KU LEUVEN

FACULTY OF SCIENCES

Visualizing Classification Results

Nura Kawa

Supervisor: Prof. Peter Rousseeuw
Katholieke Universiteit Leuven

Master thesis submitted in fulfillment
of the requirements for the degree in
Master of Science in Statistics and Data Science

Academic year 2020-2021

Summary

When a classification model does not meet the objectives of its deployment settings, there arises a need for model interpretability. A practitioner or stakeholder would desire more information about the results of a classifier than what can be gleaned from evaluation metrics. Insight into for which type of examples and for what reasons a model makes errors can help determine ways to improve the model.

To that end, the class map was proposed by Raymaekers et al. (2021) as a visual tool that provides post-hoc interpretations of classification results. For each observation in a training or test set, the class map reflects the probability that the observation belongs to an alternative class. It also shows an observation's farness, a measure of how far it is from its own class. Both quantities are derived from a trained classifier. The class map provides useful information about the data and the classifier, from which a user could infer reasons behind a classifier's results.

Classification tasks are especially challenging when the data contains a large proportion of observations with data difficulty factors, which can be due to errors in data collection or due to class imbalance. The class map's ability to reveal the level of difficulty of observations in the data is limited by the choice of classifier. Thus, this thesis aims to adapt the class map to use a classifier-agnostic measure of where an object lies in proximity to its class. The resulting adaptation is the proposed localized class map. Farness is replaced by localized farness, which measures the probability an object lies in its class using kernel-weighted local neighborhood distances. The localized class map successfully provides explanations of any classifier. Real-world data examples show how the localized class map can identify whether a model makes errors on observations that are easy or challenging to classify, and whether the data to be classified consists of homogeneous or heterogeneous classes.

This thesis illustrates additional applications of the localized class map, including comparing the performance of different classifiers and assessing a classifier's fairness with respect to sensitive attributes in the data. The implementation of the localized class map, as well as code for examples presented in this thesis, is written in R and is publicly available.

Acronyms

ANN Approximate Nearest Neighbor.

LF Localized farness.

MDS Multidimensional Scaling.

PAC Probability of Alternative Classification.

t-SNE t-distributed Stochastic Neighbor Embeddings.

List of Figures

List of Tables

Contents

Chapter 1

Introduction

1.1 Bias in Classification

In statistical learning, the problem of classification is, given a set of training data $X = (x_1, x_2, \dots x_n)'$ and categorical labels $y = (y_1, \dots y_n)$, to find a function $f(X)$ that can identify y. The final classifier, $f^*(X)$, is typically the minimizer of a loss function that quantifies the difference between f and the data. Most classification algorithms make an assumption that f belongs to a specific class of function. For example, Linear Discriminant Analysis assumes that f is a linear function. This assumption on the form of f, or the assumptions made by a particular model, is a *bias* in the model, often called *inductive bias*. In this thesis, we use the definition of bias from Mitchell (1980), which is "any basis for choosing one generalization over another other than strict consistency with the observed training instances."

Some inductive bias is necessary for classification. Having too little inductive bias, where f is allowed to take any form, can lead to an f^* that overfits, or memorizes the training data, and cannot generalize to unseen data. Having too much inductive bias can lead to underfitting, where f^* is a poor fit to both the training and unseen data. *Hyper-parameter bias* (Hellström et al., 2020), or the user's choice of model specifications, goes hand-in-hand with inductive bias. A model can perform poorly if its hyper-parameters, such as the number of hidden layers and types of activation functions in a neural network, are poorly chosen.

The choice of classification model is often based on observed qualities of the training data X. For instance, if X contains numerical values, and the distribution of X is multivariate normal in each class, it would be suitable to fit a linear model. If X contains both categorical and numerical values, one might wish to use a decision tree instead. More generally, especially when X is very large, the choice of model becomes a compromise between its fit to the data and its complexity, or the number of parameters in the model (Fürnkranz and Kliegr, 2018).

However, if the data does not accurately represent real-world settings, it can become a source of bias in classification problems. Data collection often comes with errors, either from the mechanism used to collect it or from the source of the data (Hellström et al., 2020). This includes missing features, incorrect labeling, missing values, or poorly-defined features or labels. In some cases, the choice of the form of f will be one that is suited for error-ridden data, and the classifier will perform poorly on unseen data. A good example to illustrate this is a decision tree, which assumes f to be a nested collection of decision rules. If the data contains errors, the classifier will make rules that do not accurately represent real-world deployment settings. While these decision rules are consistent with the training data, we argue that such a model is biased, because it is not trained with data that is a true representation of its deployment settings. Therefore, we add to our definition of bias that "observed training instances" should be representative training instances. Mitchell (1980) identifies that lack of knowledge about the source of training data can lead to bias. For example, if one feature of the training data is the height in centimeters of human beings, knowledge that negative values are impossible is an important constraint on learning. Without this knowledge, one could yield a biased classifier.

In order to yield the best classification function for a given problem, it is essential to identify sources of bias. In this thesis, we use the terms

- "model-driven bias" to refer to bias that comes from the choice of model. Choosing a linear model to classify data that is not linearly separable is an example of model-driven bias.

- "data-driven bias" to refer to bias that comes from training a model with data that does not accurately represent the setting of the problem. Biased data could contain incorrect labels, imbalance, inappropriate features, or missing features. A model trained on such data could "learn" incorrect associations and perform poorly on unseen data

Error resulting from model-driven bias can be reduced by adjusting the choice or specifications of the classification model. Data-driven bias can be addressed through addressing issues in the data-collection process and appropriate cleaning and preprocessing.

1.2 The Need for Interpretable Models

When there is a mismatch between the formal objective of a classification model and the objective of its deployment settings, there arises a demand for interpretability (Lipton, 2018). For example, a classification model could be optimized for predictive performance on a held-out dataset (a test set), but the real-world purpose is for the model to provide information. Essentially, users can want more from the model than just predictions.

This is especially true for cases where the classification model makes mistakes, or misclassifications. Some users demand *transparency*, or the mechanism for how the model makes decisions, to explain the errors. Others want *post-hoc interpretations*, which explain the model's predictions without necessarily revealing how the model works. The question of trust in deploying the model without human supervision can also come into play. A user might wish to know, for which examples is the model making mistakes? Is the model misclassifying observations that are also challenging for humans to label? Or, is the model making mistakes where humans typically would not? If the answer is the latter, then a user might not wish to deploy the model.

Lipton (2018) argues that the goal of building transparent models, or models which can be contemplated all at once by a human, might be at odds with other goals, such as building models that classify observations that humans cannot. Instead, we can focus on developing post-hoc interpretations of non-transparent classification models. These explanations of a classifier's predictions might not reveal the inner workings of the classifier, but could elucidate sources of its bias. For instance, post-hoc interpretations could allow a user to know that a classifier misclassified an observation because it is very different from other observations of the same class. Or, if a classifier makes mistakes on "easy" observations, it could be poorly-chosen for the data. There is much need for methods to explain the results of classification models beyond what can be gleaned from formal evaluation metrics.

1.3 Class Maps

Visualization is a common approach to post-hoc interpretation; Lipton (2018) provides examples of a few methods that focus on visualizing the results of neural networks. In the setting of classification, visualizing aspects of the classifier's decisions could help practitioners identify sources of bias, both model-driven and data-driven. While there exist several popular visualization tools that reflect model accuracy measures, such as the ROC and F1 curves, and many methods for visualizing data, such as multi-dimensional scaling and t-SNE, there are few visualizations that consider both at the same time. To this end, Raymaekers et al. (2021) introduces the *class map*, which reflects a measure of model-driven bias against a measure of data-driven bias. The map plots, for each observation in a training or test set, the probability that the observation belongs to an alternative class against how far it is from its own class. Both quantities are derived from the trained classifier; thus, the visualization reflects where observations lie *from the perspective of the classifier*.

If the assumed form of the classification function is a poor match for the data, this is reflected in the class map. Model-driven bias is shown when a classifier yields a high probability of alternative classification to points that lie close to their own class. This implies that the classifier is unable to correctly identify core members of a class. However, the measure of how far an observation lies from its class, or *farness*, might not fully reflect the extent of data-driven bias. Farness is based on the distance of an object to its class, where the distance metric is with respect to the classifier. If the classifier is poorly chosen for the data, then the distance metric might miss trends in the data that explain data-driven bias. Furthermore, many classification algorithms, such as decision trees, neural networks, and ensembles, do not use distances or dissimilarities in the data to make decisions. Therefore, it is useful to find a measure of data-driven bias that is classifier-agnostic and to incorporate that into a post-hoc interpretation.

1.4 Data Difficulty Factors

Data-driven bias often comes hand-in-hand with *data difficulty factors*, or characteristics of data that make it challenging to classify. Generally, observations that are easy to classify are similar to the majority of their class members; they share a distinct set of characteristics with their classmates. Difficult observations are often atypical examples of their classes. Even without noise, datasets tend to contain both easy and difficult observations. However, in the presence of data-driven bias, a large number of observations, even entire classes, can have qualities that make them very difficult to classify. For example, mislabeled points can display typical characteristics of one class, but have the label of a class with very different properties. Getting an idea of the data's structure, both within and between classes, can allow us to identify the presence of difficult points. It follows that assessing data difficulty can also reveal data-driven bias.

Writing in the context of imbalanced classification, or problems where at least one class has disproportionately few observations, Napierala and Stefanowski (2015) hypothesize that the classification difficulty level of the data can be approximated by analyzing the structure of its observations. With the common assumption that observations that lie in close proximity are similar, the authors define a taxonomy of *data difficulty factors* of an observation with respect to its position in the data space. These factors are:

- *safe* observations, which are located in close proximity to the majority of observations in their class

- *borderline* observations, which are located at decision boundaries between overlapping classes
- *rare* observations, which exist in small sub-concepts far from decision boundaries and far from the majority of class members
- *outliers*, which are singular observations that are very different from their class. These can be the result of mislabeling, or of an insufficiently covered sample space

The taxonomy is ordered from easiest to hardest for a classifier to identify. The authors believe that determining an observation's placement in the taxonomy, as well as the ratio of examples in a dataset, would help practitioners assess the difficulty of the classification problem.

We believe that, in addition to the authors' use case, looking at observations in light of this taxonomy could help identify the presence of data-driven bias. For example, "outlier" examples could be the result of mislabeling. Rare observations could exist due to sampling bias, where sub-populations of a class are not represented in the data. Borderline observations could exist due to a fuzzy definition of class labels. Thus, with the taxonomy of Napierala and Stefanowski (2015) as a reference, we hypothesize that there is a link between the structure of the data and the presence of data-driven bias in classification that can be measured at the level of the individual observation.

1.5 Problem Statement

Having identified the need for interpretable classification models, we find that post-hoc explanations are a promising solution. The class map of Raymaekers et al. (2021) provides explanations of classification models by visualizing data from the perspective of the classifier. This allows practitioners to get interpretations of classification error in a way that has not been done before. However, we believe that there is value in creating a post-hoc explanation tool that explains classification bias in a way such that model-driven bias and data-driven bias are decoupled. We therefore aim to develop an extension of the class map that measures data-driven bias in a classifier-agnostic way; essentially, from the perspective of the data. With such a visualization, practitioners can accurately determine whether they must change their choice of classification model or adjust their data collection process.

In short, this thesis focuses on the following problem:

How can we adapt the class map to get explanations of classification models where model-driven bias and data-driven bias are measured independently?

Our proposed solution further serves as an extension of the class map to explain the results of classifiers that do not define a distance. We also explore further applications of the class map, such as comparing different classifiers on the same data.

1.6 Thesis Organization

We begin by defining properties of a class map and proposing the *localized class map*, our adaptation of the class map. We do so by reproducing an example from Raymaekers et al. (2021) and identifying similarities and differences between the localized class map and the class map. We briefly discuss the parametrization of the localized class map and the `R` implementation of our visualization. The second part of this paper contains examples of the localized class map used to explain classifications of real-world datasets. Each example focuses on an application of the localized class map. We first look at how localized class maps can allow for a simple comparison of base classifiers on the same data. We then show how localized classmaps are particularly useful for classification problems with heterogeneous classes. Finally, we look at an example of how class maps can visually assess fairness criteria of a classification algorithm. We end with a discussion of our main findings and ideas for further research.

Chapter 2

Methods

2.1 The Class Map

We describe the class map of Raymaekers et al. (2021) with the same illustrative example the authors present in the first two sections of their paper. We focus on defining the principles and interpretation of the class map.

Suppose we wish to build a model that classifies observations in the Floral buds data, originally used in Wouters et al. (2015). The 550 observations in this dataset come from four classes of floral buds: “branch” (49 members), “bud” (363 members), “scale” (94 members) and “support” (44 members). The counts of observations within groups indicates class imbalance: the majority of observations belong to class “bud”, leaving the remaining classes to be minority classes. After exploring the distribution of the data, we opt to classify the data with Quadratic Discriminant Analysis. A common way to visualize the results of this classification is with a stacked mosaic plot, which is displayed in Figure 2.1. This visualizes class proportions in the data (x-axis) against predicted class proportions of the model (y-axis).

