IAML – INFR10069 (LEVEL 10): Assignment #2

s1709906

PART A: 20-NEWSGROUPS [60 POINTS]

Question 1 : (10 points) Exploratory Analysis

We will begin by exploring the Dataset to get some insight about it.

(a) (5 points) Focusing first on the training set, summarise the key features/observations in the data: focus on the dimensionality, data ranges, feature and class distribution and report anything out of the ordinary. What are the typical values of the features like?

The training set has 5648 entries, which means it has 5648 different documents, with 1001 unique words (key features). It also has 8 different class labels representing each news group. After grouping the data into their respective classes, I found out that out of all the newsgroups, class 7 (`soc.religion.christian`) has significantly fewer data compared to the other classes. All of the other classes have more than 700 values, but class 7 has lesser than 500. This will cause the training to be unbalanced. By examining the mean and standard deviation of each word, I could see that the words have low TF-IDF values. The minimum TF-IDF value is 0.0 and the maximum is 1.0, and the values are spread around the mean (low s.d.).

Question 1 continued on next page...

(b) (3 points) Looking now at the Testing set, how does it compare with the Training Set (in terms of sizes and feature-distributions) and what could be the repurcussions of this?

The testing set has 1883 documents and 1001 unique words. The training set and the testing set are split 75% - 25%. This split is good because it does not overfit the model. There is a similar trend in the testing set with regards to the significantly lesser amount of data for class 7. All of the other classes have above 240 values and class 7 only has 157. The percentage of class 7 in the training set and the test set and they were both 8.34%. This is not a problem in the test set, however training a model with the unbalanced training set will yield inaccurate results.

Question 1 continued on next page...

(c) (2 points) Why do you think it is useful to consider TF-IDF weights as opposed to just the frequency of times a word appears in a document as a feature?

TF-IDF weights is a statistical measure to evaluate how important a word is to a document in a collection. The importance of a word is directly proportional to the word frequency in the document, but it is offset by the frequency of the word in the entire collection. It is useful to consider TF-IDF as opposed to just the word frequency in the document because it helps eliminate 'unimportant' words (such as stop words) that are used frequently from the feature vectors by setting their weight close to 0. It also helps avoid irrelevant and costly computations.

Question 2 : (24 points) Unsupervised Learning

We will now explore the documents in some detail by way of clustering.

(a) (2 points) The K-Means algorithm is non-deterministic. Explain why this is, and how the final model is selected in the SKLearn implementation of KMeans.

Non-deterministic means that running the algorithm several times on the same data, could give different results (test points classified differently in each iteration). This is due to the algorithm's random selection of data points as initial cluster centres. In the SKLearn implementation of KMeans, we do not explicitly select the initial cluster centres - instead, the model will select them for us. This will speed up the convergence of the algorithm.

Question 2 continued on next page...

(b) (1 point) One of the parameters we need to specify when using k-means is the number of clusters. What is a reasonable number for this problem and why?

A reasonable number of clusters would be 8, because there are 8 news groups and we want to classify which words typically belong to which group.

Question 2 continued on next page...

(c) (5 points) We will use the Adjusted Mutual Information (AMI) i.e. `adjusted_mutual_info_score` between the clusters and the true (known) labels to quantify the performance of the clustering. Give an expression for the MI in terms of entropy. In short, describe what the MI measures about two variables, why this is applicable here and why it might be difficult to use in practice. *Hint: MI is sometimes referred to as Information Gain: note that you are asked only about the standard way we defined MI and not the AMI which is adjusted for the size of the domain and for chance agreement.*

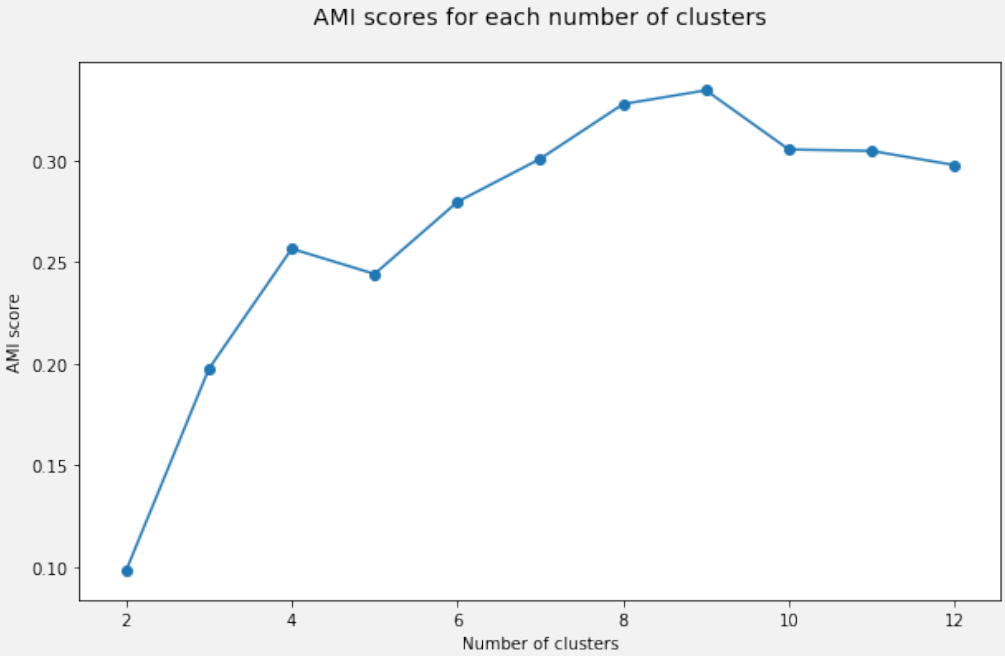
The formula for Mutual Information (MI) in terms of entropy is the following:

$$\mathrm{MI}(U,V) = \sum_{i=1}^{|U|}\sum_{j=1}^{|V|} P(i,j)\log\frac{P(i,j)}{P(i)P(j)} = H(V) - H(V|U) \tag{1}$$

where *H(V)* is the entropy of V and *H(V|U)* is a measure of what V doesn't say about U. It follows that MI is the amount of information (decrease in uncertainty) you get about one variable given knowledge of another. MI calculates the statistical dependence between the two variables. In our case, words are not independent so knowing one of them can give us more insight about the next word, and possibly about its class. MI measures require the knowledge of the true class values which are almost never available in practice or require manual assignment (annotations).

Question 2 continued on next page...

(d) (4 points) Fit K-Means objects with `n_clusters` ranging from 2 to 12. Set the random seed to 1000 and the number of initialisations to 50, but leave all other values at default. For each fit compute the adjusted mutual information (there is an SKLearn function for that). Set `average_method='max'`. Plot the AMI scores against the number of clusters (as a line plot).



