Prepared for submission to University of Cambridge

M1: Applied Data Science - Coursework Assignment

Adnan Siddiquei

University of Cambridge

E-mail: as3438@cam.ac.uk

Contents

1 Section A

This section contains the answers to the questions in Section A of the coursework assignment. Wherever the phrase 'dataset' is used in this section, it refers to `A_NoiseAdded.csv`.

1.1 Q1 - Dataset A

1.1.1 Question 1a

Fig(1) shows the combined and separated kernal density estimates of the first 20 features of the dataset. Several conclusions can be drawn from this plot. For most of the features, the KDE is concentrated around the zero value with very little variance, with the exception of 7 features: 5, 11, 13, 14, 18, 19 and 20. These features are likely to be more discriminative within classification algorithms, as they contain more variability.

1.1.2 Question 1b

The biplot in Fig(2) shows a PCA of the entire dataset, after standardisation has been applied to each feature. The coloured arrows indicate the loadings of the features which contribute more than 2% to either principal component, which implies that none of the first 20 features are particularly discriminative. An interesting observation is that the most discriminative features are very disciminative with respect to the second principal component, whereas the less discriminative features (of which there are many, many more of) are more discriminative with respect to the first principal component. As a result, the scores on the biplot for the observations w.r.t. the first two principal components separate the observations more along the first principal component, rather than the second.

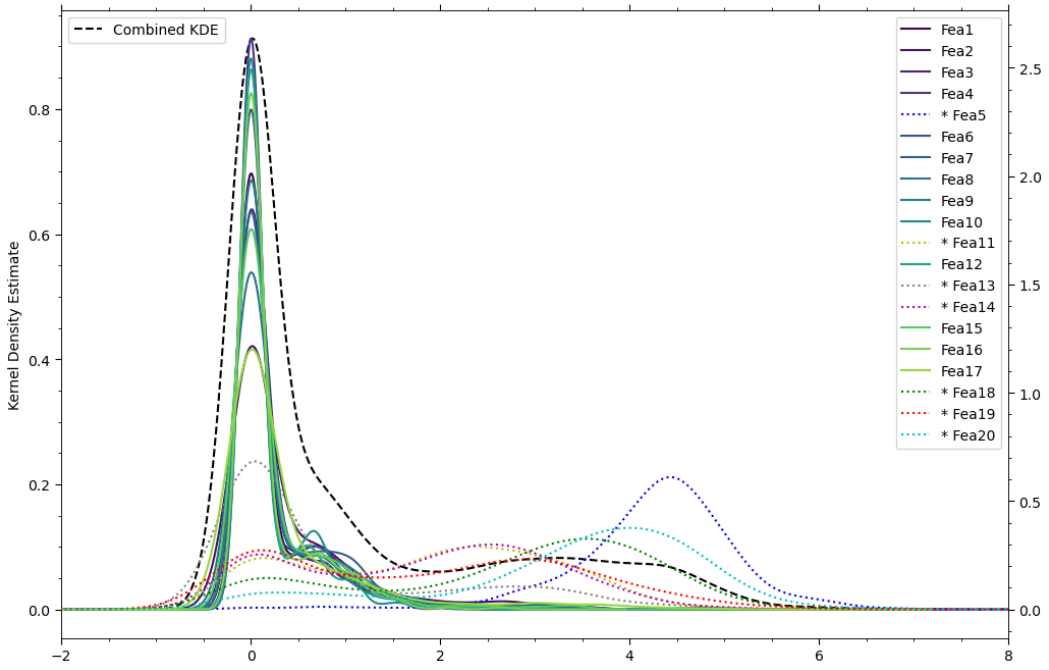

Figure 1: A kernel density estimate of the first 20 features of the `A_NoiseAdded.csv` dataset, which has 20 plots in total, with the legend and corresponding y-axis on the right. A combined KDE has also been plotted, with it's corresponding x-axis on the left. 7 of the 20 features have been highlighted in the legend with an asterisk, and have been coloured in slightly more contrasting colours and plotted with a dotted line. These features have a larger variance in their density, and are therefore more likely to be more discriminative.

