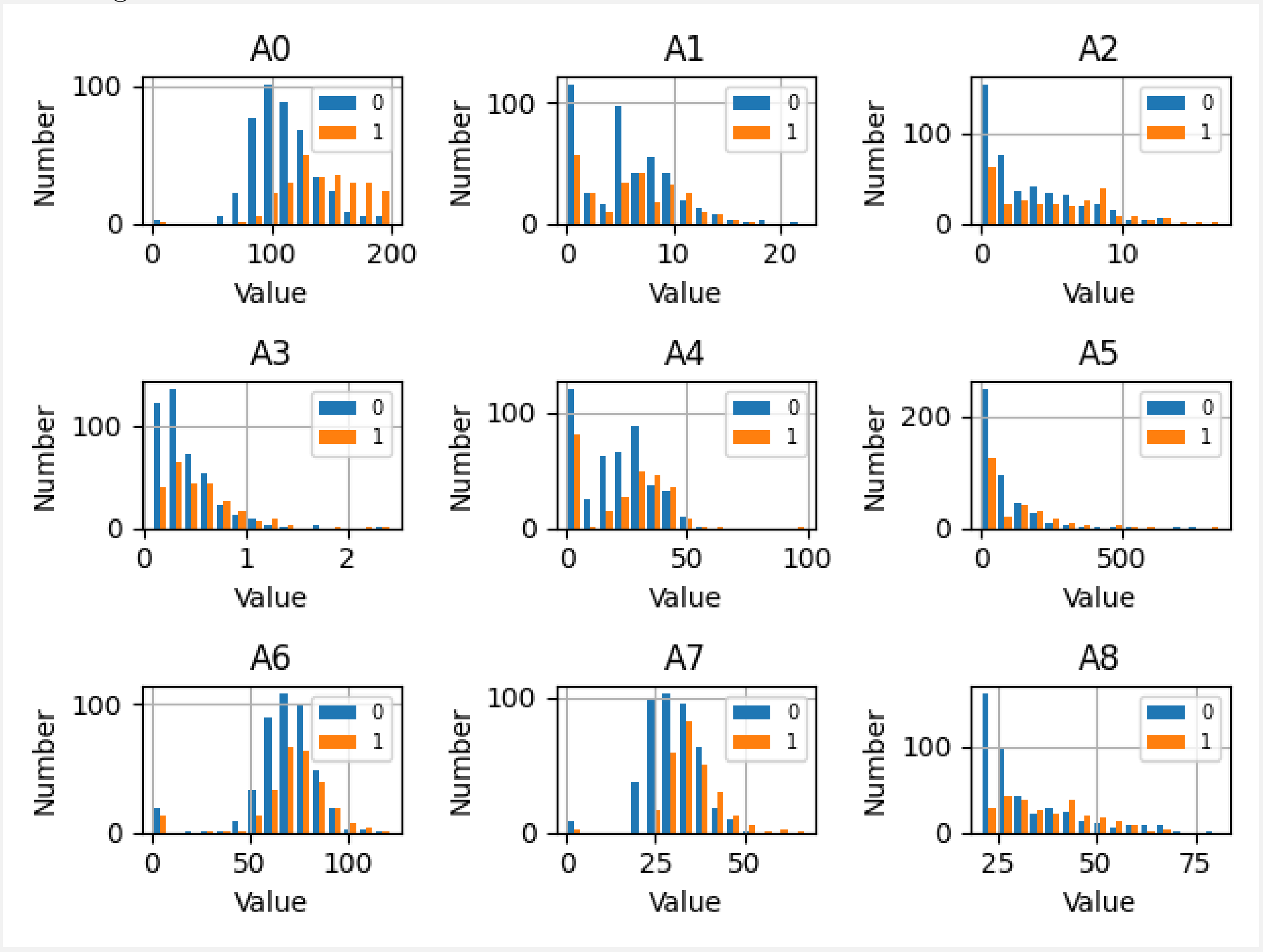
Question 1 : (80 total points) Experiments on a binary-classification data set

1.1 (9 points) We want to see how each feature in `Xtrn` is distributed for each class. Since there are nine attributes, we plot a total of nine figures in a 3-by-3 grid, where the top-left figure shows the histograms for attribute 'A0' and the bottom-right 'A8'. In each figure, you show histograms of instances of class 0 and those of class 1 using `pyplot.hist([Xa, Xb], bins=15)`, where `Xa` corresponds to instances of class 0 and `Xb` to those of class 1, and you set the number of bins to 15. Use grid lines. Based on the results you obtain, discuss and explain your findings.

The image shows how each feature in `Xtrn` is distributed for each class.



Based on the results I obtain. Some attributes are normal distribution, like A0,A6,A7, and the centroids of the class0 and class1 normal distributions of them can differ. If the difference between the centroids of class0 and class1 in an attribute is large, we can use the attribute to distinguish between class0 and class1, like A0. Besides, some attributes have an overall decreasing distribution, like A2,A3,A5,A8. There is little difference between class0 and class 1 distributions of these attributes, so I think we can not use them to distinguish between class0 and class1.

1.2 (9 points) Calculate the correlation coefficient between each attribute of `Xtrn` and the label `Ytrn`, so that you calculate nine correlation coefficients. Answer the following questions.

(a) Report the correlation coefficients in a table.
(b) Discuss if it is a good idea to use the attributes that have large correlations with the label for classification tasks.
(c) Discuss if it is a good idea to ignore the attributes that have small correlations with the label for classification tasks.

(a) Correlation coefficients are presented in the table below

Attribute	A0	A1	A2	A3	A4	A5	A6	A7	A8
Value	0.4912	0.0874	0.2273	0.2074	0.1077	0.1857	0.0763	0.3045	0.2403

(b) In my view, I think it is a good idea to use the attributes that have large correlations with the label. This is because the correlation coefficient reflects the linear correlation between two variables, the larger the absolute value of the correlation coefficient, the greater the correlation between attribute and label, so the attribute with large correlation coefficient can be used for classification tasks.

(c) As far as I am concerned, it is not a good idea to ignore the attributes that have small correlations with the label. This is because the correlation coefficient can only reflect a linear relationship. It is possible that the correlation between attribute and label is large, but not linear, so the absolute value of the coefficient will be small.

Question 1 continued on next page...

1.3 (4 points) We consider a set of instances of two variables, $\{(u_i, v_i)\}_{i=1}^N$, where N denotes the number of instances. Show (using your own words and mathematical expressions) that the correlation coefficient between the two variables, r_{uv}, is translation invariant and scale invariant, i.e. r_{uv} does not change under linear transformation, $a + bu_i$ and $c + dv_i$ for $i = 1, \ldots, N$, where a, b, c, d are constants and $b > 0, d > 0$.

The correlation coefficient for u_i, v_i:

$$r_{u_iv_i} = \frac{\sum_{i=1}^N (u_i - \bar{u})(v_i - \bar{v})}{\sqrt{\sum_{i=1}^N (u_i - \bar{u})^2}\sqrt{\sum_{i=1}^N (v_i - \bar{v})^2}}$$

We make linear transformation for u_i and v_i, so $u_i^{'} = a + bu_i$, $v_i^{'} = c + dv_i$. The correlation coefficient for $u_i^{'}, v_i^{'}$:

$$r_{u_i^{'}v_i^{'}} = \frac{\sum_{i=1}^N ((a + bu_i) - (a + b\bar{u}))((c + dv_i) - (c + d\bar{v}))}{\sqrt{\sum_{i=1}^N ((a + bu_i) - (a + b\bar{u}))^2}\sqrt{\sum_{i=1}^N ((c + dv_i) - (c + d\bar{v}))^2}}$$

$$\Rightarrow r_{u_i^{'}v_i^{'}} = \frac{bd\sum_{i=1}^N (u_i - \bar{u})(v_i - \bar{v})}{bd\sqrt{\sum_{i=1}^N (u_i - \bar{u})^2}\sqrt{\sum_{i=1}^N (v_i - \bar{v})^2}}$$

$$\Rightarrow r_{u_i^{'}v_i^{'}} = \frac{\sum_{i=1}^N (u_i - \bar{u})(v_i - \bar{v})}{\sqrt{\sum_{i=1}^N (u_i - \bar{u})^2}\sqrt{\sum_{i=1}^N (v_i - \bar{v})^2}} = r_{u_iv_i}$$

So, we can get that r_{uv} is translation invariant and scale invariant.

