Foundations of Data Analysis - SS21
Supervised learning
Lab assignment

Due date: 9:45 am on 29.04.2021

Instructions

Please follow these instructions carefully, so that I can focus on grading and providing helpful feedback. Failing to follow the instructions will result in a penalty and you may be asked to resubmit.

- As reference material, you may only use the sources mentioned in this assignment sheet, the course material, and the documentation for `numpy`, `matplotlib` and `kfda`. Any act of plagiarism will be taken very seriously and handled according to university guidelines.
- The *only* python packages you are allowed to use are `numpy`, `matplotlib` and `kfda`.
- Your code must be well-documented and readable; see the Style Guide for Python Code for the style conventions.
- You are allowed to work on this assignment either individually or in groups. If you work in a group, be sure to include the names of all your group mates in your report. Your report must always be prepared individually!
- It is highly recommended that you typeset your assignment in LaTeX. However, you are also allowed to hand-write your answers and include them in the pdf as photos, or use any other markup language/text formatter, as long as the submission is a pdf.
- Upload your submission to Moodle as exactly *two* files, your python code and comments in the file `<last_name>.py` and your write-up containing the requested figures and question answers in `<last_name>.pdf`, replacing `<last_name>` with your last name(s).

For any questions do not hesitate to email me (Akshey Kumar) at `akshey.kumar@univie.ac.at`, or post in the dedicated lab discussion forum on Moodle.

1 Kernel Fisher discriminant analysis (KFDA)

In the lectures and pen and paper assignment you saw how a linear discriminant can be useful in classifying data. Sometimes however, there exist datasets for which a linear discriminant may be inadequate at describing the class separation. Kernel discriminant analysis generalises the concept of a linear discriminant to a non-linear setting.

The main idea is as follows. While some datasets are not linearly separable in their original feature space, there may exist some (non-linear) mapping to a higher-dimensional space where the data is separable. Once we non-linearly map the samples to this space, we may then apply regular Fisher discriminant analysis (FDA).

Often enough, it may be unfeasible to explicitly map all the samples to a higher-dimensional space, particularly if the space is infinite-dimensional. Here is where the kernel trick comes in (as we will see in later lectures). It utilises the fact that our model requires knowledge only of the inner product in the transformed space and not the original mapping itself. If $\boldsymbol{x_i}$ and $\boldsymbol{x_j}$ are samples in original features space which are mapped to $\boldsymbol{\phi(x_i)}$ and $\boldsymbol{\phi(x_j)}$ in the transformed space, then the Kernel function is defined as,

$$k(\boldsymbol{x_i}, \boldsymbol{x_j}) = \langle \boldsymbol{\phi}(\boldsymbol{x_i}), \boldsymbol{\phi}(\boldsymbol{x_j}) \rangle \tag{1}$$

where $\langle ., . \rangle$ is an inner product in the transformed space. The kernel trick can be used in FDA, since the separation ratio in the transformed space can be expressed in terms of inner products. Hence instead of explicitly transforming our feature space, we can reformulate the FDA problem in terms of the kernel function.

In this assignment you need not implement KFDA but will use a package `kfda`. The focus will be on developing your intuition on how to properly train and evaluate a classifier. In the last two tasks, you will gain some insight as to how a classifier is constructed from the Kernel discriminant.

2 Understanding the dataset (10%)

The given dataset contains two features and two classes. You can find the features stored in `features.csv` and their corresponding labels in `labels.csv`.

1. Load the features and labels in `numpy` arrays. Produce a scatter plot of the samples in their two-dimensional feature space and colour each sample according to their class label.

2. Is the data linearly separable? Can you think of a transformation of the feature space (either linear or non-linear) that renders the data separable?

3 Setting up a classifier (20%)

To implement a kernel Fisher discriminant analysis classifier, you will utilise the package `kfda` that can be found here. It is also available on PyPI and can be installed using `pip install kfda`. You can refer to the documentation while writing your code. Before you begin training the classifier, you will have to prepare the data a bit. Carry out the following tasks to set up a classifier.

1. Split the dataset into a two sets each containing 75% and 25% of the samples, respectively. Your first set is your training set and you will use samples from this set to train your classifier on. The second set shall be kept aside as a testing set and you will use it exclusively for validating your classifier. Before splitting the data into a train and test set, you must ensure that the data is properly shuffled. If this is not the case, the data in your train and test set would be from different data-generating distributions and the *i.i.d.* assumption is violated.

2. For this assignment, you will use a radial basis function (RBF) kernel for the classifier which is defined as,

$$k(\boldsymbol{x_i}, \boldsymbol{x_j}) = \exp\left(-\gamma \|\boldsymbol{x_i} - \boldsymbol{x_j}\|^2\right) \tag{2}$$

   where $\boldsymbol{x_i}$ and $\boldsymbol{x_j}$ are any two samples from the dataset and γ is a paramter. Import `Kfda` from the `kfda` package and instantiate a classifier with `Kfda(kernel='rbf', n_components=1, gamma=0.2)`. This means we choose an RBF Kernel discriminant with $\gamma = 0.2$ that projects the data along a single component.

