Stockholms universitet

A Comparative Study of Linear Discriminant Analysis and K-Nearest Neighbors for Statistical Classification

Nik Tavakolian

Kandidatuppsats i matematisk statistik
Bachelor Thesis in Mathematical Statistics

Kandidatuppsats 2019:4
Matematisk statistik
Januari 2019

www.math.su.se

Matematisk statistik
Matematiska institutionen
Stockholms universitet
106 91 Stockholm

Mathematical Statistics
Stockholm University
Bachelor Thesis **2019:4**
http://www.math.su.se

A Comparative Study of Linear Discriminant Analysis and K-Nearest Neighbors for Statistical Classification

Nik Tavakolian*

January 2019

Abstract

We study the statistical classification methods: *Linear Discriminant Analysis* and *K-Nearest Neighbors*. We also consider extensions of these methods. The purpose of the study is to obtain a better understanding of when these methods are suitable for use. The performance of classification methods is considerably affected by the characteristics of the data. Understanding how different data attributes impact the performance of these methods is therefore important for applying them effectively to real-world classification problems. In this study we examine the theoretical properties of the classifiers and evaluate their performance statistically for three classification problems, using four evaluation metrics. The classification problems were obtained from the UCI Machine learning repository [10] and we used Accuracy, Log-Loss, Precision and Recall as evaluation metrics. We found that Linear Discriminant Analysis and its extensions are suitable when the sample size is small, and that its assumption of normality is a condition for optimality and not a prerequisite for it to perform well. K-Nearest Neighbors was found to suffer from diminished performance when the class distribution of the data was skewed, or when the number of independent variables was large. A weighted extension of K-Nearest Neighbors was found to effectively alleviate the former problem.

*Postal address: Mathematical Statistics, Stockholm University, SE-106 91, Sweden. E-mail: nik.tavakolian@gmail.com. Supervisor: Chun-Biu Li and Disa Hansson.

Acknowledgements

This is a Bachelor's thesis of 15 ECTS in Mathematical Statistics at the Department of Mathematics at Stockholm University.

I want to take this opportunity to thank my family and friends for their encouragement and support. I want to give a special thanks to my girlfriend Ellinor for offering helpful advice from the start. I also want to thank my supervisors Disa Hansson and Chun-Biu Li for their valuable insight and guidance throughout the project.

Contents

1 Introduction

Statistical classification is the problem of identifying to which of a set of classes a new observation belongs, using a dataset where the class membership of each observation is given. Methods for solving this problem are called *classifiers* or *classification methods*. Many classification methods have been proposed with widely differing approaches to solving the problem, some of these are reviewed in section 4 in [1]. The broad selection of classification methods is motivated by the fact that none of the classifiers outperforms all others for every classification problem (dataset). This is implied by the *No Free Lunch Theorems for Optimization* that are presented and proven in [2].

Unfortunately there is no universal rule of thumb or quick test that will reveal the optimal classifier for a given classification problem. However, the performance of a classifier is considerably impacted by the characteristics of the underlying data, such as non-linearity and class-distribution [3]. If we can understand how different data characteristics affect the performance of different classifiers we can identify classification methods that are likely to perform well for specific problems.

In this study we will compare two classification methods: *Linear Discriminant Analysis* and *K-Nearest Neighbors*. Some extensions of these methods that modify one or more of their characteristics will also be considered. The goal is to obtain useful insight about the properties of these classifiers and the differences between them. We hope that this insight can be used to help in the process of evaluating the suitability of these methods for various classification problems.

In section 2 we will define some fundamental concepts closely related to statistical classification. In section 3 we provide a statistical framework that will serve as the foundation for section 4, where we study the classifiers and their theoretical attributes. In section 5 we compare the classification performance of the methods on real-world datasets and analyze the results using the theory from section 4. We discuss the main findings from section 5 and the limitations of the study in section 6. Lastly, we conclude the report in section 7.

2 Terminology and Key Concepts

In this section we will introduce some terminologies and a few basic concepts that are relevant to the problem of statistical classification.

2.1 Bias-Variance trade-off

The bias-variance trade-off refers to the trade-off between a model that underfits the data and a model that overfits the data. Specifically, *bias* refers to the systematic deviations of a model from the true relationship of the data. The *variance* of a model refers to the amount by which the model parameter estimates change when they are estimated using a new dataset from the same underlying population. The bias-variance trade-off is straightforward to illustrate mathematically in the regression setting where the response variable is continuous.

Let Y in $\mathbb{R}$ be the response variable and X in $\mathbb{R}^p$ be the random vector of independent variables. The true relationship between these variables is described by $Y = f(X) + \epsilon$ where $E[\epsilon] = 0$ and $\text{Var}(\epsilon) = \sigma^2$. Let $\hat{f}$ denote an estimate of f and let S denote the dataset used to fit $\hat{f}$. Given a new observation x_0 with value y_0 the expected prediction error of $\hat{f}(x_0)$ using the squared-error loss function is,

$$\begin{aligned} EPE(x_0) &= E_S[(\hat{f}(x_0) - y_0)^2] \\ &= E_S[\hat{f}(x_0)^2] - 2E_S[\hat{f}(x_0)]E_S[y_0] + E_S[y_0^2]. \end{aligned} \tag{1}$$

For a random variable X the variance is given by $\text{Var}(X) = E[X^2] - E[X]^2$. For an estimator $\hat{\theta}$ of θ the *bias* is defined as, $\text{Bias}(\hat{\theta}) = E_{x|\theta}[\hat{\theta}] - \theta$, where x is the observed data. The last term in equation (1) can now be rewritten as,

$$\begin{aligned} &(E_S[(\hat{f}(x_0)] - f(x_0))^2 + E_S[(\hat{f}(x_0) - E_S[\hat{f}(x_0)])^2] + E_S[(y_0 - f(x_0))^2] \\ &= \text{Bias}^2(\hat{f}(x_0)) + \text{Var}(\hat{f}(x_0)) + \sigma^2. \end{aligned} \tag{2}$$

In equation (2) σ^2 is the variance of the irreducible error ϵ, this term is independent of the estimate $\hat{f}(x_0)$. However, the two other terms, the bias and the variance, are directly associated with $\hat{f}(x_0)$. A model with high bias and low variance underfits the data and a model with low bias and high variance overfits the data. The goal is to find a model that balances the two to obtain low bias and low variance, so that it minimizes the expected prediction error (see also section 2.9 in [4]).

2.2 Training Data vs. Test Data

When evaluating the performance of a classifier we need to make sure that we are not testing the performance on the same dataset that was used to fit/train the classifier. This would favour classifiers with low bias with no penalty for

high variance. To estimate the prediction error accurately we need to evaluate the classifier on test data that was not used during training.

In general we only have access to one dataset, but a simple method for obtaining test data is by splitting the dataset into a training set and a test set. A classifier is then trained on the training set and evaluated using the test set. A problem with this approach is that since we are only using a subset of the original dataset to train the classifier we will likely overestimate the prediction error. This is because a classifier trained with fewer observations will perform worse in general. Another problem with this approach is that the estimated prediction error may be a skewed estimate since it depends on which observations were included in the training and test datasets. A more detailed explanation of this approach and an illustration of these problems can be found in section 5.1.1 in [1].

A different approach for obtaining test data that avoids these problems is the k-fold cross-validation (section 7.10.1 in [4]). In this procedure the dataset is divided into k equally sized portions called folds. In each iteration a fold is chosen and the remaining $k-1$ folds are used as training data. The classifier is trained, and then it is tested on the chosen fold which acts as the test data. This procedure is repeated until each fold has been chosen once.

In this study we will use the stratified 10-fold cross-validation for estimating evaluation metrics for different classifiers. Stratified means that the folds are chosen to have roughly the same class distribution. The 10 resulting estimates will be averaged to give point estimates of the metrics. Stratified 10-fold cross-validation was chosen because it provides good estimates for real-world datasets as is shown in [5]. For the sake of conciseness we will simply refer to the stratified 10-fold cross-validation as cross-validation in this report unless specified otherwise.

2.3 Parametric classifiers vs. Non-Parametric classifiers

Parametric classifiers make explicit assumptions about the functional form of the conditional probability of the classes given the predictors. With these assumptions in place only the parameters of the assumed function need to be estimated to fit the classifier. Non-Parametric classifiers on the other hand make no explicit assumptions about the functional form of the conditional probability of the classes given the predictors. Instead these methods look to estimate the conditional probability based solely on the provided data. A more detailed discussion of this concept can be found in section 2.1.2 in [1].

2.4 Error Rate

Let Y be the response variable, assuming values in the set of classes C. Let X be the random vector of independent variables assuming values in $\mathbb{R}^p$. The outcomes of these variables are the training data points (x_i, y_i) where $i = 1, ..., N$. The *training error rate* of a classifier $\hat{f}$ estimating the true relationship $Y = f(X)$ is then given by,

$$\text{Training Error Rate} = \frac{1}{N} \sum_{i=1}^{N} I(y_i \neq \hat{f}(x_i)), \tag{3}$$

where $I(x) = 1$ if x is true and $I(x) = 0$ if x is false. It is the proportion of incorrect predictions that the classifier makes on the training data. As we pointed out in section 2.2, we are generally interested in the value of performance metrics on the test data and not the training data. The *test error rate* of a classifier is defined analogously to the training error rate in equation (3), using test data instead of the training data. A more detailed explanation of this topic can be found in section 2.2.3 in [1].

3 Statistical Framework

In order to evaluate and compare statistical classification methods, we need a common framework in which different methods can be put into context. In this section we will outline that framework based on the theory detailed in section 2.4 in [4].