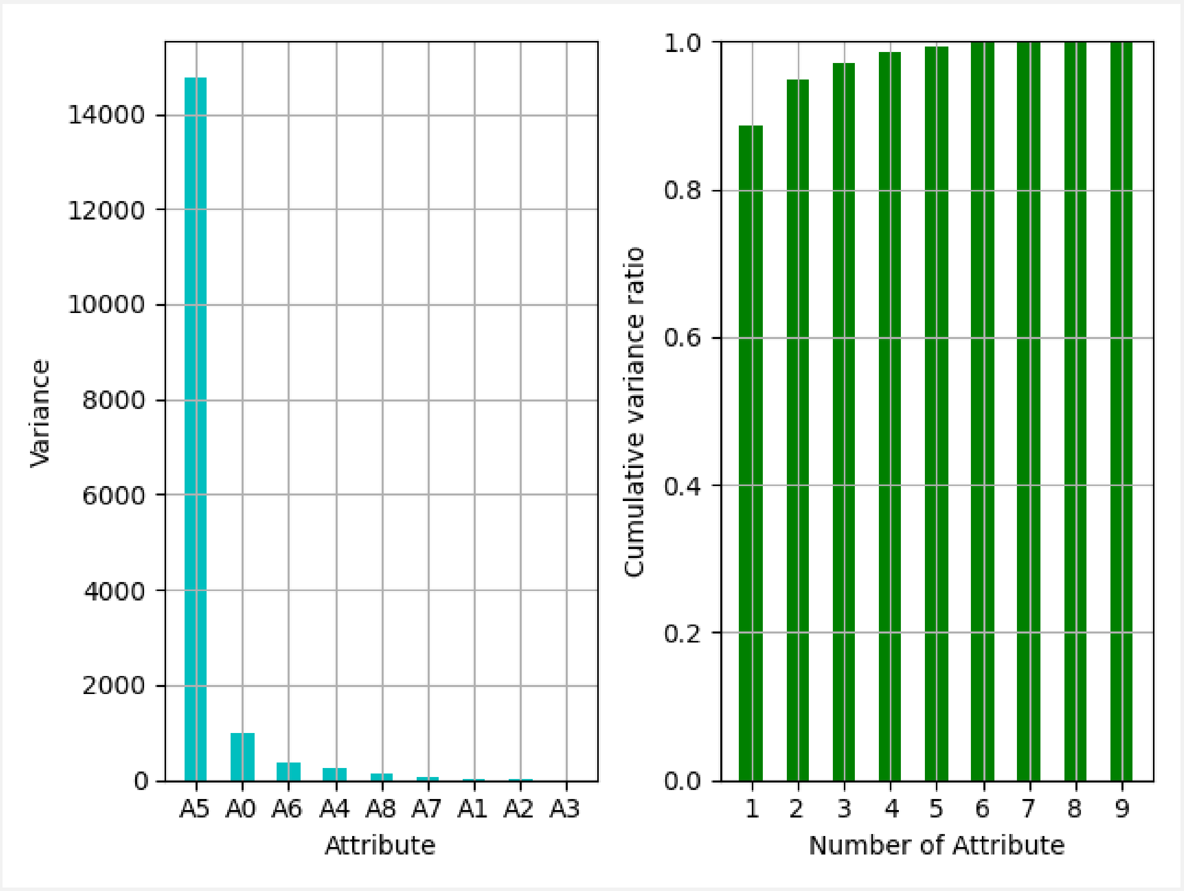
Question 1 continued on next page...

1.4 (5 points) Calculate the unbiased sample variance of each attribute of `Xtrn`, and sort the variances in decreasing order. Answer the following questions.

(a) Report the sum of all the variances.
(b) Plot the following two graphs side-by-side. Use grid lines in each plot.

- A graph of the amount of variance explained by each of the (sorted) attributes, where you indicate attribute numbers on the x-axis.
- A graph of the cumulative variance ratio against the number of attributes, where the range of y-axis should be [0, 1].

(a) The sum of all the variances is 16645.6365



(b)

1.5 (8 points) Apply Principal Component Analysis (PCA) to `Xtrn`, where you should not rescale `Xtrn`. Use Sklearn's PCA with default parameters, i.e. specifying no parameters.

(a) Report the total amount of unbiased sample variance explained by the whole set of principal components.

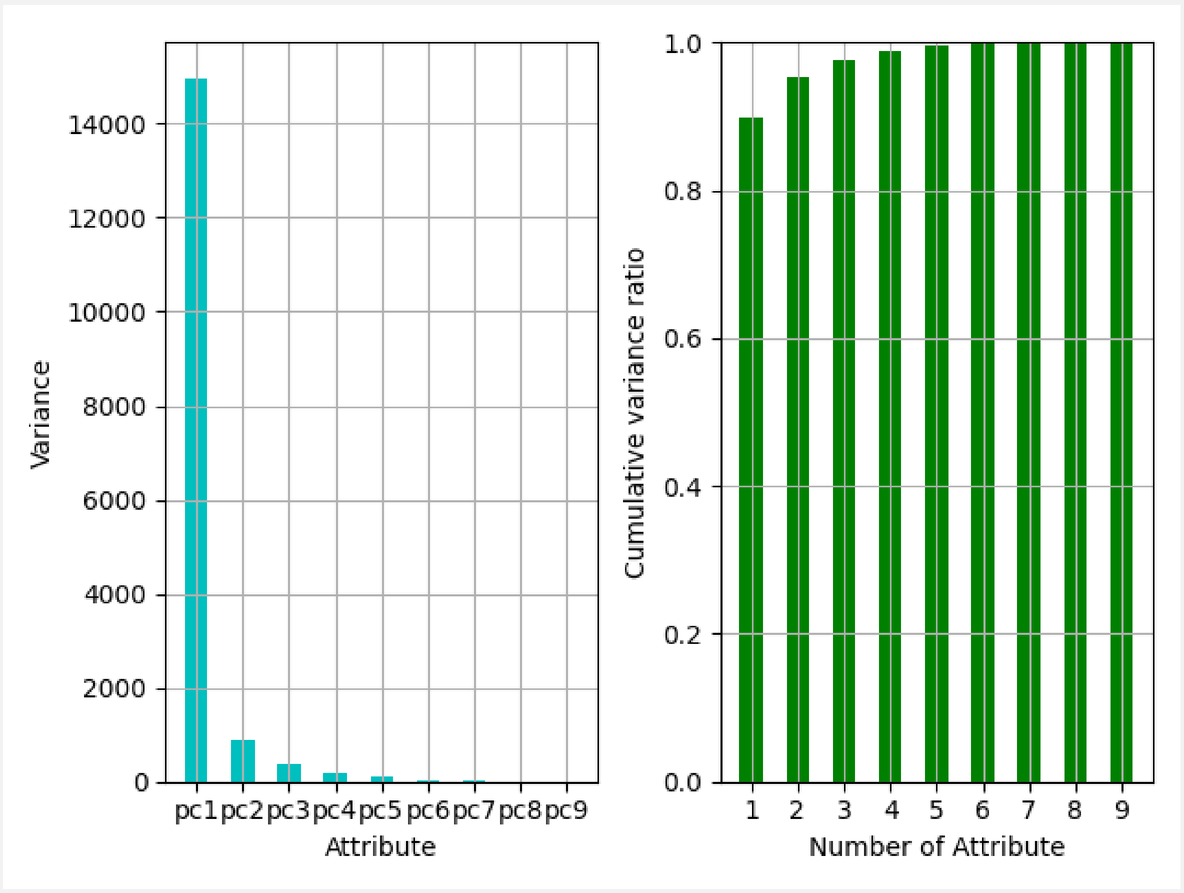
(b) Plot the following two graphs side-by-side. Use grid lines in each plot.

- A graph of the amount of variance explained by each of the principal components.
- A graph of the cumulative variance ratio, where the range of y-axis should be [0, 1].

(c) Mapping all the instances in `Xtrn` on to the 2D space spanned with the first two principal components, and plot a scatter graph of the instances on the space, where instances of class 0 are displayed in blue and those of class 1 in red. Use grid lines. Note that the mapping should be done directly using the eigen vectors obtained in PCA - you should not use Sklearn's functions, e.g. `transform()`.