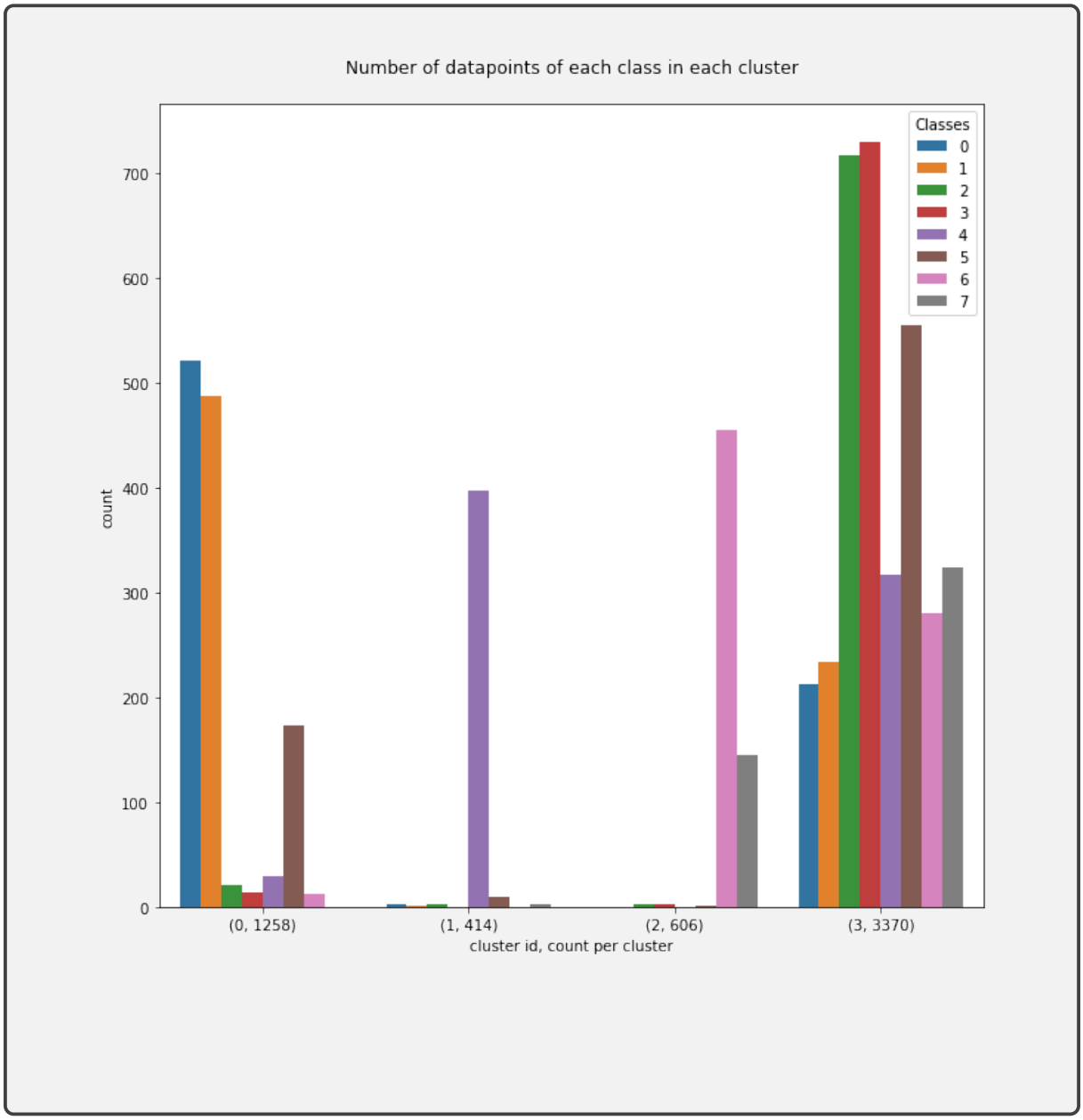
Question 2 continued on next page...

(e) (3 points) Discuss any trends and interesting aspects which emerge from the plot. Does this follow from your expectations?

The plot of AMI scores against the number of clusters shows a peak at n=9. I found this quite interesting because I expected that the AMI score will be the highest when there are 8 clusters, since there are 8 newsgroups. This might be because there are some ambiguous words that falls in more than 1 cluster and these ambiguous words are clustered together into a separate cluster, keeping the 8 other clusters relatively pure. As the number of clusters increase past 9, the AMI score decreases due to overfitting.

Question 2 continued on next page...

(f) (6 points) Let us investigate the case with four (4) clusters in some more detail. Using seaborn's `countplot` function, plot a bar-chart of the number of data-points with a particular class (encoded by colour) assigned to each cluster centre (encoded by position on the plot's x-axis). As part of the cluster labels, include the total number of data-points assigned to that cluster.



Question 2 continued on next page...

(g) (3 points) How does the clustering in Question2:(f) align with the true class labels? Does it conform to your observations in Q 2(e)?

There are 4 bigger subgroups that you can generalise from the 8 newsgroups. This is noticeable in the bar chart - the 1st cluster has the main datapoint as classes 0, 1 and 5 (Electronics), the 2nd cluster contains almost only class 4 (Cryptography), the 3rd cluster has the classes 6 and 7 (Religion). The last cluster has a mix of all the classes but its main datapoint are classes 2 and 3 (Automotive). The cryptography group is well-separated and this is well-identified by the clustering algorithm. The classes that are the hardest to determine by the cluster assignments are classes 2 and 3 which are confused with all the other classes. The plot conforms with my observations in Q(2.e) as there is a local maxima when the number of clusters is 4 and the clustering has been somewhat successful at separating the data into their bigger subgroups, however not all the clusters are pure.

Question 3 : (26 points) Logistic Regression Classification

We will now try out supervised classification on this data. We will focus on Logistic Regression and measure performance in terms of the F1 score (familiarise yourself with this score which is related to the precision and recall scores that we learnt about in class).

(a) (3 points) What is the F1-score, and why is it preferable to accuracy in our problem? How does the macro-average work to extend the score to multi-class classification?

The F1-score ((2*precision*recall)/(precision+recall)) conveys the balance between the precision and the recall. It is more preferable to use accuracy in our problem because false positives and false negatives have similar cost and we have relatively even classes. To calculate the classifier's overall F1-score combine the per-class F1-scores into a single number. The simplest way to do that is to is to compute an arithmetic mean of the per-class F1-scores, called the macro-average.

Question 3 continued on next page...

(b) (2 points) As always we start with a simple baseline classifier. Define such a classifier (indicating why you chose it) and report its performance on the **Test** set. Use the 'macro' average for the `f1_score`.

I selected the class that has the most observations (in this case it's a tie between classes 3 and 6) and used that class as the result for all the predictions. This method is theoretically expected to perform better than random guessing.
After running the baseline classifier (predicting that all the values belong to class 3), I obtained the F1-score of 0.0292.

Question 3 continued on next page...

(c) (3 points) We will now train a LogisticRegression Classifier from SKLearn. By referring to the documentation, explain how the Logistic Regression model can be applied to classify multi-class labels as in our case. *Hint: Limit your explanation to methods we discussed in the lectures.*

The LogisticRegression Classifier can fit binary, One-vs-Rest, or multinomial logistic regression. Multi-class logistic regression utilises the softmax function to get the class probabilities. This function replaces the sigmoid logistic function used in binary logistic regression. The output of a softmax function is equal to the number of classes, and they sum up to one, therefore they are equal to the probabilities of each class.