The stacked mosaic plot reveals important information about both the classifier and the data. First, it reveals the extent of class imbalance; class “bud” has the largest area in the chart. It also shows that the classifier has a high overall accuracy, as the predicted and given class are the same for most observations. The map also reveals that the classifier has the most challenge in identifying branches and supports. However, there is still the unanswered question of *why* the classifier finds difficulties with certain classes. The class map provides this answer by looking at the classifier’s view of each observation in a class.

Figure 2.2 shows the class map for observations with the ground-truth label “scale”, one of the minority classes in the data. In the stacked mosaic plot, it is represented by the third vertical rectangle. Like the stacked mosaic plot, the class map shows

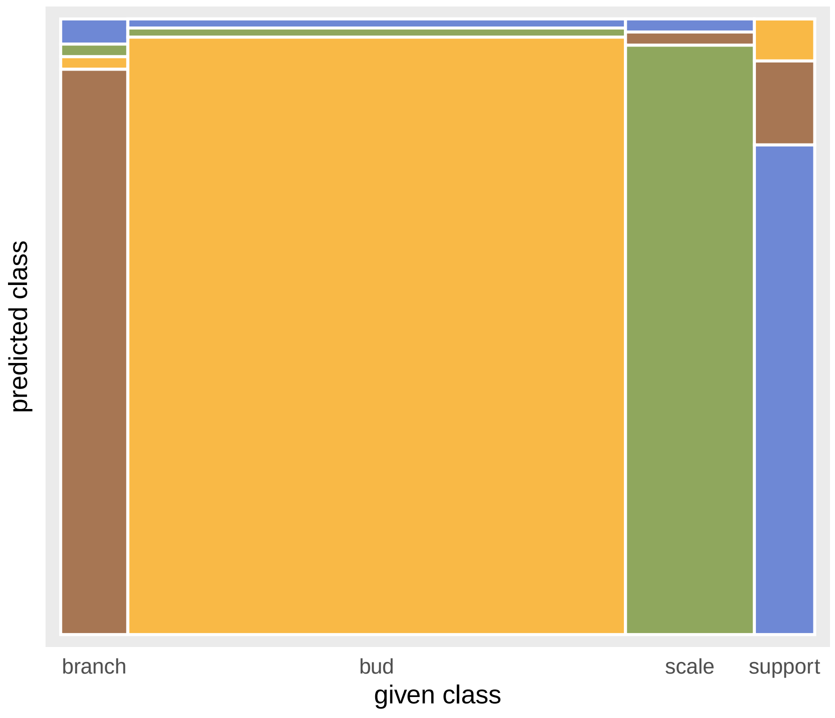

Figure 2.1: Stacked mosaic chart visualizing the classification of Floral buds data.

information from the model is on the y-axis, and information from the data on the x-axis. However, rather than visualizing proportions, the class map reveals more information by visualizing probabilities.

The y-axis is the Probability of Alternative Class (PAC), or the probability that an observation belongs to a class that is different from its given class. The x-axis is the farness, also defined as a probability, which quantifies how far an object lies from its class. Each observation is colored by the class label that the classifier assigns to it. Figure 2.2 shows, like the stacked mosaic plot, that the classifier correctly identifies the label of all but a few scales. In fact, the classifier sees the majority of scales as located safely inside of their class. The classifier finds four points to be highly likely to belong to an alternative class; all such points are located relatively far from the majority of their class. This gives us more information than does the stacked mosaic plot: that the classifier only misclassifies scales are relatively atypical. This indicates that some scales lie between decision boundaries of the classifer. On the bottom-right corner of the class map, some points are outlined in black. These are denoted as outliers, points that are far from all classes. The classifier performs well on these outliers; this means that, in the eyes of the classifier, these points still look like their class, despite being located far from its center.

This visualization can be seen as a post-hoc interpretation of the QDA model for the ground-truth class "scales." A user can get an idea of how the classifier sees the data,

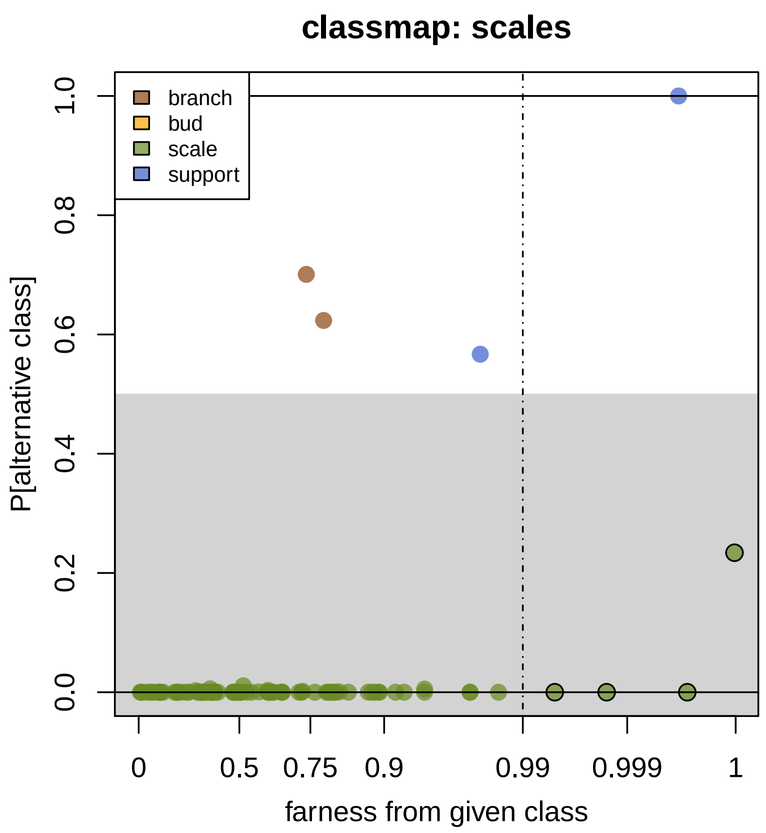

Figure 2.2: Class map of the Floral Buds data for class “scales”.

the class-level accuracy rate of the classifier, and on which observations the classifier makes mistakes.

2.1.1 Probability of Alternative Class

The first component of the class map, Probability of Alternative Classification (PAC), is the probability that an observation belongs to an alternative class, from the perspective of the classifier. It can be thought of as a measure of model-driven bias. A model that makes correct assumptions of the learning setting will make errors only on observations that are difficult to classify. A biased model sees a point that lies comfortably in its class as having a high probability of belonging to an alternative class.

Keeping the same notation as Raymaekers et al. (2021), suppose we have objects i, where $i = 1 \dots n$, and classes g, where $g = 1 \dots G$; object i has label g_i. Either directly or indirectly, the classification function f^* outputs posterior probabilities that object i belongs to each class g, which we denote as $\hat{p}(i,g)$, where $\sum_{g=1}^{G} \hat{p}(i,g) = 1$. The output of f^* is therefore

$$f^*(i) = \arg\max_g \hat{p}(i, g) \tag{2.1}$$

If the classifier predicts correctly, then this output is g_i, its ground-truth label. If not, then we can get an idea of the classifier's bias by measuring to what extent its highest posterior probability agrees with that of g_i.

To do so, we define $\tilde{p}(i)$ to be an object's maximum posterior probability for the group that is not its label g_i:

$$\tilde{p}(i) = \max(\hat{p}(i, g); g \neq g_i) \tag{2.2}$$

and compare it to $\hat{p}(i, g_i)$. If the next best posterior probability is less than that of g_i, then it means that $f^*(i) = g_i$. If not, then the classifier misclassifies object i, assigning it to an alternative label.

We define PAC as the conditional posterior probability of object i belonging to an alternative class:

$$\text{PAC}(i) = \frac{\tilde{p}_i}{\hat{p}(i, g_i) + \tilde{p}_i} \tag{2.3}$$

What does this tell us? If the classifier finds absolute certainty that object i belongs to its label g_i, then PAC$(i) = 0$. If the classifier is certain that i belongs to an alternative class, then PAC$(i) = 1$. If there is any uncertainty associated with the classifier's decision, then PAC will lie in $(0, 1)$. If the classifier finds object i to be equally likely to belong to two classes (in this scenario, the decision is usually taken randomly), then PAC$(i) = 0.5$. In short, this measure quantifies model-driven bias, or to what extent model's assumptions of the data differ from the actual data.

2.1.2 Farness

The second component of the class map, farness, measures how well an object fits its ground-truth class, in the eyes of a classifier. If item i has low farness, it is close to its given class g_i. If it is very different from its own class, it has high farness. If i has very high farness, it could potentially have been mislabeled. If a class contains many observations with mid-to high farness, then perhaps the class definition is fuzzy.

To compute farness, we begin with a distance $D(i, g_i)$ from on item i to its class g_i. D is defined with respect to the classifier. For instance, when using a k-nearest neighbor classifier, D would be the nearest-neighbor distance; when using Discriminant Analysis, D would be Mahalanobis distance.

Once D is chosen, we estimate for each group the cumulative distribution function (cdf) of $D(x, g_i)$, where x is a random object from g_i. The cdf is a function that gives the probability that a random variable is less than or equal to some value. Computing this cdf in $D(i, g_i)$ gives us the probability that the distance of x, a random member of group g_i, to its group is less than or equal to the distance of object i to its group.

$$\text{farness}(i) = P[D(x, g_i) \leq D(i, g_i)] \tag{2.4}$$

With this definition, farness is also a probability and lies in range $[0, 1]$. A key component of farness is selecting D to be that which is used by the classifier. A poorly-chosen classifier for the data could have a biased sense of the structure of the data. For instance, if a practitioner opts to use Discriminant Analysis on data that does not meet its assumptions, then the notion of farness would not reflect the true nature of the data.

2.2 The Localized Class Map

We motivate the localized classmap by looking at a different view of the Floral Buds data. Figure 2.3 displays a t-distributed Stochastic Neighbor Embeddings (t-SNE) view of the Floral buds data. t-SNE (van der Maaten and Hinton, 2008) is a nonlinear dimensionality reduction technique for visualizing large datasets. It is similar to Multidimensional Scaling (MDS), which aims to preserve the distance between data points in their original space. t-SNE focuses on preserving distances in neighborhood space; therefore, classes will appear closer together in t-SNE than in MDS. t-SNE models pairwise dissimilarities in the feature space as a joint probability distribution, where distances between points are measured with respect to a Gaussian distribution. It then finds a 2- or 3-dimensional representation of this probability distribution (replacing the Gaussian with a student t-distribution) that preserves neighborhood similarities of the feature space. The result is a map that reveals the local and global structure of the data.

From Figure 2.3 we notice not only class imbalance, but also data difficulty factors. Buds and scales tend to form tight-knit neighborhoods located in roughly the same area of the space of low-dimensional representations. Supports and branches, however, are more dispersed across the space and do not form one solid cluster. Their structure, combined with their small sample sizes, could confuse the classifier.

With this in mind, it is useful to evaluate classifier performance in the context of the local neighborhood distribution of the data; essentially, to translate the information we get from the t-SNE embeddings to a class map. To this end, we propose the *localized class map*, which compares a measure of *localized farness* computed from the data against the PAC computed from the classifier.

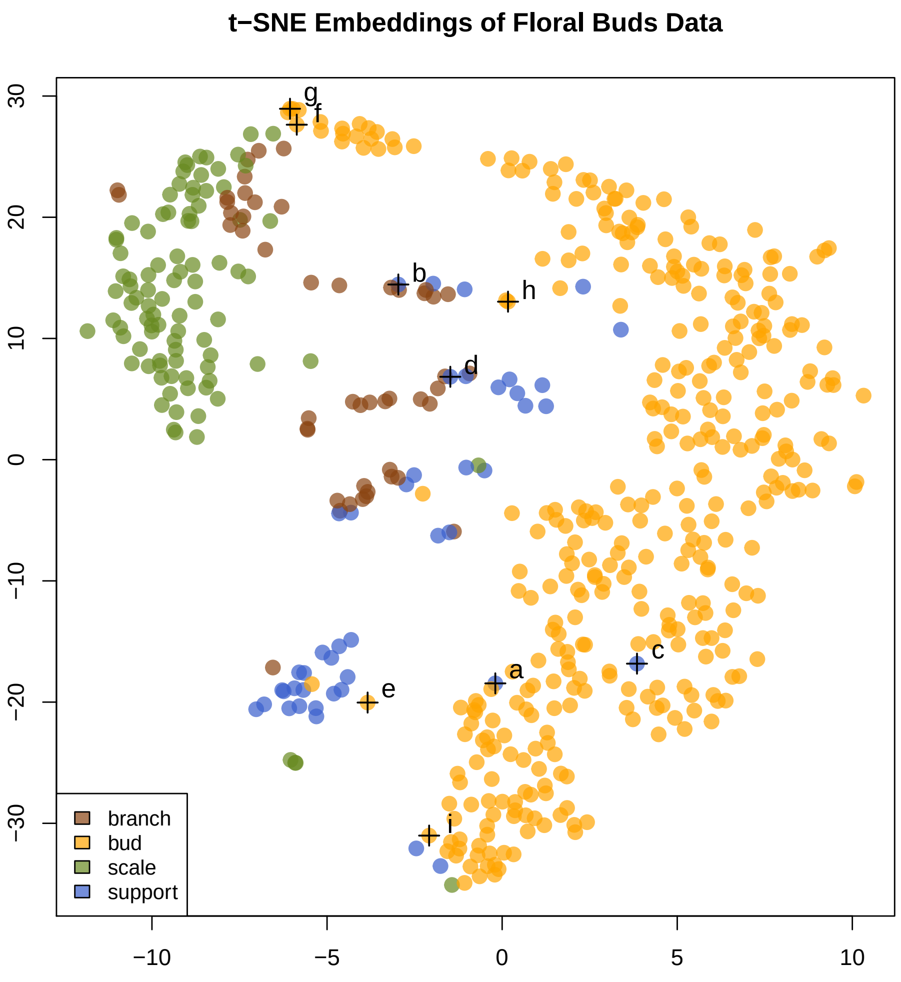

Figure 2.3: t-SNE embeddings of the Floral Buds data, colored by class. Marked points are referenced in Figure 2.4 and 2.5.

