STA 5106 Computational Methods in Statistics I

Department of Statistics

Florida State University

Class 13

October 8, 2019

Midterm

- **Midterm Project: (20 points)**

  Out: Thursday, 10/10

- **Project Report: (15 points)**

  Due: Thursday, 10/24, in class

- **Project Presentation: (5 points)**
  - 5-7 minutes for each presentation.
  - Presentation Style: Slide presentation (PPT, PDF, etc).

- **Presentation is required for PhD students in Statistics**
  - **Presenters: report (15%), presentation (5%)**
  - **Non-presenters: report (20%)**

Presenters

- **Presenters (PhD in Statistics):**
  Tingan Chen, Tianyuan Cheng, Harshita Dogra,
  Ke Han, Shuai Hao, Hanwen Hu,
  Taka Iguchi, Seyedkamyar Kazemi, Rufeng Liu,
  Pengfei Lyu, Xiaoxiao Ma, Yijia Ma,
  Sayantika Nag, Jario Pena Hidalgo, Sudipto Saha,
  Changhee Suh, Michael Wilson, Ka Chun Wong,
  Tao Xu, Zhou Xinyu

- This project is **optional** for all other students. **Email me if you choose to do it by tomorrow noon**.

Chapter 9

Statistical Pattern Recognition

9.2 Classification

Classification

- Consider the problem of classifying an observation $X \in \mathbf{R}^n$ into one of the following classes: $C_1, C_2, \ldots, C_k$.
- One can view it as the problem of (disjointly) partitioning the observation space $\mathbf{R}^n$ into k regions; if an observation falls into region i, we declare it to be of class C_i.
- Another viewpoint is that we assume there are k probability models, $f_1, f_2, \ldots, f_k$, associated with the different classes. The observation X is assumed to be a random sample from one of these probabilities.
- The problem of classification designs the partitions from the past data, so that future data can be classified automatically.

Feature Vector

- Since n is often too large in practice, it is practically impossible to analyze the observations directly.
- In most cases, one uses a mapping from $\mathbf{R}^n$ to $\mathbf{R}^d$ for a $d << n$ to map observations into a more manageable space.
- Let g: $\mathbf{R}^n \rightarrow \mathbf{R}^d$ be such a map. For $X \in \mathbf{R}^n$, the vector $x = g(X)$ is called a **feature vector** or a representation of X.
- One can broadly classify such dimension-reducing maps into two categories:

Dimension Reduction

- **1. Linear Dimension Reduction:** Let $U \in \mathbf{R}^{n \times d}$ be an orthogonal matrix, i.e. $U^T U = I_d$, and define $g(X) = U^T X$. (note: $UU^T \neq I_n$)
- An example of this linear projection is U obtained through principal component analysis (PCA).
- **2. Nonlinear Dimension Reduction:** In case the mapping g from $\mathbf{R}^n$ to $\mathbf{R}^d$ is not linear, it cannot be stated in the matrix form as earlier.
- An example of this nonlinear projection is locally linear embedding (LLE).

Feature Space

- The space of all feature vectors is called the **feature space**. This is the space in which a statistical analysis is performed to obtain pattern classification.

- Metric methods are often used to perform classification. In case the feature space is Euclidean, $\mathbf{R}^d$, one can use the *p*-norm:

$$\| x_1 - x_2 \| = \left(\sum_{l=1}^{d} | x_1(l) - x_2(l) |^p \right)^{1/p}$$

- $p = 2$ provides the usual **Euclidean** norm and is used most often.

- For $p = 1$, the resulting metric is called **Manhattan** metric.

Training Data

- **Training Data:** We will assume that we have some number of observations available for each class.
- Let $X_i^j \in \mathbf{R}^n$ denote the *j*-th observation for *i*-th class, with $j = 1, 2, \ldots, n_i$ and $i = 1, 2, \ldots, k$.
- This data is mostly labeled, i.e. with each observation we are provided the class to which it belongs.
- The primary use of this data is to develop or derive classifiers, that can then be used in future classifications.

Test Data

- **Test Data:** Let $Y \in \mathbf{R}^n$ denote a new observation that we wish to classify. In general we are not given the underlying (true) class of Y, i.e. the test data is unlabelled.
- The test data serves as a set of future observations to which a classifier is applied and tested.
- This data is not used in the development of a classifier but is often used in evaluation of a classifier.
- One applies a classifier to the test data and measures the classification performance to evaluate that classifier.

Formal Definition

- **Definition 29** A classifier is a mapping from the feature space to the set of all classes. That is,

$$\varphi : \mathbf{R}^{\mathrm{d}} \rightarrow \{C_1, C_2, \ldots, C_k\}$$

is a classifier. For any test observation Y, and its feature vector $y = g(Y)$, if $\varphi(y) = C_i$, then Y is designated to the i-th class.

- There is a large variety of classifiers that used in applications, but they can be divided into two broad categories:

1. Metric-Based Classifiers.
2. Probability-Based Classifiers.

1. Metric-Based Classifiers

- **Metric-Based Classifiers** assume that there exists a metric in the feature space to quantify difference between any two elements of $\mathbf{R}^d$.

- Given such a metric, called it $d(\cdot, \cdot)$, one can define a number of classifiers.

- Examples:

  Nearest Neighbor (NN) Classifier

  k-Nearest Neighbor (kNN) Classifier

Nearest Neighbor (NN) Classifier

- In this classifier, one computes the distance between $y = g(Y)$ and all possible training vectors $g(X_i^j)$, and assigns that class to Y that has the nearest element to y. That is,

$$\hat{i} = \arg\min_{1 \le i \le k} (\min_j d(g(Y), g(X_i^j)))$$

- Disadvantages:
  - It is slow since it requires computing distances between the test data and all the training data.
  - It is vulnerable to noise. If the noise moves a training vector from the wrong class, then the classification will be wrong.

k-Nearest Neighbor (kNN) Classifier

- ***k*-Nearest Neighbor (kNN) Classifier:** In order to make the classifier more immune to observation noise, one uses the *k*-nearest neighbor classifier.

- Instead of the nearest training data, here one finds *k* nearest training data vectors, and uses the majority class amongst them to classify *Y*.

2. Probability Based Classification

- Classification based on the largest probability.

- **Maximum-Likelihood Classifier:** Classification based on the *largest likelihood*.

$$\hat{i} = \arg\max_i (f(x \mid C_i))$$

Example: Assume a two class classification problem. For each class $f(x|C_i)$ is multivariate normal.

- **Bayesian Classifier:** Classification based on the *largest posterior probability* with given prior information.

$$\hat{i} = \arg\max_i (f(C_i \mid x)) = \arg\max_i (f(x \mid C_i) f(C_i))$$