Question 3 continued on next page...

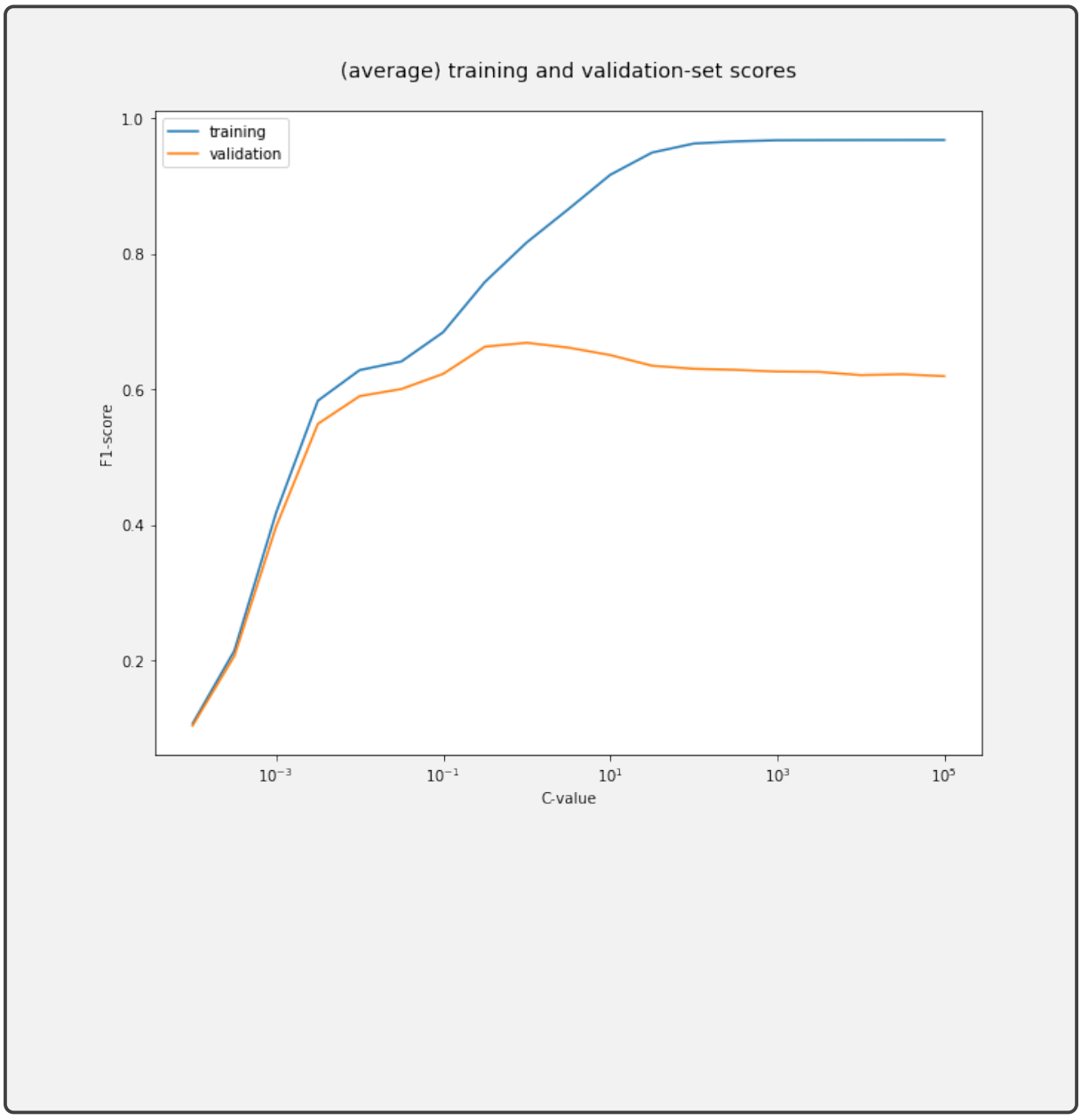
(d) (4 points) Train a Logistic Regressor on the training data. Set `solver='lbfgs'`, `multi_class='multinomial'` and `random_state=0`. Use the Cross-Validation object you created and report the average validation-set F1-score as well as the standard deviation. Comment on the result.

Average validation-set F1-score = 0.669
Standard deviation = 0.0169

The baseline classifier has significantly lower F1-score at 0.0291. Therefore, the logistic regression classifier performs significantly better than the baseline classifier. This shows a successful classification by the logistic model.

Question 3 continued on next page...

(e) (5 points) We will now optimise the Regularisation parameter C using cross-validation. Train a logistic regressor for different values of C: in each case, evaluate the F1 score on the training and validation portion of the fold. That is, for each value of C you must provide the training set and validation-set scores per fold and then compute (and store) the average of both over all folds. Finally plot the (average) training and validation-set scores as a function of C. *Hint: Use a logarithmic scale for C, spanning 19 samples between 10^{-4} to 10^{5}.*



Question 3 continued on next page...

(f) (7 points) What is the optimal value of C (and the corresponding score)? How did you choose this value? By making reference to the effect of the regularisation parameter C on the optimisation, explain what is happening in your plot from Question 3:(e) *Hint: Refer to the documentation for C in the LogisticRegression page on SKLearn.*

The most optimal value of C is 1.0 with a corresponding F1-score of 0.669. I chose this value by examining the F1-scores of the validation set and looked at the maximum, and find the corresponding value of C that yields that value. The effect of the regularisation parameter C on the optimisation is strong. This causes the model to become less sensitive to noise. However, at some point it becomes insensitive to important features. This is shown by the F1-score being low on small C values. Then the F1-score increases when C increased and reached its peak at 1.0, and after that it continues to decrease until it reaches a (somewhat) plateau.

Question 3 continued on next page...

(g) (2 points) Finally, report the score of the best model on the test-set, after retraining on the entire training set (that is drop the folds). *Hint: You may need to set* `max_iter = 200`. Comment briefly on the result.

The F1-score on the test set is 0.675 after using the optimal C-value and retraining on the entire training set. This value is higher than the average validation-set F1-score (0.669) using 10 folds that I calculated in Q(3.d) and the average validation-set F1-score (0.669) that I calculated in Q(3.e) while trying to optimise the value of C.

PART B: BRISTOL AIR-QUALITY [90 POINTS]

Question 4 : (30 Points) Exploratory Analysis

We will begin by exploring the Dataset to familiarise ourselves with it.

(a) (6 points) Summarise the key features/observations in the data: describe the purpose of each column and report (briefly) also on the dimensionality/ranges (ballpark figures only, and how they compare across features) and number of sites, and identify anything out of the ordinary/problematic: i.e. look out for missing data and negative values. Why are the latter unreasonable in such a dataset? *Hint: Refer to the documentation for how to interpret the pollutant values.*

There are 1306758 values for each column - Date/Time, followed by the three pollutants, SiteID, and the latitude and longitude of the site. I observed that there are some samples that are either empty or have a value below 0 (negative), which is unreasonable because the values are measured in $\mu g/m^3$, and they only range over positive values. There are 18 unique sites in total that are being studied. Grouping the data into their respective SiteIDs, I noticed that there are significantly lesser values for site 15 than all the other sites, with only 2712 values, followed by site 0 with 6446. This is very few compared to sites 1, 5, 6, 7, 14, 16, which has more than 100000 values. All the other sites range between 10000 and 100000. For the pollutants, NOx has the highest mean, followed by NO and NO2.

