Model Correlation in Random Forests

Benjamin Moser

October 25, 2023

Abstract

...

Contents

1 Introduction

1.1 Overview

1.2 Contributions

1.3 Statistics

We want to reason about algorithms which act on data in a broadly applicable manner. We use statistical language for this. Consider for example a data point X that is used as input for an algorithm. Our intention is to say that this X could be essentially *any* data point from some kind of data source. In other words, we can consider X to be *random*, that is, X is a random variable (a symbol) that can take different values. Which values exactly it can take depends on the data source – the *distribution* of the data. If our data source is a fair 6-sided die, X can take values in $\Omega = \{1, \ldots, 6\}$ and the distribution of values is constant where each outcome has probability $\frac{1}{6}$, i.e. $\mathbb{P}[1] = \cdots = \mathbb{P}[6] = \frac{1}{6}$.

Definition 1.3.1 *A probability space is a triple $(\Omega, \Sigma, \mathbb{P})$ where*

- *Ω is an arbitrary set modelling the sample space i.e. the set of all possible outcomes.*
- *Σ is a σ-algebra over Ω, modelling the set of events.*
- *$\mathbb{P}$ is a function $\Sigma \to [0, 1]$ such that $\mathbb{P}(\Omega) = 1$ and $\mathbb{P}\left(\bigcup_{i=1}^{\infty} A_i\right) = \sum_{i=1}^{\infty} \mathbb{P}(A_i)$ for a countable collection of (pairwise disjoint) sets in Σ, and models the probability measure.*

In the following, we assume an underlying probability space implicitly.

Definition 1.3.2 *A random variable is a quantity that depends on a random event, i.e. a function $\Omega \to M$ (commonly, we have $M = \mathbb{R}$).*

Further, we not only want to reason about the behaviour of an algorithm with respect to one point X but *many* such points. A basic notion is the *expected value* of X. Further, since X is considered random, $f(X)$ is also a random variable and we can talk about the expected value of functions of random variables.

Definition 1.3.3 *The probability density function f_X of a random variable X is a nonnegative function such that*

$$\mathbb{P}(a < X < b) = \int_a^b f_X(x)dx$$

Definition 1.3.4 *The expected value (expectation) of random variable X is defined as*

$$\mathbb{E}[X] := \int x \cdot f_X(x)\, dx$$

where the integral is over the support of X.

- Note that a function $g(X)$ of a random variable X is a random variable itself and $\mathbb{E}[g(X)] = \int g(x) f_X(x)\, dx$.
- To emphasize the random variable the function is dependent on, we sometimes mention it in the subscript and write $\mathbb{E}_X[g(X)]$.

Lemma 1.3.1 *The following facts apply to expected values.*

- *A basic property of expected values is that they are linear: For any two random variables X, Y and a constant α it holds that $\mathbb{E}[X+Y] = \mathbb{E}[X] + \mathbb{E}[Y]$ and $\mathbb{E}[\alpha X] = \alpha \mathbb{E}[X]$.*
- *The rule of iterated expectations states that, for any function r: $\mathbb{E}[r(X,Y)] = \mathbb{E}[\mathbb{E}[r(X,Y)|X]]$ and, in particular, $\mathbb{E}[X] = \mathbb{E}[\mathbb{E}[X|Y]]$*
- *Jensen's Inequality: Let X be a random variable such that $\mathbb{P}(a \leq X \leq b) = 1$. If $g : \mathbb{R} \to \mathbb{R}$ is convex on $[a,b]$, then $g(\mathbb{E}[X]) \leq \mathbb{E}[g(X)]$.*
- *If X and Y are independent, then $\mathbb{E}[XY] = \mathbb{E}[X]\,\mathbb{E}[Y]$*

A random variable is *discrete* if its set of outcomes is a countable set $\{x_i, \ldots, x_n\}$ with probabilities $\{p_i, \ldots, p_n\}$. The probability density function then is $f_X(x_i) = \mathbb{P}(X = x_i) = p_i$. The expected value of a discrete random variable is given by a sum.