2.2.1 Localized Farness

The definition of localized farness draws from an idea presented in Napierala and Stefanowski (2015), whose taxonomy of data difficulty is described in Section 1.4. For imbalanced classification problems, the authors propose a kernel-based local neighborhood approach to determine where a minority class observation lies in the data difficulty taxonomy. The authors first measure an object's level of data difficulty by computing the probability that an observation lies in the minority class. The probability is measured using an epanechnikov kernel weighting function centered at the object, which assigns weights to neighbors based on their distance from the object. The weighted proportion of neighbors belonging to the minority class becomes an observation's probability of belonging to its class. The category of difficulty is defined according to cutoff points of this probability; safe observations have high values, while outliers have very low values. We adapt the authors' measure of data difficulty to become a measure of *localized farness*, where an object's farness from its class is measured within its local neighborhood.

Definition

Revisiting the settings where we defined farness, we have objects i, where $i = 1 \dots n$, and classes g, where $g = 1 \dots G$; object i has ground-truth label g_i. For each i, we compute the distance d_{ij} from i to another object j, for all n objects in the dataset, obtaining a set of distances:

$$d_i = (d_{i1}, d_{i2}, \dots, d_{in}) \tag{2.5}$$

where $d_{ii} = 0$. The distance function can be any function that is appropriate to the data. For example, the Euclidean distance is a popular choice for data that consists of numerical coordinates. The Jaccard distance is useful for binary data.

We weight each of these distances with a kernel weighting function, w, with width ϵ_i that is specific to i:

$$w(d_{ij}, \epsilon_i) = \frac{1}{\epsilon_i} K\left(\frac{d_{ij}}{\epsilon_i}\right), \tag{2.6}$$

where K is the ephanechnikov kernel:

$$K(z) = \frac{3}{4}(1 - z^2)\mathbb{1}|z| \leq 1 \tag{2.7}$$

The kernel weighting function assigns weights to other objects based on their distance to i; larger weights are assigned to closer objects. We call the area around i within ϵ_i distance as a neighborhood, and objects located within this neighborhood are neighbors of i. If the distance of an object j to i is larger than the width of the kernel, ϵ_i, the weighting function assigns d_{ij} value 0.

The kernel-weighted sum of distances of neighbors belonging to class g_i is $p(g_i|i)$ - the probability of i belonging to g_i:

$$p(g_i|i) = \sum_{j=1}^{n} (w(d_{ij}, \epsilon_i)\mathbb{1}(g_j = g_i) \tag{2.8}$$

$p(g_i|i)$ takes values in $[0, 1]$. If i has no neighbors belonging to its group, g_i, within ϵ_i distance, then $p(g_i|i) = 0$. If i is located in a region where all other observations within

ϵ_i distance belong to g_i, then $p(g_i|i) = 1$. The probability that i belongs to groups other than g_i can be computed with the same definition, i.e. $p(g|i), \quad g = 1 \dots G$. Because of the kernel weighting function, $\sum_{g=1}^{G}(p(g|i)) = 1$. Thus, Localized farness (LF) of an object i is the probability that i does not belong to its true class, or

$$\text{LF}(i) = 1 - p(g_i|i) \tag{2.9}$$

Thus, an object lying in a neighborhood comprised entirely of its own class has LF$(i) = 0$, and an object lying in a neighborhood with points of only other classes has LF$(i) = 1$. If i lies in a mixed neighborhood, it will have a value of LF in range $(0, 1)$.

Determining ϵ_i

Napierala and Stefanowski (2015) define a fixed width, ϵ, for the epanechnikov kernel weighting function used to measure difficulty of each object i. In practice, this does not work well for datasets with unequal neighborhood density. In some of the real-world datasets used by the authors, the data difficulty of some observations cannot be evaluated, as they have no neighbors within an ϵ radius.

We use a different approach: for an integer $k \geq 1$, we define ϵ_i for each i to be the distance to the k^{th}-nearest neighbor. This way, the epanechnikov kernel always uses k neareast neighbors to measure localized farness of each object i To estimate ϵ_i, we perform an Approximate Nearest Neighbor (ANN) search, which finds the k nearest neighbors for each object i. This approach not only makes the definition of localized farness usable for all datasets, but also has computational benefits, which are explained in Section 2.3.2.

2.2.2 Example

We now return to the Floral buds classification example. With the same Quadratic Discriminant Analysis model we used in Section 2.1, we produce Figure 2.4, which displays the class map (left) and localized class map (right) for class "supports." Both visualizations use the same measure of PAC. The difference between the plots is most evident in the *x*-axis: the class map shows farness, and the localized class map shows localized farness, computed with $k = 10$. Several points of interest have been labeled `a` - `d` and marked with a "+". Their corresponding locations on the *t-SNE* map are marked by the same letters and symbol in Figure 2.3.

Both maps show a roughly similar picture: that the majority of supports are correctly classified, and that most points lie close to their class. Given the many brown-colored points, the maps indicate that branches and supports are not well-separated classes.

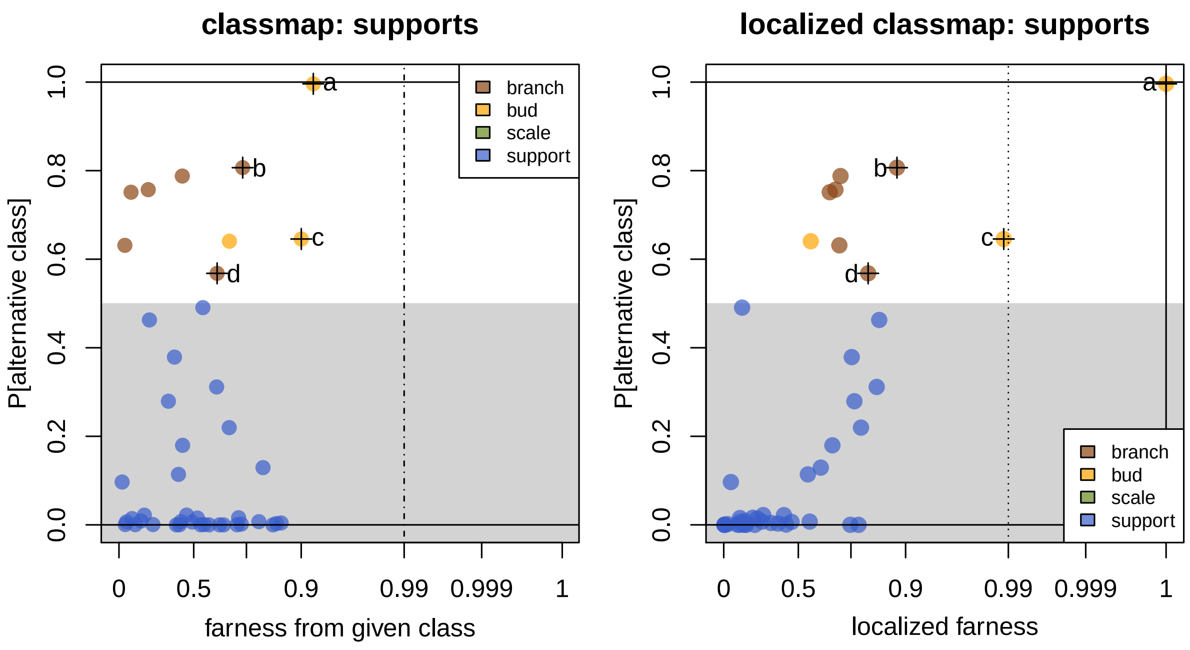

Figure 2.4: Class map (left) and localized class map (right) of class ‘supports’.

On average, misclassified points have a higher value of localized farness than that of farness. The class map gives the indication that the classifier misclassifies points that are close to their class. The localized class map, on the other hand, shows that the classifier only performs poorly on points that are locally far from their class, in other words, that live in neighborhoods with observations from different classes. We can also point out differences in selected points. Points `a` and `c` have the highest values of farness and of localized farness; this is not surprising, as Figure 2.3 show these points to be located in a neighborhood comprised entirely of buds. Points `b` and `d`, classified as branches, have comparable farness and localized farness. In the localized class map, point `b` is considered to be considerably farther from its class than point `d`, which corresponds to the picture shown in Figure 2.3: point `d` is somewhere between a small cluster of supports and branches, while point `b` has more branch neighbors than supports.

What would a localized class map look like for a more homogeneous class? We can look at Figure 2.5, which depicts the same maps as Figure 2, but for class “bud“. Looking at the t-SNE representation in Figure 2.3, most buds exist in the large, crescent-shaped area of the plot. Thus, it is not surprising to see that the QDA classifier assigns the majority of points a PAC of 0.

The class map gives a clear view of the the data from the eyes of the classifier: excluding the outliers, PAC is quite correlated with farness. Point `e` is marked as an outlier in the class map, or a point that is far from all classes. This is not the case in

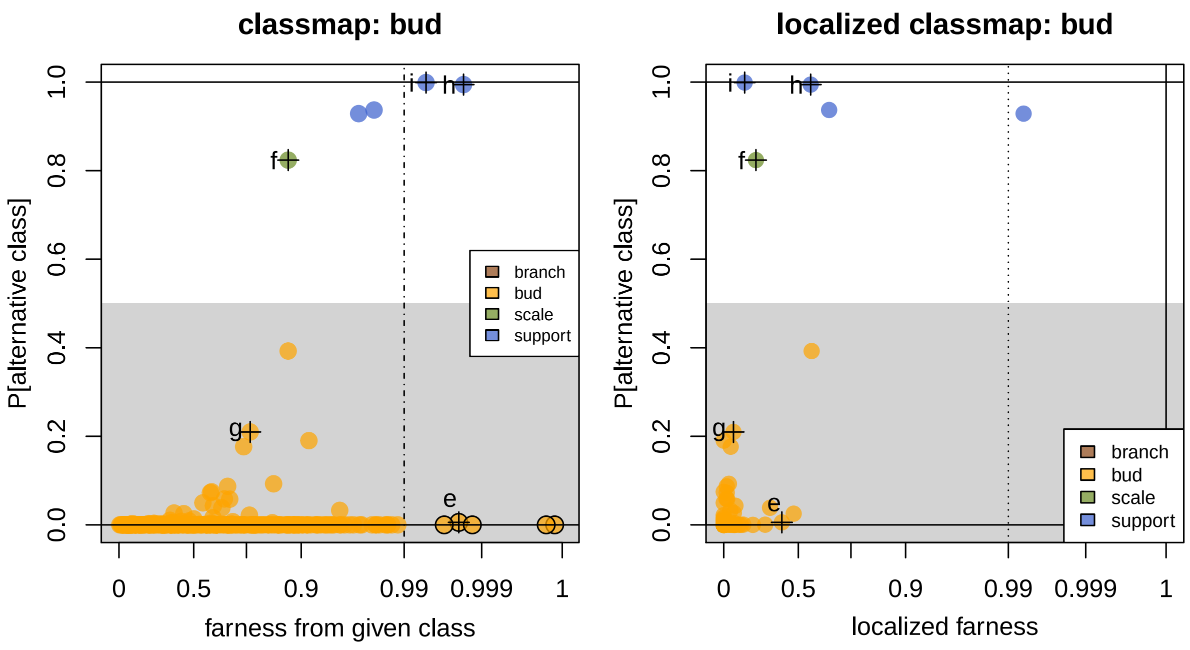

Figure 2.5: Class map (left) and localized class map (right) of class 'bud'.

the localized class map, which assigns it medium value of localized farness. Looking at Figure 2.3, both depictions have some truth. Point `e` is in a space that is far from all classes, in the sense that it is close to a cluster that contains observations from all classes, mostly supports. Points `h` and `i` (point `f`) have relatively low localized farness, despite being misclassified as a support (scale). This is because they live in mixed neighborhoods, where some of their nearest neighbors belong to their own class. The localized class map indicates, but does not make extremely evident, that these points exist on boundaries between classes.

The main difference between the two plots is that none of the correctly-classified points has a localized farness larger than 0.5, while many correctly-classified points have large farness. This highlights the difference between the two measures. From the perspective of the QDA classifier, many buds are far from their class center, but are still easy to identify if they do not exist at boundaries with other classes. Localized farness sees buds as a group where most observations live in all-bud neighborhoods, making this class relatively easy to classify.