We start by defining the statistical classification problem. Let Y be the response variable, assuming values in the set of classes C. Let X be the random vector of independent variables assuming values in $\mathbb{R}^p$. The goal in statistical classification is that of finding a function f so that,

$$Y = f(X).$$

In words, our goal is to find the function (classifier) that makes the fewest number of errors when predicting the output class Y given the corresponding input vector X. We can frame this as an optimization problem in which the goal is to minimize an objective function that quantifies the error of the classifier. In general, we want all prediction errors to be penalized equally. A function L meeting this specification can be defined as follows:

$$L(Y, f(X)) = \begin{cases} 0 & \text{if } Y = f(X), \\ 1 & \text{otherwise.} \end{cases} \tag{4}$$

We can now express the expected prediction error as,

$$EPE = E_{X,Y}[L(Y, f(X))] = \int_x \sum_y L(y, f(x))\, P(X = x, Y = y)\ dx. \tag{5}$$

Using the definition of conditional probability equation (5) can be rewritten as,

$$\begin{aligned} EPE &= \int_x P(X=x) \sum_y L(y, f(x)\,)\, P(Y=y \mid X=x)\ \ dx \\ &= E_X\big[\, E_{Y|X}[L(y, f(x)\,) \mid X=x\,]\,\big]. \end{aligned} \tag{6}$$

From equation (6) we can see that minimizing the expected prediction error is equivalent to minimizing the conditional expected value of $L(y, f(x))$ given that $X = x$. For a new observation x_0 the function f that minimizes the expected prediction error is now given by,

$$\begin{aligned} f(X=x_0) &= \underset{c \in C}{\arg\min}\ \ E_{Y|X}[L(Y,c\,) \mid X=x_0] = \underset{c \in C}{\arg\min}\ \ \Big(1 - P(c \mid X=x_0)\Big) \\ &= \underset{c \in C}{\arg\max}\ \ P(c \mid X=x_0). \end{aligned} \tag{7}$$

The result that we have obtained says that the function f that minimizes the expected prediction error is the function that, given a new observation x_0, classifies to the most probable class. This is called the Bayes classifier and we have shown that it is the optimal classifier when all misclassifications are penalized equally. The Bayes decision boundary is the border between the classes and it is defined as the set of points for which equation (7) has no unique solution c.

In practice the conditional probability $P(Y = y \mid X)$ is generally unknown. Therefore it is impossible to apply the Bayes classifier, unless we are working with data that has been sampled from a known distribution. As a result, most statistical classification methods seek to estimate $P(Y = y \mid X)$ in an attempt to approximate the optimal Bayes classifier.

4 Statistical Classification Methods

4.1 Linear Discriminant Analysis

All of the theory in this section can be found in section 4.3 in [4] unless another source is cited. Linear Discriminant Analysis (LDA) is the first statistical classification method that we will study. Let Y be the response variable, assuming values in the set of classes C. Let X be the random vector of independent variables assuming values in $\mathbb{R}^p$. The outcomes of these variables are the data points (x_i, y_i) where $i = 1, ..., N$. LDA is based on the following assumption,

$$\boldsymbol{X} \mid \boldsymbol{Y} = \boldsymbol{k} \sim \mathcal{N}(\mu_k,\, \Sigma)\,. \tag{8}$$

Assumption (8) should really be thought of as two separate assumptions. Firstly, it is assumed that the input vector of independent variables X has a multivariate gaussian distribution that is different for each output class k. Secondly, the assumption is made that all classes share a pooled covariance

matrix. As we will see assumption (8) implies a functional form for the conditional probability of the classes conditional on the predictors, making LDA a parametric classifier.

LDA aims to assign a new observation x_0 to the class k for which $P(Y = k \mid X = x_0)$ is largest. Let π_k denote $P(Y = k)$. Bayes theorem states that,

$$P(Y = k \mid X = x_0) = \frac{P(X = x_0 \mid Y = k)\,\pi_k}{\sum_{j=1}^{K} P(X = x_0 \mid Y = j)\,\pi_j}.$$

Estimating $P(X = x_0 \mid Y = k)$ and π_k gives us an estimate of $P(Y = k \mid X = x_0)$ through this relationship.

Because of assumption (8), estimating $P(X = x \mid Y = k)$ is reduced to estimating the mean vectors μ_k for all classes k and estimating the pooled covariance matrix Σ. Here x denotes the independent variables in the training data. The prior probabilities π_k also need to be estimated for all classes. Let K denote the total number of classes and let D represent the set $\{i \mid y_i = k\}$. Furthermore let N_k be the number of observations with class k. Estimates of these parameters are now given by,

$$\hat{\mu}_k = \frac{1}{N_k} \sum_{D} x_i, \tag{9}$$

$$\hat{\Sigma} = \frac{1}{N - K} \sum_{k=1}^{K} \sum_{D} (x_i - \hat{\mu}_k)(x_i - \hat{\mu}_k)^T, \tag{10}$$

$$\hat{\pi}_k = \frac{N_k}{N}. \tag{11}$$

With (9) and (10) we can now compute an estimate for $P(X = x \mid Y = k)$,

$$\hat{P}(X = x \mid Y = k) = \frac{\exp[-\frac{1}{2}(x - \hat{\mu}_k)^T \hat{\Sigma}^{-1} (x - \hat{\mu}_k)]}{\sqrt{(2\pi)^p \det \hat{\Sigma}}}. \tag{12}$$

An estimate for $P(Y = k \mid X = x)$ can now be computed using Bayes theorem,

$$\begin{aligned} \hat{P}(Y = k \mid X = x) &= \frac{\hat{P}(X = x \mid Y = k)\,\hat{\pi}_k}{\sum_{j=1}^{K} \hat{P}(X = x \mid Y = j)\,\hat{\pi}_j} \\ &= \frac{\hat{\pi}_k \exp[-\frac{1}{2}(x - \hat{\mu}_k)^T \hat{\Sigma}^{-1} (x - \hat{\mu}_k)]}{\sum_{j=1}^{K} \hat{\pi}_j \exp[-\frac{1}{2}(x - \hat{\mu}_j)^T \hat{\Sigma}^{-1} (x - \hat{\mu}_j)]}. \end{aligned} \tag{13}$$

The LDA classifier uses equation (13) in the following way: given a new observation x_0 it classifies to the class k for which $\hat{P}(Y = k \mid X = x_0)$ is maximized.

We can simplify this optimization problem by ignoring the denominator in (13) since it is shared by all classes. In this way we get,

$$\hat{P}(Y = k \mid X = x) \propto \hat{\pi}_k \exp[-\frac{1}{2}(x - \hat{\mu}_k)^T \hat{\Sigma}^{-1}(x - \hat{\mu}_k)]. \tag{14}$$

We can also use the logarithm to simplify further since it is a monotonic function. Taking the logarithm of both sides in (14) we get,

$$\begin{aligned} &\log(\hat{P}(Y = k \mid X = x)) = \log(\hat{\pi}_k \exp[-\frac{1}{2}(x - \hat{\mu}_k)^T \hat{\Sigma}^{-1}(x - \hat{\mu}_k)]) + C \\ &= \log(\hat{\pi}_k) - \frac{1}{2}(x - \hat{\mu}_k)^T \hat{\Sigma}^{-1}(x - \hat{\mu}_k) + C \\ &= \log(\hat{\pi}_k) - \frac{1}{2}\hat{\mu}_k^T \hat{\Sigma}^{-1}\hat{\mu}_k + x^T \hat{\Sigma}^{-1}\hat{\mu}_k + C'. \end{aligned} \tag{15}$$

In the last derivation of (15) the term C' is a constant with respect to k. Since we are only looking to include terms that depend on the class k, we can disregard this term. The resulting function is called the linear discriminant function,

$$\hat{\delta}_k(x) = \log(\hat{\pi}_k) - \frac{1}{2}\hat{\mu}_k^T \hat{\Sigma}^{-1}\hat{\mu}_k + x^T \hat{\Sigma}^{-1}\hat{\mu}_k. \tag{16}$$

The estimated LDA decision boundary between two classes k and j is now given by the set $\{x : \hat{\delta}_k(x) = \hat{\delta}_j(x)\}$. The points x belonging to this set are the points that solve the following equation,

$$\begin{aligned} &\hat{\delta}_k(x) = \hat{\delta}_j(x) \\ &\implies DB_{LDA} : \log\left(\frac{\hat{\pi}_k}{\hat{\pi}_j}\right) + \frac{1}{2}(\hat{\mu_j}^T \hat{\Sigma}^{-1}\hat{\mu_j} - \hat{\mu}_k^T \hat{\Sigma}^{-1}\hat{\mu}_k) + x^T(\hat{\Sigma}^{-1}\hat{\mu}_k - \hat{\Sigma}^{-1}\hat{\mu_j}) = 0. \end{aligned} \tag{17}$$

Since equation (17) is linear in x, the decision boundary that LDA produces is linear. An illustration of this is shown in Figure 1. It is important to note that while assumption (8) implies a linear decision boundary, a linear decision boundary between the classes does not imply that assumption (8) holds (see also section 1.3 in [6]).

For a classification problem with K classes, the LDA classifier is fully defined by the differences between the linear discriminant functions, $\delta_k(x) - \delta_K(x)$, between the first $K - 1$ classes. For each of these differences $p + 1$ parameters need to be estimated to define the classifier, where p is the number of independent variables. As a result the LDA classifier has $(K - 1)(p + 1)$ effective parameters.

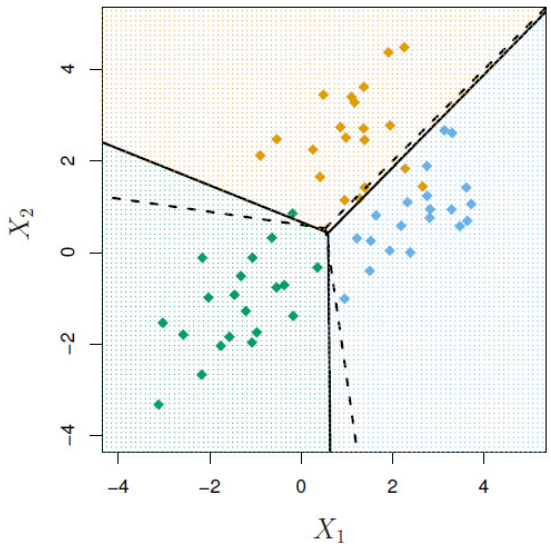

Figure 1: An example with three classes. Observations were drawn from three bivariate gaussian distributions with class specific mean vectors and a common covariance matrix. The solid black line is the LDA decision boundary and the dotted black line is the optimal Bayes decision boundary.*

4.1.1 Properties

Linear Discriminant Analysis makes strong assumptions about the distribution of the data that are unlikely to be true for real data. By doing so, however, it ensures that relatively few parameters have to be estimated to fit the model. This makes it useful in many situations. For example when the sample size is large and computational resources are limited. In other situations these assumptions can cause the classifier to make systematic errors and underfit the data.