(d) Calculate the correlation coefficient between each attribute and each of the first and second principal components, report the result in a table.

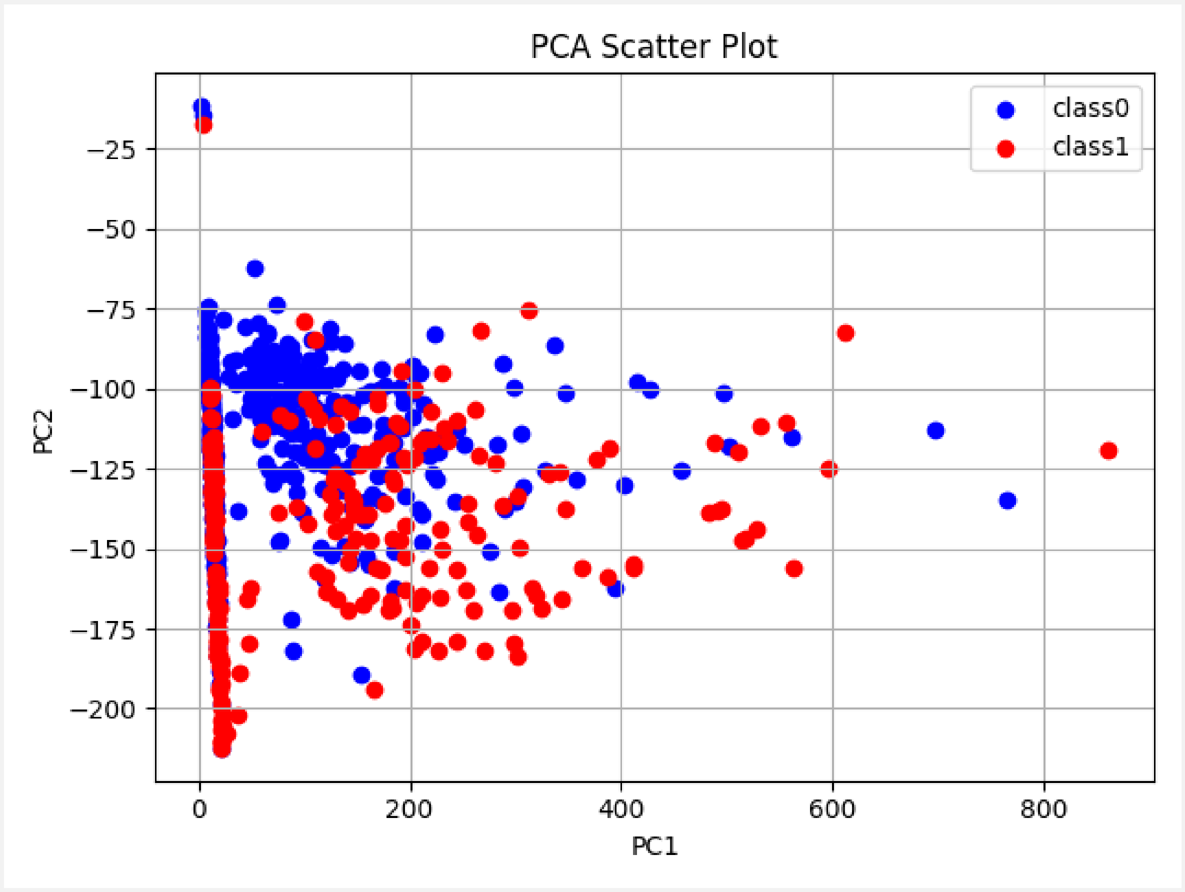
(a) The total amount of unbiased sample variance explained by the whole set of principal components is 16645.6365



(b)

(*continued from the previous page for Q1.5*)

(c)



(d) Correlation coefficient between each attribute and each of PC1 and PC2 below

Attribute	A0	A1	A2	A3	A4	A5	A6	A7	A8
PC1	0.3856	-0.0458	-0.0571	0.1858	0.4592	0.9997	0.1006	0.2323	-0.0016
PC2	-0.9143	0.0908	-0.2255	-0.0799	0.0972	0.0241	-0.2554	-0.1726	0.3734

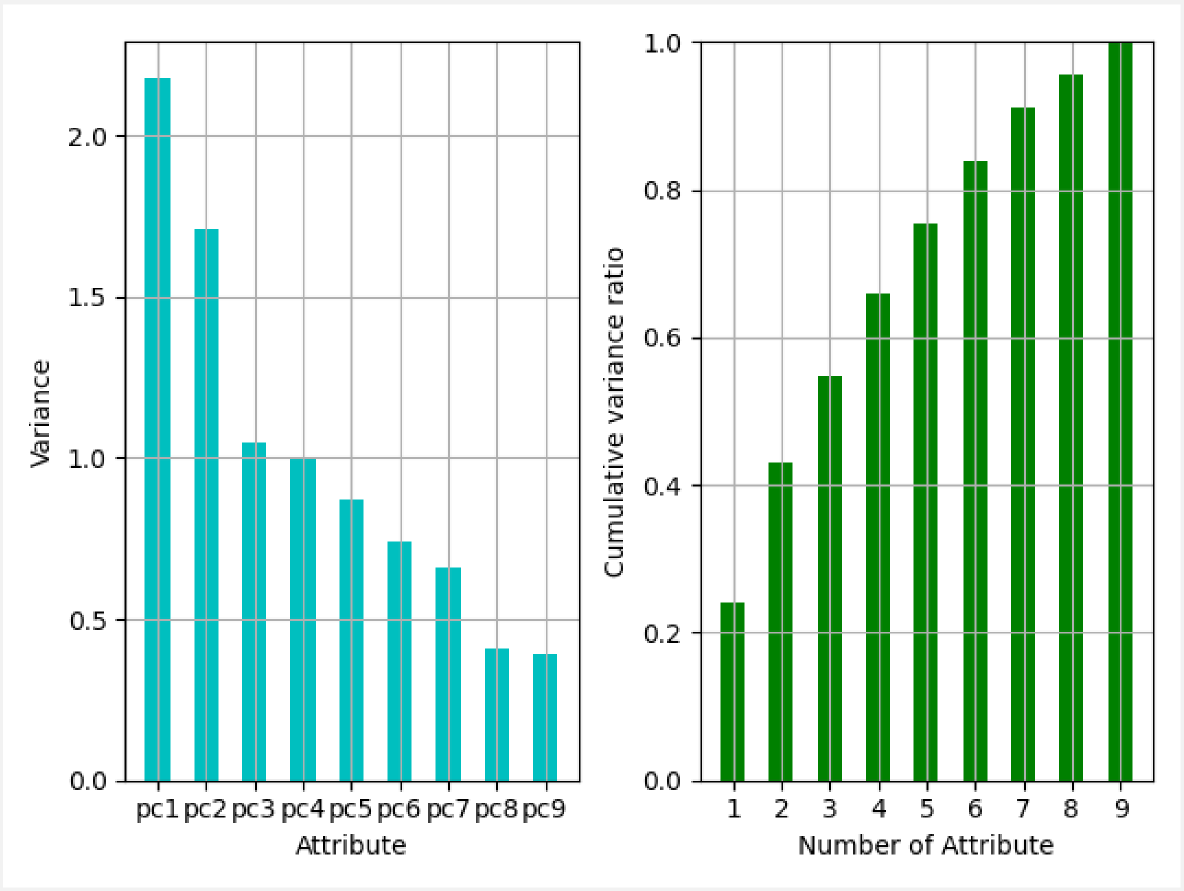
1.6 (4 points) We now standardise the data by mean and standard deviation using the method described below, and look into how the standardisation has impacts on PCA.

Create the standardised training data `Xtrn_s` and test data `Xtst_s` in your code in the following manner.