$$\mathbb{E}[X] = \sum p_i x_i$$

The expected value is a statement depending on the entire distribution. Usually, we do not know the distribution itself, but only have a limited set of samples of it. For instance, we may have a certain number of data points or run an algorithm a certain number of times and observe its output. We can use this set of samples to approximate the value of the expectation. Given that the samples are independent and identically distributed, the arithmetic mean of samples approximates the expected value.

$$\frac{1}{n} \sum_{i=1}^{n} x_i \;\to\; \mathbb{E}[X] \quad \text{as } n \to \infty$$

1.4 Supervised Learning

Our goal is to find an algorithm that is able to map objects from $\mathcal{X}$ to outcomes in $\mathcal{Y}$. Objects are described by their *features*. These are commonly numerical, so $\mathcal{X}$ can be thought of as $\mathbb{R}^d$ where d is the number of features. We will call such a representation of an object an *example*.

In *classification* problems, the outcomes are discrete among k possible outcomes and we refer to them as *labels* or *classes*. For sake of simplicity, we identify these with integers, i.e. $\mathcal{Y}$ can be thought of as the set $\{1, \ldots, k\}$. In *binary* classification, there are only two possible outcomes. Depending on what is more convenient in the mathematical context, we assume either $\mathcal{Y} = \{0, 1\}$ or $\mathcal{Y} = \{-1, 1\}$. In *regression* problems, the outcomes are continous and we refer to them as *estimates*. We can think of $\mathcal{Y}$ as $\mathbb{R}$.

The desired mapping $q : \mathcal{X} \to \mathcal{Y}$ may be nontrivial such that it is not feasible to come up with explicit rules of how to map examples to outcomes. However, we may try to algorithmically infer such a mapping from a given set of examples and their known outcomes. More specifically, we want to find a deterministic *learning algorithm*, also called *learner*, that, given a random input D, produces a mapping q_D [1]. We call q_D a *model* and note that it is dependent on D . The task of choosing and configuring

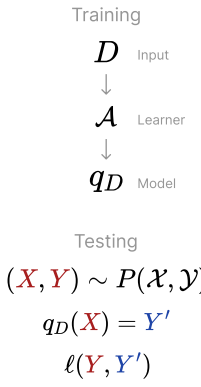

Figure 1.1: Illustration of the main components of supervised learning. A learning algorithm $\mathcal{A}$ produces a model q_D given some input D. The model is then evaluated on example-outcome pairs of the original data distribution.

1: For the statistical analysis it is essential that the learning algoritm is regarded as deterministic. However, any kind of randomness can be introduced by providing it with random input.

a learning algorithm is known as *supervised learning*, which we will be considering herein.

To analyse the problem, we presuppose a probability distribution $P(\mathcal{X}, \mathcal{Y})$ from which realisations of example-training pairs are drawn. We write this as $(X, Y) \sim P(\mathcal{X}, \mathcal{Y})$ where (X, Y) are random variables from a joint distribution. This distribution is unknown – else the problem is trivially solved already. In order for our solution to be widely applicable, we strive to make as few assumptions about P as possible.

Our given dataset $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$ can be considered a random vector D drawn from $P(\mathcal{X}, \mathcal{Y})^n$ where n is the number of data points [2]. The model q_D produced by our learning algorithm is a function $\mathcal{X} \to \mathcal{Y}$ that depends on the given dataset D. The prediction $q_D(X)$ is a random variable that depends on the random variables D and X.

To be useful, a model should not only accurately estimate outcomes for the given training examples, but also provide reasonable predictions for examples that were not part of the input to the learning algorithm We assess the quality of a single prediction with a *loss function* $\ell : \mathcal{Y} \to \mathcal{Y}$ whose value should be low if the predicted outcome is close to the true outcome. The loss at a single point x with ground-truth value y is then $\ell(y, x)$. The *risk* of model q_D is the expectation of the loss at a random example-outcome pair.

$$\text{Risk}(q_D) =_{\text{def}} \mathbb{E}_{(X,Y)\sim P}\left[\ell(Y, q_D(X))\right]$$

The quality of a given learning algorithm $\mathcal{A}$ then is the expectation over all possible inputs. This is the *expected risk*, also known as *generalisation error*:

$$\text{GE}(\mathcal{A}) =_{\text{def}} \mathbb{E}_D\left[\text{Risk}(q_D)\right] = \mathbb{E}_{(X,Y),D}\left[\ell(Y, q_D(X))\right]$$

1.5 Bias, Variance and their Effects

Since we are ultimately interested in the generalisation error of an ensemble, the question arises what forces influence it. To this end, one can strive to mathematically express the generalisation error in terms of specific meaningful quantities. A classical decomposition is the *bias-variance-decomposition*.