Question 4 continued on next page...

(b) (6 points) Repeat the same analysis but this time on a per-site basis. Provide a table with the number of samples and percentage of problematic samples (negative and missing) in each site. To report numbers, count a row which has at least one missing entry as having missing data, and similarly for negative entries. *Hint: Pandas has a handy method,* `to_latex()`*, for generating a latex table from a dataframe.*

SiteID	0	1	2	3	4	5	6	7	8
Negative	0	0	0.005	0.778	0.005	0	0.003	0.278	0
Missing	1.613	6.29	4.348	77.333	2.069	8.828	7.444	4.195	21.057
Total	6446	163111	62990	25464	74787	113952	142141	115162	43824

SiteID	9	10	11	12	13	14	15	16	17
Negative	0	0.004	0.087	0	0.016	0	0	0.014	0.002
Missing	5.301	3.59	1.904	17.485	51.461	10.532	100	6.531	6.271
Total	22071	96407	20693	45240	12423	113951	2712	154331	91053

* Negative and Missing values are in percentage (%)

Question 4 continued on next page...

(c) (4 points) Briefly summarise how the sites compare in terms of number of samples and amount of problematic samples.

Site 15 has the highest problematic values percentage of 100%. After examining the table, I can see that there are no samples for site 15 (all the problematic values are due to missing samples). The next highest problematic values percentage belongs to Site 3 at 78.1%, with a mixture of negative and missing samples. The next highest problematic values percentage belongs to site 13 (51.48%), site 8 (21.1%), site 12 (17.49%), and site 14 (10.53%). Every other site has a problematic sample percentage of less than 10%.

Question 4 continued on next page...

(d) (3 points) Given that the columns are all oxides of nitrogen and hence we expect them to be related, we will now look at correlations in our data. This will also be useful in determining how well we can predict any one of the readings from the other two. Remove the data from sites 3 and 15 and compute the **Pearson** correlation coefficient between each of the three pollutant columns on the remaining data. Visualise the coefficients between each pair of columns in a table.

	NOx	NO2	NO
NOx	1.000	0.878	0.988
NO2	0.878	1.000	0.808
NO	0.988	0.808	1.000

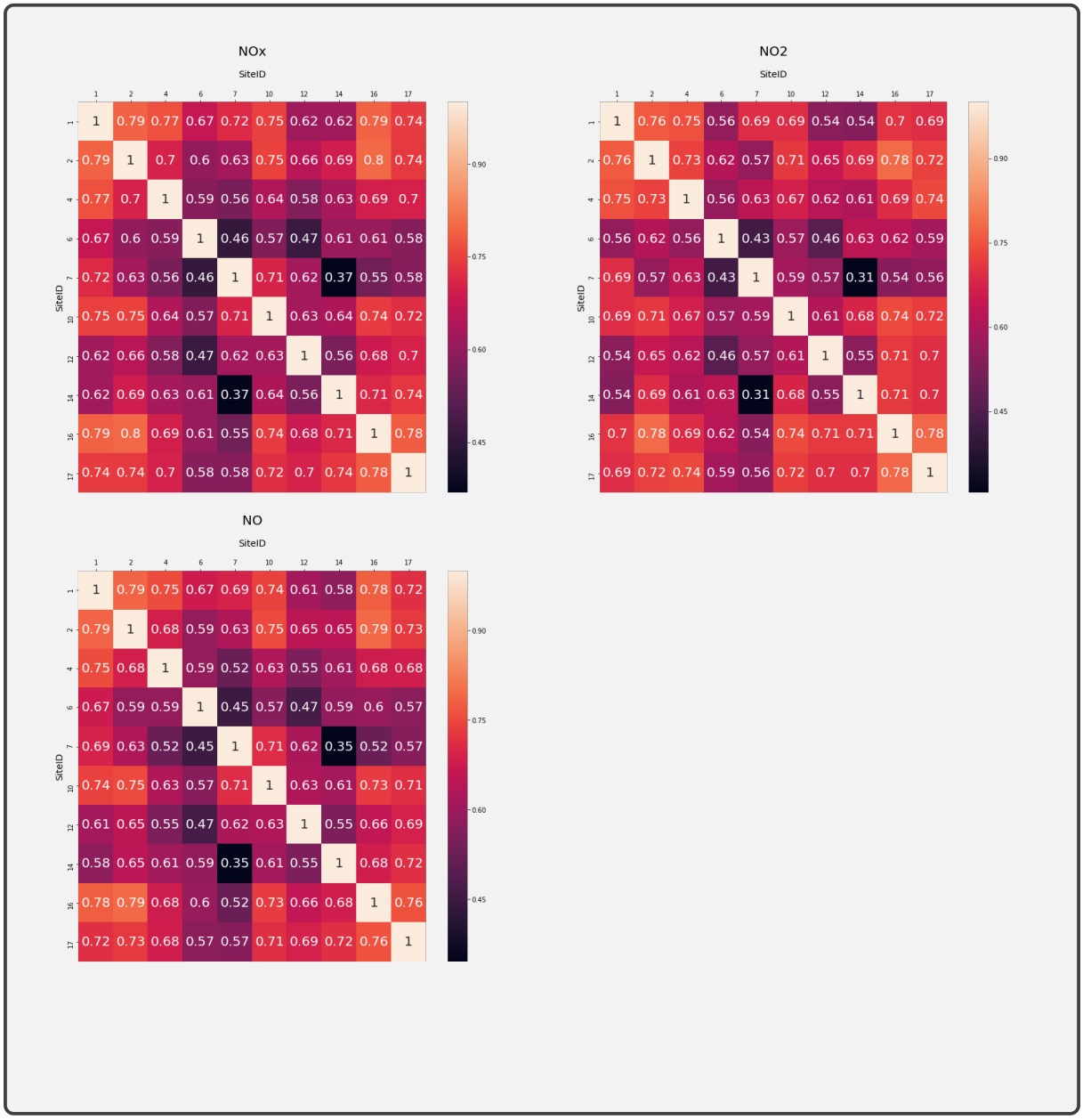
Question 4 continued on next page...

(e) (2 points) Comment on the level of correlation between each pair of pollutants.

The highest correlation is between NOx and NO, with a coefficient of 0.988. This is followed by the correlation of NOx and NO2, with a coefficient of 0.878. The lowest correlation is between NO2 and NO, with a coefficient of 0.808. Overall, I can observe that the pollutants have a strong positive correlation with each other.

Question 4 continued on next page...

(f) (5 points) For each of the three pollutants, compute the Pearson correlation between sites. *Hint: You will need to remove the 'Date Time' column and then group by the first level of the columns.* Then plot these as three heatmaps: show the values within the figures. *Hint: Use the method* `plot_matrix()` *from* `mpctools.extensions.mplext`.



Question 4 continued on next page...