2.2.3 Remarks

The goal of the localized class map is to provide post-hoc interpretations for a classifier. That is, to give a more detailed picture of where and why a classifier makes, or does not make, mistakes. In Section 1.1 we distinguish two sources of bias in classification: model-driven bias, which pertains to bias resulting from the choice of model or parameter, and data-driven bias, which is bias that the classifier inherits from being trained with poorly-chosen training data. The first quantity of the localized class map, PAC, measures model-driven bias. A well-chosen model should only assign a high PAC to observations that are very atypical of their class. That is, PAC should have a positive correlation with localized farness. The second quantity of the localized class map, localized farness, measures data-driven bias. An "easy" classification problem is one where classes are well-defined, and observations form homogeneous classes. If the distribution of localized farness of members of a class contains many values that are close to 1, this indicates potential issues with the data.

To get an idea of the bias in a trained classifier, we need to look at localized class maps for all classes present in the data. Figure 2.6 shows localized class maps of the trained QDA classifier on all observations in the Floral buds data.

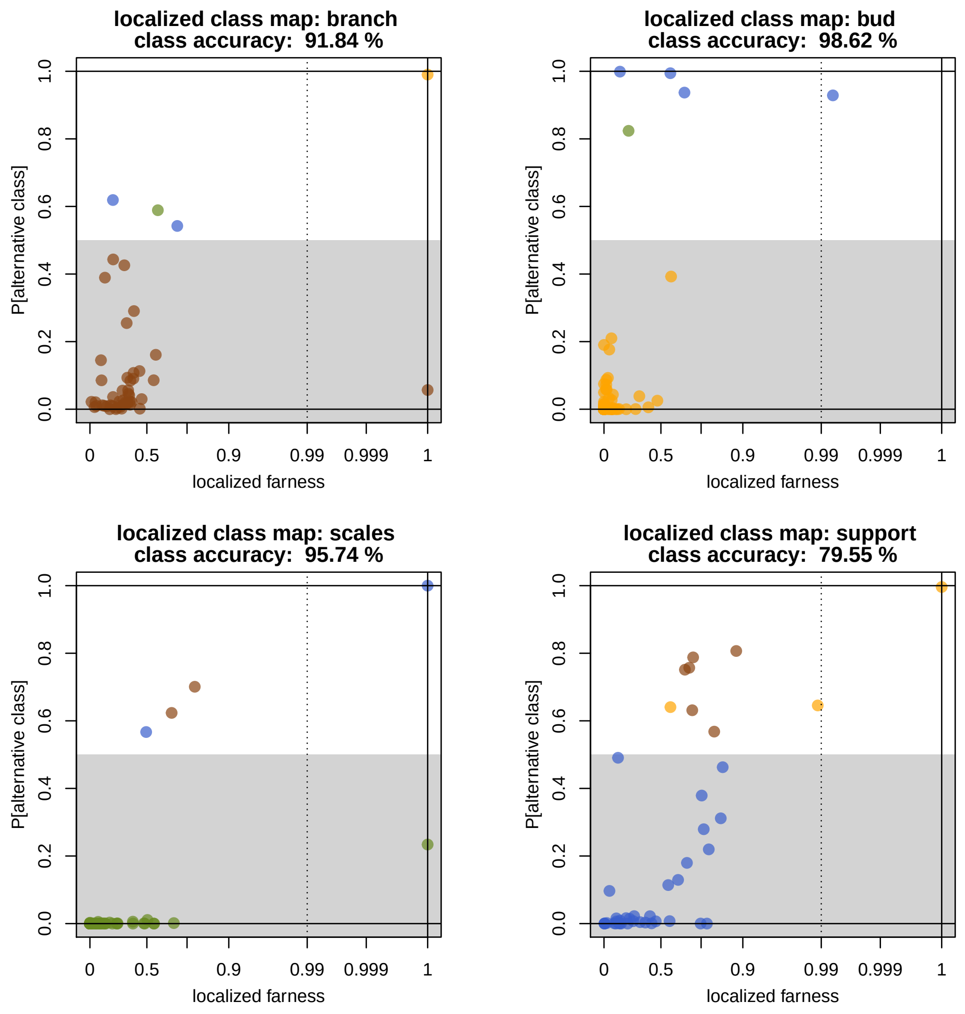

Figure 2.6: Localized class maps for all classes of the Floral buds data.

First, we notice that the classifier has an overall high accuracy rate, and PAC has a rough positive correlation with localized farness. In other words, the classifier behaves as expected: it performs poorly only on observations that are difficult to classify. Looking at the PAC values in each class map, we notice that the classifier does make some errors on points that are located close to their class. The model has some model-driven bias; it sees the data in a way that is different from how it actually is. Quadratic Discriminant Analysis assumes that observations from each class are drawn from a Normal distribution. Model-driven bias comes from the discrepancy between the model's assumptions of the data and the actual distribution of the data. Here, we do not observe too much of it. Data-driven bias would come from a scenario where a large proportion of observations have very high localized farness, indicating a potential mismatch between the ground-truth labels and the observations. Looking at localized farness, we observe that each class has only a few points with a localized farness close to 1. It is unlikely that the data has added much bias to the classifier. When looking at per-class accuracy, we see that the QDA classifier performs comparatively worse on supports. The localized classmap gives an explanation as to why: this class has relatively more difficult observations.

2.2.4 Choice of *k* in localized farness

The distribution of localized farness depends on the choice of k, which is left to the user. The choice of k mirrors the choice of perplexity in t-SNE. Like the class map, t-SNE expresses the distances between points as probabilities (in this case, conditional probabilities). The definition of this conditional probability involves centering a Gaussian distribution over each observation, i. The variance of the Gaussian, σ_i is left as a parameter for the user to select. Like ϵ_i in localized farness, there is no single value of σ_i that is optimal for all i. Rather than choosing a value for σ_i, the user chooses a value of *perplexity*, a smooth measure of the effective number of neighbors, and t-SNE performs a binary search for the set of σ_i that yields the desired perplexity. In the case of localized farness, the user chooses k.

To choose a value of perplexity, van der Maaten and Hinton (2008) suggest that users try several different values in a specified range. In the case of localized farness, we suggest that k is selected in consideration of the size of the data. A rule-of-thumb would be that k should not be too small, in order to get information from enough neighboring points, but not too large, in order to keep the neighborhood local. After using the class map on many real-world datasets, we found that a k in $[5, 10]$ produces a good map for most datasets. Generally, smaller values of k will lead to more extreme values of localized farness, while larger values of k will lead to fewer extreme values.

We illustrate this in Figure 2.7, which displays localized class maps of the Floral buds data, for values of k in $[3, 10, 40]$ and for classes "support", which is heterogeneous, and "bud", which is comparatively homogeneous.

For both classes, when $k = 3$, localized farness takes values at either extreme. Supports have localized farness that is quite spread out, while the majority of buds have LF = 0. At $k = 10$, the class map of supports has fewer values with extreme localized farness. It appears that supports, buds and branches overlap in a region, with one support residing in a bud-filled neighborhood. Looking at buds, it also appears that some buds exist at the decision boundary between supports and branches, with one outlier. However, when $k = 40$, the distribution of localized farness becomes narrower for both supports and buds. No obseravations are flagged as having high localized farness.

It is important to note that the overall trend in the class maps does not change too much with k. The model-driven bias is still apparent, but the measure of the data-driven bias changes. Therefore, we conclude that while the choice of k matters, there exist several values of k that would yield an informative localized class map.

2.3 Implementation

The class map is implemented in `R` and is available on CRAN as package `classmap`. The localized class map is also implemented in `R` and is available on Github [1]. Below, we explain the impact of specific implementation choices.

2.3.1 Compatibility with `caret` package

The localized class map is implemented to be compatible with `caret`, an R environment that streamlines the creation of predictive models. Within the `caret` framework, a user can perform data splitting, data pre-processing, model fitting, parameter tuning, and model evaluation. `caret` currently supports over 238 models[2], including many popular classification algorithms.

A user can train a model in the `caret` framework and immediately input this model, along with the data and labels the user wishes to visualize, into the function `plotClassMap()`. The function computes PAC from the model and data by calling the function `compPAC()`. It then computes localized farness directly from the data by calling `compLocalFarness()`, where the users specifies `k`. The default is set to `k = 10`. The benefit of this implementation is that a user can visualize the results of any classification model supported by `caret`, maintaining a simple, standardized framework. However, the implementation of `compLocalFarness()` is currently limited to numeric data.

[1] https://github.com/nurakawa/localized-classmap

[2] A full list of supported models can be found here https://topepo.github.io/caret/available-models.html

2.3.2 Nearest Neighbor Search with ANN Algorithm

The definition of localized farness uses, for each observation i, a set of distances to other observations, see Equation 2.5. Since the weighting function only assigns nonzero value to distances smaller than ϵ_i, the distance from object i to its k^{th} nearest neighbor, we only need to compute the distances of the k nearest neighbor for each point.

To do this, we use the ANN algorithm implemented in `R` package `FNN`, opting for the use of a *kd*-tree data structure. Given a dataset, the ANN algorithm performs an approximate nearest-neighbor search for each observation i, for which it returning a list of distances to and indices of the k nearest neighbors. We set ϵ_i to be the distance to the k^{th} nearest neighbor, and use the remaining distances to compute localized farness.

While the ANN algorithm works for any distance, the implementation supported in `FNN` is currently limited to Minkowski distance metrics. Thus, `compLocalFarness()` cannot compute localized farness for categorical data. Categorical predictors can be transformed into sets of binary predictors, but distances specific to binary data are not supported.

The advantage of using ANN is in computational efficiency. Finding exact nearest neighbors and distances for large datasets is computationally expensive. Computing nearest neighbors approximately improves the run time significantly and often yields very small errors (Arya et al., 1998). The ANN library used by the `FNN` implementation, written in `C++` has been found to be efficient for point sets as large as hundreds of thousands of points (Arya and Mount, 1998). Therefore, with this choice in implementation, we sacrifice some accuracy and a flexibility of data type, but gain advantage in efficiency.

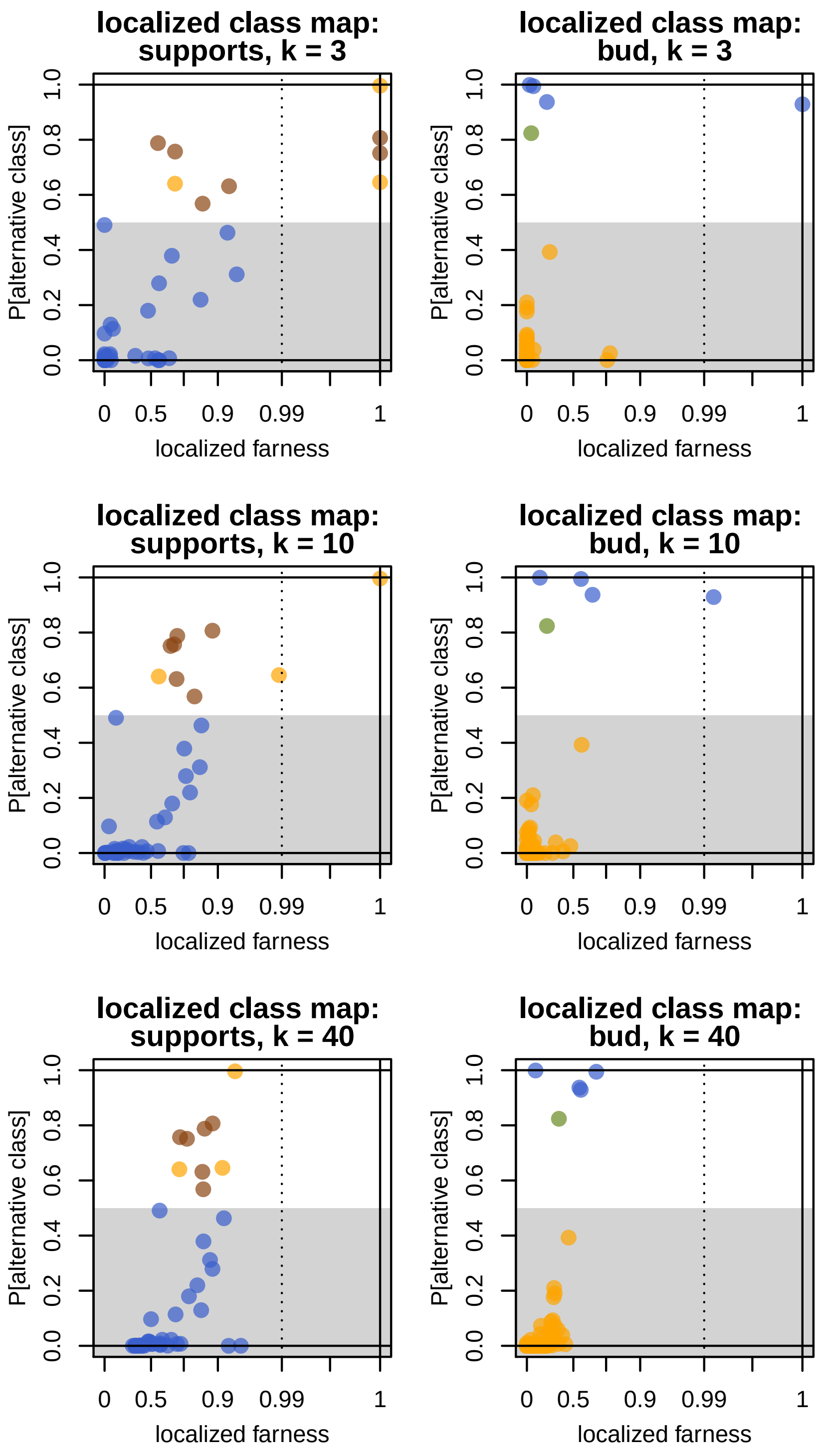

Figure 2.7: Class maps of Floral buds data with different values of k.