Informally, the generalisation error can be decomposed as

$$\text{(error)} = \text{(noise)} + \text{(bias)} + \text{(variance)}$$

The first quantity, *noise* describes inherent noise in the outcomes. It is intrinsic to the given data and can not be influenced by the choice of learner. The second quantity, *bias* is a measure of precision of the learning algorithm [3]. Intuitively, it describes how precise, on average, a model typically produced by the learning algorithm is. The third quantity, *variance* describes the spread of the learning algorithm. That is, it describes how different the models produced by the learner will be when given different realisations of the input random variable D – most prominently, different training data sets.

This decomposition is an essential tool to understand learning algorithms in general and ensembles such as Random Forests in particular. For the purpose of this thesis, it is important to understand the decomposition and its motivation in detail. We will begin by considering the widely known bias-variance-decomposition for regression using the squared-error loss $\ell(y, y') =_{\text{def}} (y - y')^2$. We will then proceed to generalise

2: The random variable D can actually encompass various sources of randomness. Most prominently, this is the training data. However, it can also reflect inherent randomness in the learning algorithm. For instance, weights in neural networks are sometimes initialised randomly. Unless otherwise stated, this is the intended meaning. If we explicitly distinguish between training data and model parameters, we denote these with the random variable Θ. We explain this formally in **??**.

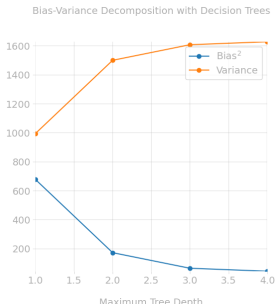

Figure 1.2: foo!

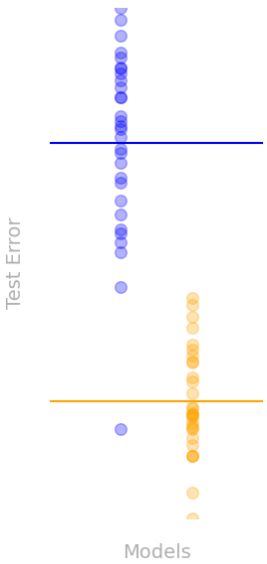

Figure 1.3: Visualising the variance of Decision Tree and Random Forest models. Each glyph corresponds to the test error of one model trained on a random subset of the full available data. The variation of the test error around the mean test error across many dataset samples is exactly the variance. Not only do Random Forests show lower test errors on average, they seem to also have lower variance. We will explain this observation in **??**

the decomposition to arbitrary loss functions. For sake of clarity, we will from now on take the dependence of $q_D(X)$ on X as understood and write only q_D.

3: Note that we are measuring the *learning algorithm* and not a produced model.

The variance of q_D with respect to the squared-error loss is commonly defined as the variance of the regressor around its expected value. This is the expected squared error between a random value (q_D) and the closest non-random value ($\mathbb{E}_D[q_D]$).

$$\mathrm{Var}_D(q_D) =_{\text{def}} \mathbb{E}_D\left[(q_D - \mathbb{E}_D[q_D])^2\right] = \min_z \mathbb{E}_D\left[(q_D - z)^2\right]$$

As such, $\mathbb{E}_D[q_D]$ is a centroid to the different realisations of q_D with respect to the loss function $\ell(y, y') = (y - y')^2$. This non-random centroid will turn out to be particularly interesting. For a random variable Z, let us denote its centroid with respect to Y as

$$Z^\star =_{\text{def}} \arg\min_z \mathbb{E}_Y\left[(Y - z)^2\right]$$

Let's consider two centroids: One with respect to the label distribution and one with respect to the model outputs.

