Question 1: Take the subset of the dataset containing your two class labels. You will use random 50/50 splits for training and testing data.
1. implement a linear kernel SVM. What is your accuracy and confusion matrix?
accuracy: 0.9,
confusion matrix svm: array([[33, 3], [4, 30]])

2. implement a Gaussian kernel SVM. What is your accuracy and confusion matrix?
accuracy: 0.9429,
confusion matrix svm: array([[34, 2],[2, 32]])

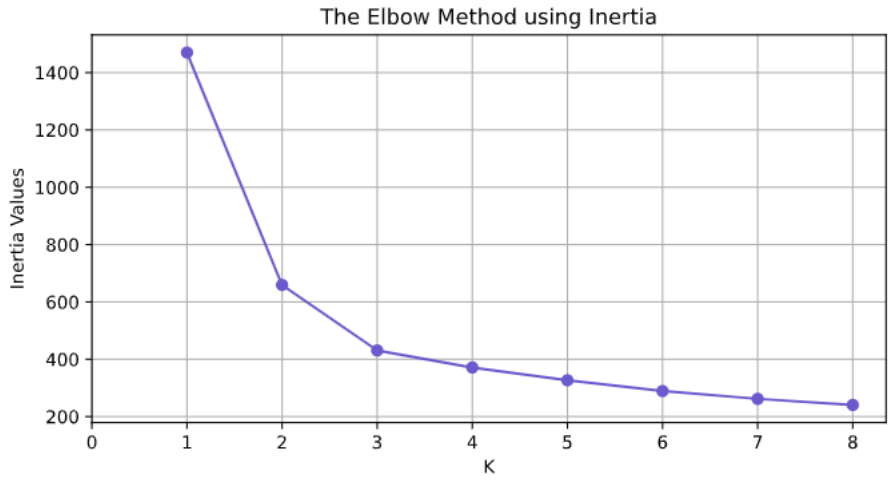
3. implement a polynomial kernel SVM of degree 3. What is your accuracy and confusion matrix?
accuracy: 0.8143,
confusion matrix svm: array([[25, 11],[2, 32]]

Question 2: Pick up any classifier for supervised learning (e.g. kNN, logistic regression, Naive Bayesian, etc).
1. use this classifier to your dataset. What is your accuracy and confusion matrix?
accuracy: 0.9286,
confusion matrix svm: array([[32, 4], [1, 33]])

2. summarize your findings in a table below and discuss your results

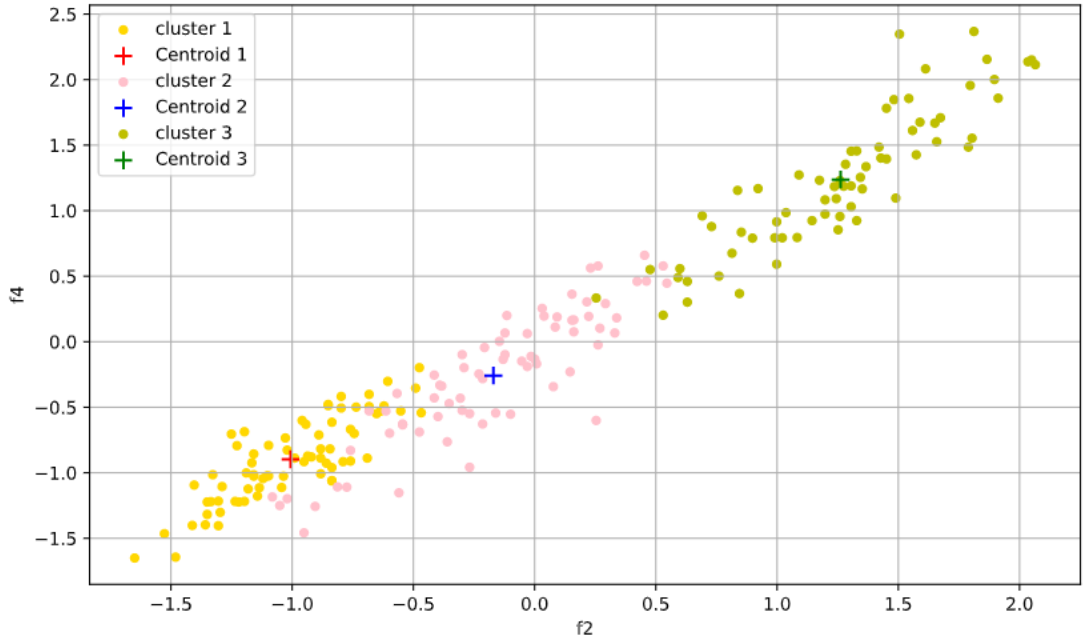
model	TP	FP	TN	FN	accuracy	TPR	TNR
linear kernel SVM	33	4	30	3	0.9	0.916667	0.882353
Gaussian kernel SVM	34	2	32	2	0.942857	0.944444	0.941176
polynomial kernel SVM	25	2	32	11	0.814286	0.694444	0.941176
knn	32	1	33	4	0.928571	0.888889	0.970588

