Shiraz University

Machine Learning course (spring 1402)

Assignment #5: kernel

Due date: 7 tir

In this assignment, you are going to be familiar with the concept and implementation of different kinds of kernels.

Kernel KNN

Recall that in the 1NN classifier (k = 1), we need to compute the **Euclidean distance** of a **test** instance to all the training instances, find the closest training instance, and find its label. The label of the nest instance will be identical **to its nearest neighbor**. This can be kernelized by observing that:

$$\|x_i - x_{i'}\|_2^2 = < x_i, x_i > + < x_{i'}, x_{i'} > - 2 < x_i, x_{i'} > = K(x_i, x_i) + K(x_{i'}, x_{i'}) - 2K(x_i, x_{i'})$$

This allows us to apply the nearest neighbor classifier to structured data objects. **Implement** the KNN classifier and kernel KNN classifier with Linear, RBF (tune the σ parameter with cross-validation), Polynomial (d = 1), Polynomial (d = 2), and Polynomial (d = 3) kernels. Report the classification **accuracy, precision, recall and F1 measure** for each data set with each classifier and compare the results. Split the data set into 70% and 30% for training and testing parts. You should report the mean of **accuracies and F1s** for **ten** individual runs. Report the running time of your code (in milliseconds) in the second table.

You should report the results of your implementation in the following tables. In other words, please copy the following tables in your report and fill them with your results.

Datasets	Accuracy and F1 of algorithm					
	1NN	1NN+Linear kernel	1NN+RBF kernel	1NN+Polynomial kernel ($d = 1$)	1NN+Polynomial kernel ($d = 2$)	1NN+Polynomial kernel ($d = 3$)
Wine						
Glass						
BreastTissue						
Diabetes						
Sonar						
Ionosphere						

Datasets	Runing Time of Algorithms					
	1NN	1NN+Linear kernel	1NN+RBF kernel	1NN+Polynomial kernel ($d = 1$)	1NN+Polynomial kernel ($d = 2$)	1NN+Polynomial kernel ($d = 3$)
Wine						
Glass						
BreastTissue						
Diabetes						
Sonar						
Ionosphere						

1. Which kernels had the best results? Why?
2. Which kernels had the same performance? Why?

Kernel Kmeans

Kernel K-means is a variant of the traditional K-means clustering algorithm that allows clustering of non-linearly separable data. In traditional K-means, the algorithm assigns each data point to a cluster based on the Euclidean distance in the original feature space. However, this approach may not work well for data that is not linearly separable.

Kernel K-means overcomes this limitation by applying the "kernel trick," which maps the data points into a higher-dimensional feature space where they might become more separable. The kernel trick involves replacing the

dot product between data points with a kernel function that implicitly computes the dot product in the higher-dimensional space.

You should **implement** this part as below:

1. Create a data using make circle and make moons libraries of sklearn in python. Then use it as your datset.

2. Implement the Kmeans and Kernel Kmeans classifier with Linear, RBF (tune the σ parameter with cross-validation), Polynomial (d = 1), Polynomial (d = 2), and Polynomial (d = 3) kernels. Report the **accuracy, precision, recall and F1 measure** measure for each data set and compare the results. Split the data set into 70% and 30% for training and testing parts.

3. Plot the original dataset and each cluster after applying kernel kmeans approach. (Plot each cluster with different colors). Do it for both train and test parts.

4. Is Kernel Kmeans a good method for separating these clusters? Why?

Kernelized PCA

Kernelized PCA (Principal Component Analysis) is an extension of traditional PCA that allows for non-linear dimensionality reduction by using kernel functions. PCA is a widely used technique for linear dimensionality reduction, but it may not be suitable for datasets with non-linear structures.

In kernelized PCA, the data points are implicitly mapped to a high-dimensional feature space using a kernel function. This mapping allows capturing non-linear relationships among the data points, enabling more effective dimensionality reduction. Kernelized PCA aims to find a low-dimensional representation of the data in the feature space while preserving the non-linear structure.

You should **implement** this part as below:

1. Create a data using make circle library of sklearn in python. Then use it as your dataset.

2. Implement the Kernelized PCA classifier with Linear, RBF (tune the σ parameter with cross-validation), Polynomial (d = 1), Polynomial (d = 2), and Polynomial (d = 3) kernels. Split the data set into 70% and 30% for training and testing parts.

3. Plot the original dataset and projection of testing data using Kernel PCA. (Plot it with different colors). Do it for both train and test parts.

4. Explain Why this method works?

Notes:

- Pay extra attention to the due date. It will not extend.
- Be advised that submissions after the deadline will not grade.
- Prepare your entire report in PDF format and include the figures and results.
- Submit your assignment using a zipped file with the name of"StdNum_FirstName_LastName".zip
- Using other students' codes or the codes available on the internet will lead to zero.