- $y^\star(X) = \arg\min_z \mathbb{E}_Y\left[(Y - z(X))^2\right] = \mathbb{E}_{Y|X}[Y]$ is the *expected label*
- $q^\star(X) = \arg\min_z \mathbb{E}_D\left[(q_D(X) - z(X))^2\right]$ is the *expected model*.

Using this notation, the well-known bias-variance decomposition for the squared-error loss is given as follows. Note that each variance term is the expected distance in terms of loss to a certain centroid.

$$\mathbb{E}_{(X,Y),D}\left[(Y - q_D(X))^2\right] = \underbrace{\mathbb{E}_{(X,Y)}\left[(Y - y^\star(X))^2\right]}_{\mathrm{Var}(Y)\ \text{("noise")}} + \underbrace{\mathbb{E}_{X,D}\left[(q^\star(X) - q_D(X))^2\right]}_{\mathrm{Var}(q)\ \text{("learner variance")}} + \underbrace{\mathbb{E}_X\left[(y^\star(X) - q^\star(X))^2\right]}_{\mathrm{Bias}^2(Y,q)\ \text{("learner bias")}} \tag{1.1}$$

This decomposition is usually derived by expanding the square [**todo**]. The cross-terms then vanish due to that $q^\star = \mathbb{E}_D[q_D]$ and $y^\star = \mathbb{E}_{Y|X}[Y]$. This is but a special case of a more general structure applying to a certain class of losses. We will provide a more general proof in lemmas 4.4.1 and 4.4.2.

The first term, $\mathrm{Var}(Y)$ is independent of D and q_D. This means we have no means of influencing it with our choice of q_D. $\mathrm{Var}(Y)$ is also referred to as *noise*, *bayes error* or *irreducible error*. The second term, $\mathrm{Var}(q)$ measures the variance of our model around its non-random centroid with respect to different realisations of the random training dataset D. This can be understood as a measure of spread of the learning algorithm with respect to different realisations of D. The third term, $\mathrm{Bias}^2(q_D, Y)$ is the distance in terms of loss between the expected classifier and the expected label. This can be thought of as a measure of precision of our learning algorithm.

Note that we developed two things: One the one hand, we derived quantities that measure the notions of bias and variance. On the other hand, by virtue of these quantities appearing in the error decomposition 1.1, we have quantified the *effect* these quantities have on the generalisation error. This distinction is often overlooked because, like here, for many commonly used losses the quantities and their effects coincide. However, in general this is not necessarily true. We are particularly interested in the 0/1-loss for classification and a decomposition of it that is independent of the target has been proven to not exist [**todo**]. We will see that, while we cannot give a bias-variance decomposition for any loss, we can give a more general bias-variance-*effect* decomposition for which the former is a special case.

The variance-effect is the expected change in loss caused by using q_D instead of the non-random centroid $q^\star$. Likewise, the bias-effect is the expected change in loss caused

by using the expected model instead of the expected label. [4] Formally:

$$\text{Variance-Effect} =_{\text{def}} \mathbb{E}_{D,Y}\left[\ell(Y, q_D) - \ell(Y, q^\star)\right]$$
$$\text{Bias-Effect} =_{\text{def}} \mathbb{E}_{D,Y}\left[\ell(Y, q^\star) - \ell(Y, y^\star)\right]$$

4: In the original publication [**james_GeneralizationsBiasVariance_**], bias-effect is called the *systematic effect*, i.e. the effect of the systematic components. However, it is clearer to call this *bias-effect*, particularly when we begin to introduce notions of diversity in 4.1.

To shorten notation, and to give the equations a shape suggestive for later, we define:

Definition 1.5.1 *(Loss-Effect) For a loss funcion ℓ, and random variables Y, Z, Z', we define the change in loss between Z and Z' in relation to Y as:*

$$LE(Z, Z') =_{def} \ell(Y, Z) - \ell(Y, Z')$$

Putting this together, we can state a generalised bias-variance decomposition. The classical bias-variance decomposition for squared error (1.1) is a special case of this (**??**).

