Analyse et Traitement de l'Information

TP4: Linear Discriminant Analysis (LDA), k-means and autoregression.

1 Fisher's Discriminant for Two Classes

Assume we are given a *training set* $\{\boldsymbol{x}_i, y_i\}_{i=1}^n$ where each $\boldsymbol{x}_i$ is a high dimensional observation in $\Re^d$ and each $y_i = 0, \ldots, (L-1)$ corresponds to a label specifying which of the L possible classes $\boldsymbol{x}_i$ belongs to. For instance, each $\boldsymbol{x}_i$ may be an image where $y_i = 0, \ldots, (L-1)$ corresponds to the image class. The goal of *classification* is to learn a mapping $f : \boldsymbol{x} \to y$ based on this labeled training set. Predictions can then be made for any future sample $\boldsymbol{x}$ with an unknown label by setting it's predicted label equal to $f(\boldsymbol{x})$.

A simple approach to classification is Fisher's linear discriminant analysis (LDA). The basic idea of LDA is to project $\{\boldsymbol{x}_i\}_{i=1}^n$ into a low dimensional space where different classes are "best" separated. The classification is then performed on this low dimensional space. Here, without loss of generality, we only discuss LDA for two classes, i.e., the number of classes is $L = 2$ where y_i can be either 0 or 1. In this case, the classifier is defined as

$$f(\boldsymbol{x}) = \begin{cases} 0 & \text{if } \boldsymbol{a}^T\boldsymbol{x} + b \geq 0; \\ 1 & \text{if } \boldsymbol{a}^T\boldsymbol{x} + b < 0, \end{cases} \tag{1}$$

where $\boldsymbol{a} \in \Re^d$ specifies a linear projection from $\Re^d$ to $\Re$ and $b \in \Re$ is a threshold to determine which of the two classes $\boldsymbol{x}$ belongs to. The learning task is to find the optimal $\boldsymbol{a}$ and b based on a given training set $\{\boldsymbol{x}_i, y_i\}_{i=1}^n$.

Rather than provide the complete derivation, we directly provide the solution of the optimal $\boldsymbol{a}$ and b and refer the reader to the textbooks [1, 2] for the detailed analysis. The optimal $\boldsymbol{a}$ with respect to $\{\boldsymbol{x}_i, y_i\}_{i=1}^n$ is given by

$$\boldsymbol{a} = S_W^{-1}(\boldsymbol{g}_0 - \boldsymbol{g}_1). \tag{2}$$

In eq.(2),

$$\boldsymbol{g}_0 = \frac{1}{N_0}\sum_{i:y_i=0} \boldsymbol{x}_i, \quad \boldsymbol{g}_1 = \frac{1}{N_1}\sum_{i:y_i=1} \boldsymbol{x}_i,$$

where N_0 and N_1 are the number of samples in class 0 and 1, respectively, and $\boldsymbol{g}_0$ and $\boldsymbol{g}_1$ are the corresponding mean vectors. S_W is a $d \times d$ *within-class scatter matrix* defined as

$$S_W = \sum_{i:y_i=0} (\boldsymbol{x}_i - \boldsymbol{g}_0)(\boldsymbol{x}_i - \boldsymbol{g}_0)^T + \sum_{i:y_i=1} (\boldsymbol{x}_i - \boldsymbol{g}_1)(\boldsymbol{x}_i - \boldsymbol{g}_1)^T.$$

Since the given classes of data are balanced, the optimal b with respect to $\{\boldsymbol{x}_i, y_i\}_{i=1}^n$ is given by

$$b = -\frac{1}{2}\boldsymbol{a}^T(\boldsymbol{g}_0 + \boldsymbol{g}_1). \tag{3}$$

Consider the binary learning problem to classify images of "3" and "7". From the MNIST dataset, sample 2000 instances of "3"s and "7"s for training and use all instances for testing. Apply PCA to reduce the dimensionality of the input data to 50 before classification. Then perform the following tasks.

1. Apply LDA to classify the complete set of testing instances. Report the accuracy using the confusion matrix[1].
2. Use the k-NN method that you implemented in TP3 to classify the testing instances with $k = 1$. Report the accuracy using the confusion matrix.
3. Describe how the two classification methods compare.