Let us start by looking at situations in which LDA is appropriate in theory:

- When the Bayes decision boundary is linear or approximately linear.
- When the number of observations is low, the simplicity of LDA prevents it from overfitting the data.
- When assumption (8) holds completely or approximately.

*Reprinted/adapted by permission from [Springer Nature Customer Service Centre GmbH]: [Springer [Introduction to Statistical Learning] by [Trevor Hastie, Gareth James, Robert Tibshirana, and Daniela Witten] [COPYRIGHT] (2013)

- When the number of observations is high and computational resources are limited, it might not be feasible to use methods with more parameters because of computational limitations.

Some situations when LDA may not be appropriate or when other classifiers may perform better in general are:

- When the Bayes decision boundary is non-linear.
- When many observations are available and more flexible methods are computationally feasible.
- When assumption (8) is violated.

We have already seen that the assumptions of LDA result in a linear decision boundary between the classes. This is not always a good approximation of the Bayes decision boundary, which can be very irregular for real world problems. Applying LDA for problems with a clearly non-linear class separation will therefore result in a model with high bias. If the optimal decision boundary is approximately linear LDA may perform quite well. In these situations it may be preferred over more flexible classifiers which are more likely to overfit the data.

In situations where the number of observations is low, the data may only be able to reliably give a rough estimate of the decision boundary, since the variance of such an estimate would be high. Using classifiers that have more parameters and that support non-linear decision boundaries can easily lead to overfitting in these cases. LDA can perform well in these situations. It offers a simple estimate of the decision boundary that may be successful in approximating the general relationship between the classes without capturing idiosyncrasies in the data.

When the number of observations is high, more information about the shape of the decision boundary can be reliably obtained. Since the decision boundary is often non-linear for real world classification problems, other classifiers generally perform better in these situations. However if computational resources are limited, more flexible approaches might be too computationally complex to be feasible. In these situations LDA can be appropriate if the Bayes decision boundary is not too non-linear.

When assumption (8) holds approximately for a given problem, LDA will perform very well. This is because LDA is asymptotically efficient under this assumption [6]. This means that there is no other classifier that will perform better asymptotically (as $N \to \infty$) if the assumption holds.

4.1.2 Extensions

The primary limitation of LDA is that it produces a linear decision boundary. As we pointed out in the previous section, this can be a poor approximation for real-world classification problems. Several extensions of LDA have been proposed with weaker assumptions to allow for non-linear decision boundaries. Some of these are detailed in sections 12.4-12.7 in [4].

A natural extension of LDA arises when the assumption of a common covariance matrix for all classes is relaxed and each class is allowed to have a unique covariance matrix. Assumption (8) is then replaced by the following assumption,

$$\boldsymbol{X} \mid \boldsymbol{Y} = \boldsymbol{k} \sim \mathcal{N}(\mu_k, \Sigma_k)\,. \tag{18}$$

The consequence of this that the discriminant function from (16) changes to,

$$\hat{\delta}_k(x) = -\frac{1}{2}\log|\hat{\Sigma}_k| - \frac{1}{2}(x-\hat{\mu}_k)^T\hat{\Sigma}_k^{-1}(x-\hat{\mu}_k) + \log(\hat{\pi}_k). \tag{19}$$

The equation for the estimated decision boundary between any two classes k and j is now given by,

$$DB_{QDA}:$$
$$\log\left(\frac{\hat{\pi}_k}{\hat{\pi}_j}\right) + \frac{1}{2}\left(\log\left(\frac{|\hat{\Sigma}_j|}{|\hat{\Sigma}_k|}\right) + (x-\hat{\mu}_j)^T\hat{\Sigma}_j^{-1}(x-\hat{\mu}_j) - (x-\hat{\mu}_k)^T\hat{\Sigma}_k^{-1}(x-\hat{\mu}_k)\right) = 0 \tag{20}$$

This is a quadratic function in x and so the decision boundary is quadratic. The method that uses this decision boundary is unsurprisingly called Quadratic Discriminant Analysis (QDA) and is the most direct extension of LDA. The QDA classifier is fully defined by the differences between the quadratic discriminant functions (equation (19)) for the first $K-1$ classes. For each of these differences $(p(p+3)/2+1)$ parameters need to be estimated to define the classifier. As a result the number of effective parameters for the QDA classifier are given by $(K-1)(p(p+3)/2+1)$. The decision boundaries of LDA and QDA are shown for a two-class problem in Figure 2.

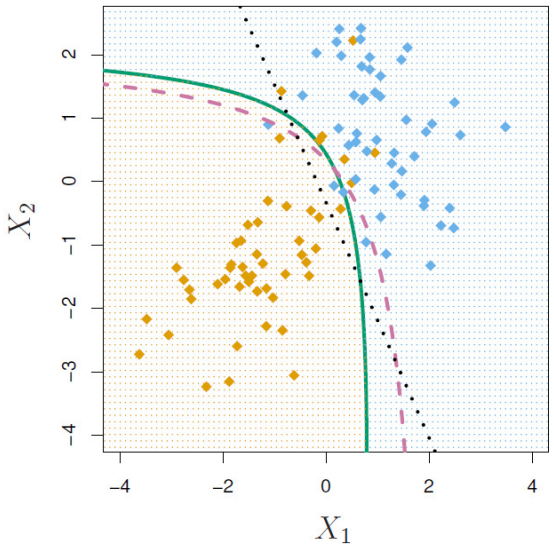

Figure 2: An example with two classes. Observations were drawn from two bivariate gaussian distributions with class specific mean vectors and covariance matrices. The dotted black line is the LDA decision boundary and the solid green line is the QDA decision boundary. The optimal Bayes decision boundary is the purple dashed line.[†]

If LDA seems to underfit the data for a given problem and QDA seems to overfit, a compromise between the two methods can be obtained by defining the covariance matrix for a given class as a weighted average between the LDA covariance matrix, and the QDA covariance matrix. This approach is called Regularized Discriminant Analysis (RDA); the covariance matrix is then given by,

$$\hat{\Sigma}_k(\alpha) = \alpha\hat{\Sigma}_k + (1-\alpha)\hat{\Sigma}, \;\; 0 \leq \alpha \leq 1. \tag{21}$$

When $\alpha = 0$ this method is equivalent to LDA and when $\alpha = 1$ it is equivalent to QDA. When choosing α, the goal is to minimize the test error rate of the resulting classifier. In practice the test error rate is commonly estimated for different values of α with cross-validation.

[†]Reprinted/adapted by permission from [Springer Nature Customer Service Centre GmbH]: [Springer [Introduction to Statistical Learning] by [Trevor Hastie, Gareth James, Robert Tibshirana, and Daniela Witten] [COPYRIGHT] (2013)

4.2 K-Nearest Neighbors

The second classification method we are going to study is the K-Nearest Neighbors (KNN). Unlike LDA it does not make any explicit assumptions on the distribution of the data, making it a non-parametric classifier. All of the theory in this section can be found in section 13.3 in [4] unless another source is cited.

Given a new observation x_0 the KNN classifier finds the k closest points to x_0 and uses these to estimate $P(Y = j \mid X = x_0)$ locally for each class j, according to the following rule,

$$\hat{P}(Y = j \mid X = x_0) = \frac{k_j}{k}. \tag{22}$$

Here k_j is the number of neighbors that belong to class j among the k neighbors. It then classifies x_0 to the class j for which $\hat{P}(Y = j \mid X = x_0)$ is the largest. If there is a tie between classes the class is chosen among them at random. This is called the majority voting scheme/rule, since it assigns the new observation to the majority class among the neighbors.

One ambiguity in the above formulation of the KNN scheme is the measure of distance used for determining the k closest points. Different measures can be used, but the most commonly used measure is the Euclidean distance, as evidenced by its usage on page 465 in [4] to define the KNN scheme. The Euclidean distance d_{th} between two observations $x_t = (x_{t1}, ..., x_{tp})$ and $x_h = (x_{h1}, ..., x_{hp})$ is given by,

$$d_{th} = \sqrt{\sum_{i=1}^{p} (x_{ti} - x_{hi})^2}. \tag{23}$$

The number of effective parameters for K-Nearest Neighbors is N/k where N is the number of observations and k is the number of neighbors (page 15 in [4]). This can be understood when considering the situation in which the neighborhoods are not overlapping. Then there would be N/k neighborhoods and for each of these, one parameter would have to be estimated, the neighborhood mean. The number of neighbors k is the only explicit parameter of K-Nearest Neighbors. Lower values of k yields a model with higher variance, since the number of effective parameters increases as k decreases. For the same reason higher values of k gives a model with greater bias. An illustration of this relationship can be found on page 466 in [4]. The value of k that is optimal for a given problem is usually determined by estimating the test error rate and choosing the value of k for which the test error rate is minimized.

4.2.1 Properties

Since K-Nearest Neighbors is a non-parametric classifier, it is very adaptable compared to parametric methods like LDA that perform poorly if their assumptions about the data do not approximately coincide with reality. Furthermore since KNN estimates $P(Y = y \mid X)$ locally for each new observation, it can support highly non-linear decision boundaries between the classes. However as we will see there are situations where KNN will perform poorly and when other methods will be more suitable.

Let us start by looking at some situations in which the KNN classifier is appropriate in theory:

- When the Bayes decision boundary is non-linear and irregular and/or when parametric methods cannot adequately approximate its shape.
- When the number of observations is high and KNN is computationally feasible.

Here are some situations when KNN may not be appropriate or when other classifiers may perform better in general:

- When the number of observations is low.
- When the number of observations is high but computational resources are limited and cannot support KNN.
- When the class distribution is skewed.
- When the number of independent variables is high.

KNN is a memory-based classifier which means that it stores the training data in memory and accesses it every time a new observation is to be classified. As the training data grows in size this can become computationally unfeasible if computer memory is limited. Because of this, KNN is most appropriate when there are enough observations in the data to support a complex decision boundary, but not so many that it becomes computationally difficult or impossible to use. LDA does not suffer from this problem since it fits a model, as a result only the estimated model parameters need to be stored for classification of new observations.