Theorem 1.5.1 *(Bias-Variance-Effect-Decomposition [**james_GeneralizationsBiasVariance_**]) For any loss function L, it holds that*

$$\mathbb{E}_{(X,Y),D}\left[\ell(Y, q_D)\right] = \underbrace{\mathbb{E}_Y\left[\ell(Y, y^\star)\right]}_{\textit{noise}} + \underbrace{\mathbb{E}_{(X,Y)}\left[LE(q^\star, y^\star,)\right]}_{\textit{bias-effect}} + \underbrace{\mathbb{E}_{(D,Y)}\left[LE(q_D, q^\star,)\right]}_{\textit{variance-effect}}$$

$$\textit{for} \quad LE(q^\star, y^\star) = \ell(Y, q^\star) - \ell(Y, y^\star)$$

$$LE(q_D, q^\star) = \ell(Y, q_D) - \ell(Y, q^\star)$$

This decomposition holds for *any* loss function ℓ since, by linearity of expectation, the individual terms on the right-hand-side simply cancel out. Further, this decomposition is independent of the definitions of $y^\star$ and $q^\star$.

1.6 Classifier Margins

A basic tool in the analysis of classifiers is the notion of margins. The margin can be thought to represent the classifier's confidence in its prediction. $\mathbb{P}[k|x]$ describes the probability estimated by the classifier that X is of class k.

Definition 1.6.1 *(Classifier margins [**tibshirani_ElementsStatisticalLearning_2017**]) The margin for class k of an example X is the difference between the model's confidence that X is of class k and the next-best class:*

$$m(x, y) =_{def} \mathbb{P}[y|X] - \max_{j \neq y} \mathbb{P}[j|X]$$

- *The vector $m(x) = [m_1(x), \ldots, m_K(x)]^\top$, where K is the total number of classes, is called a margin vector iff its components sum to zero.*
- *For a pair (X, y) of example and true outcome, the model's prediction is correct iff $m_y(X) > 0$*

Similarly, we can define a notion of margin for classification ensembles deciding by majority voting.

Definition 1.6.2 *(Ensemble margins for majority voting [**breiman**]) The margin for class y of an example x is the difference between the number of member votes for class Y and the number of votes for the next-best class.*

$$mr(x, y) =_{def} \frac{1}{M}\sum_{i=1}^{M} \mathbb{1}\left[q_i = y\right] - \max_{j \neq y} \frac{1}{M}\sum_{i=1}^{M} \mathbb{1}\left[q_i = j\right]$$

If ensemble members are parameterised by Θ, from the statistical point of view, we can also write

$$\mathrm{mr}(x, y) = \mathbb{P}_\Theta\left[q_\Theta = y\right] - \max_{j \neq y} \mathbb{P}_\Theta\left[q = j\right]$$

where the probabilities are conditioned on x. For binary classification under the 0/1-loss, the ensemble margin is linearly related to the ratio of incorrect members $\frac{1}{M}\sum_{i=1}^{M} \ell_{0/1}\left(y, q_i(x)\right) \approx \mathbb{P}_\Theta\left[q \neq y\right]$.

$$\begin{aligned}\mathrm{mr}(x, y) &= \mathbb{P}_\Theta\left[q = y\right] - \mathbb{P}_\Theta\left[q \neq y\right] \\ &= \left(1 - \mathbb{P}_\Theta\left[q \neq y\right]\right) - \mathbb{P}_\Theta\left[q \neq y\right] \\ &= 1 - 2\mathbb{P}_\Theta\left[q \neq y\right]\end{aligned}$$

So, $\mathrm{mr}(x, y) = 1 - 2\frac{1}{M}\sum_{i=1}^{M} \ell_{0/1}\left(y, q_i(x)\right)$.

1.7 Bregman Divergences and Centroids

To measure the difference between predicted and ground-truth outcomes, we use a loss function ℓ. The choice of loss function depends on the data domain, the learning task and computational considerations.

Often, learners are analysed with respect to a specific loss, such as the squared-error loss [**scornet_ConsistencyRandomForests_2015**] for regression or the 0/1-loss [**theisen_WhenAreEnsembles_2023**] or the KL-divergence [**webb_EnsembleNotEnsemble_2019**] for classification. The well-known bias-variance decomposition for the squared-error loss is usually shown directly in teaching materials [**tibshirani_ElementsStatisticalLearning_2017**, **weinberger_Lecture12Bias_**]. The question then arises which properties are specific to the loss function and which are part of a more general structure.

