E0-270: Assignment 1

Due date: March 29, 2023

REPORT

1. Introduction:

1.1. PCA:

Principle Component Analysis is a mathematical method which is used to reduce the dimensionality of a very high dimensional data by preserving as much of the original information as possible.

First, we identify the vector, such that the variance of the projected data (on that vector) is maximum. This vector becomes our first principle component. Then we identify the next vector which is perpendicular to the first component and the variance of the projected data (on this vector) is maximum. This vector becomes our second principal component.

We iteratively do the above process till we get our required number of components and then we finally project our original data on the space of our principle components. Now we use this projected data with reduced dimensions for our further analysis.

1.2. SVM:

Support Vector Machine algorithm finds the best linear separating hyper-plane for a binary class classification problem. For finding this plane, we maximize the margin (min distance of a point from the hyper-plane) such that all points are correctly classified. For solving this problem, we fix the margin to a unit value and optimise on “w” (vector perpendicular to the hyper-plane). This works because we can have different (scaled) values of “w” vector for the same hyper-plane.

Sometimes, there could be cases when solving above optimisation problem would not be possible because such plane may not exist which classifies all the points correctly. To tackle this problem we subtract a positive error term from 1 and now this becomes our new margin.

Now we optimise (minimise) for both the w vector and summation of all error terms.

2. Methodology:

2.1. Task:

- Apply PCA on provided mnist training and testing dataset.
- Use the above transformed data on multi-class SVM algorithm to classify the data points on testing data in one of the 10 digits.
- Plot different performance metrics of multiclass SVM with respect to different number of principle components used for dimension reduction.

2.2. Implemented Ideas:

- Normalised every dimension (i) by using the formula :
  (2*(i-i.min())/(i.max()-i.min())) – 1 .
  And, If i.min() == i.max() then skipped that dimension.
- Implemented binary soft SVM using gradient descend and stochastic descend (used single as well as a batch of data points).
- Implemented PCA by first calculating eigen values and eigen vectors and then choosing principle components (eigen vectors). whose corresponding eigen values were maximum.
- Implemented multiclass SVM by taking transformed data from PCA algorithm as input by training 10 binary SVMs (each for one class).
  In case, more than one binary SVMs are classifying a data point to their respective numbers , used the maximum distance of that point from the hyper plane to resolve the tie.
  In case, none of the binary SVMs are classifying a data point to their respective numbers, used the minimum distance of that point from the hyper plane to resolve the tie.
- Plotted accuracy, recall, precision and f1 score with respect to increase in number of principal components used.

3. Analysis:

3.1. Tie resolution while using Multi-class SVM:

Here we have used soft binary SVMs for multi-class classification. Following is the flow of implemented logic:

- Sent all the data points one by one to each binary SVM (for every digit).
- Now a Binary SVM of a particular digit will say whether a given datapoint is that particular digit or not.
- Now there could be 3 cases:
    a) Only one binary SVM is saying that a given datapoint belongs to its digit. In this case we straight away predicted that particular digit for this datapoint.
    b) More than one SVM are say that it belongs to their respective digits. Now in order to break the tie , we calculated the distance of the point from each hyper-plane. Now larger is the distance , more accurate will that particular SVM be. Hence we predicted that digit whose SVM's plane distance was maximum from that point.
    c) None of the SVM classifies a point to their respective digits. Now here we have to find which SVM has the least accuracy for this point. This could be found by again comparing the distances , which ever has the least distance will be the least confident hence we predicted that SVMs digit whose distance is coming out to be the least.

3.2. Drawbacks:

- Comparing distances is not that efficient because we can't say that in case of tie maximum distance SVM is most confident or minimum distance SVM least confident.
- There is a chance that the actual label is 0 but the SVM of 0 digit didn't classify it correctly. Then we will surely get that point wrong.

3.3. Scope:

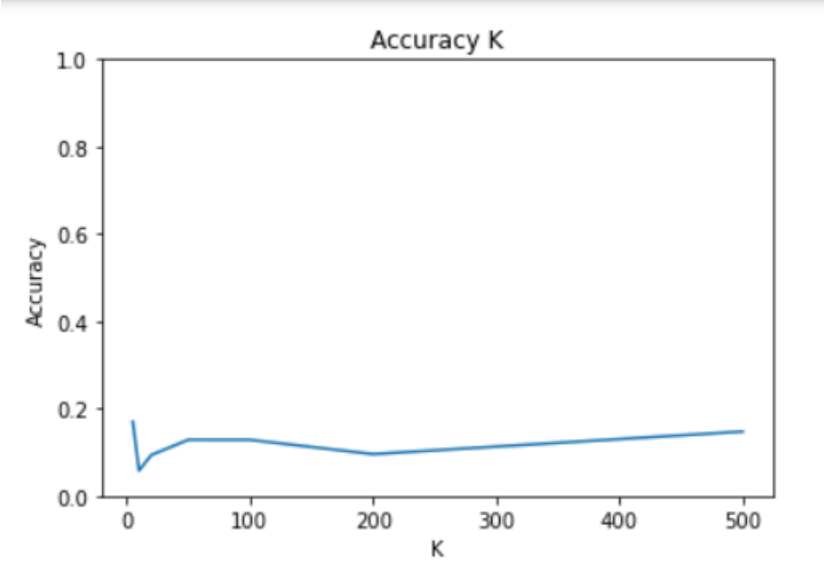
- Instead of using distance, we can use one of the performance metrics (depending on aim) to resolve ties.