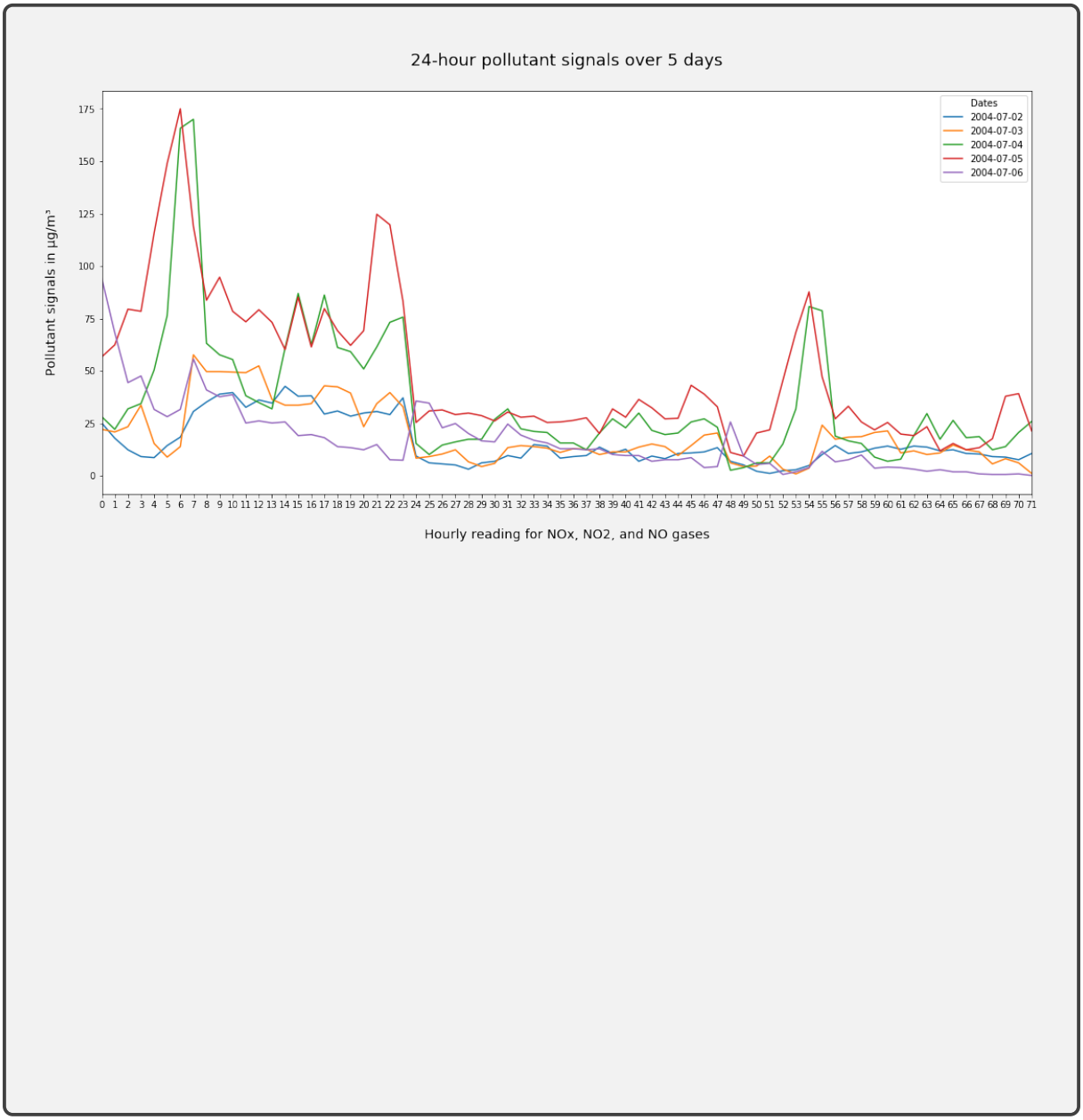
(g) (4 points) Comment briefly on your observations from Question 4:(f): start by summarising the results from the NO gas and then comment on whether the same is observed in the other gases or if there is something different.

For NO gas, the highest correlation is between site 2 and 16 (0.79), followed by 1 and 16 (0.78), then 16 and 17 (0.76). The lowest correlation is between site 7 and 14 (0.35), followed by site 7 and 6 (0.45), then 6 and 12 (0.47). These trends are also spotted in NOx and NO2. In fact, the correlation between the sites for each pollutant varies very slightly, as the heatmap looks rather identical at first glance. The lowest correlation between sites can usually be found on NO2, and the highest correlation between sites can usually by found on NOx.

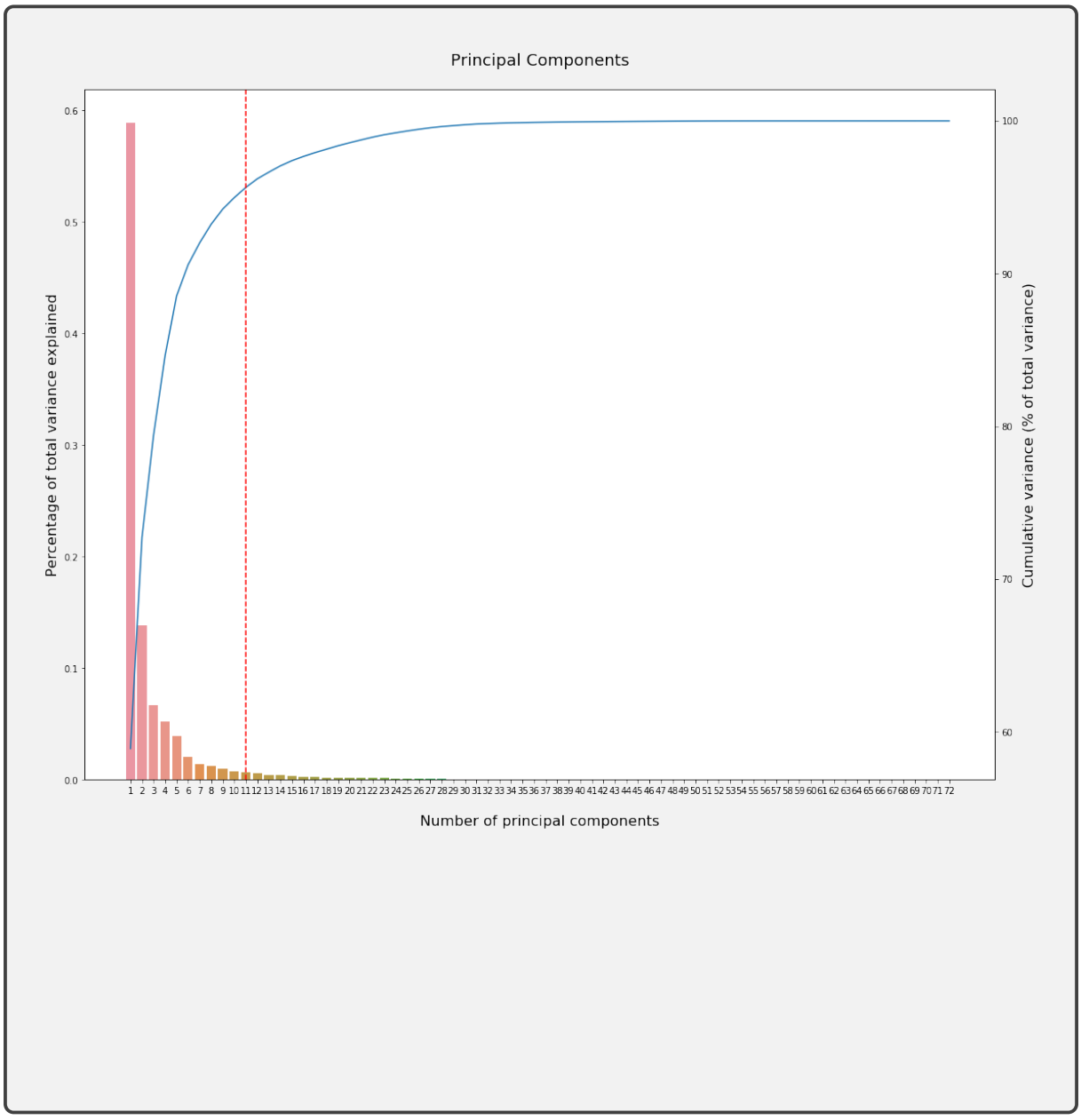
Question 5 : (19 Points) Principal Component Analysis