1.1.3 Questions 1c and 1d

Fig(3a) shows the contingency table for two k-means clusterings of the dataset with $k = 8$ and $k = 3$. The two clusterings (`kmeans_1` and `kmeans_2`) were formed by training two k-means models on half of the dataset, and then the other half were mapped onto the learned clusters. Standardisation was applied to each feature before clustering as k-means is sensitive to feature scaling, this is because k-means works by computing a centroid for each cluster such that the sum of the squared Euclidean distances between each centroid and the observations in the cluster is minimised. Therefore, without standardisation, features with larger variances and scale will dominate the clustering. Predictions are then made by assigning new observations to the cluster whose centroid is closest to the observation [1]. Crucially, the labels assigned to each cluster from each k-means computation are arbitrary, therefore, the labels in `kmeans_2` were mapped back to the labels in `kmeans_1` by finding which centroids in each clustering were closest to each other. This step was crucial, otherwise the contingency table would be meaningless.

For $k = 8$, 59% of the observations lie on the leading diagonal, which indicates that the two clusterings were not very similar or stable, as a large proportion of the observations were assigned to different clusters in each clustering. `kmeans_1` clustered 85% of the observations into clusters 1, 2 and 3. `kmeans_2` clustered 86% of the observations into clusters 1 and 2. This indicates that both clusterings identified most of the data lives within a small set of clusters, which was expected as the dataset is labelled and so it is known that there are