We will now define a family of loss functions, called *Bregman divergences*, that encompasses many widely used loss functions in supervised learning (see table 1.1).

Definition 1.7.1 *(Bregman Divergence [**todo**]) The Bregman divergence $B_\phi(p, q)$ is defined based on a generator function ϕ as follows:*

$$B_\phi(p, q) := \phi(\mathbf{p}) - \phi(\mathbf{q}) - \langle \nabla\phi(q), (\mathbf{p} - \mathbf{q})\rangle$$

where $\langle\cdot,\cdot\rangle$ is the inner product, $\nabla\phi(\mathbf{q})$ is the gradient vector of ϕ at $\mathbf{q}$ and $\phi : \mathcal{S} \to \mathbb{R}$ is a strictly convex function on a convex set $\mathcal{S} \subseteq \mathbb{R}^k$ such that it is differentiable on the relative interior of $\mathcal{S}$.

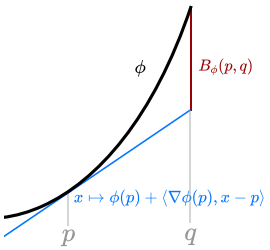

Figure 1.4: Given a strictly convex generator ϕ, the Bregman divergence for points p, q is the difference between the linear approximation around p and ϕ at the point q.

Bregman centroids A common measure is the "statistical" variance $(X - \mathbb{E}[X])^2$. One can see that $\mathbb{E}[X]$ is in fact the minimizer of the expected squared distances to realisations of X:

$$\mathbb{E}[X] = \arg\min_z \mathbb{E}_X\left[(X - z)^2\right]$$

As such, $\mathbb{E}[X]$ is a *centroid* of the realisations of X with respect to the squared error. This variance of a random variable around its centroid will be a very basic building block in our analysis of the generalisation error of ensembles. Concretely, we will explain that, except for the bias term, the well-known bias-variance decomposition in fact considers only such variances:

- The variance of the label around the expected label – this commonly referred to as *label noise* or *irreducible noise*.
- The variance of a concrete model q_D trained on a dataset D around the expected model – this is commonly referred to as the *(learner) variance*.

Further, we will proceed to show that a crucial component to ensemble learning will be the variance of a learner parameterised by Θ around its expected model with respect to the distribution of Θ. We will see that all these quantities can indeed be expressed as variances due to a specific property of Bregman divergences. For the $0/1$-loss, this takes a different form.

In order to talk about variances with respect to Bregman divergences, we need a notion of a centroid with respect to a Bregman divergence. Bregman divergences are in general not symmetric and hence there is a *left* and *right* centroid.

Lemma 1.7.1 *(Left and right Bregman centroids, [**pfau_GeneralizedBiasVarianceDecomposition_**])* *Let B_ϕ be a bregman divergence of generator $\phi : S \to \mathbb{R}$. For a random variable Y on S, it holds that*

- *the left Bregman centroid is* $\arg\min_z \mathbb{E}_z\left[B_\phi(z, X)\right] = (\nabla\phi)^{-1}\mathbb{E}\left[\nabla\phi(X)\right]$
- *the right Bregman centroid is* $\arg\min_z \mathbb{E}_z\left[B_\phi(X, z)\right] = \mathbb{E}[X]$

The left Bregman centroid is the expected value in the dual space implied by $\nabla\phi$. Due to this, we also write

$$\mathcal{C}[X] =_{\text{def}} (\nabla\phi)^{-1}\mathbb{E}\left[\nabla\phi(X)\right]$$

The left Bregman centroid will be important for considering the variance of a learner around its parametrisation Θ.

