KNN Algorithm

Optimization, feature weighting

Instructed by Dr.Azimifar

Reza Tahmasebi

saeed77t@gmail.com, Stu ID: 40160957

Table of Contents

1 What are the methods that take care of storage taking in KNN?

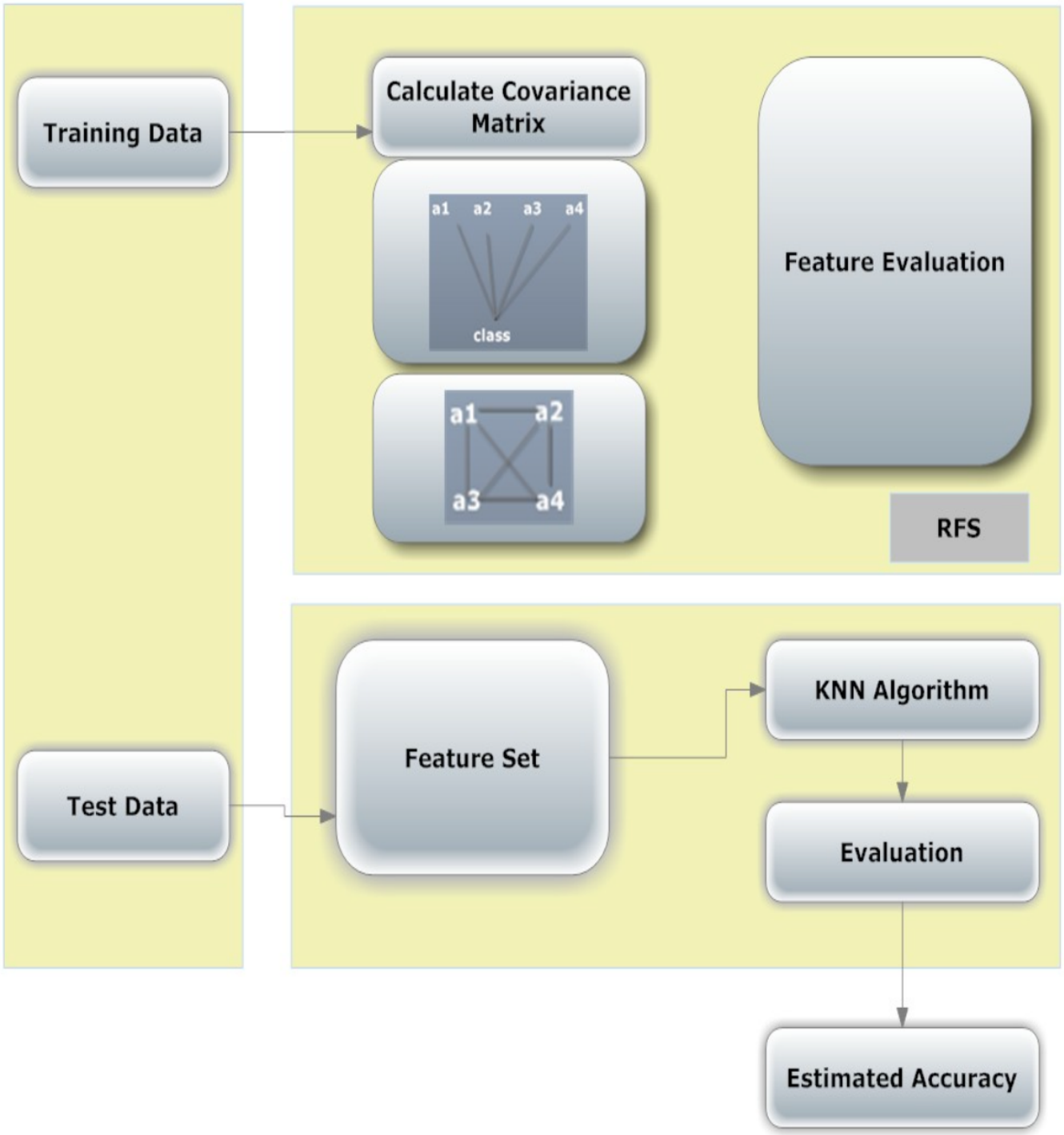
The classification task in data mining is commonly referred to as supervised learning, in which a specified set of classes are known, and training sample objects are assigned with the appropriate type. The classification technique aims to build a model with the best generalization capability. For the supervised learning task, here represent data as a table of training samples, also known as instances, described by features and their classes. Features are also called attributes. Generally, we require two sets of data tables, a training dataset, and a test dataset. We train the classification algorithm on the training dataset and test the algorithm using the test dataset. Decision Tree, Naïve Bayes, and Nearest Neighbor are various classification techniques. K- Nearest Neighbor is one of the simple and well-known classification techniques in which distance is measured between the input point and all other records of the dataset. The K-Nearest Neighbor's class label is the input point's class label.

