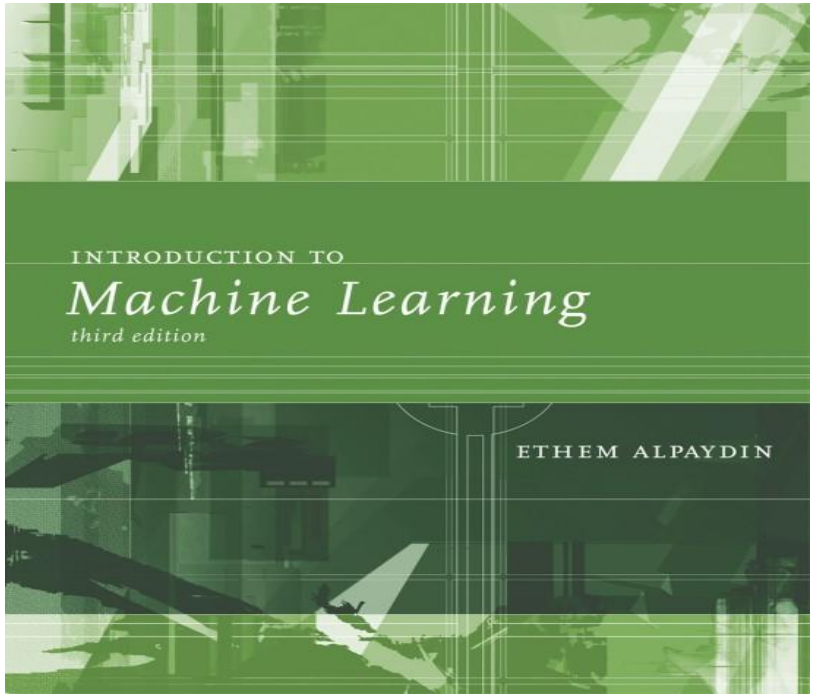
Unit 5 - Dimensionality Reduction, Mining different types of data

- **Dimensionality Reduction - Introduction, Subset Selection, PCA (Principal Component Analysis) – Technique, Examples.**

- Mining different types of data: Mining the World Wide Web - Page Rank 41 Algorithm, Text mining, Mining Time Series Data, Ensemble methods-Increasing the Accuracy.

Text Books

- Ethem Alpaydin, “Introduction to Machine Learning”, Second Edition, MIT Press, Prentice Hall of India (PHI) Learning Pvt. Ltd. 2010



Dimensionality Reduction

Topics

- **Introduction**
- Subset Selection
- PCA (Principal Component Analysis) Technique and Examples

Introduction

- **Dimensionality Reduction** is the task of reducing the number of inputs; this can reduce noise and improve the performance of machine learning algorithms.
- **Dimensionality Reduction Technique - Principal Component Analysis**, an algorithm for realigning our data in the direction of the most variance.
- **Dimensionality Reduction Technique - Singular Value Decomposition**, is a matrix factorization technique, use to approximate original data and thereby reduce its dimensionality.

Introduction

- **Example Cricket Match**
  - need to follow the position of the ball on the playing field.
  - Behind the scene, we are converting the million pixels on the monitor into a three-dimensional image showing the ball's position on the playing field, in real time.
  - Reduced the data from one million dimensions to three dimensions.
  - In this sports match example, we are presented with millions of pixels, but it's the ball's three-dimensional position that's important - Dimensionality Reduction.

Introduction

- The complexity of any classifier or regressor depends on the number of inputs.
- This determines both the **time and space complexity** and the necessary number of training examples to train such a classifier or regressor.
- In this chapter, we discuss feature selection methods that choose a subset of important features pruning the rest and feature extraction methods that form fewer, new features from the original inputs

Introduction

- In classification or regression, observation data contain information are taken as inputs and fed to the system for decision making.
- We should not need feature selection or extraction as a separate process.
- The classifier (or regressor) should be able to use whichever features are necessary, discarding the irrelevant.

Introduction

- Several reasons why we are interested in reducing dimensionality as a separate preprocessing step:
  - In most learning algorithms, the complexity depends on the number of input dimensions, *d*, as well as on the size of the data sample, *N*, and for reduced memory and computation, we are interested in reducing the dimensionality of the problem. Decreasing *d* also decreases the complexity of the inference algorithm during testing.
  - When an input is decided to be unnecessary, we save the cost of extracting it.
  - Simpler models are more robust on small datasets. Simpler models have less variance, that is, they vary less depending on the particulars of a sample, including noise, outliers, and so forth.
  - When data can be explained with fewer features, we get a better idea about the process that underlies the data and this allows knowledge extraction.
  - When data can be represented in a few dimensions without loss of information, it can be plotted and analyzed visually for structure and outliers.