KNN makes no explicit assumptions about the distribution of the data. Therefore it can approximate complex and irregular decision boundaries between the classes. When the number of observations is low this is an undesirable trait in general. This is because, as we mentioned in section 4.1.1, when the sample size is small the data can only support simple decision boundaries between the classes. KNN will likely overfit the data in these situations since it does not assume an approximate shape for the decision boundary, and the data does not contain enough information about its true shape.

KNN has an interesting property that makes it inadvisable for classification problems with a large number of independent variables. We will illustrate this property with an example. Suppose we have a dataset with 50 people and we want to classify their sex with one explanatory variable, the height of each person. Using KNN we can classify the sex of a new person by considering the k people in the dataset closest in height, and deciding the sex through a majority vote. Now consider a situation where we have 20 explanatory variables instead, such as height, weight, eye color, etc. The probability that a new person will have similar characteristics to k others, among 50 people, and across 20 variables, is much smaller than the probability that they are close in height alone. Across 20 independent variables the concept of a nearest neighbor becomes almost meaningless when there are 50 observations in the dataset. In general the closest neighbors might be very far away in high dimensions and might not be a good basis for estimating the local conditional probability in (22). This problem is called *the curse of dimensionality*. A more detailed illustration of the problem can be found in section 2.5 in [4].

Another problem with KNN is that if the class distribution is skewed in the training data the majority voting scheme heavily favors the more prevalent classes [7]. This is because KNN does not differentiate between points that are closer and points that are farther away within neighborhoods. This is especially problematic if k is large, since some points may be much closer to the new observation than others within the neighborhood.

4.2.2 Extensions

Many extensions of the KNN classifier have been proposed, some of these are detailed in section 13.4 in [4]. Most of them look to address the problems that we outlined in the previous section. In essence there are two aspects that can be modified in this classification method to obtain improved results: the measure of distance and the voting rule. In the standard method that we have described, Euclidean distance is used and the voting rule is majority vote among the neighbors. In this section we will look at how changing the distance measure and the voting rule affects the outcomes of this method.

As mentioned in the previous section, KNN with majority voting performs poorly when the class distribution is skewed. By weighing the votes of each neighbor based on its distance from the query point, we can alleviate this problem in theory. A natural way of doing this is by using the inverse-distance between the points as weights. Formally the weight of a point x_i in the neighborhood of a new observation x_0 is then given by,

$$w_i = \frac{1}{d_{i0} + \epsilon}. \tag{24}$$

Where we have used the Euclidean distance as defined in (23), and where ϵ is a small real-valued number that prevents division by zero in the case when

$d_{i0} = 0$. Modifications of KNN that are based on adding weights to the neighbors are called weighted K-Nearest Neighbors (wKNN) methods. A detailed review of different weighing schemes for KNN can be found in [8].

Another problem with the KNN method is the curse of dimensionality which was explained in the previous section. Alleviating this problem is more difficult, but the choice of distance measure can affect the performance of KNN in higher dimensions. The Manhattan distance between two observations $x_t = (x_{t1}, ..., x_{tp})$ and $x_h = (x_{h1}, ..., x_{hp})$ is defined as,

$$d_{th} = \sum_{i=1}^{p} |x_{ti} - x_{hi}|. \tag{25}$$

In [9] it is shown that the Manhattan distance is more appropriate than the Euclidean distance in higher dimensions. That is, it yields a KNN classifier that does not suffer as much from the curse of dimensionality.

5 Data Analysis

In this section we are going to evaluate the performance of the classification methods we have presented so far on three real-world datasets. Specifically we will include the following methods in the analysis:

- Linear Discriminant Analysis (LDA)
- Quadratic Discriminant Analysis (QDA)
- K-Nearest Neighbors (KNN)
- K-Nearest Neighbors with inverse-distance weights (wKNN)

We will use the Euclidean distance for both nearest neighbor approaches. The three datasets we are going to use for the analysis have been downloaded from the University of California Irvine Machine Learning Repository [10]. The datasets have been chosen to be of different sizes to avoid bias towards any of the methods. Furthermore, two of the datasets have binary response variables and one has a categorical response variable with 7 levels.

5.1 Methodology

We will assess the classification performance of the methods using 4 evaluation metrics: accuracy, precision, recall and log-loss. The definitions of the former three can be found in [11]. We will use the log-loss (cross-entropy) as defined on page 209 in [12]. In section 5.1.1 we define and discuss the accuracy and the log-loss. In section 5.1.2 we do the same for the precision and recall. High values are desirable for the the accuracy, precision and recall; lower values are desirable for the log-loss. These metrics will be computed for each method in the following way:

1. A Non-parametric Bootstrap of the dataset is performed with I bootstrap samples.
2. Cross-validation is used to estimate the value of the evaluation metric for each bootstrap sample.
3. A 95% bootstrap confidence interval for the metric is determined by considering the I bootstrap estimates and using the 2.5 percentile as the lower bound of the interval and the 97.5 percentile as the upper bound.

We will plot the point estimates of the metrics together with the 95% confidence intervals for all methods in the same plot. We will use the visual representation of the confidence intervals to determine if the differences between the methods are statistically significant. If the confidence intervals of two methods do not overlap, the difference between them is considered to be statistically significant for that metric.

For each classification problem, we will compute the number of effective parameters for the methods. We will use this information as an indicator of the complexity of the classifiers. Fewer effective parameters suggest that the classifier is simpler and less flexible.

The value of k, the number of neighbors for the nearest neighbor approaches, will be determined by using cross-validation on the datasets to evaluate the test error rate for values of k between 1 and 100. The value of k for which the test error rate is the lowest is then chosen. This procedure will be performed separately for KNN and wKNN. The associated cross-validation curves for all datasets can be found in the Appendix.

The number of bootstrap samples used for estimating the confidence intervals will be determined independently for each dataset. The highest possible number will be used given the constraints of the available computational resources.

For the Breast Cancer dataset that we will analyze in section 5.2, we will use the Royston H test for multivariate normality to check whether the assumptions of LDA and QDA are satisfied. The details of this test can be found in [13].

5.1.1 Accuracy and Log-Loss

The *accuracy* of a classifier $\hat{f}$ for a given classification problem is defined as:

$$\text{Accuracy} = 1 - \text{Error Rate} = \frac{1}{N}\sum_{i=1}^{N} I(y_i = \hat{f}(x_i)). \tag{26}$$

It is the proportion of correct predictions made by the classifier. The accuracy assumes values between 0 and 1. A classifier with an accuracy of 0 makes no correct predictions, while a classifier with an accuracy of 1 makes no incorrect predictions. We will use the accuracy as an evaluation metric since it has a simple interpretation, and because it is natural indicator of the predictive power of a classifier.

One shortcoming of the accuracy, however, is that it does not account for the uncertainty associated with classification to a given class. For a binary classification problem with classes A and B, suppose the correct label for a given observation is class A. If classifier 1 correctly predicts class A, with an associated probability of 0.51, and classifier 2 also correctly predicts class A, but with probability 0.99, the resulting effect on the accuracy score will be the same for both classifiers. This is because the accuracy only distinguishes between correct and incorrect predictions. Classifier 2 clearly performs better in this situation, however, since it predicts the label correctly with greater certainty than classifier 1.

To account for this uncertainty in prediction we will use the log-loss as a complementary measure of predictive power alongside the accuracy. The *log-loss* for a classification problem with N observations and K classes is defined as:

$$\begin{aligned} \text{Log-Loss} &= -\frac{1}{N}\sum_{i=1}^{N}\sum_{k=1}^{K} y_{ik} \ln \hat{P}(Y = k \mid X = x_i), \\ \text{where } y_{ik} &= \begin{cases} 1 & \text{if } y = k \text{ for observation } i, \\ 0 & \text{otherwise.} \end{cases} \end{aligned} \tag{27}$$

The log-loss and the accuracy share the same denominator, N, the total number of predictions/observations. The numerators are different however. The numerator of (26) is simply a count of the number of correct predictions made; the numerator of (27) is a weighted count of the correct predictions, where the weights quantify the uncertainty associated with each classification. Because of this the log-loss can be thought of as a more nuanced version of the accuracy that provides information about the uncertainty associated with each classification. It can be a particularly useful metric when the differences in accuracy between methods is small or indistinguishable. The log-loss assumes values between 0 and ∞. A lower value of the log-loss is desirable since it indicates that a classifier is making correct predictions with a high degree of certainty.

5.1.2 Precision and Recall

We will use the accuracy and the log-loss to evaluate the overall performance of the classifiers for specific classification problems. It is also of interest to evaluate the class-specific performance of the methods. In doing so we can discover if the classifiers are biased in their predictions towards any of the classes. We will use the precision and recall to evaluate class-specific performance. Before we define these metrics we will define some terminologies related to class-specific performance.

Suppose that we have a binary classification problem with classes A and B.

- A **true positive** is a correct classification to class A.
- A **false positive** is an incorrect classification to class A.
- A **true negative** is a correct classification to class B.
- A **false negative** is an incorrect classification to class B.

The *precision* of a classifier for class A is defined as,

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}. \tag{28}$$

It takes on values between 0 and 1 and is the ratio between the number of correct predictions to class A and the total number of predictions to class A. A high precision means that the false positive rate is low. This indicates that the classifier rarely classifies to class A incorrectly.

The *recall* of a classifier for class A is defined as,

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}. \tag{29}$$

It assumes values between 0 and 1 and is the ratio between the number of correct predictions to class A and the total number of observations belonging to class A. A high recall means that the false negative rate is low. This indicates that the classifier rarely misclassifies observations that belong to class A.

In our analysis we will use the precision and recall in conjunction to evaluate the distribution of the errors made by the classifiers. If the recall is considerably lower than the precision, for example, we can conclude that the classifier predicts false negatives more often than false positives. Similar values of the precision and recall indicate that the classifier does not systematically predict more false positives than false negatives or vice versa.