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler().fit(Xtrn)
Xtrn_s = scaler.transform(Xtrn)      # standardised training data
Xtst_s = scaler.transform(Xtst)      # standardised test data
```

Using the standardised data `Xtrn_s` instead of `Xtrn`, answer the questions (a), (b), (c), and (d) in 1.5.

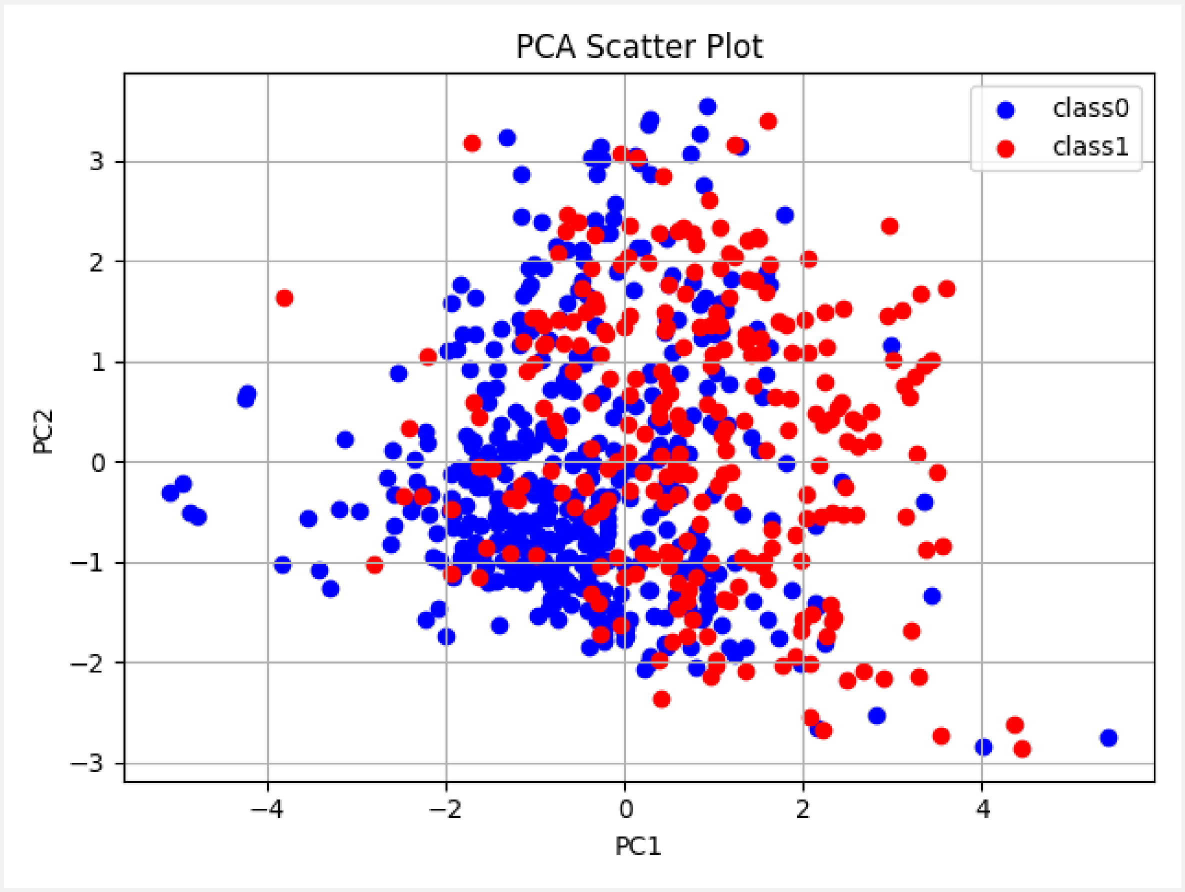
(a) The total amount of unbiased sample variance explained by the whole set of principal components is 9.0129



(b)

(*continued from the previous page for Q1.6*)

(c)



(d) Correlation coefficient between each attribute and each of PC1 and PC2 below:

Attribute	A0	A1	A2	A3	A4	A5	A6	A7	A8
PC1	0.6007	0.0573	0.2680	0.3657	0.6230	0.6299	0.5229	0.6512	0.3529
PC2	0.1774	0.1000	0.7600	-0.2076	-0.4660	-0.3698	0.2242	-0.1684	0.7812

1.7 (7 points) Based on the results you obtained in 1.4, 1.5, and 1.6, answer the following questions.

(a) Comparing the results of 1.4 and 1.5, discuss and explain your findings.
(b) Comparing the results of 1.5 and 1.6, discuss and explain your findings and discuss (*using your own words*) whether you are strongly advised to standardise this particular data set before PCA.

(a) The results of 1.4 and 1.5 are the same except that attributes have been replaced with PC. This is because the principle of calculating variance is the same.

(b) The total amount of unbiased sample variance explained by the whole set of principal components with `Xtrn`(16645.6365) is far greater than `Xtrn_s`(9.0129). The variance has changed greatly because the data are compressed into a smaller range, so the variance is smaller.
Compared to the variance curve obtained with `Xtrn`, the variance curve of the PC obtained with `Xtrn_s` falls more gently, and the maximum and minimum values of the PC variance do not differ much. This is because all the data in `Xtrn_s` are compressed into a small range and the variance obtained is not too large.
I do not think we should standardise this particular data set before PCA. For example, in order to achieve a cumulative variance ratio of 0.9, we only need to take one eigenvector(PC1) before standardising the data. However, after standardising the data, we need to take seven eigenvectors, which increases the computational effort significantly, but does not improve the model's performance.

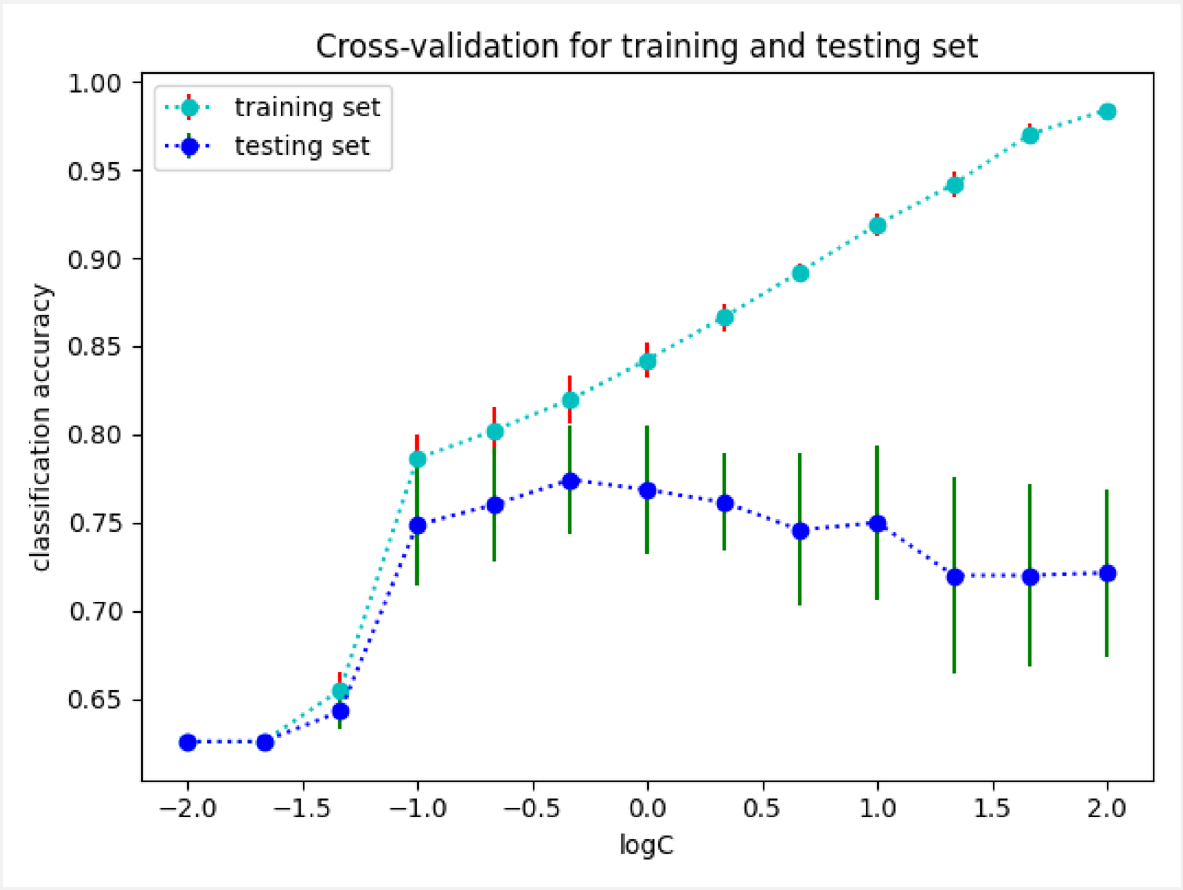
Question 1 continued on next page. . .

1.8 (12 points) We now want to run experiments on Support Vector Machines (SVMs) with a RBF kernel, where we try to optimise the penalty parameter C. By using 5-fold CV on the standardised training data `Xtrn_s` described above, estimate the classification accuracy, while you vary the penalty parameter C in the range 0.01 to 100 - use 13 values spaced equally in log space, where the logarithm base is 10. Use Sklearn's `SVC` and `StratifiedKFold` with default parameters unless specified. Do not shuffle the data.

Answer the following questions.

(a) Calculate the mean and standard deviation of cross-validation classification accuracy for each C, and plot them against C by using a log-scale for the x-axis, where standard deviations are shown with error bars. On the same figure, plot the same information (i.e. the mean and standard deviation of classification accuracy) for the training set in the cross validation.
(b) Comment (in brief) on any observations.
(c) Report the highest mean cross-validation accuracy and the value of C which yielded it.
(d) Using the best parameter value you found, evaluate the corresponding best classifier on the test set { `Xtst_s`, `Ytst` }. Report the number of instances correctly classified and classification accuracy.

(a) Cross-validation classification accuracy:



(b) As the logC increases, the accuracy of the training set gradually increases with a small standard deviation, while the accuracy of the test set increases first and then decreases with a large standard deviation all the time.

(c) The highest mean cross-validation accuracy: 0.7743 Value of C: 0.4642

(d) Number of instances correctly classified: 75 Classification accuracy: 0.75

1.9 (5 points) We here consider a two-dimensional (2D) Gaussian distribution for a set of two-dimensional vectors, which we form by selecting a pair of attributes, A4 and A7, in `Xtrn` (NB: not `Xtrn_s`) whose label is 0. To make the distribution of data simpler, we ignore the instances whose A4 value is less than 1. Save the resultant set of 2D vectors to a Numpy array, `Ztrn`, where the first dimension corresponds to A4 and the second to A7. You will find 318 instances in `Ztrn`.

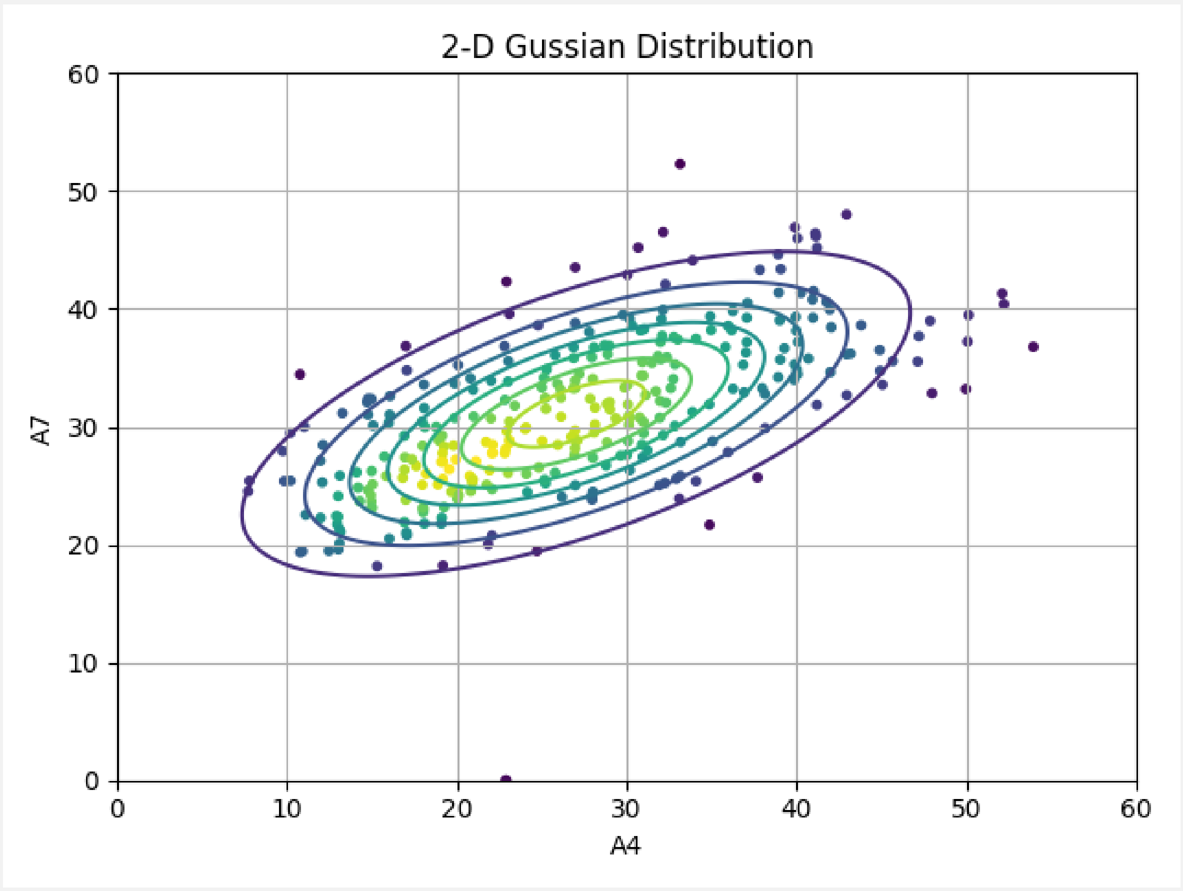
Using Numpy's libraries, estimate the sample mean vector and unbiased sample covariance matrix of a 2D Gaussian distribution for `Ztrn`. Answer the following questions.

(a) Report the mean vector and covariance matrix of the Gaussian distribution.