Chapter 3

Results

3.1 Comparing Different Classifiers on the Same Data

The class map is a useful tool to compare several classifiers on the same data; Section 8 of Raymaekers et al. (2021) provides an example. By plotting class maps of different classifiers on the same data, a machine learning practitioner can get different views of the data, each from the perspective of the classifier. The localized class map can also be used to compare different classifiers, in a different manner. Because localized farness is classifier-agnostic, we can directly compare the PAC values output from different classifiers. This would give us a clearer sense of model-driven bias. It also provides a visual aid in selecting the best classifier for the data, as well as considering the potential impact of data pre-processing to improve a classifier's accuracy.

3.1.1 The Yeast Data

The yeast data is a benchmark dataset available on the UCI Machine Learning Repository (Dua and Graff, 2017). It contains 1484 observations of yeast specimen described by 8 attributes. The goal is to predict the localization site of the yeast protein. In the original data, there are 10 different sites of localization, making it a mutliclass problem. We convert it into a binary problem, where the goal is to identify the specimen that are localized in site "ME2", labeling the remaining specimen as "OTHER".

After relabeling, this is a dataset that has very high class imbalance; only 3.4% of the data belong to the minority class "ME2". Figure 3.1, a t-SNE plot of the data, indicates potential data difficulty factors in minority observations. Some observations exist in small clusters. Others exist in two- or three-member sub-concepts, which Napierala and Stefanowski (2015) calls *rare* examples, and still others exist as singleton observations embedded in neighborhoods of the opposite class.

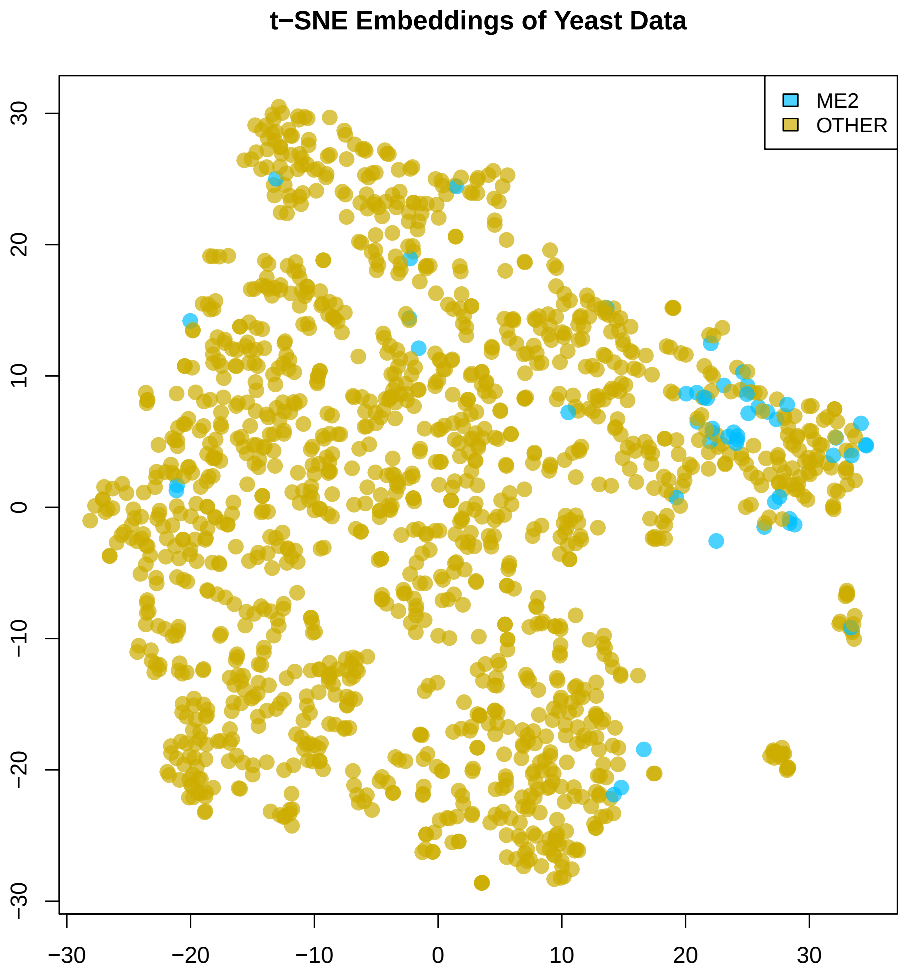

Figure 3.1: t-SNE plot of yeast dataset.

3.1.2 Classification Algorithms

After visualizing the data and identifying data difficulty factors, we opt for classification algorithms that make no assumption of the distribution of the data. We randomly split the data into 70% train, 30% test. The imbalance ratio is the same in both sets. We use the training set to train the following classification models:

- **C4.5** (Quinlan, 1993), a classification algorithm that builds a decision tree. It does not use a distance, but rather a measure of information gain. In its most simple form, the algorithm recursively searches for the feature that, when used to split the data into subsets, yields the largest information gain. It avoids overfitting by "pruning" the trees it builds. We use the open-source Java implementation of this algorithm, called `J48`[1], which is supported in `caret`.

[1]The J48 algorithm information can be found here
http://santini.se/teaching/ml/2016/Lect_03/Lab02_DecisionTrees.pdf

- **Multi-layer Perceptron** (MLP), a feed-forward Artificial Neural Network. This classifier approximates a nonlinear function of the data by expressing it as a weighted sum of inputs, then applying an activation to yield a decision. MLPs are prone to overfitting if not regularized appropriately. We use the method denoted `"mlp"` in `caret`, which is implemented in `R` package `RSNNS`.
- **k-nearest neighbor classification** (KNN), a method that finds the k-nearest neighbors of an observation and estimates for it a class probability that is the proportion of the class membership of its group. The measure of localized farness is related to classification by k-nearest neighbors.

The performance of each classifier on the train and test sets is displayed in the tables below. In this example, the focus is on building a classifier with high Sensitivity, or the True Positive Rate, which measures how well the classifier identifies members of the minority class. Specificity, or the True Negative Rate, measures how well the classifier detects the majority class. Balanced Accuracy is the average of Sensitivity and Specificity. Accuracy, which measures how well the classifier performs on the entire dataset, is a misleading metric in imbalanced classification. In this case, a classifier that predicts all observations to belong to class "OTHER" would have a high overall accuracy but no ability to detect the minority class.

	C4.5	MLP	KNN
Accuracy	0.978	0.9779	0.971
Balanced Accuracy	0.734	**0.801**	0.610
Sensitivity	0.472	**0.611**	0.222
Specificity	0.996	0.991	0.998

Table 3.1: Train set performance metrics of C4.5, MLP and KNN classifiers.

The tables give a good understanding of performance of each algorithm. Looking at the train set results in Table 3.1.2 (above), the classifiers all have very high Specificity, but relatively low sensitivity. The MLP performs best on the train set. However, it seems to overfit, as the test set performance is quite poor. Table 3.1.2 (below) reveals that all the base classifiers have low Sensitivity on the test set, with the best performance from C4.5.

	C4.5	MLP	KNN
Accuracy	0.966	0.955	0.966
Balanced Accuracy	**0.629**	0.591	0.532
Sensitivity	**0.267**	0.200	0.067
Specificity	0.991	0.981	0.998

Table 3.2: Test set performance metrics of C4.5, MLP and KNN classifiers.

What the tables do not reveal is an explanation behind the performance. For the C4.5 model, we can get an explanation by plotting the nodes of the final decision tree it creates, to see where the algorithm determines splits. However, this is not always a feasible option when classifying large datasets, where the model of choice is often ensembles of trees instead of a single tree. The MLP provides no explanation for its decisions - it is seen as a “black-box” model. The decisions of the KNN classifier are best explained by looking at the data, as it makes decisions based on local neighborhoods.

To get an explanation behind the performance of each model, we can plot localized class maps on the training data (Figure 3.2) and test set (in the Appendix as Figure A.1). The top row of Figure 3.2 is the majority class, “OTHER”, while the bottom row is the minority class, “ME2”. Each row has the same localized farness, computed with $k = 10$. Therefore, we can directly compare the PAC values from different models.

3.1.3 Results

Figure 3.2 depicts the localized class maps of each classifier.

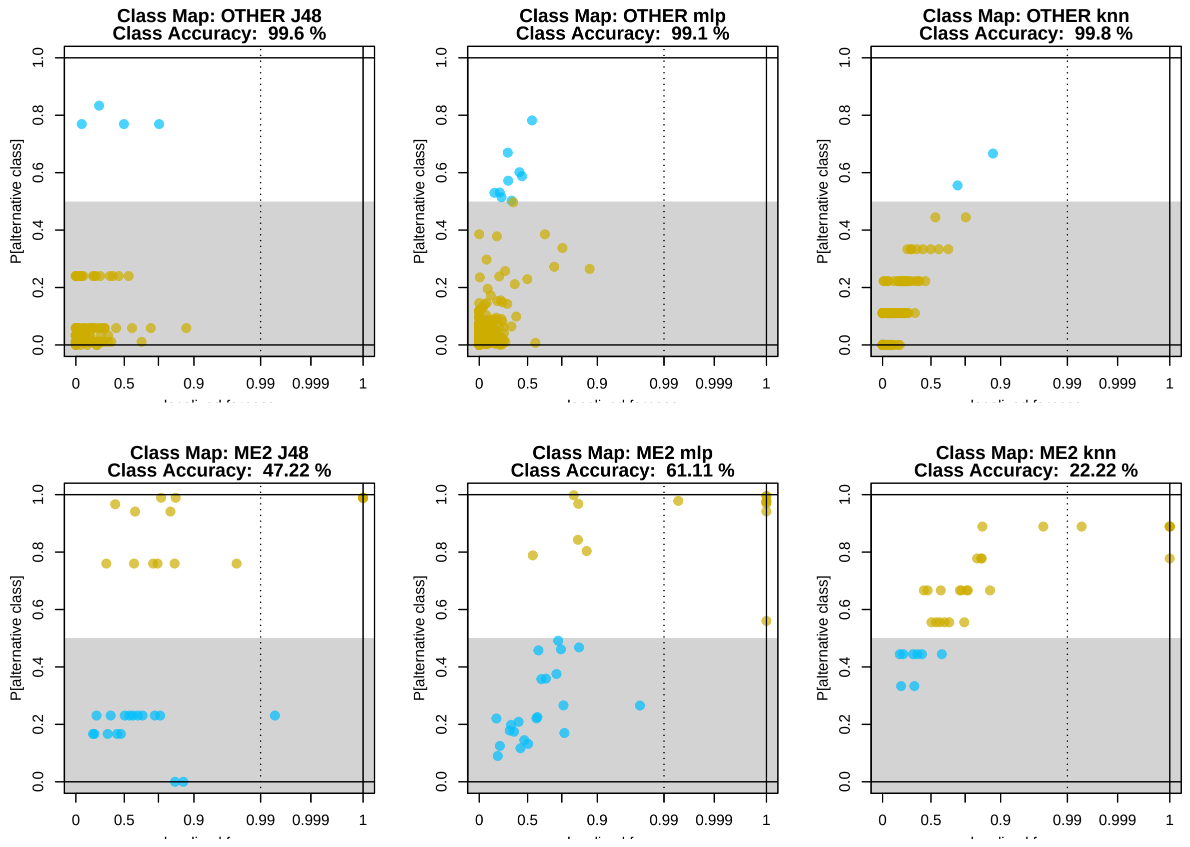

Figure 3.2: Localized class maps of train observations of the Yeast dataset for classifiers C4.5 (left), MLP (center), KNN (right).

First, we can ask the question *Can the classifiers' errors be attributed to data-driven bias?* In the majority class, "OTHER", we notice that the data are relatively homogeneous - no points have localized farness above 0.9. In class "ME2", all points have localized farness larger than 0, indicating data difficulty factors. However, there are only a few points with maximal LF. It does not seem that many points contain erroneous values or incorrect labels. Rather, the only possible source of data-driven bias is the class imbalance. For the purpose of this example, we chose standard, out-of-the-box classifiers and did use a fine grid of values to tune parameters. If we had taken into account knowledge of the source of the training data, i.e., that there is extreme class imbalance, we could have chosen classifiers that are specifically developed for imbalanced classification, such as classifiers that support class weighting.

Now, we compare the PAC, or measure of model-driven bias, across classifiers. C4.5 seems to have a lot of inductive bias, as it does not made decisions in line with localized farness. The form of the resulting f^* might be too simple. The exact opposite phenomenon is observed by the KNN classifier (using cross-validation, the k was selected to be 9). Its decisions are strongly correlated with localized farness. It only misclassifies the points that are the farthest away from their class. This classifier has almost no inductive bias; it overfits to the data. The MLP lies between C4.5 and KNN - it has a weak correlation with localized farness. It is not surprising that it has the best performance on the minority class in the training data.