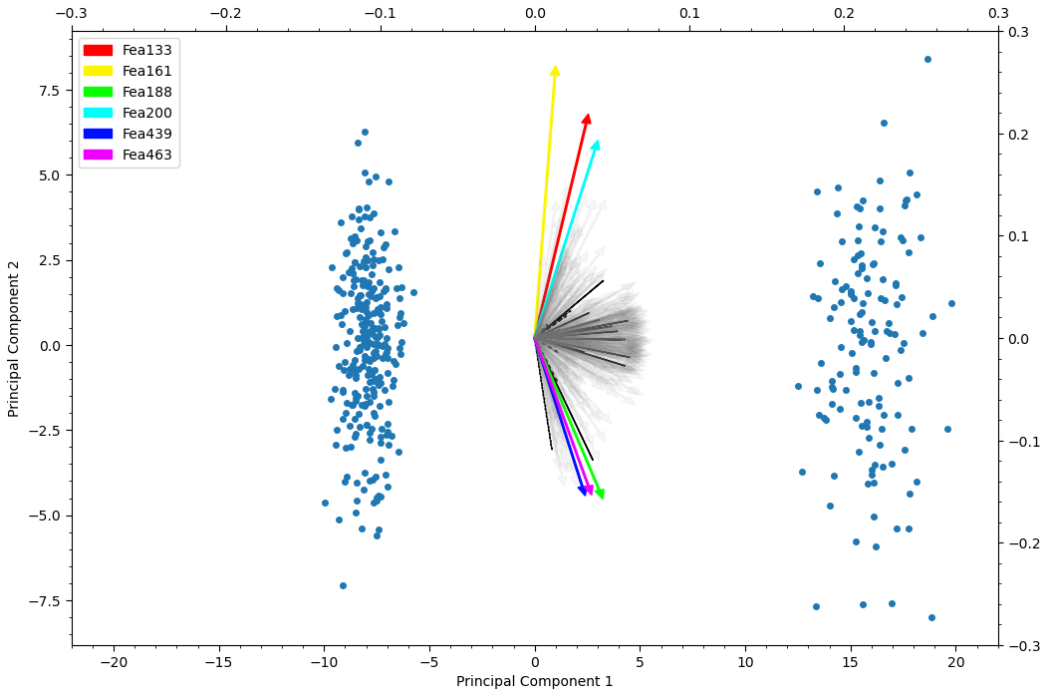

Figure 2: A biplot of the first two principal components in the `A_NoiseAdded.csv` dataset. The coloured arrows indicate the first two principal component loading vectors for every feature which contributes to either principal component more than 2%, these are the most discriminative features. Every other loading vector has been plotted in very light grey so their general directions and magnitudes are visible. The loading vectors for the first 20 features have also been plotted in darker black lines, with the dotted black lines corresponding to the dotted KDEs in Fig(1). The loading vectors use the right and top axis of the plot. The blue dots indicate the scores for each observation in the dataset for the first two principal components. The scores use the left and bottom axis of the plot. All four axis are symmetric about the origin for ease of comparison.

only 3 clusters - both clusterings identified at least 2 similar clusters (1 and 2). However, the presence of very small clusters indicates that there may be outliers present in the dataset or smaller clusters that the large $k = 8$ is overfitting to.

Fig(3b) shows the contingency table for two k-means clusterings of the dataset with $k = 3$. A larger proportion of the observations lie on the leading diagonal compared to $k = 8$, which is expected because the number of clusters is both smaller and equal to the actual number of clusters in the dataset. `kmeans_2` identified 3 distinct clusters whereas `kmeans_1` only identified 2 distinct clusters with a few remnant observations in assigned to cluster 2.

1.1.4 Questions 1e

Fig(4) shows the k-means clusterings on the PCA plot shown in Fig(2). The PCA indicates that there are two clusters in the dataset when it is reduced to the first two principal components (which generally explain a large amount of the variance in a dataset). Fig(4) provides subtle evidence in favour of this as well, note that clusters 1 and 3 are stable, but cluster

	Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 5	Cluster 6	Cluster 7	Cluster 8	Total
Cluster 1	**43**	0	9	8	0	4	14	1	**79**
Cluster 2	0	**185**	0	0	0	0	0	0	**185**
Cluster 3	0	87	**0**	0	0	0	0	0	**87**
Cluster 4	12	0	2	**5**	0	3	6	0	**28**
Cluster 5	1	0	0	0	**0**	0	0	0	**1**
Cluster 6	6	0	0	0	0	**0**	0	0	**6**
Cluster 7	12	0	0	0	1	1	**7**	0	**21**
Cluster 8	1	0	0	0	0	0	0	**0**	**1**
Total	**75**	**272**	**11**	**13**	**1**	**8**	**27**	**1**	

(a) A contingency table for two k-means clusterings of the `A_NoiseAdded.csv` dataset, with $k = 8$, the default `scikit-learn` value. 240 of the 408 observations lie on the leading diagonal.

	Cluster 1	Cluster 2	Cluster 3	Total
Cluster 1	**155**	117	0	**272**
Cluster 2	0	**0**	10	**10**
Cluster 3	0	0	**126**	**126**
Total	**155**	**117**	**136**	

(b) A contingency table for two k-means clusterings of the `A_NoiseAdded.csv` dataset, with $k = 3$. 281 of the 408 observations lie on the leading diagonal.

Figure 3: Contingency tables for two k-means clusterings of the `A_NoiseAdded.csv` dataset with number of clusters $k = 3$ and $k = 8$. Each feature in the dataset was standardised before clustering. `kmeans_1` totals are on the right and `kmeans_2` totals are on the bottom.