For the Forest Cover dataset that we will analyze in section 5.4, the response variable is not binary. The precision and recall for one class in a multiclass (more than 2 classes) problem do not provide comprehensive information about the performance of a classifier for all classes. It is not obvious how the precision and recall scores should be defined to provide valuable information about the distribution of the predicted classes for a multiclass problem. We could use the arithmetic mean of the class specific scores to summarize the performance of a classifier. One problem with this, however, is that the scores of more prevalent classes are not weighted more than less prevalent classes. Since we are generally interested in minimizing the error rate of a classifier, this is an undesirable property. Instead we will use micro-averaging of the precision and recall scores across all classes [11]. With micro-averaging more prevalent classes are weighted more heavily. For a classification problem with K classes the *micro-averaged precision* is defined as,

$$\text{Precision}_\mu = \frac{\sum_{i=1}^{K} \text{True Positives}_i}{\sum_{i=1}^{K} (\text{True Positives}_i + \text{False Positives}_i)}, \tag{30}$$

and the *micro-averaged recall* is defined as,

$$\text{Recall}_\mu = \frac{\sum_{i=1}^{K} \text{True Positives}_i}{\sum_{i=1}^{K} (\text{True Positives}_i + \text{False Negatives}_i)}. \tag{31}$$

5.1.3 Normalization

Because the nearest neighbor approaches use the distance between observations as the basis for classification, the relative magnitude of the independent variables has a considerable impact on the classification performance of these methods (see also page 165 in [1]). Independent variables on a larger scale will have a greater effect on the distance between the observations. For instance, suppose we have two independent variables in a dataset of countries: population size and life expectancy. Life expectancy will assume values that are roughly in the range [0, 100] while population size will assume values in the millions. In this situation, the nearest neighbor approaches will fail to account for the difference in scale and will almost exclusively use the population size to determine the closest neighbors. To solve this problem we will normalize the data before applying the nearest neighbor methods. Let x_{ij} denote the elements of the feature matrix, where $i = 1, ..., N$ are the observations and $j = 1, ..., p$ are the independent variables. We will replace the vectors $x_j = \{x_{1j}, ..., x_{Nj}\}$ with the normalized vectors $x'_j = \{x'_{1j}, ..., x'_{Nj}\}$ where,

$$x'_j = \frac{x_j}{||x_j||}. \tag{32}$$

Here $||x_j||$ is the Euclidean norm of the vector x_j.

5.2 Breast Cancer Dataset

The first dataset we are going to analyze consists of 569 observations of tumors from breast cancer patients. The data is labelled with information about whether a tumor is malignant or benign. The class distribution is skewed with 357 benign tumors and 212 malignant tumors. For each tumor a digitized image of a fine needle aspirate (a sample of the cell nuclei) was analyzed. Characteristics of the cell nuclei (area, radius, etc.) were recorded for each tumor. Ten characteristics were recorded in total. For each of these, the mean, standard error, and largest value was computed to account for the variation between individual cell nuclei. As a result the dataset has 31 columns for each patient. One column specifying whether a tumor is malignant or benign and 30 columns describing the characteristics of the tumors cell nuclei and their spread. The classification problem is that of classifying a given tumor as malignant or benign, based on the characteristics of the tumor, as described by the 30 explanatory variables (all real-valued).

5.2.1 Results and Analysis

The number of bootstrap samples used with this dataset is 1000. Using cross-validation, the estimated optimal values of k that will be used are 19 for KNN and 23 for wKNN. The associated cross-validation curves can be found in section 8.1 in the Appendix. The number of effective parameters for each method are shown in Table 1. We see that QDA has considerably more

effective parameters than the other methods for this classification problem. This is because the number of effective parameters for QDA grows quickly with the number of independent variables in the dataset, and this dataset contains a large number of independent variables.

Method	#Parameters
LDA	31
QDA	496
KNN	30
wKNN	25

Table 1: The number of effective parameters for each method with the Breast Cancer Dataset.

For this dataset the Royston H test for multivariate normality was performed on the independent variables corresponding to each class. This was done to test whether the LDA and QDA assumptions of normality are satisfied for this classification problem. Multivariate normality was rejected for both classes with p-values of 6.41×10^{-99} for the class malignant and 3.81×10^{-124} for the class benign. These result imply that the assumptions of LDA and QDA are clearly violated for this classification problem.

Let us now look at the accuracy scores of the different methods for this classification problem. The mean accuracy for each method and the associated 95% confidence intervals are shown in Figure 3.

From Figure 3 we can see that all methods achieve a high accuracy for this problem, above 0.90. The only statistically significant difference between the accuracy scores of the methods is that KNN has a lower accuracy than wKNN. Since wKNN only differs from KNN in that it uses weights in the voting rule, this must be the reason for the discrepancy. The motivation for wKNN over KNN, as described in section 3.2.2, is that when the class distribution is skewed, KNN is biased towards the more prevalent class when making a classification. For this dataset there are 68.4% more observations in the class benign compared to malignant. This skew in the class distribution explains why wKNN achieves a higher accuracy than KNN for this classification problem.

For a more nuanced evaluation of the predictive power of the methods we will now look at the log-loss scores. In Figure 4 the mean log-loss scores are shown for the methods along with 95% confidence intervals for the true values of the log-loss.

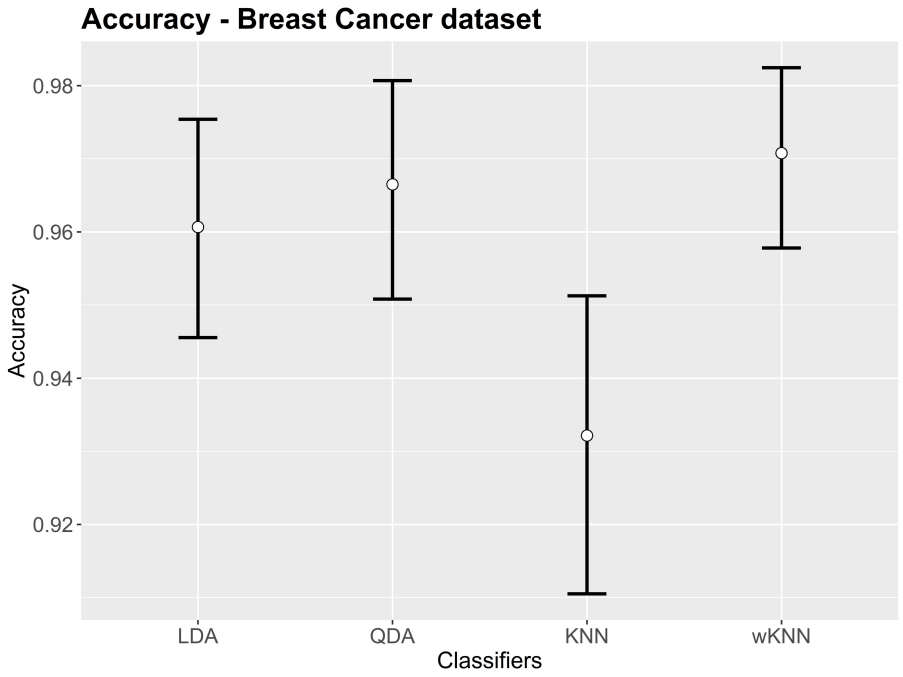

Figure 3: A plot of the mean accuracy (white dots) for each method across all bootstrap samples for the Breast Cancer dataset. The capped black lines are the 95% confidence intervals of the accuracy for each method.

From Figure 4 we can see that the only statistically significant difference between the log-loss scores of the methods is that LDA has a lower log-loss score than QDA and KNN. This indicates that when taking the uncertainty of the classifications into account, LDA performs better than QDA and KNN.

A plausible explanation for the difference in the log-loss score between LDA and QDA is that, since LDA has fewer parameters than QDA, it generalizes well for this classification problem, while QDA overfits the data. This explanation seems reasonable since QDA has considerably more effective parameters than LDA. For the difference between LDA and KNN, the most likely explanation seems to be that KNN suffers from the skewed class distribution that was illustrated in Figure 3, where wKNN achieved a higher accuracy than KNN.

All precision scores and recall scores were computed for the class malignant. For this classification problem the differences between the precision scores of the methods are not statistically significant. Let us look at the recall scores for the methods. The recall score mean for each method along with 95% confidence intervals for the true recall scores are shown in Figure 5.

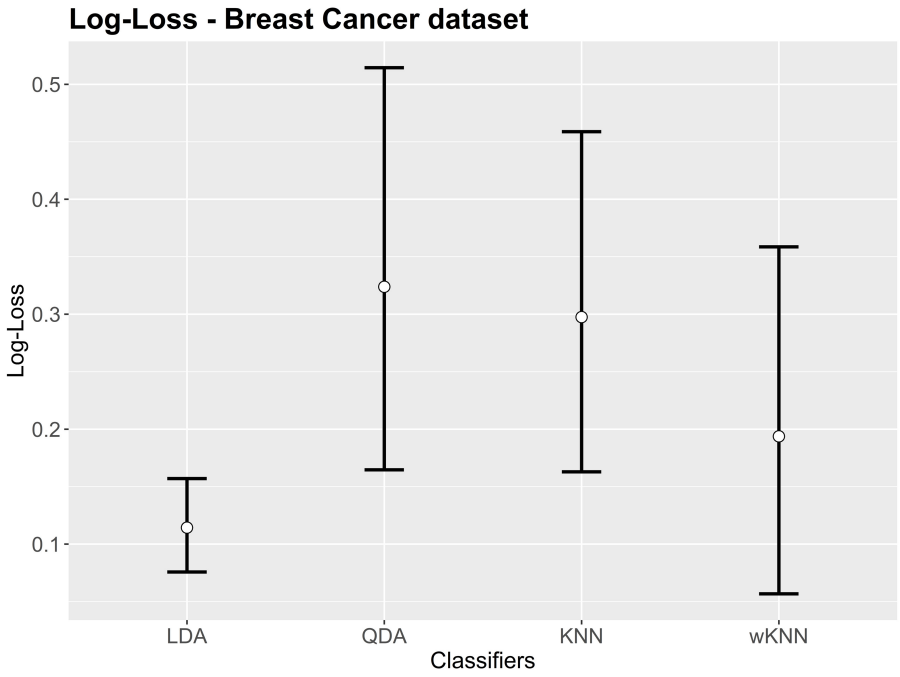

Figure 4: A plot of the mean log-loss (white dots) for each method across all bootstrap samples for the Breast Cancer dataset. The capped black lines are the 95% confidence intervals of the log-loss for each method.

The only statistically significant difference between the recall scores of the methods is that QDA has a higher recall score than KNN. From our previous analysis in this section, we know that KNN seems to suffer from the skewed class distribution, so that it is biased towards classifying tumors as benign. In this context, this bias translates to a high false negative rate. Therefore the low recall score of KNN is to be expected.