Table 1.1: Examples of commonly used loss functions that are Bregman divergences [**clustering with bregman divergences**, **wood23**]

Divergence $B_\phi(p, q)$	Generator $\phi(q)$	Domain S	Loss function
$(p-q)^2$	q^2	$\mathbb{R}$	Squared Error
$x\log\left(\frac{x}{y}\right) + (1-x)\log\left(\frac{1-x}{1-y}\right)$	$x\log x + (1-x)\log(1-x)$	$[0,1]$	Logistic loss
$\frac{x}{y} - \log\left(\frac{x}{y}\right) - 1$	$-\log x$	$\mathbb{R}_{>0}$	Ikura-Saito distance
$\|\| x - y \|\|^2$	$\|\| x \|\|^2$	$\mathbb{R}^d$	Squared Euclidean distance
$(x-y)^\top A(x-y)$	$x^\top A y$	$\mathbb{R}^d$	Mahalanobis distance
$\sum_{j=1}^d x_j \log_2\left(\frac{x_j}{y_j}\right)$	$\sum_{j=1}^d x_j \log_2 x_j$	d-simplex	KL-divergence
$\sum_{j=1}^d x_j \log\left(\frac{x_j}{y_j}\right) - \sum_{j=1}^d (x_j - y_j)$	$\sum_{j=1}^d x_j \log x_j$	$\mathbb{R}^d_{\geq 0}$	Generalized I-divergence
$\sum_{j=1}^d x_j \log x_j$	$\sum_{j=1}^d x_j \log x_j$	$\mathbb{R}_{\geq 0}$	Poisson loss

Generalised variance and Bregman information One can define a generalised measure of variance according to a Bregman divergence. As with the case for the squared error, this is the expected distance of a random point to a centroid.

Definition 1.7.2 *The variance around the right Bregman centroid is also known as the Bregman information* $I_\phi(X)$ *[***banerjee_ClusteringBregmanDivergences_2004***].*

$$I_\phi(X) =_{def} \mathbb{E}_X \left[B_\phi \left(X, \mathbb{E}_X \left[X \right] \right) \right]$$

For example, let $X = \{X_1, \ldots, X_n\} \subset \mathbb{R}^d$. Then the squared Euclidean distance corresponds to the sample variance [**banerjee_ClusteringBregmanDivergences_2004**].

$$B_\phi \left(p, q \right) = || \ p - q \ ||^2 \ \rightarrow \ \ I_\phi(X) = \frac{1}{n} \sum_{i=1}^{n} (X_i - \mathbb{E} \left[X \right])^2$$

For the KL-divergence, the bregman information is the *mutual information*. Consider a random variable X over probability distributions with probability measure p [**banerjee_ClusteringBregmanDivergences_2004**].

$$B_\phi \left(u, v \right) = \sum_{j=1}^{d} u_j \log \left(\frac{u_j}{v_j} \right) \ \rightarrow \ \ I_\phi(X) = \sum_{i=1}^{n} \sum_{j=1}^{m} p \left(u_i, v_j \right) \log \frac{p \left(u_i, v_j \right)}{p \left(u_i \right) p \left(v_j \right)}$$

2 Ensemble Learning

Ensemble Learning is the method of training M individual models $q_1, \ldots, q_M$ for a given task and aggregating their output by an aggregation function $\bar{q}$ to form an ensemble prediction [**zhou_EnsembleMethodsFoundations_2012**]. The individual models $q_1, \ldots, q_M$ are referred to as *members*. When all members are constructed using the same learning algorithm, we call it a *homogeneous* ensemble. The learning algorithm is then also referred to as the *base learner*. Otherwise the ensemble is *heterogeneous*. When analysing the generalisation error, we refer to $\bar{q}$ as the *ensemble model*. When considering how to actually combine ensemble predictions, we also refer to $\bar{q}$ as the *combiner*.

2.1 Notation

In the concrete approximation setting, i.e. given an actual ensemble of learners, we consider $\bar{q}$ and $q_1, \ldots, q_M$ to be functions $\mathcal{X} \to \mathcal{Y}$. In the statistical setting, $\bar{q}, q_1, \ldots, q_M$ are random variables dependent on a random variable X that represents the query example. Further, combiner and members depend on the input D to the ensemble construction algorithm. For sake of brevity, this dependence is implicit in the notation and we write $q_i =_{\text{def}} q_i(X; D)$.