After looking at localized class maps of the training set performance, we get insight into how the classifier performs on the data. This explains much of the test set results. Since the MLP classifies in line with localized farness, it performs poorly on the minority class of the test set. The C4.5 classifier, on the other hand, has rules that are not in line with localized farness, which yields poor performance on the training set, but prevents it from overfitting to unseen data. It is important to note that in this example, the MLP correctly classifies exactly one more observation than does the C4.5 classifier; thus, their test-set performances are quite close. The KNN classifier performs abysmally on the minority class in the test set, because the minority class have few neighbors of the same class.

After comparing these classifiers using localized class maps, machine learning practitioners can make informed decisions on what to do next. They can begin with a KNN classifier and perform sub-sampling to improve the results. Or, they can add regularization to the MLP to reduce its overfitting, Or, they can opt for decision trees but instead use ensemble techniques to improve their performance. The localized class map sheds light on the decisions that algorithms make with respect to the data, and are useful comparison tools.

3.2 Binary Classification with Ambiguous Data

Most classification algorithms work well when the training and testing data consist of observations in well-separated classes. However, many classification problems are those where classes are not well-separated. If we were to visualize the feature space of such data, we would see regions with more than one class (Trappenberg and Back, 2000). For example, the task of classifying images of handwritten digits has well-defined, well-separated classes. The task of classifying observations into sentiment categories, however, has overlapping classes. Sentiment exists on a spectrum and has some subjectivity involved in its identification. Observations could convey different levels of sentiment at once - for example, text data could contain both positive and negative words. This would lead to ambiguous data, where many observations are only loosely matched to their label.

When data is homogeneous, the class map is useful for identifying mislabeled points. When classes are well-defined, mislabeled points would have a high farness in the eyes of the classifier. However, in the presence of ambiguous data, it could be difficult to get a sense of how well an object really lies inside its class. In this scenario, it might be beneficial to instead use a localized class map. Instead of comparing an object to all of its class members, many of which might be poorly labeled, we only look at an object's local neighborhood. This way, the measure of how closely an object matches its label is less impacted by heterogeneity in the class. If an object has high localized farness, this means that it has a different label from nearly all k neighboring objects that are similar to it in structure. This measure has a higher likelihood of revealing poorly-labeled points. Therefore, we get a more clear measure of data-driven bias. We illustrate this by reproducing an example from Raymaekers et al. (2021), where the authors classify book reviews as “positive” or “negative”.

3.2.1 The Amazon Book Reviews Data

The Amazon book reviews data is a benchmark dataset used in Prettenhofer and Stein (2010) for sentiment classification. It contains 4,000 English-language book reviews from Amazon's website. A user who reviews a book also ranks it from 1 to 5 stars. Reviews with a high ranking (4 or 5 stars) are labeled as “positive”, and the reviews with low ranking (1 or 2 stars) are labeled as “negative”, and the 3-star reviews are discarded. Intuitively, ranking and review sentiment should have a positive correlation. However, there is plenty of room for ambiguity in this labeling scheme. A high-ranking review could still be negative, for example if the reviewer enjoyed the book but has complaints about Amazon's delivery service. There are also neutral reviews, which could belong to both categories.

3.2.2 Results

The data is split into training (2000 points) and testing (2000 points) sets; both sets consist of 50% positive reviews, making them perfectly balanced. We use the same classification model and parametrization as Raymaekers et al. (2021). We construct a kernel matrix of the book reviews using the spectrum string kernel, which matches strings of length p. Using cross validation, p was selected to be 7. We then used it to train a linear Support Vector Machine classifier, where parameters were chosen to match the original model; we achieved 82% classification accuracy.

Figure 3.3 shows the resulting localized class map of the test set of positive book reviews, where the k in localized farness is set to 7, although values of k in range $[5, 10]$ showed good results.

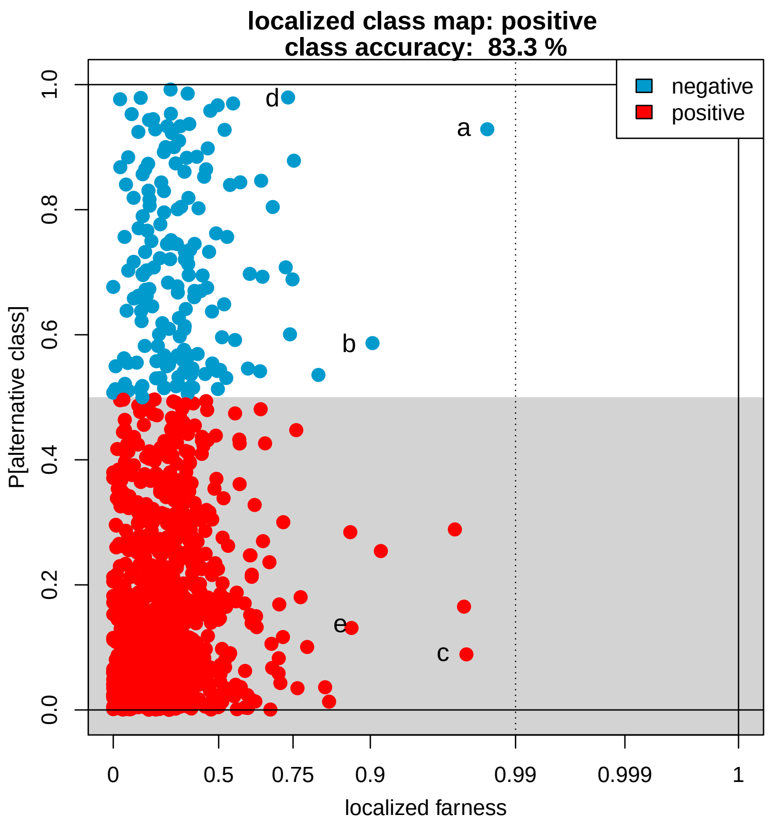

Figure 3.3: Class map of positive book reviews in the test set

Excerpts from the text of marked labels are presented in Table 3.3. The map shows that the majority of positive book reviews have a localized farness in range $[0, 0.5]$, with few examples having higher value. The PAC is not correlated with localized farness; the classifier has confusion even with points that live in homogeneous neighborhoods.

By looking at the model's performance on the training set, shown in the Appendix as Figure B.1 and Figure B.2, we identified the issue of overfitting, as the model has 100% training set accuracy; this indicates model-driven bias. Regardless, given that several observations have high local farness, we can look into the possibility of data-driven bias.

marked	excerpts from the text
a	"if you are the kind of person who want to quickly assimilate and regurgitate the matter for cissp, then dont even bother." "and you'd expect this kind of book to live on your shelf for a long long time...but the quality of paper will make that unlikely." "hence i am giving 4 stars to a book which otherwise would deserve 6 star"
b	"the book's main drawback comes in its unappealing characters. most readers like either sympathetic characters or disgusting villains. this book lacked both." "but any long-time fan will enjoy the book"
c	"I like the book very muc"
d	"the ending just wasn't believable." "wait for it in the bargain bin"
e	"this book lacked substance and depth" "so many missing pieces that could have explained more clearly what led up to the revolution"

Table 3.3: Excerpts from book reviews with a positive label, identified in Figure 3.3.

Point a has relatively high PAC and high localized farness. It is labeled as positive, but the review has some complaints about the quality of the book's paper. The review also contains some negative sentiment, but not directly about the book. The ambiguity present in the text is likely what makes it difficult to classify. Point b is similar to a; in fact, this review seems quite neutral overall. Example c is very positive, but has a high local farness; it happens to be the shortest review in the test set. It was also flagged by the class map in Raymaekers et al. (2021). Points d and e are interesting; it seems that they could have been mislabeled. While they contain some neutral information, the excerpts included in Table 3.3 make the reviews lean negative. Indeed, they have a relatively high localized farness. However, while point d is seen as highly likely to be negative by the classifier, point e is seen by the classifier as positive.

Figure 3.4 displays the class map of the negative reviews in the test set, also with $k = 7$. In contrast to the postive reviews, in which the majority of observations form a vertical band, the negative reviews are more spread out across the localized class map. This class also contains many points with high local farness. We have selected a few points of interest and included excerpts from their corresponding text in Table 3.4.

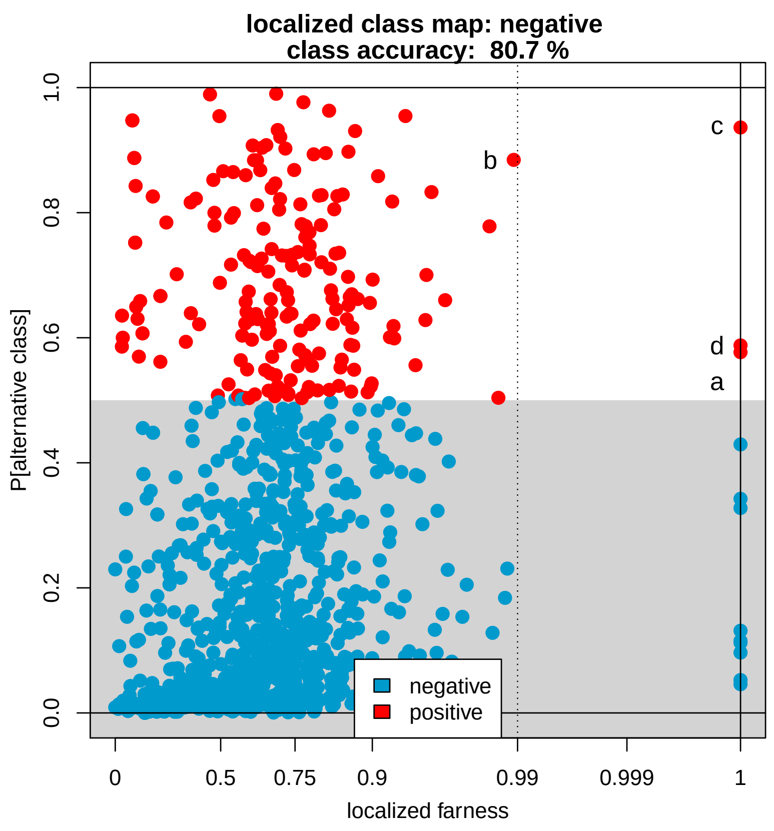

Figure 3.4: Class map of negative book reviews in the test set

marked	excerpts from the text
a	“rather than wooldridge’s fear and resentment-mongering, i’d recommend the upcoming boook by free speech radio news host...”
b	“this book is the best i ever read.it tells about all the animals that live there.a short man wacks tree,all the animals are sad that their family will live without tree’s”
c	“this book is an awesome reference tool. it arrived in perfect condition !”
d	“however, the whole matter is subject to the particular perspective preference of the readers personal comprehension.”

Table 3.4: Excerpts from book reviews with a negative label, identified in Figure 3.4.

Here, points `a` and `d` have nearly the same PAC - slightly above 0.5 - and a localized farness of 1. Both reviews are more nuanced than negative. They also are both very long - well above the third quartile of review length. The extreme length, as well as their nuanced text, might be what places them in neighborhoods that contain similar objects but with positive labels. Points `b` and `c` appear to have been mislabeled. The excerpts in Table 3.4 are the reviews in their entirety. Point `b` does contain the negative word "sad", but, unless the author of the review was writing in a sarcastic tone, the phrase "this book is the best I have ever read" indicates positive sentiment. Point c seems unequivocally positive, and certainly seems to have been mislabeled.

In this example, localized class maps yielded similar results to the class maps in Raymaekers et al. (2021). However, there are some differences that make localized class maps especially useful in the case that the label is a category that leaves room for ambiguity. The distribution of localized farness in the positive and negative classes reflects each class's level heterogeneity. The localized maps also identified some mislabeled test points in each class. It is likely that these mislabeled points do not seem so far from their class when comparing it to all members, but they stand out in their local neighborhoods. After using localized farness to measure data-driven bias, it is evident that a direction of model improvement would be to check the quality of the data.

3.3 Assessing Fairness in Classification

Friedman and Nissenbaum (1996) define the term *algorithmic bias* as pertaining to systems that systematically discriminate against certain groups in favor of others. For example, classification systems can consistently have worse accuracy for observations with the attribute `gender == female` than `gender == male`. *Fairness* of algorithms is modifications towards correcting for algorithmic bias. While there are many definitions of fairness across disciplines, this thesis defines fairness as parity of algorithmic performance across levels of sensitive groups. Sensitive groups are subsets of the data split by level of an attribute with sensitive information, such as gender or race.

How can we assess if a classifier is fair? While there exist several different definitions and methods, we focus on measuring a classifier's adherence to the fairness criteria described in Barocas et al. (2019). For simplicity, we consider the following scenario: A, Y and R are random variables. A is a sensitive attribute, such as gender or race, that takes values a and b. Y is the ground-truth labels of the data, which can take values 0 or 1. R is the result of the classifier, which also takes values 0 or 1. A classifier is considered fair if it satisfies *Independence*, *Separation*, and *Sufficiency*.