2 k-means Clustering

Classification is a *supervised* problem, where labeled (training) data is used to label the unlabeled (testing) data.
Clustering is an *unsupervised* problem, where the objective is to group a given (unlabeled) dataset $\{\boldsymbol{x}_i\}_{i=1}^n$ into k clusters, so that similar samples appear in the same cluster, and dissimilar samples are in different clusters.

The most basic clustering algorithm, k-means, aims to partition the input data into k clusters in the most compact way, in the sense that the following objective function is minimized

$$J(\boldsymbol{y}) = \sum_{j=1}^{k} \sum_{i:y_i=j} ||\boldsymbol{x}_i - \boldsymbol{g}_j||^2, \tag{4}$$

where

- $\boldsymbol{y} = (y_1, \dots, y_n)$ represents a clustering scheme. Each y_i is an integer taking values from $\{1, 2, \dots, k\}$, indicating which cluster the corresponding sample $\boldsymbol{x}_i$ belongs to;
- $\boldsymbol{g}_j$ is the center of cluster j so that

$$\boldsymbol{g}_j = \frac{1}{N_j} \sum_{i:y_i=j} \boldsymbol{x}_i \quad (N_j \text{ is the number of samples in cluster } j)$$

Therefore, $J(\boldsymbol{y})$ means the sum of the square distances from each sample to its associated cluster center. Minimizing $J(\boldsymbol{y})$ means to take each cluster *compact*, in the sense that a sample should be near to its cluster center. To minimize this $J(\boldsymbol{y})$, the k-means algorithm is given as follows

- Randomly sample 2000 instances "3"s and "7"s from the MNIST dataset. Perform k-means with $k = 2, 3, \dots, 10$. Compute J defined in eq.(4) for each k. Plot J varying with k. Describe the effect of k over J.

[1] https://en.wikipedia.org/wiki/Confusion_matrix

ici on peut tout importer d'un coup

ou utiliser k means from library

```
1: procedure K-MEANS($\{\boldsymbol{x}_i\}_{i=1}^n, k$)  ▷ As inputs: a set of samples and the number of clusters
2:     ▷ As outputs: $\{y_i\}_{i=1}^n$. $y_i \in \{1, 2, ..., k\}$ specifies $\boldsymbol{x}_i$ belongs to which of the $k$ clusters.
3: Randomly initialize each $y_i^0$ to one of $\{1, 2, ..., k\}$; t=0;
4: repeat
5:     for $j \leftarrow 1 \rightarrow k$ do    ▷ Compute the centroid of each cluster
6:         $\boldsymbol{g}_j \leftarrow \sum_{i:y_i^t=j} \boldsymbol{x_i}/N_j$;
7:     for $i \leftarrow 1 \rightarrow n$ do    ▷ Assign each point to the closest centroid
8:         $y_i^{t+1} \leftarrow \text{argmin}_j ||\boldsymbol{x}_i - \boldsymbol{g}_j||$;
9:     $t \leftarrow t + 1$;
10: until $\forall i, y_i^t = y_i^{t-1}$
11: Output $\{y_i\}_{i=1}^n$
```

- For the case $k = 2$ in the above experiment, give the confusion matrix.
- Sample 2000 instances of "3"s and "5"s (rather than "3"s and "7"s). Perform k-means with $k = 2$. Build the confusion matrix. Based on your results, explain the difference between these two clustering problems: (1) "3" vs. "7"; (2) "3" vs. "5".

3 Time series prediction *

You will find two `csv` data files in the `data` folder:

- `data_mintemp.csv` contains daily minimum temperature data with two columns: `"Date"` and `"Temp"`.
- `data_cons.csv` contains daily electrical consumption, wind and solar electricity production with three columns: `"Date"`, `"Consumption"`, `"Wind+Solar"`.

Questions

1. Import the data contained in these files and create three datasets with the format $[(t, \mathbf{x}_t)]_{t \in N_t}$ where t represents all possible times for the dataset (`"Date"`), and $\mathbf{x}_t$ could be either the corresponding `"Temp"`, `"Consumption"`, or `"Wind+Solar"`.

   Hint: you could use `pandas.read_csv` with the right parameters for `path`, `header`, `index_col`, and `usecols`.