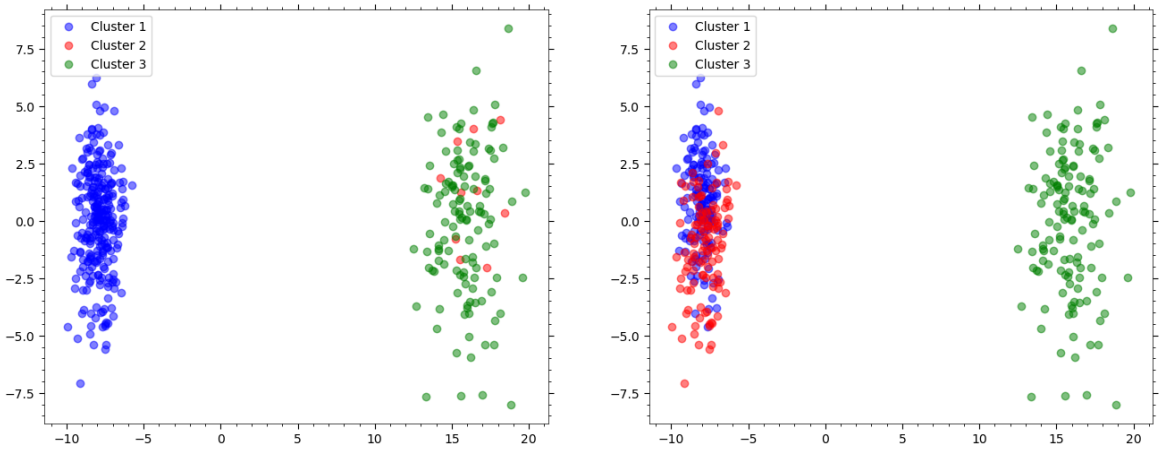

Figure 4: The k-means clusterings performed with $k = 3$, shown on the contingency table in Fig(3b), plotted on the first two principal components of the dataset shown in Fig(2). The left plot is `kmeans_1` and the right plot is `kmeans_2`.

2 jumps between the two groups in the PCA plots, whilst only capturing observations from either one of the groups but never both. The fact that cluster 2 never captures data from both groups in the PCA plot indicates that it is likely capturing a subset of the data from one of the PCA groups, based on random initialisation of the k-means algorithm. Its inability to capture data from both PCA groups at once, and the clear separation of clusters 1 and 3 indicate that there may only be two clusters in the dataset, as opposed to the three clusters

	Count
1	179
2	157
4	72
Missing	20
Total	**428**

(a) Raw `B_NoiseAdded.csv` dataset, before pre-processing. There are 428 observations with 20 missing labels and 20 duplicated observations (40 observations involved in the duplication).

	Count
1	177 (-3, 5, -12, 8)
2	163 (-3, 4, -5, 10)
4	68 (-4, 1, -3, 2)
Missing	0
Total	**408**

(b) `B_NoiseAdded.csv` after pre-processing, no missing labels or duplicated observations. The 4 numbers in the brackets indicate how the count in each classification changed due to, in order: 1 and 2 are counts exiting and entering (respectively) the class after correcting for mislabelling; 3 - count exiting after dropping duplicates; 4 - count entering after imputation of missing labels.

Figure 5: Summary of classifications for the `B_NoiseAdded.csv` dataset before and after pre-processing.

that the labels indicate.

Performing k-means before PCA has the advantage of being able to visualise the clusters separation in the original feature space, whilst performing PCA before k-means has the advantage of being able to reduce computational load and identify clusters along the first two principal components (which often explain the most variance in the dataset). Performing PCA first also provides a visual way to determine the number of clusters in the dataset if is is unknown as PCA will separate data along the two most discriminative axis. If the number of clusters is unknown, performing PCA first would be a good idea. As to which is better, it can depend on the dataset and the number of features. It can often turn out that the first two principal components do not actually explain much of the variance, and as such, in these cases it can be better to perform k-means first.

1.2 Q2 - Dataset B

Fig.(5a) shows the summary of classifications for the `B_NoiseAdded.csv` dataset. It was identified that the dataset contained 20 duplicated observations (a total of 40 observations involved in the duplication), and 10 of these duplicates contained different labels across the two duplicates. See Appendix A for the full list of duplicates. The correct assignment for this mislabelling was determined using multinomal logistic regression on the labelled data to predict the labels on the mislabelled data. Multinomal logistic regression was then used again to predict the labels for the 20 missing observations, and Fig.(5b) shows the new summary of classifications.