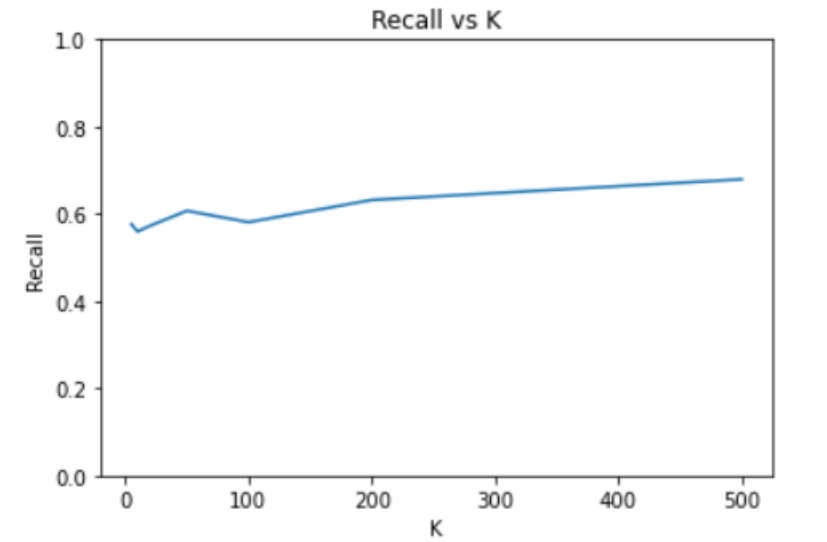
4. Performance Plots:

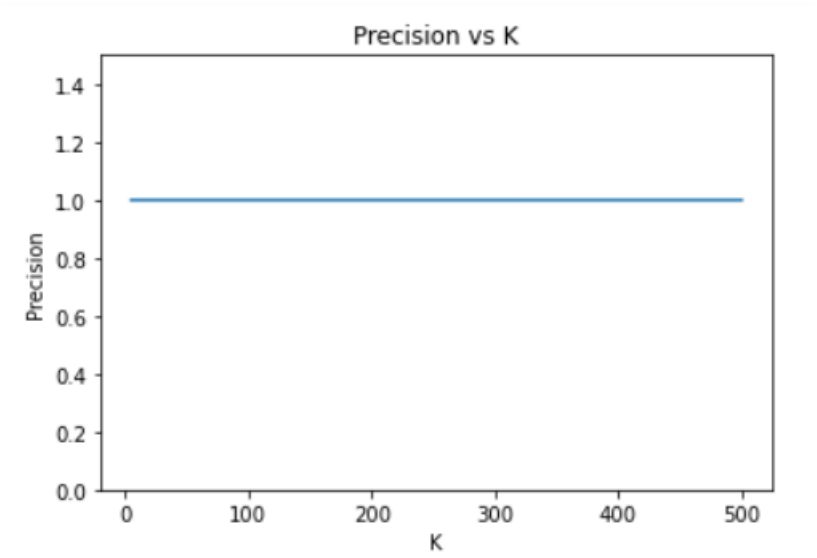
Iterations = 1000

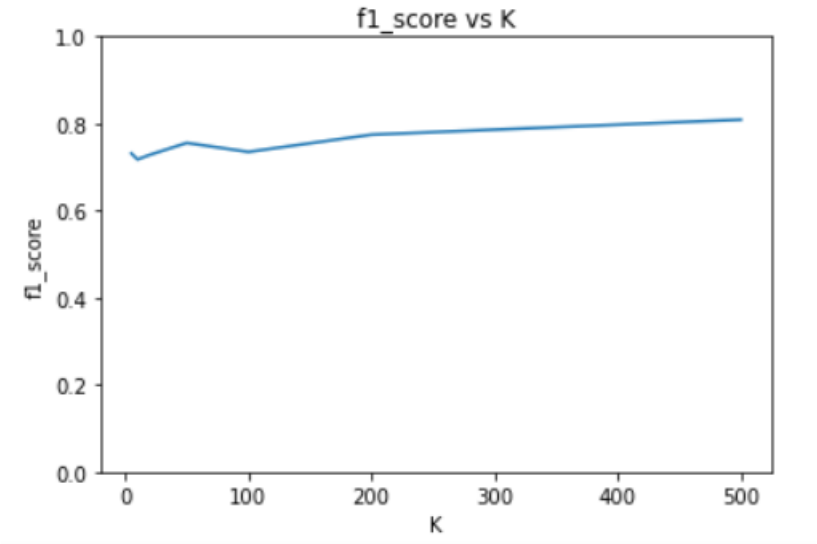
Learning rate = 0.01

C = 10






Precision vs K
Precision
K
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4
0
100
200
300
400
500


f1_score vs K
f1_score
K
0.0
0.2
0.4
0.6
0.8
1.0
0
100
200
300
400
500