5.3 Mushroom Dataset

The second dataset we are going to analyze consists of 8124 hypothetical samples from 23 species of gilled mushrooms in the Agaricus and Lepiota families. Each species is labelled as either poisonous or edible, 4208 of the mushroom samples are edible and 3916 are poisonous. There are 22 explanatory variables that specify different characteristics of the mushrooms such as their size, shape and color. All of these are categorical variables. The classification problem is that of classifying a new sample as poisonous or edible based on the characteristics of the sample as described by the explanatory variables.

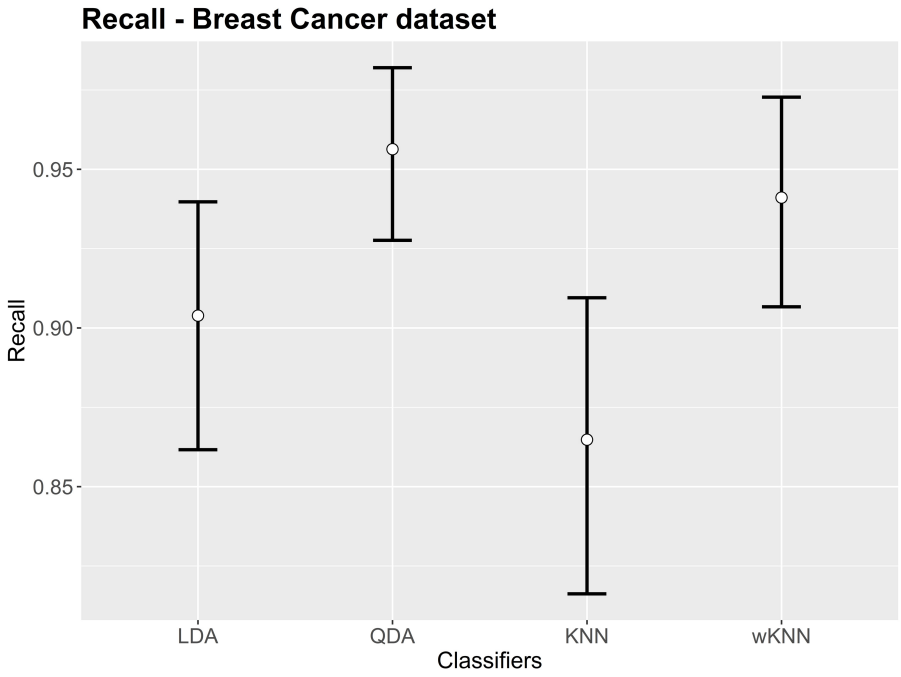

Figure 5: A plot of the mean recall (white dots) for each method across all bootstrap samples for the Breast Cancer dataset. The capped black lines are the 95% confidence intervals of the recall for each method.

5.3.1 Results and Analysis

The number of bootstrap samples used with this dataset is 100. Using cross-validation the estimated optimal value of k that will be used is 1 for both KNN and wKNN. The associated cross-validation curves can be found in section 8.2 in the Appendix. Since this means that both methods will classify to the class of the closest neighbor, KNN and wKNN will be equivalent for this classification problem. Since all of the variables in the dataset are categorical, the LDA and QDA assumptions of normality are clearly violated for this classification problem. The number of effective parameters for each method are shown in Table 2. From Table 2 we see that the nearest neighbor approaches have considerably more effective parameters for this classification problem than LDA and QDA. Furthermore QDA has more effective parameters than LDA.

In Figure 6 the accuracy scores that the different methods achieved for this classification problem are shown. We can see that all methods achieved a very high accuracy for this classification problem, above 0.999 for all methods. For QDA, KNN and wKNN, the upper bound of the 95% confidence interval for the accuracy score is 1.0; for KNN and wKNN the lower bound is also 1.0.

Method	#Parameters
LDA	23
QDA	276
KNN	8124
wKNN	8124

Table 2: The number of effective parameters for each method with the Mushroom dataset.

This means that the accuracy of the nearest neighbor methods can be said to be optimal for this classification problem.

The only statistically significant difference between the accuracy scores of the methods is that LDA has a slightly lower accuracy than KNN and wKNN. Since the nearest neighbor approaches have considerably more parameters, it seems LDA underfits the data slightly in comparison. A natural explanation for this seems to be that the normality assumption of LDA, assumption (8), is violated for this classification problem. However, since QDA does not perform significantly worse than the nearest neighbor approaches in terms of accuracy this explanation is unsatisfactory, given that the normality assumption of QDA is also violated. A more plausible explanation for the slightly inferior performance of LDA seems to be that the Bayes decision boundary for this problem is non-linear. This is further supported by the fact that the nearest neighbor methods had the lowest test error rate with 1 neighbor ($k = 1$). Since this is the most flexible variation of these classifiers, it suggests that the Bayes decision boundary is irregular and non-linear.

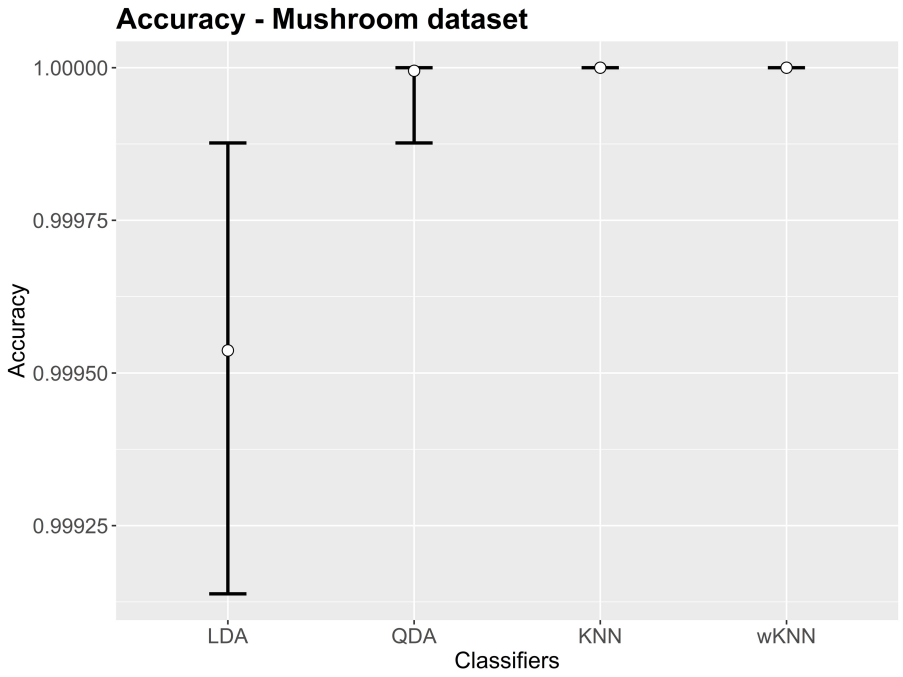

Figure 6: A plot of the mean accuracy (white dots) for each method across all bootstrap samples for the Mushroom dataset. The capped black lines are the 95% confidence intervals of the accuracy for each method.

Lets now see how the methods performed in terms of the log-loss score. In Figure 7 the log-loss scores are summarized for each method. Like with the accuracy score the only statistically significant difference between the log-loss scores of the methods is that LDA performs worse than the nearest neighbor methods. It is reasonable to conclude that the reasons for the difference between log-loss scores are the same as for the difference in the accuracy scores. This is because the log-loss can be thought of as a weighted version of the accuracy score as we explained in section 5.1.1.

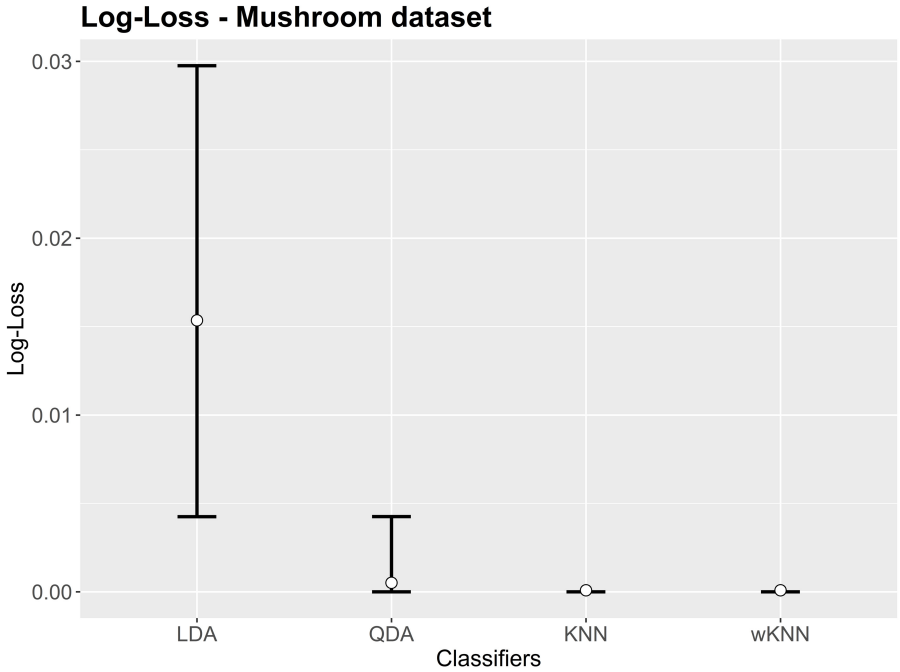

Figure 7: A plot of the mean log-loss (white dots) for each method across all bootstrap samples for the Mushroom dataset. The capped black lines are the 95% confidence intervals of the log-loss for each method.

All precision scores and recall scores were computed for the class poisonous. In terms of precision, all methods obtained an optimal estimated precision of 1.0. All lower and upper bounds for the 95% confidence intervals were also 1.0. Thereby all methods seem to avoid false positives completely. For this classification problem this means that the methods will not classify a new mushroom as poisonous by mistake.

The recall scores for the methods are shown in Figure 8. We can see that all methods have high recall scores, above 0.997. The only statistically significant difference between the methods is that LDA has a slightly lower recall score than KNN and wKNN. Since there is no difference in the precision score between the methods, we conclude that the relative bias of LDA is in the direction of classifying to the class edible.

