0.8764
	SVM	0.708	0.5509	0.5191	0.4948
20%	Random Forest	0.9635	0.9591	0.9522	0.9555
	SVM	0.781	0.7593	0.6691	0.6873
10%	Random Forest	0.9343	0.917	0.9327	0.9242
	SVM	0.8102	0.7851	0.7503	0.7633

- The performances obtained by the models on the cleaned data with Kernel Density detector are the following:

Injection	Model	Accuracy	Precision	Recall	F1-score
50%	Random Forest	0.8889	0.8768	0.8148	0.8393
	SVM	0.7222	0.3714	0.4815	0.4194
40%	Random Forest	0.9024	0.8576	0.8576	0.8576
	SVM	0.7805	0.3902	0.5	0.4384
30%	Random Forest	0.9286	0.9545	0.875	0.9048
	SVM	0.7143	0.3571	0.5	0.4167
20%	Random Forest	0.8727	0.9054	0.86	0.8664
	SVM	0.6545	0.689	0.63	0.6131
10%	Random Forest	0.9344	0.9004	0.9341	0.9153
	SVM	0.8033	0.7348	0.7123	0.7219

- The performances obtained by the models on the cleaned data with Isolation Forest detector are the following:

Injection	Model	Accuracy	Precision	Recall	F1-score
50%	Random Forest	0.9714	0.9821	0.9375	0.9576
	SVM	0.7714	0.3857	0.5	0.4355

40%	Random Forest	0.9512	0.9024	0.9024	0.9024
	SVM	0.8049	0.4231	0.4714	0.4459
30%	Random Forest	0.9583	0.9531	0.9531	0.9531
	SVM	0.6667	0.3333	0.5	0.4
20%	Random Forest	0.9636	0.9542	0.9542	0.9542
	SVM	0.8545	0.8684	0.7542	0.7868
10%	Random Forest	0.9516	0.9375	0.9634	0.9477
	SVM	0.9194	0.9268	0.8926	0.9065

    - As a general note, Random Forest always outperformed SVM. After cleaning the outliers, we noticed that, in the case of the Kernel Density detector, for low percentages of outliers (10%, 20%, 30%) SVM improved, while for the high ones it worsened. Instead, Random Forest does not show a similar pattern, actually sometimes it improves, sometimes it does not. Regarding Isolation Forest detector, it always improves the performances, both with respect to the initial dataset and the one cleaned through Kernel Density outlier detection.

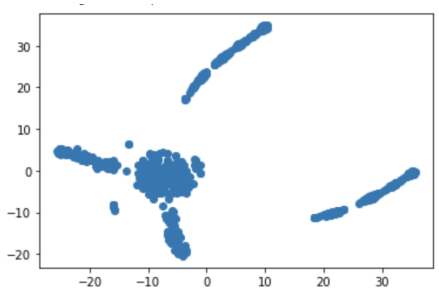
House Dataset

- SETUP CHOICES
    - As ML techniques for the classification task, we opted for Random Forest with 10 base classifiers and SVM. We decided to choose the first one because it is an ensemble method that allows us to improve the performance by using multiple base learners. The second one was chosen because it is effective even in high dimensional spaces.
    - Given that the assigned task was classification, the metrics chosen to evaluate the performances are Accuracy, Precision, Recall and F1-score. All of these metrics were chosen to have a complete overview of how the model performed. Furthermore we plotted the confusion matrix to see where the models were mistaking more.
    - To detect the outliers, we tried several attempts and the best performing methods proved to be Robust Z-Score as basic method and Hierarchical Clustering (Clustering-based methods) with single linkage as advanced one.
- PIPELINE IMPLEMENTATION

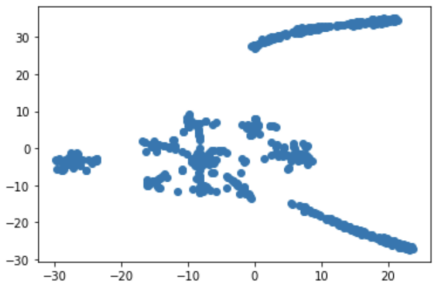
First of all we performed data profiling to have a better understanding of the data at hand. From this step we noticed that data contained 74 features, both numerical and categorical, so we decided to perform dimensionality reduction.

Before doing so we used Label Encoder to convert categorical features to numerical ones. In particular, this has been chosen because we didn't want to further increase the number of features with One Hot Encoding. After this step, data has been normalized using Robust Scaler, especially to avoid sensitivity to outliers. To reduce the dimensionality, we first used PCA (Principal Component Analysis) to reduce the number of feature to 50 but keeping at the same time the cumulative variance greater than 90%, to then apply t-SNE (t-Stochastic Neighbour Embeddings) to reduce the final features to 2, keeping all the information. By plotting the reduced data, we found that outliers were clearly separated from the original data.