Generally, there are multiple ways to handle missing labels. One such way is model

	Fea58	Fea142	Fea150	Fea233	Fea269	Fea299	Fea339	Fea355	Fea458	Fea466	Fea491
138	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan
143	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan
231	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan
263	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan
389	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan

(a) The samples and features with missing data.

	Fea58	Fea142	Fea150	Fea233	Fea269	Fea299	Fea339	Fea355	Fea458	Fea466	Fea491	classification
138	0.0	4.13	0.0	1.07	0.35	0.0	1.84	0.0	0.57	0.0	0.0	2.0
143	0.0	4.02	0.0	1.0	0.56	0.0	2.02	0.0	0.58	0.0	0.0	2.0
231	0.0	4.55	0.0	0.66	0.0	0.05	2.33	0.0	1.33	0.0	0.0	1.0
263	0.17	4.31	0.0	0.71	0.12	0.0	1.86	0.0	1.08	0.44	0.0	4.0
389	0.0	4.47	0.0	0.7	0.0	0.0	2.61	0.0	0.33	0.0	0.0	4.0

(b) The imputed data.

Figure 6: Samples and features with missing data, for the `C_MissingData.csv` dataset.

based imputation, which was used here. This is where an appropriate model is trained on the labelled data to predict the missing labels. Another option could be to ignore the data with the missing labels, if the sample size is sufficiently large enough. Model based imputation has the advantage of using all the data, however it can introduce bias if the model is not a great fit for the data. Ignoring the data is advantageous in that it won't create bias if the sample size is large enough.

Missing at random (MAR) is when the likelihood that a label is missing is independent of the label itself, and missing not at random (MNAR) is when the likelihood of the label missing is in some way correlated with the value of the label. Looking at the 4th number in the brackets of Fig.(5b), the missing labels in each class as percentages of the total count in the respective classes are 4.5%, 6.1% and 2.9% (for 1, 2, and 4 respectively). This gives no significant indication of MNAR.

1.3 Q3 - Dataset C

1.3.1 Questions 3a and 3b

Fig(6a) shows all the features that have missing values, and all the samples which are missing those features. There are a few ways to handle missing data. A straightforward method is static imputation where every sample missing a given feature, is imputed with the same value for that feature, such as the mean of that feature. This is computationally inexpensive but can reduce the variance in the dataset and introduce bias if there are many missing values. Another method is model based imputation where an applicable model is chosen to estimate the value of the missing feature for each sample. An example is the K Nearest Neighbours approach, where a sample missing a feature is imputed with the mean of the K nearest neighbours. This can reduce the bias introduced and leave the variance unaffected, but highly dimensional data can often lead to less meaningful nearest neighbours, as distance can become less meaningful with more dimensions [2].

Multiple imputation is where the missing data is imputed using a probabilistic model mutliple times, to create multiple datasets. The multiple imputed values can then either be averaged or the multiple datasets can then be individually analysed, Multiple imputation

helps capture the uncertainty in what the missing data is, which is useful in the case that there is a large amount of missing data.