(b) Make a scatter plot of the instances and display the contours of the estimated distribution on it using Matplotlib's contour. Note that the first dimension of `Ztrn` should correspnd to the x-axis and the second to y-axis. Use the same scaling (i.e. equal aspect) for the x-axis and y-axis, and show grid lines.

(a) mean vector: $\begin{bmatrix} 27.0209 & 31.0932 \end{bmatrix}$

covariance matrix: $\begin{bmatrix} 95.1411 & 41.4700 \\ 41.4700 & 46.6934 \end{bmatrix}$

(b) The scatter plot of the instances and the contours of the estimated distribution:



1.10 (7 points) Assuming naive-Bayes, estimate the model parameters of a 2D Gaussian distribution for the data `Ztrn` you created in 1.9, and answer the following questions.

(a) Report the sample mean vector and unbiased sample covariance matrix of the Gaussian distribution.

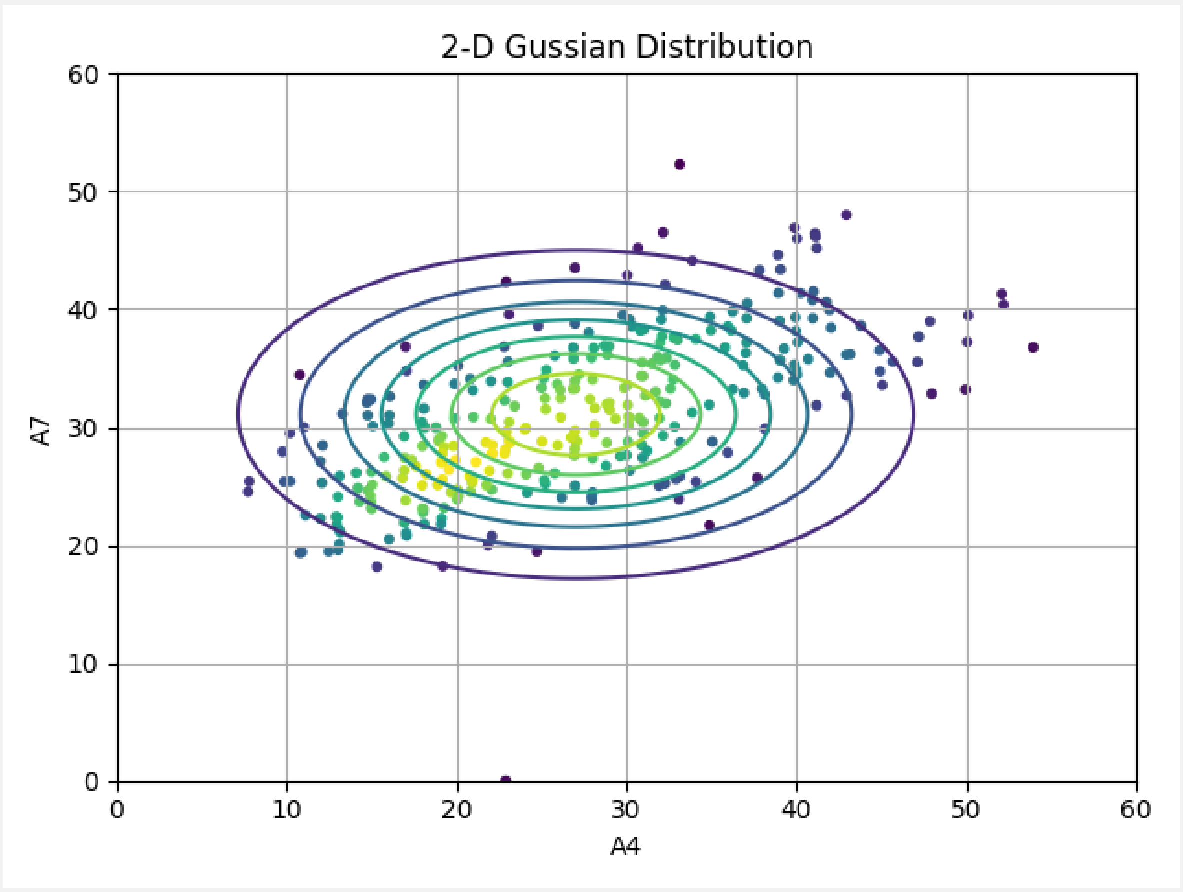
(b) Make a new scatter plot of the instances in `Ztrn` and display the contours of the estimated distribution on it. Note that you should always correspond the first dimension of `Ztrn` to x-axis and the second dimension to y-axis. Use the same scaling (i.e. equal aspect) for x-axis and y-axis, and show grid lines.

(c) Comparing the result with the one you obtained in 1.9, discuss and explain your findings, and discuss if it is a good idea to employ the naive Bayes assumption for this data `Ztrn`.

(a) mean vector: $\begin{bmatrix}27.0209 & 31.0932\end{bmatrix}$

covariance matrix: $\begin{bmatrix}95.1411 & 0\\ 0 & 46.6934\end{bmatrix}$

(b) The scatter plot of the instances and the contours of the estimated distribution:



(c) Comparing the result with the one I obtained in 1.9, the two axes of the estimated distribution contour in 1.10 are parallel to the x and y axes respectively. This is because we have assumed naive Bayes and there is no association between the two attributes
In my view, I do not think it is a good idea to employ the naive Bayes assumption. Because the original 2D Gaussian distribution without naive Bayes assumption can better reflect the distribution of the data.

1.11 (10 points) We now consider classification with logistic regression, for which we use the standardised training data `Xtrn_s` created in 1.6. Use Sklearn's `LogisticRegression` with default parameters except for specifying '`max_iter=1000`' and '`random_state=0`'. Use Sklearn's `StratifiedKFold` with default parameters. Do not shuffle the data.

(a) Using 5-fold CV on the training set, report the mean and standard deviation of cross-validation accuracy.
(b) We consider a simple feature selection that chooses eight attributes out of the nine, i.e. dropping a single attribute. Using 5-fold CV on the training set, for each choice of attribute to drop, report the mean and standard deviation of cross-validation accuracy in a table, and report the attribute which gave the highest mean cross-validation accuracy when it was omitted.
(c) Discuss and explain your findings.

(a) mean of cross-validation accuracy: 0.7714
standard deviation of cross-validation accuracy: 0.0436

(b) The mean and standard deviation of cross-validation accuracy:

Attribute omitted	A0	A1	A2	A3	A4	A5	A6	A7	A8
mean	0.6914	0.7657	0.7829	0.7557	0.7743	0.7700	0.7629	0.7557	0.7686
standard deviation	0.0298	0.0464	0.0466	0.0395	0.0395	0.0374	0.0308	0.0439	0.0390

The attribute which gave the highest mean accuracy when it was omitted: A2

(c) After omitting A0, both the mean and standard deviation of the accuracy are the lowest. After omitting A2, both the mean and standard deviation of the accuracy are the highest. This shows that in the classification task, A0 has the largest contribution and A2 has the smallest contribution.

Question 2 : (90 total points) Experiments on an image data set of handwritten letters

2.1 (5 points)

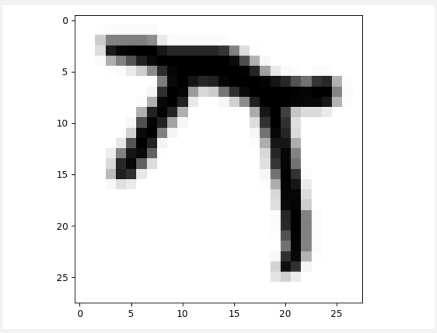
(a) Report (using a table) the minimum, maximum, mean, and standard deviation of pixel values for each `Xtrn` and `Xtst`. (Note that we mean a single value of each of min, max, etc. for each `Xtrn` and `Xtst`.)

(b) Display the gray-scale images of the first two instances in `Xtrn` properly, clarifying the class number for each image. The background colour should be white and the foreground colour black.

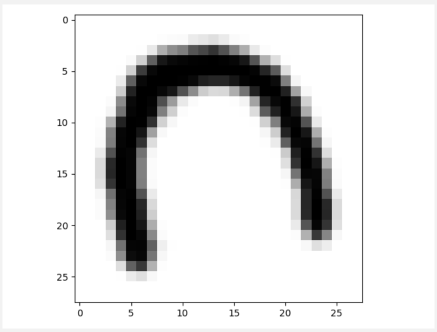
(a) Pixel values for each `Xtrn` and `Xtst`:

Data set	minimum	maximum	mean	standard deviation
`Xtrn`	0	1	0.1774	0.3350
`Xtst`	0	1	0.1756	0.3335

(b) class number: 10



class number: 2



2.2 (4 points)

(a) Xtrn_m is a mean-vector subtracted version of Xtrn. Discuss if the Euclidean distance between a pair of instances in Xtrn_m is the same as that in Xtrn.

(b) Xtst_m is a mean-vector subtracted version of Xtst, where the mean vector of Xtrn was employed in the subtraction instead of the one of Xtst. Discuss whether we should instead use the mean vector of Xtst in the subtraction.