Introduction

- There are two main methods for reducing dimensionality:
  - Feature Selection
  - Feature Extraction
- In ***Feature Selection***, we are interested in finding k of the d dimensions that give us the most information and we discard the other $(d - k)$ dimensions.
- ***Subset Selection*** is a feature selection method.

Introduction

- In ***Feature Extraction***, we are interested in finding a new set of *k* dimensions that are combinations of the original *d* dimensions.
  - These methods may be supervised or unsupervised depending on whether or not they use the output information.
  - ***Principal Components Analysis* (PCA)** and *Linear Discriminant Analysis* (LDA), which are both linear projection methods, unsupervised and supervised respectively.
  - PCA bears much similarity to two other unsupervised linear projection methods, namely, *Factor Analysis* (FA) and *Multidimensional Scaling* (MDS).
  - As examples of *nonlinear* dimensionality reduction - *Isometric feature mapping* (Isomap) and *Locally Linear Embedding* (LLE).

Why to reduce dimensionality?

- Reducing data from more than one million values to the three relevant values.
- Much easier to work with data in fewer dimensions.
- The relevant features may not be explicitly presented in the data. Often, we have to identify the relevant features before we can begin to apply other machine learning algorithms.
- In dimensionality reduction, first preprocess the data. After preprocessing the data, can proceed with other machine learning techniques.

Why to reduce dimensionality?

- Displaying data isn’t the only problem with having a large number of features.
- A short list of other reasons we want to simplify our data includes the following:
  - Making the dataset easier to use
  - Reducing computational cost of many algorithms
  - Removing noise
  - Making the results easier to understand

Topics

- Introduction
- **Subset Selection**
- PCA (Principal Component Analysis) Technique and Examples

Subset Selection

- In subset selection, we are interested in finding the best subset of the set of features.
- The best subset contains the least number of dimensions that most contribute to accuracy.
- We discard the remaining, unimportant dimensions.
- Use a suitable error function
- Used in both regression and classification problems.
- There are 2^d possible subsets of d variables, but we cannot test for all of them unless d is small and we employ heuristics to get a reasonable **(but not optimal)** solution in reasonable **(polynomial)** time.

Subset Selection

- There are two approaches: *forward selection and backward selection.*
- In ***Forward Selection***, we start with no variables and add them one by one, at each step adding the one that decreases the error the most, until any further addition does not decrease the error.
- In ***Backward Selection***, we start with all variables and remove them one by one, at each step removing the one that decreases the error the most, until any further removal increases the error significantly.
- In **either case**, checking the error should be done on a validation set distinct from the training set because we want to test the generalization accuracy.
- With more features, generally we have lower training error, but not necessarily lower validation error.

Subset Selection

Let us denote by F, a feature set of input dimensions, $x_i, i = 1, \ldots, d$. $E(F)$ denotes the error incurred on the validation sample when only the inputs in F are used. Depending on the application, the error is either the mean square error or misclassification error.

In *sequential forward selection*, we start with no features: $F = \emptyset$. At each step, for all possible x_i, we train our model on the training set and calculate $E(F \cup x_i)$ on the validation set. Then, we choose that input x_j that causes the least error

$$j = \arg\min_i E(F \cup x_i)$$

and we

add x_j to F if $E(F \cup x_j) < E(F)$

We stop if adding any feature does not decrease E. We may even decide to stop earlier if the decrease in error is too small, where there is a user-defined threshold that depends on the application constraints, trading off the importance of error and complexity. Adding another feature introduces the cost of observing the feature, as well as making the classifier/regressor more complex.

Subset Selection

This process may be costly because to decrease the dimensions from d to k, we need to train and test the system $d+(d-1)+(d-2)+\cdots+(d-k)$ times, which is $\mathcal{O}(d^2)$.