Independence requires that the sensitive characteristic is independent from the result of the classifier:

$$P(R = 1|A = a) = P(R = 1|A = b) \tag{3.1}$$

Separation requires that both levels of the sensitive attribute experience the same true positive rate and the same false positive rate:

$$\begin{aligned} P(R = 1|Y = 1, A = a) = P(R = 1|Y = 1, A = b) \\ P(R = 1|Y = 0, A = a) = P(R = 1|Y = 0, A = b) \end{aligned} \tag{3.2}$$

Suffiency requires that the probability of having label $Y = 1$ is equal for any two observations with different levels of A that were predicted to belong to the same group. In binary classification, this means parity of positive/negative predictive values across all levels of the sensitive attribute.

$$\begin{aligned} P(Y = 1|R = 1, A = a) = P(Y = 1|R = 1, A = b) \\ P(Y = 1|R = 0, A = a) = P(Y = 1|R = 0, A = b) \end{aligned} \tag{3.3}$$

One application of the localized class map is in showing that a classifier satisfies

Independence and/or *Separation*. By plotting a localized class map for each level of a sensitive attribute, we can visually identify whether this attribute has a relationship with the classifier's result. We can also assess whether the classifier has equal true positive rate and false positive rate across levels of the sensitive attribute. More importantly, we can get an explanation as to why a classifier might or might not uphold this property.

3.3.1 The Adult Data

The Adult Dataset is a popular fairness benchmark dataset, available from the UCI ML Repository [2]. The data is extracted from the 1994 US Census Bureau Database. It describes several attributes of census participants, such as education level, occupation, country of origin, gender and race. The prediction goal of this dataset is to determine whether participants' yearly income exceeds $50,000. In this example, we consider gender to be a sensitive attribute.

The data is already split into train and test sets, which have roughly equal proportions of observations across sub-groups. We relabel the outcome, income bracket, to be "ge50k" or "lt50k" ($\geq \$50,000$ or $< \$50,000$). We also transform categorical features into sets of binary features.

3.3.2 Gender Bias in Income Bracket Classification

We train a penalized logistic regression classifier on training observations of the Adult dataset, using the `R` package `glmnet` implementation supported by `caret`. The model achieved an overall accuracy of 83.5%, with 92% specificity and 54% sensitivity. Low sensitivity is in part due to class imbalance: only 24% of training observations come from the high income bracket.

In this example, the focus is not on obtaining high sensitivity, but rather fairness of the classifier across gender groups. Figure 3.5 displays two localized class maps of test observations belonging to the high income class, divided by gender. From these localized class maps, we ask: *Is the classifier fair?*

First, we check for Independence. In this case, it means that the probability of being classified as `ge50k` is equal for males and females. Comparing the localized class maps, we notice that for female examples, especially those with with LF < 0.5, the proportion of misclassified points is higher than the proportion of correctly-classified points. For male examples, the proportion looks roughly equal. Visually, it looks like the results of the classifier and the category of gender are not independent.

Next, we consider Separation,which requires that the true positive rate and false

[2]https://archive.ics.uci.edu/ml/datasets/adult

positive rates are equal across gender groups. In Figure 3.5, the true positive rate is the ratio of purple-colored points to orange-colored points. Visually, we see a difference across genders. To assess the false positive rate, we can plot the localized classmaps for observations with ground truth label `lt50k` and perform the same comparison.

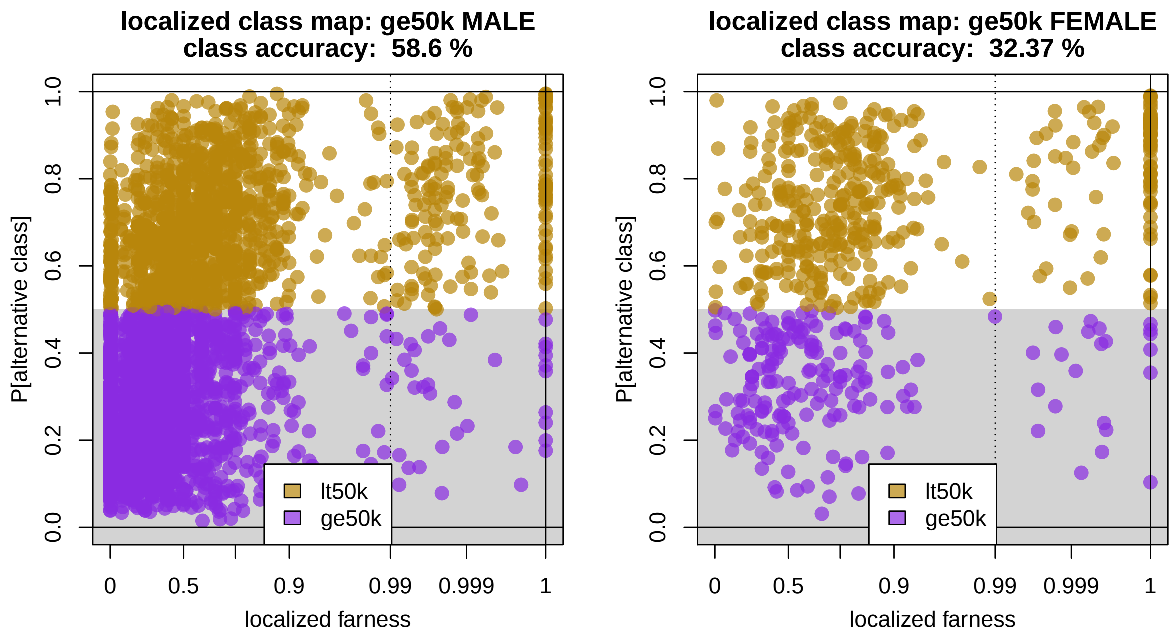

Figure 3.5: Localized class maps of test observations of the Adult dataset for class `ge50k`, for sub-groups Male (left) and Female (right).

Identifying whether a classifier meets the conditions of Independence and Separation can be done without visualization. What the localized class map can provide is an explanation as to why or why not a classifier meets these criteria. Looking at the number of examples in each gender group, we see that there seem to be more male examples than female examples. Observing the distribution of localized farness across gender groups, we notice comparatively fewer Female observations have LF < 0.5; such observations have a very high misclasification rate. Looking to the distribution of PAC across groups, although Females seem to have a higher PAC on average, the classifier has roughly the same trend: it performs poorly on observations with low LF and with high LF; PAC is not correlated with LF. Some the error can be attributed to model-driven bias, but the error that leads to unfairness seems to come from data-driven bias.

We confirm our observation by looking at the training data. Figure 3.6 is a stacked barchart showing gender and income bracket. The lefthand bar is that of male observations, and the righthand bar is that of female observations. Immediately, we wee that there are nearly twice as many male individuals than female individuals in the training set. On top of that, the proportion of high-income participants (colored purple) is much higher among males. As a result, a classifier can "learn" that a member of the female gender group has a lower chance of being high-income.

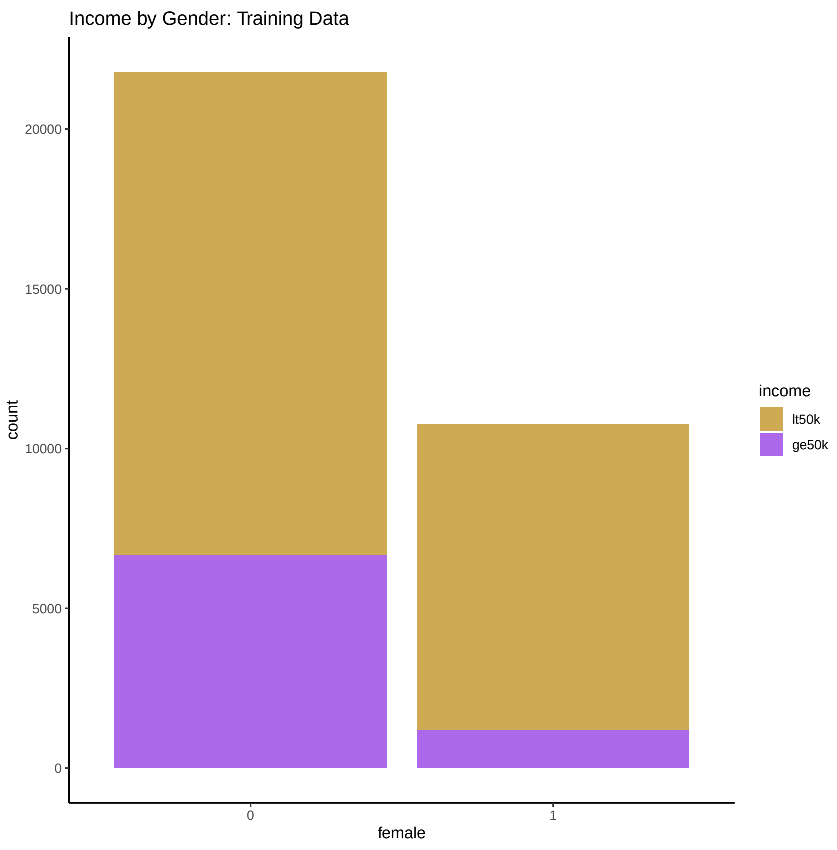

Figure 3.6: Grouped barchart of Adult training set.

However, our classifier, a sparse linear model, assigns a coefficient of 0 to the binary attribute `female` (this attribute takes value 1 if the observation is female). This means that the classification function does not use an observation's gender to make a prediction. Then, why is this classifier not fair across gender groups? Barocas et al. (2019) show that removing a sensitive attribute from data does not lead to fair classification, as there can exist other attributes that correlate with the sensitive attribute. As the localized class maps in Figure 3.5 show a difference in localized farness of gender groups to be a source of bias, a way to make the classifier fair could be through data pre-processing. We could sub-sample the data in a way that each gender group has the same sample size and same distribution of incomes.

It is important to note that localized class maps cannot only visually indicate whether a classifier meets the criteria of Independence and Sufficiency; it is not meant as a rigorous answer. Its main use is to give explanations as to why these criteria are, or are not, met. Because PAC and localized farness are computed independently, the localized class map is well-suited for identifying and explaining discrepancies in classifier performance across groups. Furthermore, fairness is a complex issue in classification, and in machine learning as a whole. Even if an algorithm meets all three criteria we defined in this example, it can still show behaviors that could deem it to be unfair (Barocas et al., 2019). There is much space for exploration on the topic of fairness, and the role that post-hoc interpretations such as the localized class map can play in assessing fairness of an algorithm.

Chapter 4

Discussion

When a classification model does not meet the objective of its deployment settings, there arises a need for explanations of its results. One tool that provides this in the form of post-hoc interpretations is the class map of Raymaekers et al. (2021). The class map visualizes data from the viewpoint of a trained classifier, allowing a user to assess performance at the class level and at the level of individual observations. It shows, for each object in a dataset, the probability that it belongs to an alternative class (PAC) and a measure of how far it lies from its own class (farness).

Errors that a classifier makes stem from its bias, which we defined as assumptions that a statistical learning model makes that is not based on strict adherence to training data; we noted that such training data is expected to accurately represent real-world settings. We divided sources of bias into model-driven, or bias from the choice of model, and data-driven, or bias a model inherits from exposure to unrepresentative training data. We believed that, in the presence of model-driven bias, the class map's measure of farness might not reveal potential sources of data-driven bias. This thesis aimed to adapt the class map such that model-driven bias and data-driven bias are defined independently. Our result was the localized class map, which used a classifier-agnostic measure of farness. It also served as a post-hoc interpretation tool for classifiers that do not define a distance metric.

We began by defining the components of the class map and the localized class map. Then, using examples from real-world data, we demonstrated the difference between the results of the localized class map and the class map. We then explored three use cases of the localized class map, each on a different real world dataset. We compared the results of different classifiers on the Yeast dataset, which contains extreme class imbalance; we explained the classification results of the Amazon book reviews data, which consistent of many ambiguous observations; and, we used the localized class map to visually assess whether a sparse linear classifier of the Adult dataset was fair with respect to gender.

Main Findings

We hypothesized that there exists a link between the local structure of data and the presence of data-driven bias in classification. Additionally, we believed that visualizing a measure of data-driven bias computed using local neighborhood information would be advantageous; replacing farness with such a measure would yield class maps with explanations that are better-suited for difficult data scenarios. To test our hypothesis, we developed examples with four real-world datasets to see if localized farness allowed us to identify data-driven bias. We also considered scenarios for model interpretability that are specifically suited for localized class maps.

In our first example, we classified the Floral buds data with Quadratic Discriminant Analysis and visualized the results of each class. We were successfully able to match observations with high localized farness to points on a t-SNE plot of the data that exist in mixed-class neighborhoods. This showed that the localized class map translates neighborhood information we get from t-SNE into a post-hoc explanation tool. For two classes of the Floral buds data, “supports” and “buds”, we compared the localized class map with the class map. While each map showed slight differences, they gave roughly similar pictures of the data. With this example, we found that the localized class map gives similar post-hoc interpretations as the class map, as it successfully identifies challenging points in the data.