(a) Suppose p, q is a pair of instances in Xtrn, p_m and q_m are the values they need to be subtracted from. So, $p - p_m$ and $q - q_m$ is a pair of instances in Xtrn_m.

$$d(p,q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \cdots + (p_n - q_n)^2} \tag{1}$$

$$\begin{aligned} d(p - p_m, q - q_m) &= \sqrt{((p_1 - p_{1m}) - (q_1 - q_{1m}))^2 + ((p_2 - p_{2m}) - (q_2 - q_{2m}))^2 + \cdots} \\ &= \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \cdots + (p_n - q_n)^2} \\ &= d(p,q) \end{aligned} \tag{2}$$

Therefore, we can get that the Euclidean between a pair of instances in Xtrn_m is the same as that in Xtrn.

(b) I think we should not use the mean vector of Xtst in the subtraction. We use the mean value of Xtrn in the subtraction when using Xtrn in training, so when testing on Xtst, we should also subtract the mean value of Xtrn.
This is because the means of different Xtst datasets vary, and two datasets with different means and the same distribution of Xtst will produce the same result. The result will be wrong.

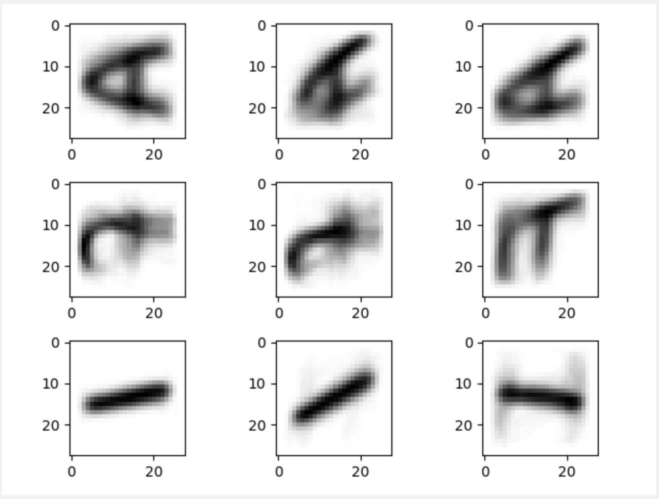
Question 2 continued on next page. . .

2.3 (7 points) Apply k-means clustering to the instances of each of class $0, 5, 8$ (i.e. 'A', 'F', 'I') in `Xtrn` with $k = 3, 5$, for which use Sklearn's `KMeans` with `n_clusters=`k and `random_state=0` while using default values for the other parameters. Note that you should apply the clustering to each class separately. Make sure you use `Xtrn` rather than `Xtrn_m`. Answer the following questions.

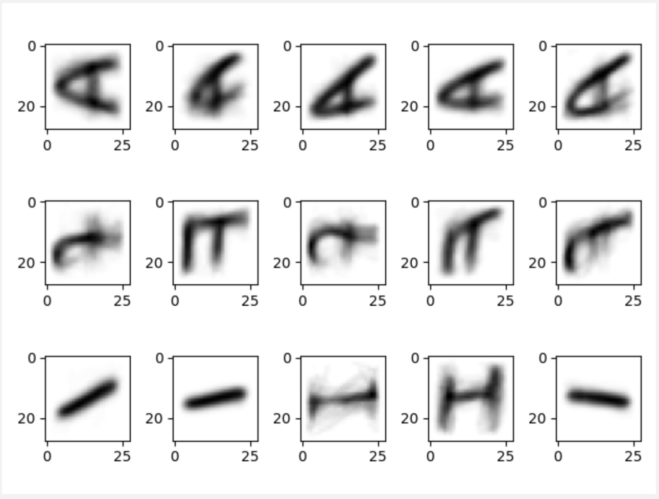
(a) Display the images of cluster centres for each k, so that you show two plots, one for $k = 3$ and the other for $k = 5$. Each plot displays the grayscale images of cluster centres in a 3-by-k grid, where each row corresponds to a class and each column to cluster number, so that the top-left grid item corresponds to class 0 and the first cluster, and the bottom-right one to class 8 and the last cluster.

(b) Discuss and explain your findings, including discussions if there are any concerns of using this data set for classification tasks.

(a) Images of cluster centres for 3:



Images of cluster centres for 5:



(b) When apply k-means clustering to the instances of one class, the features of grayscale images of cluster centres will vary. For example, when k=5 for class 8, some `I` look like `H`, which is also difficult to classify for human. Therefore, I think this data set does not do a good job of classifying similar letters.

2.4 (5 points) Explain (using your own words) why the sum of square error (SSE) in k-means clustering does not increase for each of the following cases.

(a) Clustering with $k + 1$ clusters compared with clustering with k clusters.
(b) The update step at time $t + 1$ compared with the update step at time t when clustering with k clusters.

(a) The sum of square error(SSE) is:

$$SSE = \sum_{i=1}^{k} \sum_{p \in C_i} (p - m_i)^2 \tag{3}$$

with k clusters, C the set of objects in a cluster, m the center point of a cluster.

SSE uses Euclidean distance as a clustering function between variables. The optimal solution is found in the direction of one variable at a time. So, this ensures that the SSE decreases at each iteration, eventually converging the SSE. The extreme case is that the SSE is 0 when k and the number of data points are equal.

(b) This is because as the cluster is constantly updated and the centroid is changing, the SSE slowly becomes smaller and converges to a value.

Question 2 continued on next page. . .

2.5 (11 points) Here we apply multi-class logistic regression classification to the data. You should use Sklearn's `LogisticRegression` with parameters '`max_iter=1000`' and '`random_state=0`' while use default values for the other parameters. Use `Xtrn_m` for training and `Xtst_m` for testing. We do not employ cross validation here. Carry out a classification experiment.

(a) Report the classification accuracy for each of the training set and test set.
(b) Find the top five classes that were misclassified most in the test set. You should provide the class numbers, corresponding alphabet letters (e.g. A,B,...), and the numbers of misclassifications.
(c) For each class that you identified in the above, make a quick investigation and explain possible reasons for the misclassifications.

(a) Classification accuracy on training set: 0.9161
Classification accuracy on testing set: 0.7223

(b) Top five classes that were misclassified most:

class number	alphabet letter	number of misclassification
11	L	53
17	R	48
8	I	42
10	K	38
13	N	36

(c) As the top five handwritten words are similar to other words, there are high recognition error rate on them. For example, L is similar to I and C, R is similar to B and K, N is similar to Z.

Question 2 continued on next page...