One aspect which we have not yet explored is the temporal nature of the data. That is, we need to keep in mind that the readings have a temporal aspect to them which can provide some interesting insight. We will explore this next.

(a) (1 point) Plot the first 5 lines of data (plot each row as a single line-plot).



(b) (5 points) We will focus first on data solely from Site 1. Extract the data from this site, and run PCA with the number of components set to 72 for now. Set the `random_state=0`. On a single graph plot: (i) the percentage of the variance explained by each principal component (as a bar-chart), (ii) the cumulative variance (line-plot) explained by the first n components: (*Hint: you should use `twinx()` to make the plot fit*), *and*, (iii) mark the point at which the number of components collectively explain at least 95% of the variance (using a vertical line). *Hint: Number components starting from 1.*



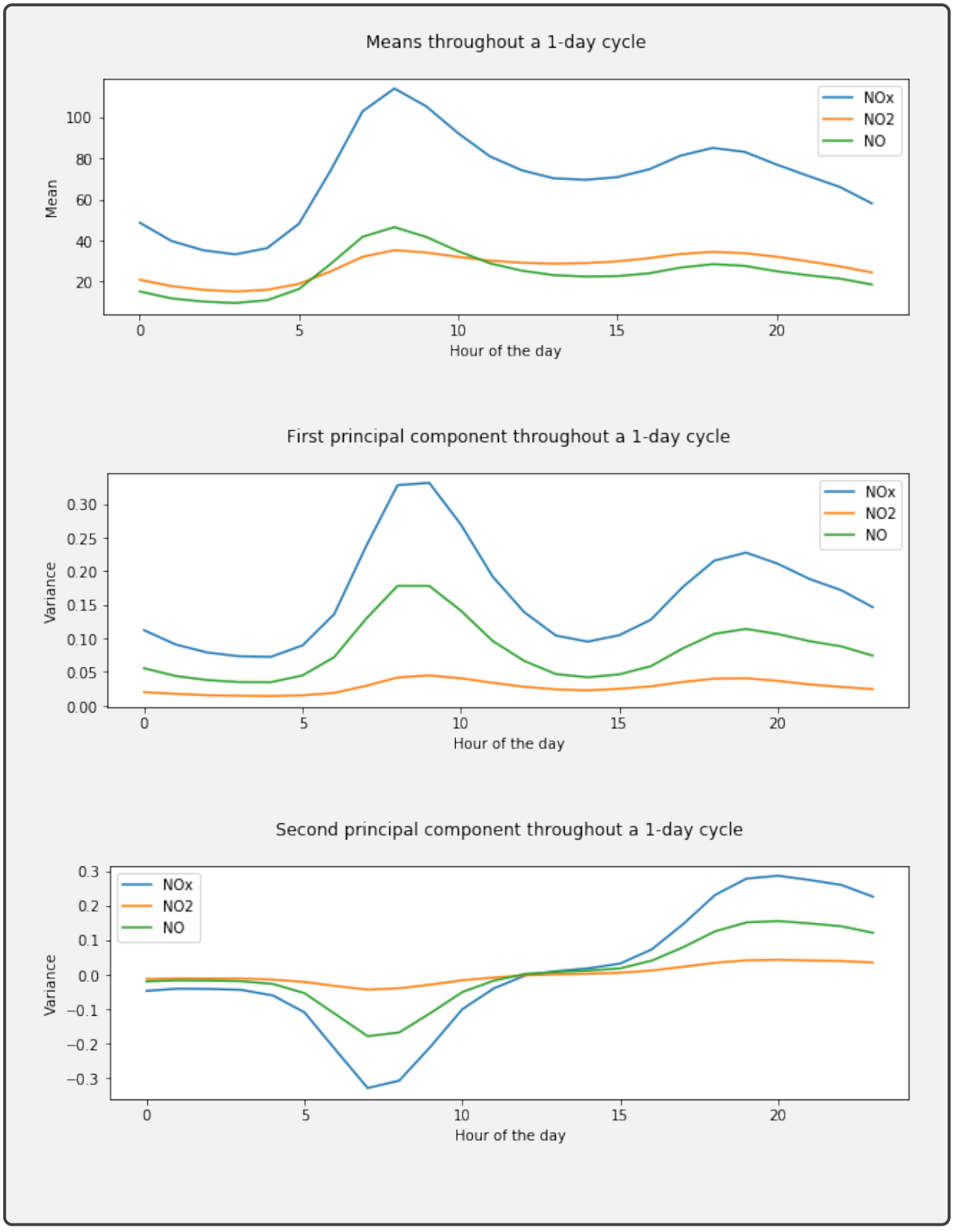
Question 5 continued on next page...

(c) (2 points) Interpret and summarise the above plot.

The percentage of total variance explained reduces exponentially as the number of principal components increase. The highest percentage of total variance explained is when the number of principal components is 1. The number of components which collectively explain at least 95% of the variance is 11. The first principal component accounts for almost 60% of the data. This is as expected because the first principal component accounts for as much of the variability in the data as possible, and all the components after that account for as much of the remaining variability as possible.

Question 5 continued on next page...

(d) (5 points) Generate three figures, one for the mean and one for each of the first 2 principal components: in each, plot the mean/component as three lines, one for each pollutant throught one day cycle. *Hint: You will need to reshape the components appropriately.*



Question 5 continued on next page...