In our second example, we classified the Yeast data with three different classifiers. We found that, because localized farness is computed independently of the classifier, we could use the localized class map to directly compare the PAC values output from each classifier. Our third example reproduced the Amazon book reviews example of Raymaekers et al. (2021). Instead of showing class maps, we showed localized class maps. Using the localized class maps, we successfully identified mislabeled points in the test set of the Amazon book reviews. This example highlighted the value of using localized farness in cases of ambiguous data, or data that contains many observations that do not perfectly fit one label.

From our final example, we found that the localized class map could be used to visually check if a classifier was fair with respect to a sensitive attribute. As in the second example, we showed the value of the localized class map as a comparison tool. However, a key finding in the final example was that we could use the localize class map to visually identify differences in the data-driven bias across levels of a sensitive attribute (Male and Female gender) as a reason for discrepancy in class-level sensitivity.

In all of the examples, we discussed how to use the localized class map to determine if the error of a classifier comes from model-driven bias, data-driven bias, or both. From this, we made suggestions of how to potentially improve each model. Overall, we found our hypotheses to be valid: that we can measure an observation's classification difficulty from local neighborhood information, and visualizing such a measure could provide explanation of model-driven bias.

Limitations and Further Research

In this thesis, we limited our exploration of sources of bias to model-driven and data-driven. Mitchell (1980) identifies three sources of bias that we did not directly consider in our definition: intended use of the learned generalizations, bias towards simplicity, and bias towards finding an analogy with previously-learned generalizations. Bias towards simplicity is especially interesting to consider. If a practitioner compares localized class maps of different classifiers, he or she might infer that the better model is one that fits more closely to the data (for example, a practitioner might prefer the MLP over the C4.5 classifier of the Yeast data in Figure 3.2). However, such a model might be more complex, and therefore more computationally expensive to fit. The interpretation of a localized class map is that a simpler model contains more model-driven bias; however, it does not reflect any costs of complexity. It would be interesting to consider localized class maps with an expanded definition of sources of bias, where a practitioner can decide the measures of comparison based on the use-case of the classification.

Furthermore, data-driven bias can be challenging to define and identify. We considered data-driven bias to be bias which a classifier inherits from training data that poorly represents the real-life deployment settings; Hellström et al. (2020) identifies several cases where this occurs. Our measure of data-driven bias, localized farness, is not guaranteed to identify such data. First, if an observation only deviates in a small way - for example, only one of its attributes is misspecified - it might not be identified with high local farness. Second, the success of localized farness depends on its parametrization. While we showed that the choice of k does not drastically change the overall pattern, it does make a noticeable difference in identifying points with high localized farness. Depending on a practitioner's choice of k, he or she can get a substantially different view of the data. More research needs to be done on finding the right choice for k.

Another limitation to localized farness is the need for a human being to determine cutoffs. A practitioner must manually identify points of interest on a localized class map and look at the raw data to identify where it provides challenges. This is shown in the Amazon book reviews example. While we successfully identified mislabeled points, we had to manually determine which points were of interest in the localized class map of positive examples (Figure 3.3), as none of the observations had extremely large values of localized farness. Furthermore, we had to read the reviews and use our human understanding of sentiment to determine why these points were locally far. It would be an improvement if localized farness could be developed in a more standardized way, where we could set cutoffs that make identification of challenging points automatic.

Finally, one of the most important applications of the localized class map is to not only give users explanations of classification models, but also insights into how to improve a poorly-performing classifier. More research needs to be done to find some general "best practices" drawn from localized class maps. This would especially be applicable for large-scale classification tasks.

Conclusion

Visualization is a powerful tool for providing post-hoc explanations of classification results. The challenge of developing such a tool is to successfully identify and measure sources of bias in classification. The proposed localized class map is a modification of the class map of Raymaekers et al. (2021). For each observation in a dataset, it shows its posterior probability of belonging to an alternative class, from the perspective of the classifier. In the horizontal direction, it shows how far an object is from its class with respect to its local neighborhood, agnostic of the classifier. This visualization allows a practitioner to identify whether classification error can be attributed to the type of model (model-driven bias) or the quality of the data (data-driven bias). Further applications include comparing results of different classifier, identifying mislabeled points in ambiguous data, and visually confirming fairness criteria of a classifier.

Bibliography

Arya, S. and Mount, D. (1998). Ann: library for approximate nearest neighbor searching.

Arya, S., Mount, D. M., Netanyahu, N. S., Silverman, R., and Wu, A. Y. (1998). An optimal algorithm for approximate nearest neighbor searching fixed dimensions. *J. ACM*, 45(6):891–923.

Barocas, S., Hardt, M., and Narayanan, A. (2019). *Fairness and Machine Learning*. fairmlbook.org. http://www.fairmlbook.org.

Dua, D. and Graff, C. (2017). UCI machine learning repository.

Friedman, B. and Nissenbaum, H. (1996). Bias in computer systems. *ACM Trans. Inf. Syst.*, 14(3):330–347.

Fürnkranz, J. and Kliegr, T. (2018). The need for interpretability biases. In Duivesteijn, W., Siebes, A., and Ukkonen, A., editors, *Advances in Intelligent Data Analysis XVII*, pages 15–27, Cham. Springer International Publishing.

Hellström, T., Dignum, V., and Bensch, S. (2020). Bias in machine learning what is it good (and bad) for? *CoRR*, abs/2004.00686.

Lipton, Z. C. (2018). The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery. *Queue*, 16(3):31–57.

Mitchell, T. M. (1980). The need for biases in learning generalizations. In Shavlik, J. W. and Dietterich, T. G., editors, *Readings in Machine Learning*, pages 184–191. Morgan Kauffman. Book published in 1990.

Napierala, K. and Stefanowski, J. (2015). Types of minority class examples and their influence on learning classifiers from imbalanced data. *Journal of Intelligent Information Systems*, 46(3):563–597.

Prettenhofer, P. and Stein, B. (2010). Cross-Language Text Classification using Structural Correspondence Learning. In Hajič, J., Carberry, S., Clark, S., and Nivre, J., editors, *48th Annual Meeting of the Association of Computational Linguistics (ACL 2010)*, pages 1118–1127. Association for Computational Linguistics.

Quinlan, J. R. (1993). *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Raymaekers, J., Rousseeuw, P. J., and Hubert, M. (2021). Class maps for visualizing classification results. *Technometrics*, page 1–24.

Trappenberg, T. and Back, A. (2000). A classification scheme for applications with ambiguous data. volume 6, pages 296–301.

van der Maaten, L. and Hinton, G. (2008). Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605. Accessible via: http://jmlr.org/papers/v9/vandermaaten08a.html.

Wouters, N., De Ketelaere, B., Deckers, T., De Baerdemaeker, J., and Saeys, W. (2015). Multispectral detection of floral buds for automated thinning of pear. *Comput. Electron. Agric.*, 113(C):93–103.

Appendices

Appendix A

Localized Class Maps of the Yeast data

Figure A.1 shows localized class maps of the test set of the Yeast data. An important note is that, for minority class “ME2” the C4.5 (labeled as J48) classifier has a higher accuracy than the MLP classifier. The former correctly identifies four points, while the latter correctly identifies three. Sometimes, points in the localized class map perfectly overlap. Thus, while the accuracy looks the same for the C4.5 and MLP classifiers, they are in fact different.

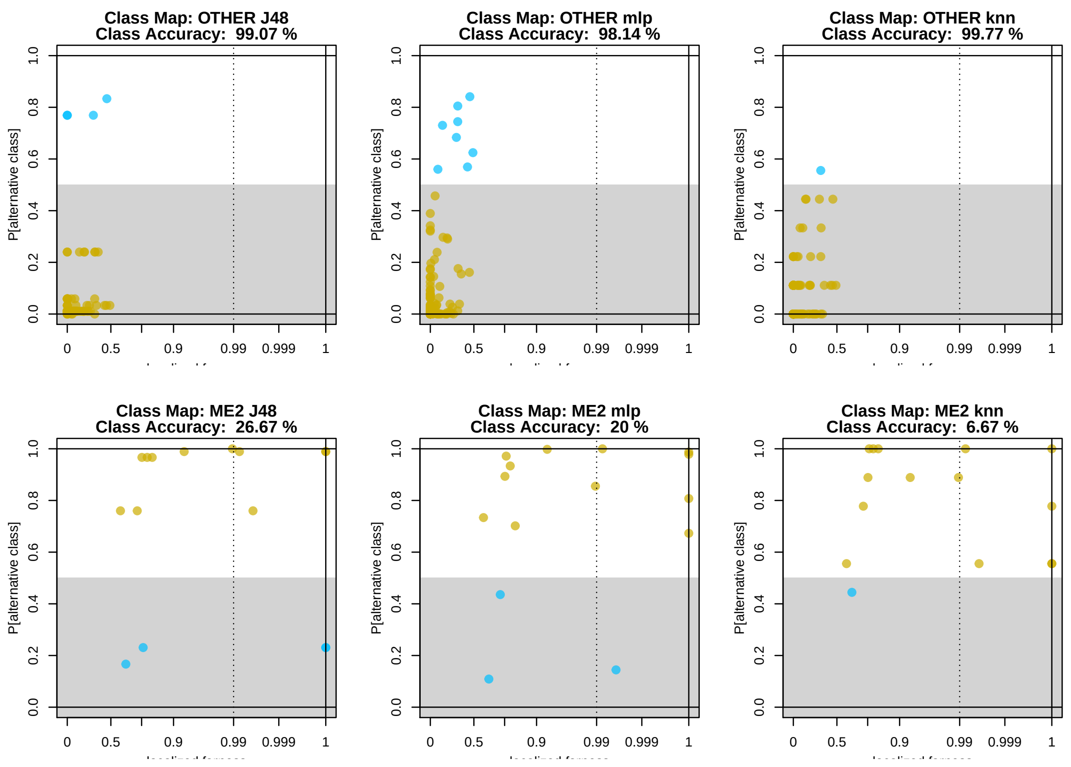

Figure A.1: Localized class maps of the test for classes ME2 and OTHER, using three different base classifiers.

Appendix B

Localized Class Maps of the Amazon Book Reviews Data

Figure B.1 and Figure B.2 show the localized class maps of the positive and negative reviews of the training data. The classifier overfits to the training data, and most observations have PAC = 0.

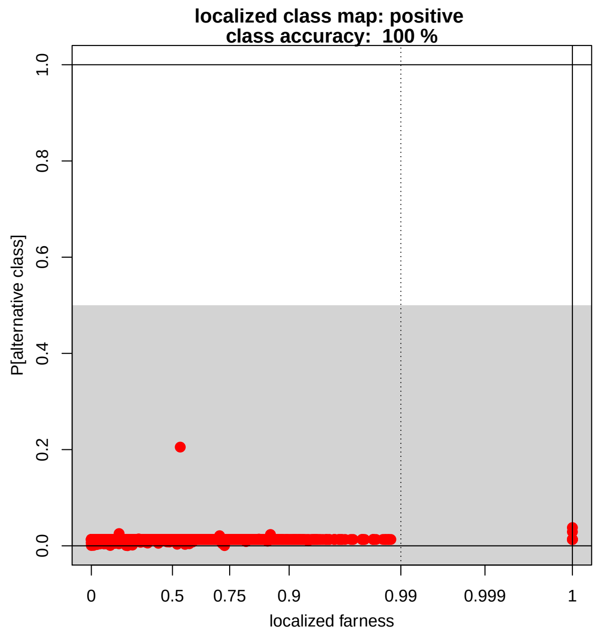

Figure B.1: Localized class maps for the training data of the Amazon book reviews, with ground truth class positive.

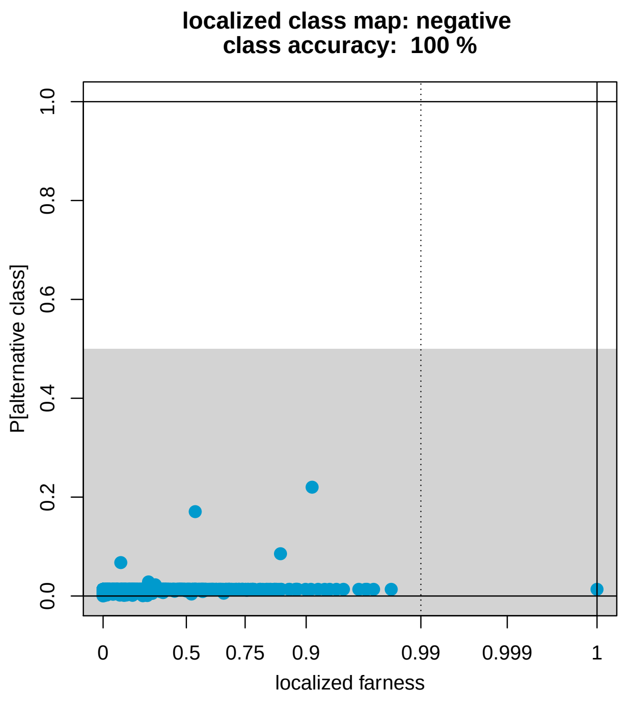

Figure B.2: Localized class maps for the training data of the Amazon book reviews, with ground truth class positive.

AFDELING
Straat nr bus 0000
3000 LEUVEN, BELGIË
tel. + 32 16 00 00 00
fax + 32 16 00 00 00
www.kuleuven.be