It is known that if too much irrelevant information is present in the training or test data, the learning and prediction become more complex and inaccurate. The process of identifying relevant features and removing outside elements from the data is known as feature selection, which is also known as attribute selection, subset selection, or variable selection. The feature selection technique gives the advantage of reducing the amount of data needed, reducing execution time, and improving accuracy for prediction in classification problems.

As the answer, propose a modified k-nearest neighbor algorithm with relevant feature selection (*RFS-KNN*), which automatically selects the relevant features and removes irrelevant features of the dataset.

1.1 Description of Modified k-Nearest Nearest Neighbor Algorithm for Relevant Feature Selection (RFS- KNN)

Rashmi Agrawl developed a modified k- Nearest Neighbor Algorithm for Relevant Feature Selection (RFS-KNN). The algorithm does not require any input on the number of features to be selected and hence adopts a filter approach. The algorithm works on the concept that if two features are highly correlated (either positively or negatively), their importance in predicting the class label is negligible, so these features are irrelevant in classification. On the other hand, if features are highly correlated with the class label, they take a prime role in predicting the class label. Also, if a feature's variance is less, the population will exhibit almost the same characteristics; if it is more, it will show different parts. The figure below represents the framework of RFS-KNN. In the RFS-KNN, relevant features from a dataset are selected using the RFS module, and then the dataset with selected features is passed to the KNN algorithm for prediction.



The framework of the Relevant Feature Selection KNN (RFS-KNN) Algorithm

Based on the facts discussed above, the variance-covariance matrix of all features, including the class label of the training data set, has been built. The following table shows the variance-covariance matrix of the Bupa dataset having six features. The values at the main diagonal (shown in the boldface) represent the variance of the features.

Table1: Variance-Covariance matrix of the Bupa dataset

	A1	**A2**	**A3**	**A4**	**A5**	**A6**	**Class**
A1	**15.56533**	0.40427	7.70553	3.41859	26.58191	3.50465	-0.16055
A2	0.40427	**294.93847**	29.58121	22.99432	99.43678	6.58170	-0.61912
A3	7.70553	29.58121	**348.58854**	119.01427	371.88663	15.10548	-0.22643
A4	3.41859	22.99432	119.01427	**83.93156**	163.94010	8.96761	0.97402
A5	26.58191	99.43678	371.88663	163.94010	**1498.98131**	49.77327	2.88583
A6	3.50465	6.58170	15.10548	8.96761	49.77327	**10.11532**	0.04180
Class	-0.16055	-0.61912	-0.22643	0.97402	2.88583	0.04180	0.24701

2 What is feature weighting for KNN?

As we know, feature weighting techniques are in two main categories. 1- performance bias methods. 2- preset bias method.

2.1 What is the performance bias method?

Performance bias happens when one group of subjects in an experiment (for example, a control group or a treatment group) gets more attention from investigators than another group. The difference in care levels results in systematic differences between groups, making it difficult or impossible to conclude that a drug or other intervention caused an effect, as opposed to the level of care. A similar bias is verification bias, where outcomes are more likely to be found in treatment groups due to investigators knowing which person is in which group.

Performance bias can also mean that participants can change their responses or behavior if they know which group they are allocated to. For example, participants might take up their protein intake if a weight loss study investigates whether a high-protein diet reduces weight. This particular type of bias is also called the set of Hawthorne effects.

Performance bias is a significant threat to internal validity. Internal validity is a measure of how good your results are or how confident you are that the outcome of your experiment is due to a single independent variable.

This bias is more likely to happen if investigators know which group a participant is in. It can be minimized or eliminated by blinding, which prevents the investigators from knowing who is in the control or treatment groups. If blinding is used, there still may be differences in care levels, but these are likely to be random and not systematic, which should not affect outcomes.

2.2 What is the preset bias method?

This method uses a function to specify the number of weights and measures each feature's data like correlation, etc. They run with low computational complexity, so the results are fast.

3 The algorithms that work with KNN

3.1 Bucket sort

Bucket sort, also known as bin sort, is a sorting algorithm that divides an array's elements into several buckets. The buckets are then sorted one at a time, either using a different sorting algorithm or by recursively applying the bucket sorting algorithm. Bucket Sort is a sorting algorithm that is commonly used in computer science. Bucket Sort works by distributing the elements of an array into several buckets. Each bucket is then sorted individually, either using a different sorting algorithm or by recursively applying the bucket sorting algorithm.

3.2 K-d tree

A K-Dimensional Tree (also known as K-D Tree) is a space-partitioning data structure for organizing points in a K-Dimensional space. This data structure acts similarly to a binary search tree, with each node representing data in the multi-dimensional space. The K-Dimensional Tree was first developed in 1975 by Jon Bentley. The purpose of the tree was to store spatial data to accomplish the following:

Nearest neighbor search.

Range queries.

Fast look-up.

K-D Trees can guarantee a Log2(n) depth, where n is the number of points in the set.

Since this data structure takes place in a multi-dimensional space, this data structure is beneficial right now. Some modern applications of a K-D Tree could range from astrophysical simulation to computer graphics to even data compression. This data structure also works exceedingly fast thanks to being similar in performance to a Binary Search Tree.