2.6 (20 points) Without changing the learning algorithm (i.e. use logistic regression), your task here is to improve the classification performance of the model in 2.5. Any training and optimisation (e.g. hyper parameter tuning) should be done within the training set only. Answer the following questions.

(a) Discuss (using your own wards) three possible approaches to improve classification accuracy, decide which one(s) to implement, and report your choice.
(b) Briefly describe your implemented approach/algorithm so that other people can understand it without seeing your code. If any optimisation (e.g. parameter searching) is involved, clarify and describe how it was done.
(c) Carry out experiments using the new classification system, and report the results, including results of parameter optimisation (if any) and classification accuracy for the test set. Comments on the results.

(a)
- use `solver` in `LogisticRegression` in the optimization problem. For example, we can use `solver='saga'` to improve classification accuracy
- use `penalty='l1' or 'l2'` to constrain the parameters so that the model is less likely to be over-fitted and a more generalizable model can be obtained.
- use hyper parameter tuning(adjusting `C`) to improve classification accuracy.

In this section, I choose `solver='saga'`, `penalty='l1'` and adjust the value of `C` to improve the classification performance.

(b) We set some parameters in `LogisticRegrssion` to optimise the performance of the model. We set `solver='saga'`, which is stochastic optimization algorithm using linear convergence. We set `penalty='l1'`, which means that the parameters of the model satisfy the Laplace distribution. Besides, we set `tol=0.1` and change the regularization factor `C` from 0.01 to 100 to get the best performance of the model. To be more specific, we apply `C=0.01, 0.1, 1, 10, 100` to the model.

Question 2 continued on next page...

(*continued from the previous page for Q2.6*)

(c) The performance of new classification system:

the value of C	accuracy for test set
0.01	0.2265
0.1	0.7138
1	0.7519
10	0.7492
100	0.7492

As the value of C increases, the accuracy of the test set first increases and then decreases. The classification accuracy of the model reaches a maximum at around C=1. As C increases, the classification accuracy of the test set converges to a value(0.7492). This is because C is the inverse of the regularization factor, which tends to 0 as C tends to infinity. In addition, we can select multiple C values around C=1 and apply them to the model to get the best C value to make the model work best.

Question 2 continued on next page...

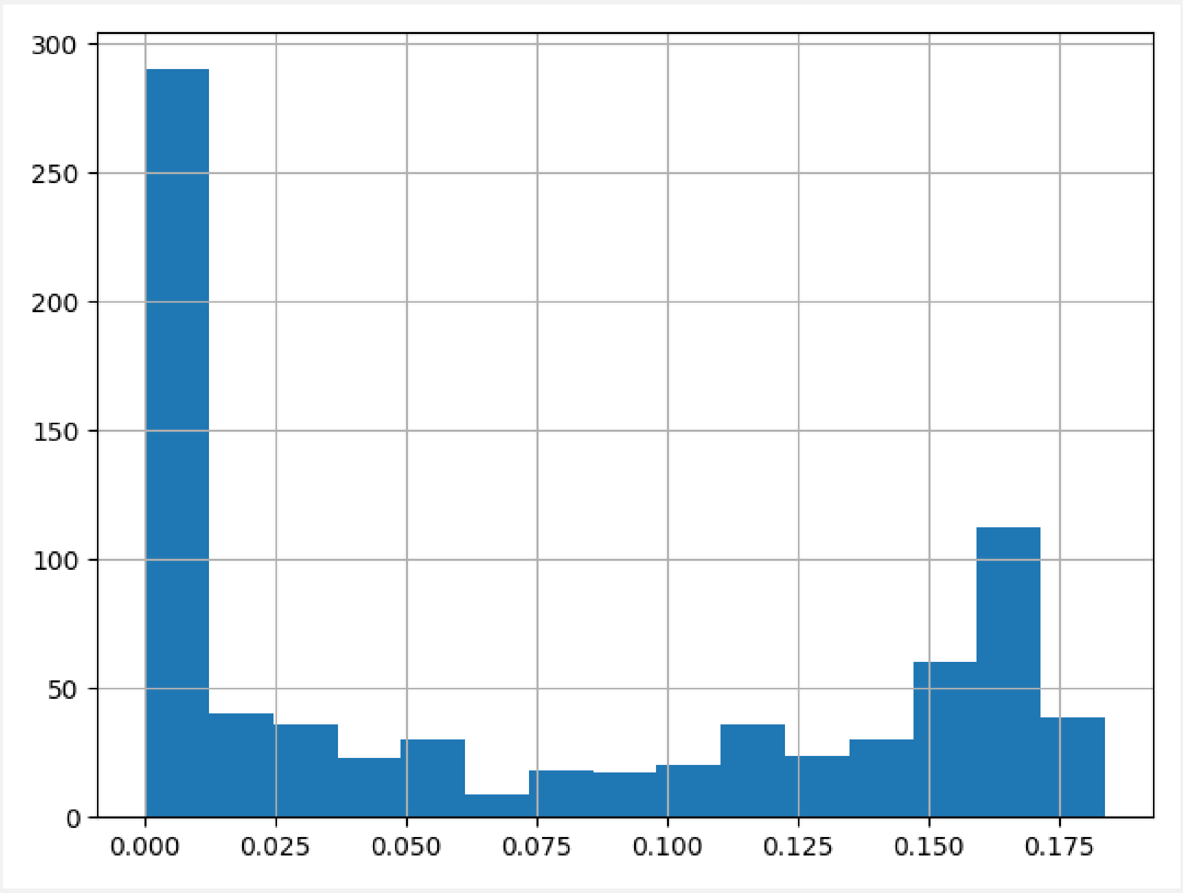
2.7 (9 points) Using the training data of class 0 ('A') from the training set `Xtrn_m`, calculate the sample mean vector, and unbiased sample covariance matrix using Numpy's functions, and answer the following.

(a) Report the minimum, maximum, and mean values of the elements of the covariance matrix.
(b) Report the minimum, maximum, and mean values of the diagonal elements of the covariance matrix.
(c) Show the histogram of the diagonal values of the covariance matrix. Set the number of bins to 15, and use grid lines in your plot.
(d) Using Scipy's `multivariate_normal` with the mean vector and covariance matrix you obtained, try calculating the likelihood of the first element of class 0 in the test set (`Xtst_m`). You will receive an error message. Report the main part of error message, i.e. the last line of the message, and explain why you received the error, clarifying the problem with the data you used.
(e) Discuss (using your own words) three possible options you would employ to avoid the error. Note that your answer should not include using a different data set.

(a) the elements of the covariance matrix:
minimum: -0.0975 maximum: 0.1838 mean: 0.0017

(b) the diagonal elements of the covariance matrix:
minimum: 0 maximum: 0.1838 mean: 0.0723

(c) The histogram of the diagonal values of the covariance matrix:



Question 2 continued on next page. . .

(*continued from the previous page for Q*)

(d) Error:
numpy.linalg.LinAlgError: singular matrix
Why:
This is because the input matrix is a singular matrix and the current matrix is not invertible, its determinant is 0. This means that the system of equations you are trying to solve does not have a unique solution.

(e)
- use `np.linalg.pinv`, which leverages SVD to approximate initial matrix
- when we use `multivariate_normal.pdf()`, we should set `allow_singular=True`
- use `numpy.linalg.lstsq` to invert singular matrices

Question 2 continued on next page...

2.8 (8 marks) Instead of Scipy's `multivariate_normal` we used in 2.7, we now use Sklearn's `GaussianMixture` with parameters, `n_components=1, covariance_type='full'`, so that there is a single Gaussian distribution fitted to the data. Use { `Xtrn_m`, `Ytrn` } as the training set and { `Xtst_m`, `Ytst` } as the test set.