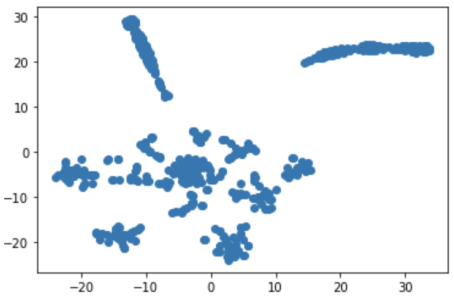
- Reduced 50% injected data



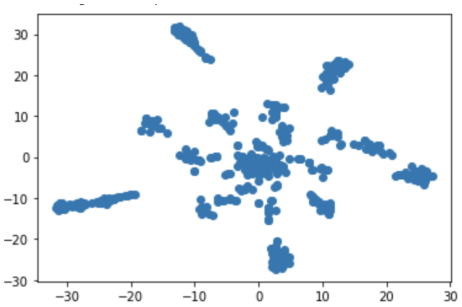
- Reduced 40% injected data



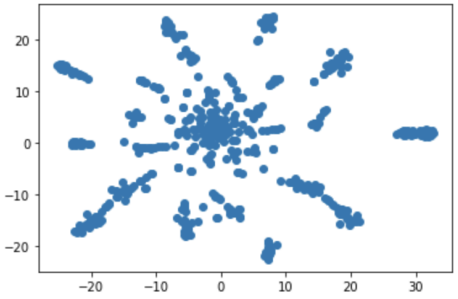
- Reduced 30% injected data



- Reduced 20% injected data



- Reduced 10% injected data

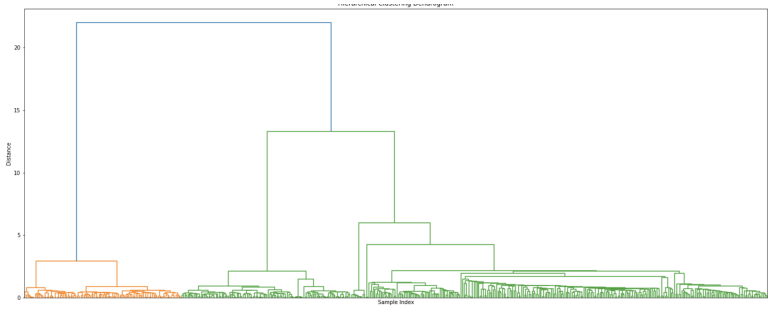


After these initial steps, we performed the initial classification with the techniques mentioned above. Then, we moved to outlier detection. Using the Z-score method, we needed to choose an adequate threshold, which was set to 2.9 after several attempts. As an output, we had the following number of outliers:

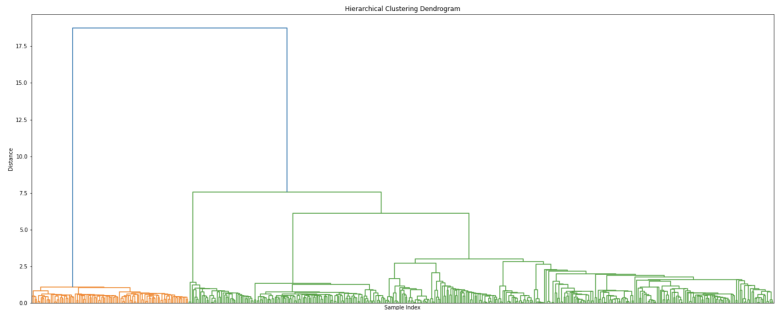
- 50% injection: 261 (52,1%)
- 40% injection: 147 (29,3%)
- 30% injection: 131 (26,1%)
- 20% injection: 64 (12,7%)
- 10% injection: 27 (5,4%)

For the second technique used, Hierarchical Clustering, we needed to choose the number of clusters to consider, which ended up to be 3, after analyzing the dendograms.