(e) (6 points) Focusing on the mean and first principal component, are there any significant patterns which emerge throughout the day? *Hint: Think about car usage throughout the day.* What is different when interpreting the mean versus the first component? *Hint: Do peaks signify the same thing in both cases?* Looking at the principal components only, are there any significant differences between the pollutants? Why could this be happening? *Hint: You can refer to one of the limitations of PCA.*

There is an observable pattern in the mean plot - pollutant levels are lowest at 3 AM, has a global maximum at 8 AM, reached a local minima at 2 PM and reaches a local maxima at 6 PM. This corresponds to the general car usage throughout the day, where people commute in the morning to work/school/etc. and comes home in the evening. The peak in the mean shows the highest average measurement for a pollutant. However, the peak in the first component shows the maximum variance throughout the 24 hours. The first component follows a pattern similar to the mean despite them signifying different things. This is because the pattern in the mean is greatly affected by the pollutant values on work days (more work days than weekends) and the variance is caused by a different pattern that emerges on weekends (people not commuting as much). Looking at the principal components, I noticed that the value of NO2 does not vary as much as the other pollutants.

Question 6 : (41 points) Regression

Given our understanding of the correlation between signals and sites, we will now attempt to predict the NOx level for Site 17 given the value at the other sites. We will evaluate our models using the Root Mean Squared Error (RMSE) i.e. the square root of the mean_squared_error score by sklearn.

(a) (2 points) First things first: since we are dealing with a supervised task, we will need to split our data into a training and testing set. Furthermore, since some of our regressors will involve hyper-parameter tuning, we will also need a validation set. Use the `multi_way_split()` method from `mpctools.extensions.skext` to split the data into a Training (60%), Validation (15%) and Testing (25%) set: use the ShuffleSplit object from sklearn for the `splitter`. Set the random state to 0. *Hint: The method gives you the indices of the split for each set, which can then be applied to multiple matrices.* Report the sizes of each dataset.

Training set: 8937, validation set: 2234, testing set: 3724

Question 6 continued on next page...

(b) (4 points) Let us start with a baseline. By using only the y-values, what baseline regressor can you define (indicate what it does)? Implement it and report the RMSE on the training and validation sets. Interpret this relative to the statistics of the data.

The baseline regressor that I used assigns the central tendency measure as the result for all predictions. The central tendency measure that I chose is the mean.

The RMSE of the baseline regressor on the training set is: 79.714
The RMSE value of the baseline regressor on the validation set is: 80.205

Question 6 continued on next page...

(c) (3 points) Let us now try a more interesting algorithm: specifically, we will start with LinearRegression. Train the regressor on the training data and report the RMSE on the training and validation set, and comment on the relative performance to the baseline.

The RMSE value for the training set is: 39.835
The RMSE value for the validation set is: 41.127
From these values we can observe that the Linear Regression algorithm does significantly better than the baseline regressor.

Question 6 continued on next page...

(d) (5 points) We want to explore further what the model is learning. Explain why in Linear Regression, we cannot just blindly use the weights of the regression coefficients to evaluate the relative importance of each feature, but rather we have to normalise the features. By referring to the documentation for the LinearRegression implementation in SKLearn, explain what the normalisation does and how it helps in comparing features. Will this affect the performance of the Linear Regressor?

The magnitude of the regression coefficient indicates how much a unit-change in the independent variable affects the target variable, therefore it depends on the magnitude of the independent and dependent variable to begin with. If the two attributes are not on the same scale, we have to normalize them so that we can compare them. When you set `normalize=True`, the regressor's X values will be normalized before regression. The values are normalized by subtracting the mean and dividing by the l2-norm.

Question 6 continued on next page...

(e) (5 points) Retrain the regressor, setting `normalize=True` and report (in a table) the ratio of the relative importance of each feature. Which is the most/least important site? How do they compare with the correlation coefficients for Site 17 as computed in Question 4:(f), and why do you think that is?

SiteID	Correlation Coefficient
6	-0.006799
2	0.009947
7	0.059388
10	0.079779
14	0.094507

SiteID	Correlation Coefficient
1	0.115556
12	0.129110
4	0.134139
16	0.161828

The most important site is site 16 (0.162) and the least important site is site 6 (-0.007). These values correspond to the values computed in Q(4.f). This is because if the variable is strongly positively correlated to the target variable, it will have a big weight in linear regression, and affect the predicted value more.

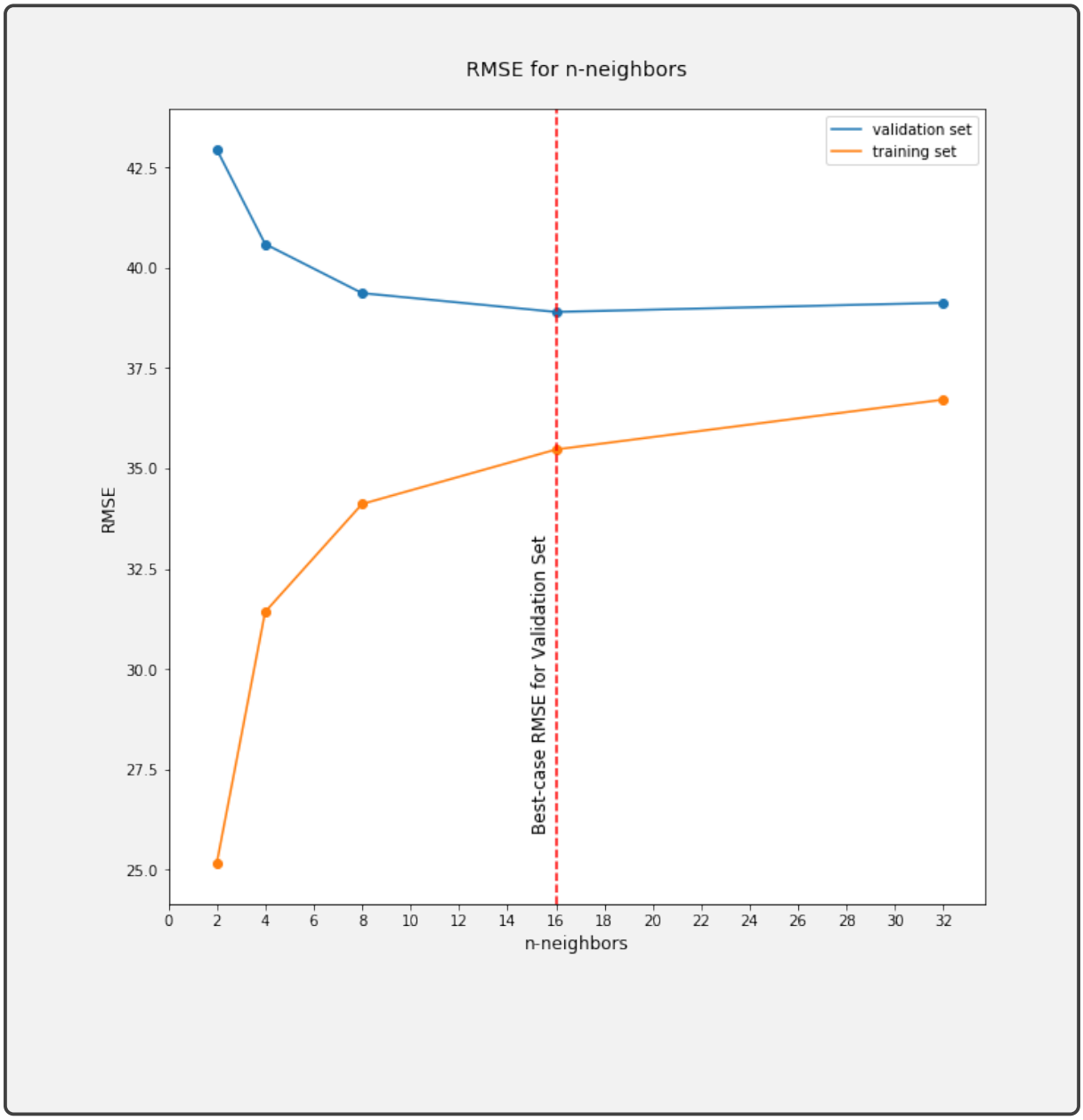
Question 6 continued on next page...