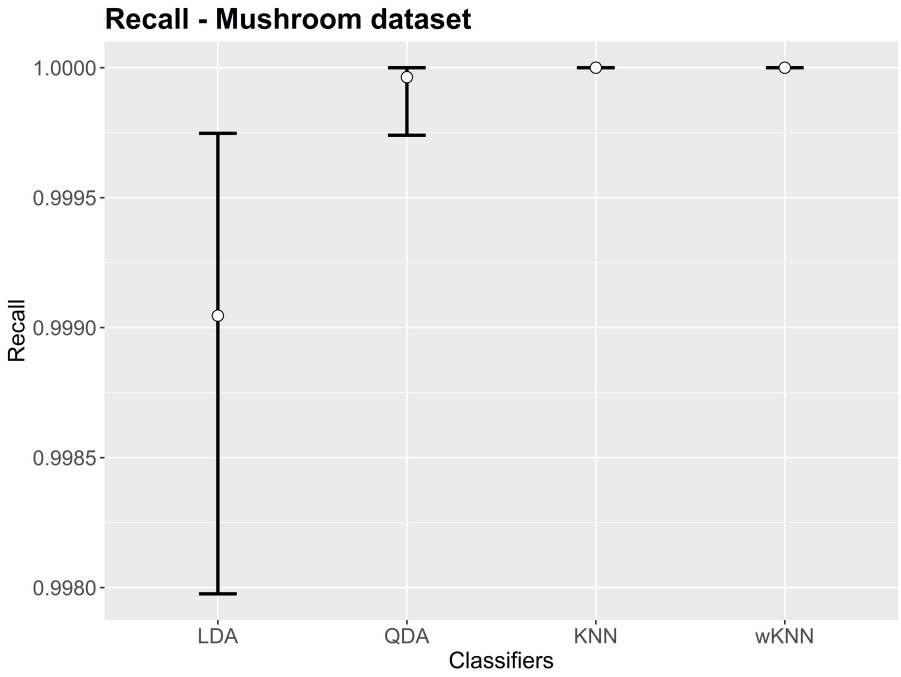

Figure 8: A plot of the mean recall (white dots) for each method across all bootstrap samples for the Mushroom dataset. The capped black lines are the 95% confidence intervals of the recall for each method.

5.4 Forest Cover Dataset

The third dataset we are going to analyze consists of 581 012 observations where each observation represents a 30x30 meter forest area. The data is labelled with the forest cover type in the area and there are 7 types in total. For each observation 12 cartographic variables (Elevation, Slope, etc.) were recorded. Two of these are categorical variables and the rest are real-valued. The classification problem is that of predicting the forest cover type given the cartographic variables for a new observation.

The two categorical variables, soil type and wilderness area, have 4 and 40 categories respectively. In this dataset each category is represented by a dummy variable resulting in 44 additional columns for these two variables alone. Thereby, the inclusion of these two variables increases the number of independent variable columns in the dataset from 10 to 54.

Because LDA and QDA assume normality for the independent variables, the inclusion of categorical variables that clearly violate this assumption will likely be unfavourable to these methods. We have already evaluated the performance

of these methods with the Mushroom dataset, which only had categorical variables. To obtain a more diverse selection of classification problems we will therefore modify the Forest Cover dataset by excluding the categorical variables.

5.4.1 Results and Analysis

The number of bootstrap samples used with this dataset is 30. Using cross-validation the estimated optimal value of k that will be used is 1 for both KNN and wKNN. The associated cross-validation curves can be found in section 8.3 in the Appendix. Like with the Mushroom dataset from section 5.3, this implies that KNN and wKNN will be equivalent for this classification problem. The number of effective parameters for each method are shown in Table 3. We can see that the nearest neighbor approaches have considerably more effective parameters than LDA and QDA for this classification problem.

Method	#Parameters
LDA	66
QDA	396
KNN	581012
wKNN	581012

Table 3: The number of effective parameters for each method with the Forest Cover Dataset.

Since this is a multiclass problem we will use micro-averaging of the precision and recall scores. For this classification problem the performance of the different methods had the same ranking order across all metrics. The nearest neighbor methods performed significantly better than the discriminant analysis classifiers and LDA performed slightly better than QDA. An illustration of this relationship can be seen in Figure 9, which shows the accuracy scores of the methods and the associated 95% confidence intervals.

A reasonable explanation for the significant difference in performance between the nearest neighbor approaches and the discriminant analysis classifiers is that the latter underfit the data for this classification problem. This seems plausible since the number of effective parameters for KNN and wKNN is considerably higher than for the other methods. This could be because the assumptions of normality that LDA and QDA are based upon are not good approximations of the distribution of the data. It could also be because the Bayes decision boundary for this problem is highly irregular so that the decision boundaries of LDA and QDA fail to capture its shape. The latter hypothesis is supported by the fact that the most flexible version of KNN and wKNN (with 1 neighbor) minimized the estimated test error rate for this classification problem.

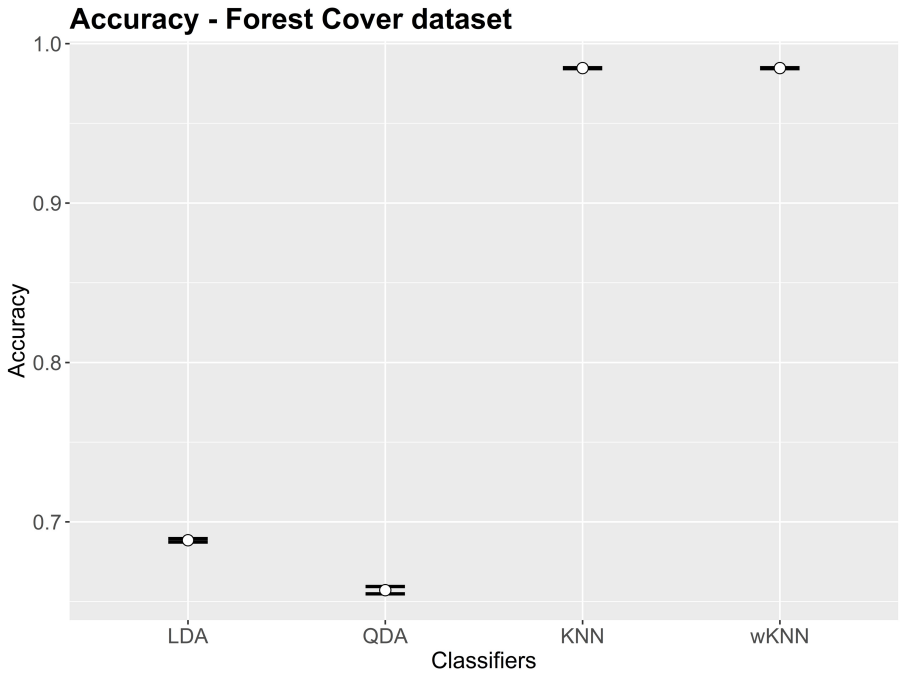

Figure 9: A plot of the mean accuracy (white dots) for each method across all bootstrap samples for the Forest Cover dataset. The capped black lines are the 95% confidence intervals of the accuracy for each method.

Explaining the difference in performance between LDA and QDA is more difficult. Since QDA is a more flexible classifier with more effective parameters than LDA, we expect it to perform better for this classification problem, given that the much more flexible nearest neighbor approaches perform well. Therefore, the slight edge in performance that LDA achieves over QDA across all metrics seems counter intuitive. A plausible explanation for this difference is an open problem that will be addressed in future studies.

6 Discussion

In this section we will summarize the main findings of the analysis from section 5 for the three datasets that were included. We will also discuss our methodology and its limitations.

For the Breast Cancer dataset we found that all classes had good predictive power. We also observed that wKNN had better predictive power than KNN, implying that KNN suffers from the skewed class distribution.

For the Mushroom dataset we found that all classes had very good predictive power. We also observed that LDA had slightly lower predictive power than the nearest neighbor approaches which, performed optimally for this classification problem. QDA did not perform significantly worse than the nearest neighbor methods which suggests that the assumption of normality made by both LDA and QDA was not the reason for LDA underfitting the data. Instead the linear decision boundary of LDA seems to be reason for its relatively poor performance for this classification problem.

For the Forest Cover dataset we found a large discrepancy in performance between the discriminant analysis approaches and the nearest neighbor methods, the latter had significantly higher predictive power. These results suggest that the Bayes decision boundary is irregular and non-linear for this classification problem. Furthermore, LDA performed slightly better than QDA across all metrics.

The results we have obtained imply some general guidelines for the applicability of the studied classification methods:

- The assumptions of normality made by LDA and QDA should be viewed as conditions for optimality as oppose to a strict requirements for using these classifiers.
- When the class distribution is skewed KNN suffers significantly in terms of predictive power. wKNN alleviates this problem effectively and should be used instead of KNN in these situations.

In our analysis we focused primarily on the relative differences in performance between the methods, as oppose to the absolute differences. The motivation for this is that absolute differences in performance are only meaningful in the context of specific classification problems. For instance, consider the Mushroom dataset from section 5.3. We observed that there was a slight difference in accuracy between the nearest neighbor methods and LDA. This difference of roughly 0.05% could be meaningful if we are looking to use this classifier at a large scale, and if the risks associated with misclassification are considerable. For this problem in particular, the consequences of misclassification could be that mushrooms are classified as edible when they are poisonous, which is a very undesirable outcome. For a different

classification problem this difference might be virtually meaningless. The purpose of our study has been to gain insight into the properties of the classification methods and the differences between them. Therefore we have chosen to treat the datasets as generic classification problems, and to place an emphasis on relative differences in our analysis.

We compared the performance of the classification methods using three datasets. An attempt was made to diversify this selection, by using datasets of different sizes and removing the categorical variables in the Forest Cover dataset. Although these measures likely helped in reducing the bias of our results, the limited number of datasets is a source of considerable bias. This bias limits the extent to which our findings can be reliably generalized to a variety of classification problems. Further study into the differences between these classifiers, using a larger sample of datasets, would facilitate our understanding.

The number of bootstrap samples used for computing the confidence intervals of the evaluation metrics was limited by computational resources. Using fewer bootstrap samples results in more biased estimates of the confidence intervals [14]. It is possible that more accurate confidence intervals would have changed some of the results of our analysis. For instance, some of the observed differences in the evaluation metrics of the classifiers could have been deemed statistically insignificant, if we had obtained different estimates of the confidence intervals.