Subset Selection

- This is a **Local Search** procedure and does not guarantee finding the optimal subset, namely, the minimal subset causing the smallest error.
- For example, xi and xj by themselves may not be good but together may decrease the error a lot, but because this algorithm is greedy and adds attributes one by one, it may not be able to detect this. It is possible to generalize and add multiple features at a time, instead of a single one, at the expense of more computation.
- We can also backtrack and check which previously added feature can be removed after a current addition, thereby increasing the search space, but this increases floating search complexity.
- In **Floating Search** methods, the number of added features and removed features can also change at each step.

Subset Selection

In *sequential backward selection*, we start with F containing all features and do a similar process except that we remove one attribute from F as opposed to adding to it, and we remove the one that causes the least error

$$j = \arg\min_i E(F - x_i)$$

and we

remove x_j from F if $E(F - x_j) < E(F)$

We stop if removing a feature does not decrease the error. To decrease complexity, we may decide to remove a feature if its removal causes only a slight increase in error.

- Subset selection is supervised in that outputs are used by the regressor or classifier to calculate the error.

Topics

- Introduction
- Subset Selection
- **PCA (Principal Component Analysis) Technique and Examples**

Principal Component Analysis

- There are dimensionality reduction techniques that work on labeled and unlabeled data.
- Here focus on unlabeled data because it’s applicable to both types.
- The first method for dimensionality reduction is called principal component analysis (PCA).
  - The dataset is transformed from its original coordinate system to a new coordinate system.
  - The new coordinate system is chosen by the data itself.
  - The first new axis is chosen in the direction of the most variance in the data.
  - The second axis is orthogonal to the first axis and in the direction of an orthogonal axis with the largest variance.
  - This procedure is repeated for as many features as in the original data.
  - The majority of the variance is contained in the first few axes.
  - Therefore, ignore the rest of the axes, and reduce the dimensionality of the data.

Principal Component Analysis

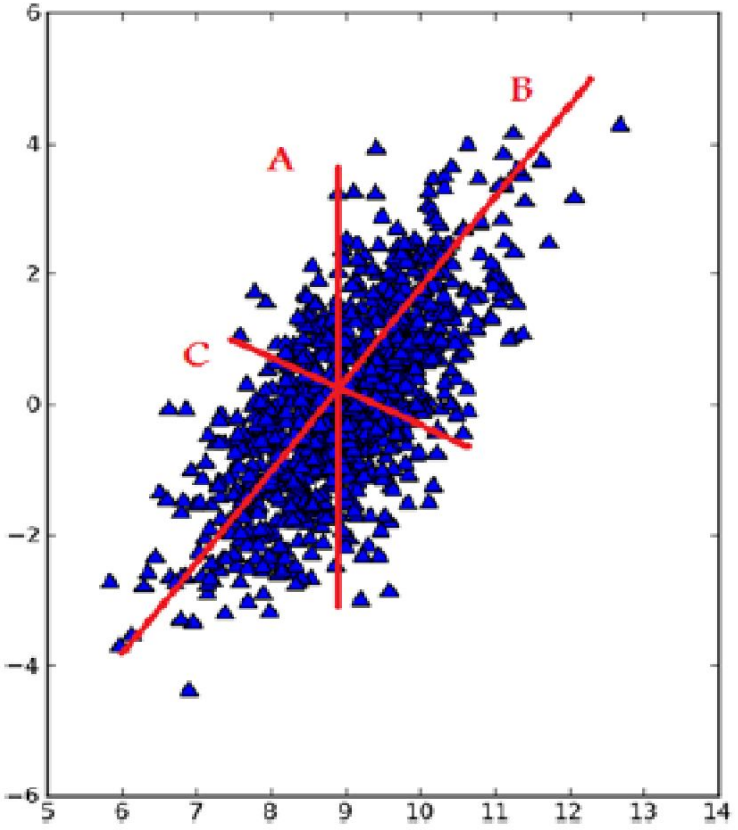


Figure 13.1 Three choices for lines that span the entire dataset. Line B is the longest and accounts for the most variability in the dataset.

- Line B - The largest variation - a line covering the data points, the longest possible line could draw.
- After choosing the axis covering the most variability, choose the next axis, which has the second most variability, provided it's perpendicular to the first axis.
- line C would be our second axis. With PCA, we're rotating the axes so that they're lined up with the most important directions from the data's perspective.