1.3.2 Question 3c

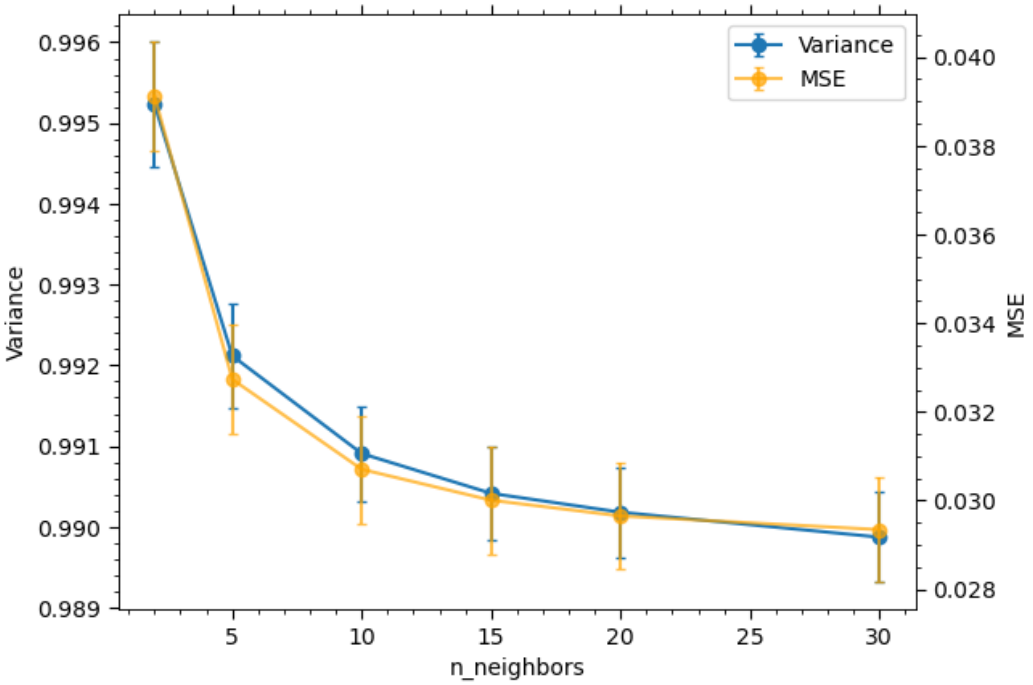

Figure 7: Variance and MSE of KNN imputed data, for different values of k. The variance is shown as a percentage of the original dataset, and the MSE is the mean squared error of predicted values against the true values.

	Var(KNN) / Var(orig) %	Var(mean) / Var(orig) %	Var(KNN) / Var(mean) %	KS(orig, KNN) p-val
Fea58	98.84	98.77	100.06	1.000000
Fea142	98.88	98.77	100.10	1.000000
Fea150	98.79	98.77	100.01	1.000000
Fea233	98.81	98.77	100.03	1.000000
Fea269	99.03	98.77	100.26	1.000000
Fea299	98.80	98.77	100.03	1.000000
Fea339	98.90	98.77	100.13	1.000000
Fea355	98.78	98.77	100.00	1.000000
Fea458	98.89	98.77	100.11	1.000000
Fea466	98.90	98.77	100.13	1.000000
Fea491	98.78	98.77	100.01	1.000000

Figure 8: A comparison of the variances of each feature after imputation. Column 1 shows the variance of each feature after KNN imputation, as a percentage of the original dataset. Column 2 shows this for mean imputation. Column 3 compares column 1 against column 2. Column 4 shows the p-value of the Kolmogorov-Smirnov test comparing the original dataset to the KNN imputed dataset.

Fig(7) was used to optimise the value of k for the KNN imputation, and $k = 15$ was chosen based on this simulation as further increase in k did not significantly increase accuracy (a reduction in MSE).

Fig(6b) shows the imputed data, with aforementioned KNN approach. However, PCA was performed first to reduce the dimensionality of the dataset, before the nearest neighbours were identified. Fig(8) shows some summary and comparative statistics of the imputed data, which identifies that imputing the missing data did not significantly change the distributions of the features.