3. Use the `fit` method to train the classifier on the the samples from the training set. To examine the performance of your classifier, use the `score` method to estimate the score on samples from both the train and test set separately. Are the train and test scores equal? If not, can you give a reason why they differ?

4 Learning curves (25%)

Here, you will investigate the learning curves and choose a suitable value for the Kernel parameter γ while taking care not to overfit or underfit the classifier. These topics will be covered in detail in the lectures on model validation. Nevertheless, you may attempt this section before the lecture — you will be adequately guided here.

1. Evaluate the train and test scores[1] of the classifier as a function of the number of training samples (taken from the training set) and plot the

[1] To get a good estimate of the train score, you must sample a sufficiently high number of times from the training set (with replacement) and find the mean train score for a given number of training samples.

learning curve. Along the $x-$axis, plot the number of training samples and along the $y-$axis plot the train and test scores. Thus you will have two curves per plot. Interpret your observations from the learning curves. (If necessary, you may refer section 11.3 in Shalev-Shwartz's book, *Understanding Machine Learning: From Theory to Algorithms*).

2. Plot one learning curve for each of the six kernel parameters

$$\gamma \in \{0.0001, 0.001, 0.01, 0.1, 0.15, 0.3\} \tag{3}$$

Based on the learning curves, what value of γ would you select for the classification task? Justify your answer and give your interpretation of what is going on in each learning curve.

3. Using your chosen value of parameter γ, determine the labels that your classifier predicts for samples in the following set, $\{(0,1)^T, (2,3)^T, (7,0)^T\}$.

5 In the projected space (25%)

Now that you've found a well-trained classifier, let us examine how it works a bit more closely. To do this, we will investigate our data in the projected space.

1. Project the original features onto the kernel Fisher discriminant by using the `transform` method.

2. Plot a histogram of the projected data with class labels 0 and 1 coloured differently. Now repeat this process with each of the values of γ given in Task 4.2. Explain the effect of varying γ on the histogram and give reasons.

3. Recall the separation ratio from the pen and paper assignment on Fisher discriminant analysis. The discriminant is a result of maximising this ratio. The separation ratio R is defined as the ratio of the between-class variance to the within-class variance.

$$R = \frac{D^2}{S} \tag{4}$$

where D is the distance between the projected means of each class and S is the sum of the variance of each class (of the projected data).

Write a function `separation_ratio(projected_data,labels)` whose parameters are one-dimensional arrays containing the projected data and their labels and returns the separation ratio R for the projected data. Use this function to estimate the separation ratio obtained for each of the values of gamma in Task 4.2 above.

6 The decision boundary (20%)

The aim of this section is to develop an understanding of how a classifier is constructed out of the projected data. For this purpose, we will plot the (non-linear) decision boundary in the original feature space.

1. Estimate each of the class means μ_0 and μ_1 in the projected space and plot their mid-point $\mu = (\mu_0 + \mu_1)/2$ in the histogram plot. This mid-point is the decision boundary of your classifier in projected space. (It makes sense that the decision boundary is a point since the projected space is one-dimensional.) Now that you have the decision boundary in projection space, you can find it in the original feature space. Although an analytic solution may not always be possible, you can numerically estimate the decision boundary as follows.

2. Create a meshgrid of data points over a region of the original feature space. For example, you may choose a square grid with vertices $(2.5, -15.5)$, $(13.5, -15.5)$, $(2.5, -4.5)$, $(13.5, -4.5)$. Project these points along the kernel fisher discriminant using the `transform` method. Now create a 3-d surface plot with the xy-plane[2] as the original feature space and the $z - axis$ as the value in the projected space. What can you infer from the shape of this surface plot? *Hint*: You may find `numpy.meshgrid`, `numpy.reshape` useful for this task.

3. From the grid, find each data point $\boldsymbol{x'}$ whose projected values $q(\boldsymbol{x'})$ lie within a threshold τ of the decision boundary point μ.

$$|q(\boldsymbol{x'}) - \mu| < \tau \tag{5}$$

   For example, you can set the threshold $\tau = 0.01$. Create a scatter plot with the points $\boldsymbol{x'}$ that satisfy the above relation. You may consider the procedure outlined above as a good way to approximate a decision boundary when an analytic solution is not known. Use the same procedure to plot each decision boundary that results from the values of γ in Task 4.2. Observe how the decision boundaries vary for values of γ that lead to overfit and underfit models.

[2] In this task x, y, z refer only to the conventional geometric directions in the 3-d plot and are not to be confused with the samples