When comparing the performance of the classifiers, we considered the possibility that the assumption of normality made by LDA and QDA did not hold. With the Breast Cancer dataset we performed the Royston H test for multivariate normality and found that the data violated this assumption. With the Mushroom dataset we could assert that since its variables are categorical it clearly violates the assumptions of LDA and QDA. We were unable to determine whether these assumptions were met for the Forest Cover dataset however. Because of computational limitations multivariate normality testing was not possible for this dataset due to its large size.

7 Conclusion

The objective of our study has been to gain a better understanding of the classification methods Linear Discriminant Analysis (LDA) and K-Nearest Neighbors (KNN), as well as extensions of these methods. The purpose of the study has been to help potential users of these classifiers in evaluating their applicability for different classification problems.

For each class, LDA and QDA assume a multivariate gaussian distribution for the independent variables. This is a strong assumption that limits the applicability of these methods in theory. However, our experimental results suggest that even when this assumption does not hold, LDA and QDA can perform well. In light of these results, we recommend that potential users of these classifiers should not dismiss them entirely when this assumption is violated.

The simplicity of the discriminant analysis methods makes them appropriate when the sample size of the data is small. In these situations the data will not contain enough information to support a complex decision boundary between the classes. Even though the nearest neighbor classifiers may achieve lower bias they will likely also suffer from higher variance under these circumstances. We recommend that users consider LDA or QDA instead of the nearest neighbor methods when the sample size is small.

K-Nearest Neighbors is a versatile classifier in general since it does not make explicit assumptions about the distribution of the data. In theory, it performs well when enough observations are available to support a complex decision boundary. Our experimental results support this finding. For the Forest Cover dataset the nearest neighbor methods outperformed LDA and QDA with a large margin.

The predictive power of the nearest neighbor classifiers diminishes as the number of independent variables increases. This limitation should be considered when applying these classifiers. Another problem with K-Nearest Neighbors is that when the majority voting scheme is used, it will be biased towards the more prevalent class if the class distribution is skewed. This tendency of KNN was supported by our experimental results and we found that wKNN alleviates the problem effectively. Because of this, we recommend potential users of KNN to examine the class distribution and, if it is not roughly symmetrical, to use wKNN instead.

8 Appendix

8.1 Breast Cancer Dataset

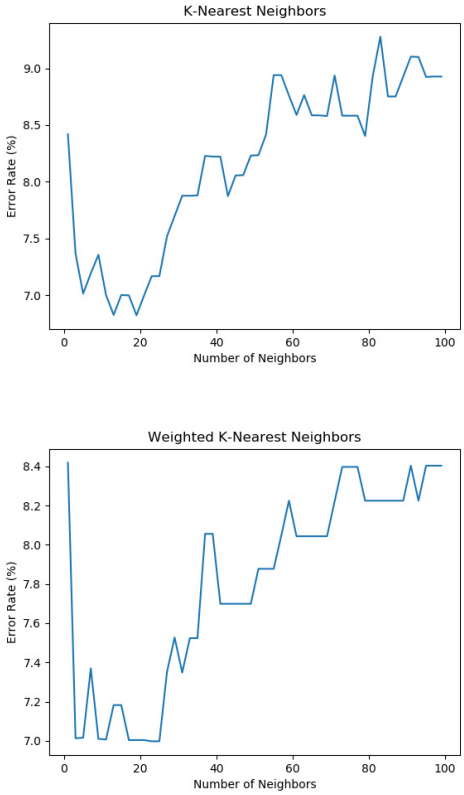

Figure 10: Cross-validation curves for KNN (top) and wKNN (bottom) for the Breast Cancer dataset, showing the test error rate for values of k between 1 and 100.

8.2 Mushroom Dataset

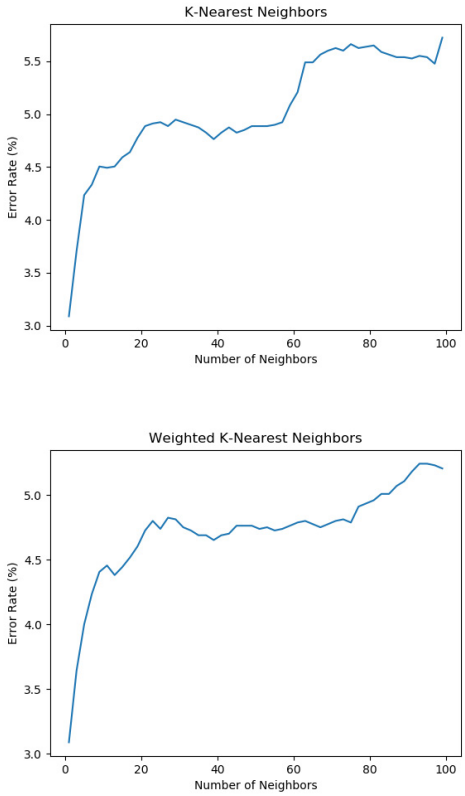

Figure 11: Cross-validation curves for KNN (top) and wKNN (bottom) for the Mushroom dataset, showing the test error rate for values of k between 1 and 100. The characteristic U-shape of cross-validation curves is not seen in these figures. This might have interesting implications that should be explored in future studies.

8.3 Forest Cover Dataset

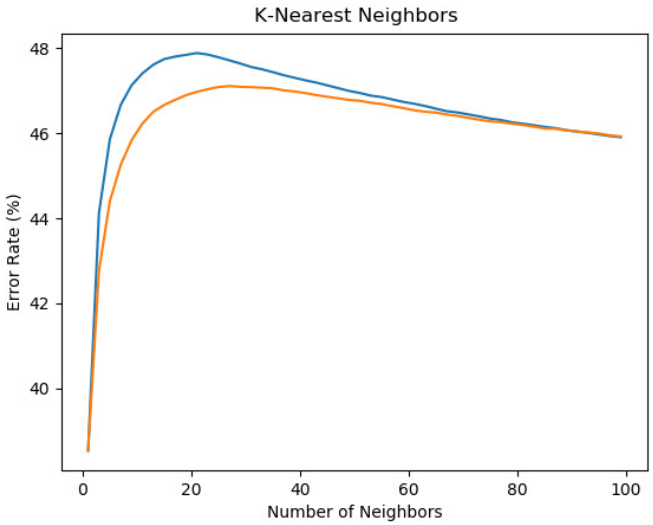

Figure 12: Cross-validation curves for KNN (blue) and wKNN (orange) for the Forest Cover dataset, showing the test error rate for values of k between 1 and 100. The characteristic U-shape of cross-validation curves is not seen in this figure. This might have interesting implications that should be explored in future studies.

References

[1] Trevor Hastie, Gareth James, Robert Tibshirana, and Daniela Witten. *An Introduction to Statistical Learning*. Germany, Berlin: Springer; 2013

[2] William G. Macready, David H. Wolpert. *No Free Lunch Theorems for Optimization*. IEEE Transactions on Evolutionary Computation. (1997); 1(1): p.67-82.
Available from: doi:10.1109/4235.585893
[Accessed 2nd December 2018]

[3] Melody Y. Kiang. *A comparative assessment of classification methods*. Decision Support Systems. (2003); 35(4): p.441-454.
Available from: doi:10.1016/S0167-9236(02)00110-0
[Accessed 2nd December 2018]

[4] Jerome Friedman, Trevor Hastie and Robert Tibshirana. *The Elements of Statistical Learning*. 2nd ed. Germany, Berlin: Springer; 2009

[5] Ron Kohavi. 1995. A study of cross-validation and bootstrap for accuracy estimation and model selection. In Proceedings of the 14th international joint conference on Artificial intelligence - Volume 2 (IJCAI'95), Vol. 2. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1137-1143.

[6] Andrew Ng. CS229 Lecture notes part IV: Generative Learning algorithms. [Lecture] Stanford University.
Available from:
`https://see.stanford.edu/materials/aimlcs229/cs229-notes2.pdf`
[Accessed 2nd December 2018]

[7] Songbo Tan. *Neighbor-weighted K-nearest neighbor for unbalanced text corpus*. Expert Systems with Applications. (2005); 28(4): p.667-671. . (2016); 48(2): p.331–378.
Available from: doi:10.1016/j.eswa.2004.12.023
[Accessed 3rd December 2018]

[8] Zoltan Geler, Mirjana Ivanović, Vladimir Kurbalija, Miloš Radovanović. *Comparison of different weighting schemes for the kNN classifier on time-series data*. Knowledge and Information Systems. (2016); 48(2): p.331–378.
Available from: doi:10.1007/s10115-015-0881-0
[Accessed 2nd December 2018]

[9] Charu C. Aggarwal, Alexander Hinneburg, and Daniel A. Keim. 2001. On the Surprising Behavior of Distance Metrics in High Dimensional Spaces. In Proceedings of the 8th International Conference on Database Theory (ICDT '01), Jan Van den Bussche and Victor Vianu (Eds.). Springer-Verlag, Berlin, Heidelberg, 420-434.

[10] Dua, D. and Karra Taniskidou, E. (2017). UCI Machine Learning Repository. Irvine, CA: University of California, School of Information and Computer Science.
Available from: `http://archive.ics.uci.edu/ml`
[Accessed 15th November 2018]

[11] Guy Lapalme, Marina Sokolova *A systematic analysis of performance measures for classification tasks*. Information Processing and Management. (2009); 45(4): p.427-437.
Available from: doi:10.1016/j.ipm.2009.03.002
[Accessed 15th November 2018]

[12] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Germany, Berlin: Springer; 2006

[13] Patrick Royston. *Some Techniques for Assessing Multivariate Normality Based on the Shapiro-Wilk W*. Journal of the Royal Statistical Society, Series C (Applied Statistics). (1983); 32(2): p.121-133.
Available from: doi:10.2307/2347291
[Accessed 2nd December 2018]

[14] Russell Davidson James G. MacKinnon. *Bootstrap Tests: How Many Bootstraps?*. Working Papers 1036, Queen's University, Department of Economics. Available from:
`https://ideas.repec.org/p/qed/wpaper/1036.html`
[Accessed 2nd December 2018]