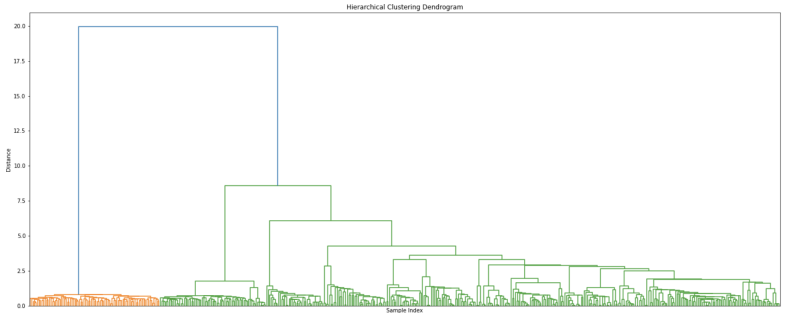
- 50% injected data



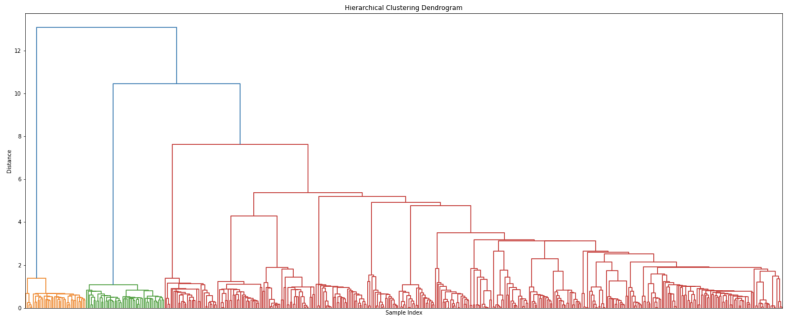
- 40% injected data



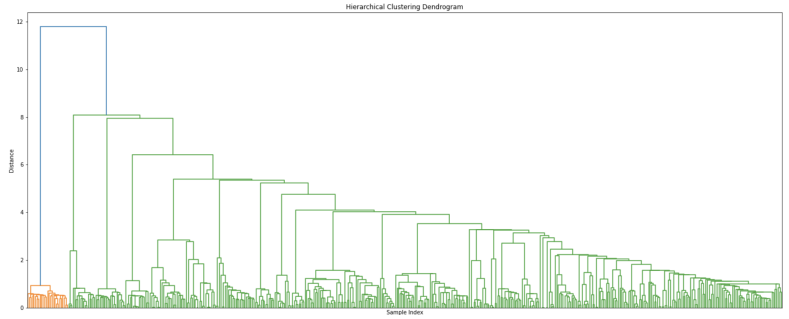
- 30% injected data



- 20% injected data



- 10% injected data



The outliers were identified as all the points inside the clusters with the longest distance of merging. The resulting outliers were the following:

- 50% injection: 220 (43,9%)
- 40% injection: 150 (29,9%)
- 30% injection: 158 (31,5%)
- 20% injection: 92 (18,4%)
- 10% injection: 45 (9,0%)

As it can be noticed from the previous data, the advanced technique performed better than the basic one.

Finally, we removed the outliers and we fitted the models again.

- RESULTS
  - The results obtained by the models on the original injected data are the following:

Injection	Model	Accuracy	Precision	Recall	F1-score
50%	Random Forest	0.8812	0.375	0.3429	0.3533
	SVM	0.8317	0.1663	0.2	0.1816
40%	Random Forest	0.8614	0.3088	0.2778	0.2867
	SVM	0.802	0.1337	0.1667	0.1484
30%	Random Forest	0.8317	0.4566	0.3571	0.3763
	SVM	0.802	0.2005	0.25	0.2225
20%	Random Forest	0.8812	0.4452	0.3905	0.3976
	SVM	0.8416	0.1683	0.2	0.1828
10%	Random Forest	0.8911	0.3294	0.3477	0.3381
	SVM	0.8713	0.1743	0.2	0.1862

  - The performances obtained by the models on the cleaned data with Robust Z-score detector are the following:

Injection	Model	Accuracy	Precision	Recall	F1-score
50%	Random Forest	0.8333	0.2899	0.3175	0.303
	SVM	0.875	0.2917	0.3333	0.3111
40%	Random Forest	0.8333	0.303	0.303	0.303
	SVM	0.9167	0.3056	0.3333	0.3188
30%	Random Forest	0.7568	0.2543	0.2595	0.2514
	SVM	0.7568	0.1514	0.2	0.1723
20%	Random Forest	0.8182	0.361	0.3	0.3117

	SVM	0.75	0.15	0.2	0.1714
10%	Random Forest	0.8333	0.303	0.2778	0.2833
	SVM	0.75	0.125	0.1667	0.1429

- The performances obtained by the models on the cleaned data with Hierarchical Clustering detector are the following:

Injection	Model	Accuracy	Precision	Recall	F1-score
50%	Random Forest	0.8621	0.2155	0.25	0.2315
	SVM	0.8621	0.2155	0.25	0.2315
40%	Random Forest	0.8333	0.3647	0.3333	0.3406
	SVM	0.7778	0.1556	0.2	0.175
30%	Random Forest	0.8	0.2121	0.2333	0.2222
	SVM	0.8571	0.2143	0.25	0.2308
20%	Random Forest	0.7561	0.155	0.2	0.1746
	SVM	0.7561	0.1512	0.2	0.1722
10%	Random Forest	0.7391	0.3524	0.2549	0.2684
	SVM	0.7391	0.1478	0.2	0.17

- As a general note, Random Forest always outperformed SVM in the original data. After cleaning the outliers, we noticed that for all the injections with both outlier detection techniques, SVM and Random Forest reached similar performances. The performance data do not show a technique being better than the other, in fact some metrics increase and other decrease with respect to the same context. In some cases, even the unclean injected data perform better than the one where outliers are removed.