(f) (5 points) It might be that with non-linear models, we may get better performance. Let us try to use K-Nearest-Neighbours. Train a KNN regressor with default parameters on the training set and report performance on the training and validation set. *Hint: it might be beneficial to set* `n_jobs=-1` *to improve performance.* How does it compare with Linear Regression in terms of performance on both sets? What is a limitation of the KNN algorithm for our dataset?

KNN regression RMSE on the training set: 32.437 | validation set: 40.307
Linear Regression RMSE on the training set: 39.835 | validation set: 41.127
From these values we can observe that the KNN regression performs better than linear regression. The limitation of the KNN algorithm on our dataset is that it is computationally costly to calculate the distance between a new point point and all the existing points, hence the performance of the algorithm degrades.

Question 6 continued on next page...

(g) (4 points) The KNN regression allows setting a number of hyper-parameters. We will optimise only one: the number of neighbours to use. By using the validation set, find the optimal value for the `n_neighbours` parameter out of the values [2, 4, 8, 16, 32]. Plot the training/validation RMSE and indicate (for example with a line) the best value for `n_neighbours`.



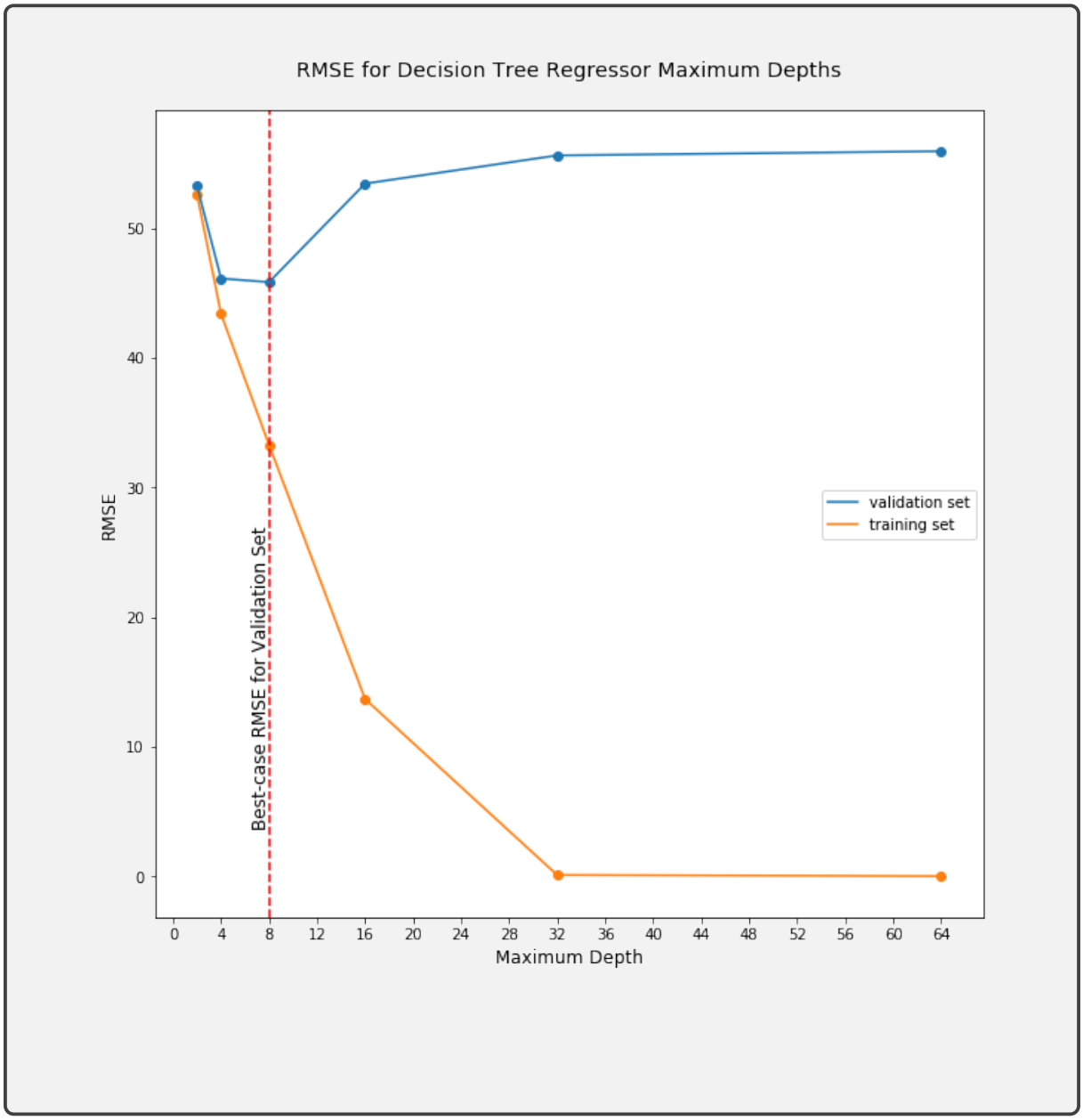
Question 6 continued on next page...

(h) (1 points) What is the best-case RMSE performance on the validation set for KNN?

The best-case RMSE performance on the validation set is 38.902. This is obtained when `n_neighbors` is set to 16.

Question 6 continued on next page...

(i) (4 points) Let us try one last regression algorithm: we will now use DecisionTreeRegressor. Again, the algorithm contains a number of hyper-parameters, and we will optimise the depth of the tree. Train a series of Decision Tree Regressors, optimising (over the validation set) the `max_depth` over the values [2, 4, 8, 16, 32, 64]. Set `random_state=0`. Plot the training/validation RMSE and indicate (as before) the best value for `max_depth`.



Question 6 continued on next page...

(j) (3 points) What is the best-case RMSE performance on the validation set? What do you notice from the plot about the performance of the Decision Tree Regressor?

The best-case RMSE on the validation set is 45.844, with a corresponding `max_depth` value set at 8. For the training set, increasing max depth reduces the RMSE to zero. For the validation set, increasing max depth overfits the training data without capturing useful patterns and cause RMSE to increase.

Question 6 continued on next page...

(k) (5 points) To conclude let us now compare all the models on the testing set. Combine the training and validation sets and retrain the model from each family on it: in cases where we optimised hyper-parameters, set this to the best-case value. Report the testing-set performance of each model in a table *Hint: You should have 4 values.*

Regressor	RMSE on Testing Set	Regressor	RMSE on Testing Set
Baseline	78.952130	KNN	37.984965
Linear	40.509027	Decision Tree	43.068871