Question 3: Take the original dataset with all 3 class labels.
1. for k = 1, 2, . . . , 8 use k-means clustering with random initialization and defaults. Compute and plot distortion vs k. Use the ”knee” method to find the best k.



Best k is 3

2. re-run your clustering with best k clusters. Pick two features fi and fj at random (using python, of course) and plot your datapoints (different color for each class and centroids) using fi and fj as axis. Examine your plot. Are there any interesting patterns?



Yes, there are three clusters.

3. for each cluster, assign a cluster label based on the majority class of items. For example, if cluster Ci contains 45% of class 1 (”Kama” wheat), 35% of class 2 (”Rosa” wheat) and 20% of class 3 (”Canadian” wheat), then this cluster Ci is assigned label 1. For each cluster, print out its centroid and assigned label.

cluster 1,
label is : 1
centroid is
[-0.14111949 -0.17004259 0.4496064 -0.25781445 0.00164694 -0.66191867 -0.58589311]

cluster 2,
label is : 2
centroid is
[1.25668163 1.26196622 0.56046437 1.23788278 1.16485187 -0.04521936 1.29230787]

cluster 3,
label is : 3
centroid is
 [-1.03025257 -1.00664879 -0.9649051 -0.89768501 -1.08558344 0.69480448 -0.62480856]

4. consider the following multi-label classifier. Take the largest 3 clusters with label 1, 2 and 3 respectively. Let us call these clusters A, B and C. For each of these clusters, you know their means (centroids): μ(A), μ(B) and μ(C). We now consider the following procedure (conceptually analogous to nearest neighbor with k = 1): for every point x in your dataset, assign a label based on the label on the nearest (using Euclidean distance) centroid of A, B or C. In other words, if x is closest to center of cluster A, you assign it label 1. If x is closest to center of cluster B, you assign it class 2. Finally, if x is closest to center of cluster C, you assign it class 3. What is the overall accuracy of this new classifier when applied to the complete data set?

accuracy for k mean is 0.919

5. take this new classifier and consider the same two labels that you used for SVM. What is your accuracy and confusion matrix? How does your new classifier (from task 4) compare with any classifiers listed in the table for question 2 above?

cluster 1
label is : 1
centroid is
 [1.04333519 1.02420543 0.71857247 0.89137341 0.98474652 -0.64824175 0.2394117]

cluster 2
label is : 3
centroid is
 [-0.7165073 -0.70337   -0.49347748 -0.61214801 -0.67627171 0.44517807 -0.16441526]

accuracy is 0.9071
confusion matrix is [[57 13] [0 70]]

	TP	FP	TN	FN	accuracy	TPR	TNR
linear kernel SVM	33	4	30	3	0.9	0.916667	0.882353
Gaussian kernel SVM	34	2	32	2	0.942857	0.944444	0.941176
polynomial kernel SVM	25	2	32	11	0.814286	0.694444	0.941176
knn	32	1	33	4	0.928571	0.888889	0.970588
k mean	70	13	57	0	0.907143	1	0.814286

It`s performance looks present the average of SVMs and Knn.