2. For each of these three datasets, perform the following tasks (you could create a function for these tasks and apply it on each dataset):

   (a) For $k \in \{7, 30, 365\}$, create a new feature of the dataset named `"SMA_k"` which corresponds to the simple moving average with a window of size k defined as the simple mean of the k previous values:

   $$(\texttt{SMA\_k})_t = \frac{\sum_{i=0}^{k-1} \mathbf{x}_{t-i}}{k}$$

   The dataset will now have 5 columns: `"Date"`, `"value"`, `"SMA_7"`, `"SMA_30"`, and `"SMA_365"`.

(b) Plot the data along with their three simple moving averages and give an interpretation.

(c) Create a new feature of the dataset named `"value-365"` (`"value"` being the correct considered data) which corresponds to the serie $[\mathbf{y}_t] = [\mathbf{x}_t - (\texttt{SMA\_365})_t]_{t \in N_t}$

(d) Create a new feature of the dataset named `"SMA_30(value-365)"`, which corresponds to the simple moving average of the previous serie `"value-365"`, with a window of size 30, this new serie is $[\mathbf{z}_t]_{t \in N_t}$.

(e) Create a new feature of the dataset named `"value-365-30"` (`"value"` being the correct considered data) which corresponds to the serie $[\mathbf{y}_t - \mathbf{z}_t]_{t \in N_t}$. The dataset will now have 8 columns.

(f) Plot in the same figure the series `"value"`, `"SMA_365"`, `"SMA_30(value-365)"`, and `"value-365-30"`. Explain the mathematical link between these series and give a clear interpretation to them (you should use the same terminology as in slides ATI.06).

3. For the first dataset only, which contains the daily minimum temperature data, with only the original two columns, perform the following tasks:

   (a) For $k \in \{365, 182, 91\}$, create a new feature column which is the serie of temperature lagged by k days. You could use the pandas function `shift`, and get the autocorrelation $Cor(\mathbf{x}_t, \mathbf{x}_{t-k})$ between the original and lagged series.

   (b) On the same figure, plot the tree previous autocorrelation points and the global autocorrelation curve (for any possible k): the x-axis corresponds to k and the y-axis to the correlation. You could use the `pandas.plotting.autocorrelation_plot` function.

   (c) Give an interpretation to this figure.

4. Define a function which takes as arguments ($\mathbf{y}_{t-1}$, $\mathbf{a}$, σ), and returns the predicted value y_t (scalar) defined in slide 17 of ATI.6 where:

   - $\mathbf{y}_{t-1}$ is a vector of length d: the d previous values of the serie,
   - $\mathbf{a}$ is a vector of length d: the Yule-Walker coefficients of deepness d,
   - σ is a scalar: the constant in the autoregressive formula.

5. Define a function which takes as arguments (**time_serie**, *deep*, *n_pred*, *n_train*) and returns the predicted time serie (of length *n_pred*), where:

   - **time_serie** is a time serie (i.e. a pandas dataframe or serie) of length more than $n\_train + n\_pred$,
   - *deep* is the deepness value for autoregression formula and for the Yule-Walker coefficients (it must be smaller than *n_train*),
   - *n_pred* is the number of values of the serie to predict: with indexes $n\_train \rightarrow n\_train + n\_pred - 1$,

- n_train is the number of the first values of the serie to use for the training: with indexes $0 \rightarrow n\_train - 1$.

. You can use the previous function and the function `statsmodels.api.regression.yule_walker`.

6. For the first dataset only, which contains the daily minimum temperature data, for each $deep \in \{7, 30, 365\}$ predict the last year of data ($length(\textbf{time\_serie}) - 365$ and $n\_pred = 365$) and find the Mean Square Error (MSE) with the real values of the last year of the serie. Comment on you results.

Assessment

Please archive your report and codes in "Prénom Nom.zip" (replace "Prénom" and "Nom" with your real name), and upload to "`Upload TPs/TP4 LDA, k-means and AR`" on `https://moodle.unige.ch` before Monday, November 9 2020, 23:59 PM. Note, the assessment is mainly based on your report, which should include your answers to all questions and the experimental results.

4 Supplements

1. Explain the k-means algorithm.
2. Explain LDA.

References

[1] Christopher M.Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NY, USA, 2006.

[2] Kevin P Murphy. *Machine learning: a probabilistic perspective*. MIT Press, Cambridge, MA, 2012.