Principal Component Analysis

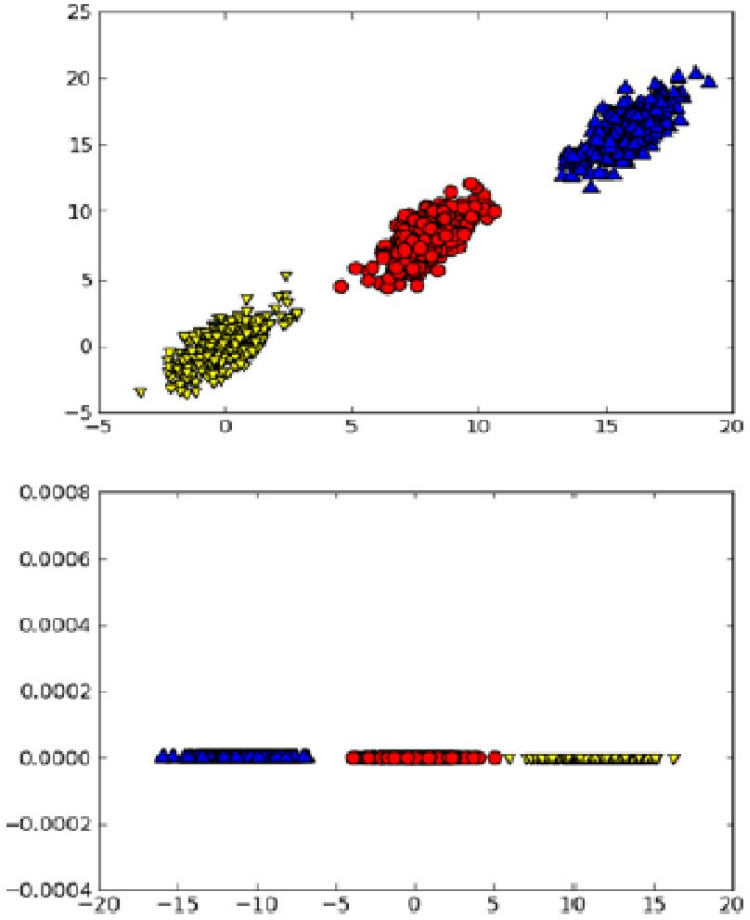


Figure 13.2 **Three classes in two dimensions. When the PCA is applied to this dataset, we can throw out one dimension, and the classification problem becomes easier.**

- To separate the classes - decision tree. Remember that decision trees make a decision based on one feature at a time.
- SVM could get better separation of the classes with a hyperplane.
- The SVM may give us better margin than the decision tree, but the hyperplane is harder to interpret.
- By doing dimensionality reduction with PCA on the dataset, we can have the best of both worlds: we can have a classifier as simple as a decision tree, while having margin as good as the support vector machine.

Principal Component Analysis

- In figure 13.2, we have only one axis because the other axis was just noise and didn’t contribute to the separation of the classes.
- This may seem trivial in two dimensions, but it can make a big difference when we have more dimensions.
- We take the first PC to be in the direction of the largest variability of the data.
- The second PC will be in the direction of the second largest variability, in a direction orthogonal to the first PC.
- We can get these values by taking the covariance matrix of the dataset and doing eigenvalue analysis on the covariance matrix.

Principal Component Analysis

- Once we have the eigenvectors of the covariance matrix, we can take the top N eigenvectors.
- The top N eigenvectors will give us the true structure of the N most important features.
- We can then multiply the data by the top N eigenvectors to transform our data into the new space.

Principal Component Analysis

Pros: Reduces complexity of data, indentifies most important features

Cons: May not be needed, could throw away useful information

Works with: Numerical values

Pseudocode for transforming out data into the top N principal components

- *Compute the mean*
- *Compute the covariance matrix*
- *Find the eigenvalues and eigenvectors of the covariance matrix*
- *Sort the eigenvalues from largest to smallest*
- *Take the top N eigenvectors*
- *Transform the data into the new space created by the top N eigenvectors*

Summary

- Dimensionality reduction techniques allow us to make data easier to use and often remove noise to make other machine learning tasks more accurate.
- It’s often a preprocessing step that can be done to clean up data before applying it to some other algorithm.
- The most widely used method is principal component analysis.
- Principal component analysis allows the data to identify the important features. It does this by rotating the axes to align with the largest variance in the data. Other axes are chosen orthogonal to the first axis in the direction of largest variance.

Summary

- Eigenvalue analysis on the covariance matrix can be used to give us a set of orthogonal axes.
- The PCA algorithm loads the entire dataset into memory.