1.3.3 Question 3d and 3e

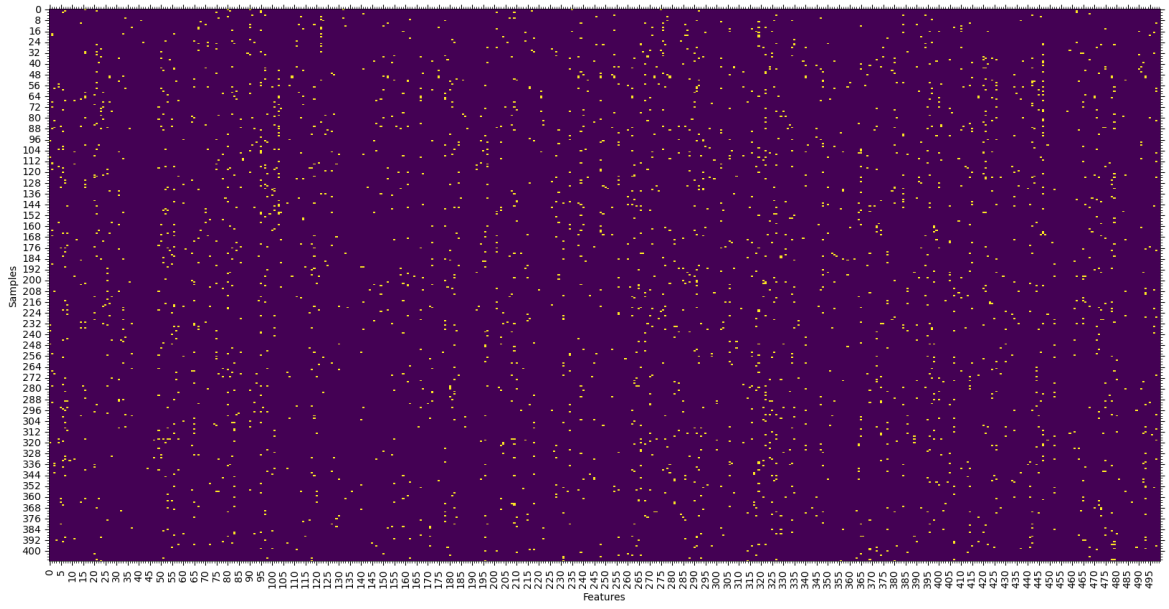

Figure 9: A heatmap of the outliers in the `C_MissingFeatures.csv` dataset. The orange marks indicate an outlier value, 2904 outlier values were identified.

Outlier values were identified by standardising the data feature-wise, and then identifying any values which were more than 3σ away from the mean. Fig(9) shows the outliers in the dataset, which shows that the outliers are fairly uniformly distributed across the dataset.

These outliers were treated as missing values and imputed using the same dimensionality reduction and KNN imputation approach as utilised in Question 3c for missing values. The justification behind this is similar as that for Question 3c, the dataset is clustered and a distance based approach is therefore likely to be fruitful, and utilising dimensionality reduction allowed for identification of meaningful nearest neighbours. Outliers were removed iteratively, with the dataset being re-standardised and outliers being re-computed after each iteration. The algorithm iterated 8 times, until only 0.27% of the data lay outside of 3σ, which is the expected proportion of data outside of 3σ for a normal distribution.

Fig(10) shows the mean and standard deviation of the 81 most discriminative features (identified from the PCA loadings) before and after imputing the outliers. The variance of the features were reduced after imputation, which was expected, and the reduction in variance in the features with means close to zero was more significant as these features were more sparse and therefore more likely to have outliers.