(a) Train the model using the data of class 0 ('A') in the training set, and report the log-likelihood of the first instance in the test set with the model. Explain why you could calculate the value this time.

(b) We now carry out a classification experiment considering all the 26 classes, for which we assign a separate Gaussian distribution to each class. Train the model for each class on the training set, run a classification experiment using a multivariate Gaussian classifier, and report the number of correctly classified instances and classification accuracy for each training set and test set.

(c) Briefly comment on the result you obtained.

(a) log_likelihood for the first sample in the test set: -1712612.7436
Why:
In `GaussianMixture`, we set `reg_covar=1e-6(default)`, which means non-negative regularization will be added to the diagonal of covariance and allow to assure that the covariance matrices are all positive. So we do not need to worry about the issue of singular matrix.

(b) number of correctly classified instances in training: 7800
classification accuracy on training: 1.0
number of correctly classified instances in testing: 1803
classification accuracy on testing: 0.6935

(c) While the classification accuracy on the training set is high at 1, the classification accuracy on the test set is only 0.6935. This indicates that the multivariate Gaussian classifier does not achieve a good classification result and needs to be enhanced.

Question 2 continued on next page...

2.9 (6 points) Answer the following question on Gaussian Mixture Models (GMMs).

(a) Explain (using your own words) why Maximum Likelihood Estimation (MLE) cannot be applied to the training of GMMs directly.

(b) The Expectation Maximisation (EM) algorithm is normally used for the training of GMMs, but another training algorithm is possible, in which you employ *k*-means clustering to split the training data into clusters and apply MLE to estimate model parameters of a Gaussian distribution for each cluster. Explain the difference between the two algorithms in terms of parameter estimation of GMMs.

(a) If we apply MLE to the training of GMMs directly, the problem is that we have a lot of data, but we do not know which Gaussian model the data comes from. Therefore, we cannot solve for the joint probability of the samples, and we also cannot find the likelihood function.

(b) As for k-means, it only relies on one component, which is the mean of each cluster, to get the centroids of each cluster. If two clusters have overlapping means and different covariance matrices, we can not separate them with k-means. Besides, k-means use Euclidean to calculate the mean.
In terms of Expectation Maximisation(EM), clusters are defined by their means and their variance, and it also works if clusters are overlapping. In addition, EM use the probability of X belonging to a cluster to categorise data.

Question 2 continued on next page. . .

2.10 (15 points) We now extend the classification with a separate multivariate Gaussian model for each class that we performed in 2.8 to one with Gaussian Mixture Model (GMM) per class. To achive this, change the number of mixture components in Sklearn's `GaussianMixture`. To simplify the experiment, do not use cross validation, but instead use the test set as a validation set. Use `random_state=0` when you call Sklearn's `GaussianMixture`.

(a) Run experiments with GMMs for $k = 1, 2, 4, 8$ (where k is the number of mixture components).

  (i) Report classification accuracy for each of the training set and test set in a single table.

  (ii) Describe and briefly explain your findings.

(b) Using GMMs with $k = 2$, optimise the parameter '`reg_covar`' for the test set.

  (i) Report the result, i.e. the highest test-set accuracy and the value of the parameter value of `reg_covar` that yields it.

  (ii) Briefly discuss the '`reg_covar`' parameter in the context of this data set.

(a) Your Answer for (a) Here

(i)

value of k	1	2	4	8
accuracy of training set	1	1	1	1
accuracy of testing set	0.6935	0.7350	0.7619	0.8108

(ii) As the value of k increases, the classification accuracy of the training set is always 1, and the classification accuracy of the test set is constantly improving.
As k in `GaussianMixture` increases, there are more mixture components and the model is able to generalize more features in the image being classified. Therefore the classification accuracy of testing set will get higher and higher with increasing k.

(b) Your Answer for (b) Here

(i) The highest test-set accuracy: 0.8765
The value of `reg_covar`: 0.1

(ii) We use `reg_covar` to add non-negative regularization to the diagonal of covariance, so the covariance matrices are all positive and we do not need to worry about singular matrix error in `GaussianMixture`. As the value of `reg_covar` increases from 1e-6 to 100, the accuracy of the test-set increases and then decreases, reaching its highest value at around 0.1.