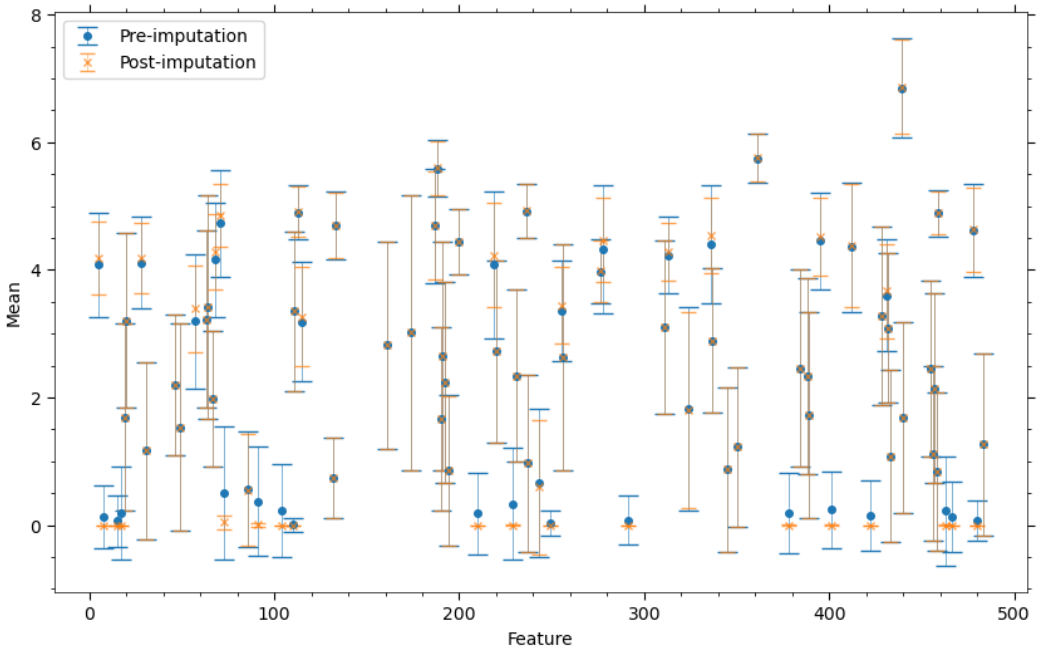

Figure 10: The mean and standard deviation of the 81 most discriminative features in the original `C_MissingFeatures.csv` dataset, before and after outlier imputation.

2 Section B

2.1 Q4 - Baseline Dataset

2.1.1 Question 4a

Classification decision trees utilise recursive binary splitting to split the feature-space into high dimensional rectangles in order to predict the outcome class given a feature set. In each iteration, an existing rectangle is split into two new rectangles, by splitting on a feature and a threshold value for that feature. The end tree will have some number of internal nodes N_n, representing splits in the feature space, and $N_n + 1$ terminal nodes which represent the final classification, which would be the modal class in that rectangle. The quality of the split can be measured by a criterion such as the Gini index, and at each split, the feature to split on and the threshold would be chosen by whichever split decreases the Gini index the most. The Gini index is defined as [3]:

$$G = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk}) \tag{2.1}$$

where $\hat{p}_{mk}$ is the proportion of training samples in the mth region that are from the kth class. A region m that contains mainly a single class will have a small Gini index for that region.

Decision trees suffer from high variance as the trained model is highly dependent on the training data. Bagging and random forests are ensemble methods that reduce the variance of decision trees. With bagging, multiple decision trees are trained on different bootstrapped samples of the training data, and the classification is then decided by a majority vote of the trees. Random forests reduce the variance further by forcing an internal node to split

on a random subset of the features, resulting in less correlated trees, and therefore a larger decrease in variance.

Two hyperparameters of random forests `max_depth` and `max_features`, the former is the maximum number of internal nodes per tree, and the latter is the number of features to consider when splitting. Heuristically, `max_depth` is typically set to $\sqrt{N}$ where N is the number of features.

2.1.2 Question 4b

A Q2: Duplicate Observations in `B_NoiseAdded.csv`

The duplicated pairs in `B_NoiseAdded.csv`, the number is the sample number: [[74 146], [220 291], [147 409], [44 193], [66 253], [28 260], [384 396], [188 249], [83 198], [175 311], [166 424] [344 389], [117 297], [351 352], [120 359], [119 382], [100 173], [30 101], [46 107], [210 305]]

References

[1] scikit-learn developers, *scikit-learn k-means documentation*. Available at: https://scikit-learn.org/stable/modules/clustering.html#k-means [Accessed: 13-Dec-2023].

[2] Bellman, R. E. (1957). *Dynamic Programming*. Princeton University Press, Princeton, NJ, USA.

[3] James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). *An Introduction to Statistical Learning: with Applications in R*. Springer, New York.