In the original supervised learning setup (see section 1.4), we considered D to be input to the learning algorithm, particularly including any training data or randomness. Now we are considering multiple learning algorithms. Each of these individual learners receives its own input. For instance, each learner may be trained on a random subset of the available training data (this is *bootstrapping*, see section **??**). To this end, we consider D to be a random vector of random variables $(D_i, \ldots, D_M)$. Due to the law of total expectation, one can write

$$\mathbb{E}_D\left[\ell(y, q(x; D_i))\right] = \mathbb{E}_{D_i}\left[\ell(y, q(x; D_i))\right]$$

even if there are dependencies between the components of D. This means that for an individual member q_i, an expectation over D reduces to an expectation over D_i. This justifies considering expectations over D in general. In some cases, it is clearer to distinguish between input training data and other random parametrisation of the individual learner. In this case, we take D to be the training data and Θ to be a random variable representing other parameters. One can then write $\bar{q} = \mathbb{E}\left[q(x; \Theta)\right]$.

2.2 Methods

There are three main variants of ensemble learning [**mienye_SurveyEnsembleLearning_2022**]:

- *Parallel*: All members are trained independently. The outputs of all members are then aggregated to form the ensemble prediction.
- *Stacking* or *Meta-Learning*: All members are trained independently. The member outputs serve as input data for another learning algorithm, which then provides the ensemble prediction.
- *Sequential*: Members are trained in sequence. The output of the previous ensemble member informs the construction of the next member.

Random Forests [**breiman_RandomForests_2001**] are an example of parallel ensemble construction. M decision trees are constructed independently and the tree's predictions are aggregated by a kind of mean (see section 3.1). A classical example for sequential ensemble construction is *Boosting* [**schapire_BoostingFoundationsAlgorithms_2012**]. In boosting algorithms, the ensemble combiner $\bar{q}$ is not a mean but a (weighted) sum $\bar{q} = \sum_{i=1}^{M} \alpha_i q_i$. The first member q_1 provides a base prediction. Successive members are then trained to predict not an output value but *increments* (*pseudo-targets*) to the base prediction such that the sum $\alpha_1 q_1 + \alpha_2 q_2 + \dots$ moves towards a more precise prediction. In this thesis, we will focus on the Random Forest learner and variations of it. Although it can be seen as a parallel ensemble construction method, we will see that the distinction between parallel and sequential approaches blurs (see **??**).

2.3 Motivation

We will now review some arguments that motivate ensemble learning. We do this to provide context to the results in section **??**, which also clearly show when and how ensemble learning is beneficial.

The arithmetic mean combiner can be seen as approximating an expectation over member models, i.e. $\bar{q} = \mathbb{E}_\Theta [q_\Theta]$. This motivated abe to invoke Jensen's inequality . For a loss function ℓ that is convex in its first argument, it holds that

Jensen's inequality, in a probabilistic setting, states that, for a function $\phi : \mathbb{R} \to \mathbb{R}$ and a random variable X

ϕ convex $\to \phi(\mathbb{E}[X]) \leq \mathbb{E}\left[\phi(X)\right]$

$$\underbrace{\ell(\mathbb{E}_\Theta [q_\Theta(X)], Y)}_{\text{"ensemble loss"}} \quad \leq \quad \mathbb{E}_\Theta \left[\underbrace{\ell(q_\Theta(X), Y)}_{\text{"member loss"}} \right]$$

and thus

$$\mathbb{E}_\Theta [\ell(q_\Theta(X), Y)] - \ell(\mathbb{E}_\Theta [q_\Theta(X)], Y) \geq 0$$

This shows that, for convex loss functions and the arithmetic mean combiner, the ensemble loss is always smaller-equal than the average member error. abe interpret this *Jensen gap*, i.e. the difference between these quantities, as a measure of ensemble improvement.

Variance reduction for squared-error regression A common motivation for using ensembles over single models is that the combination of models reduces the variance as compared to the expected variance of a single model. Further, the bias is not affected, i.e. the ensemble bias is the same as the bias of any member. In the regression setting, under the squared-error loss and the arithmetic mean combiner, this can be seen as follows [**stackexchange**]. Assume $q_1, \dots, q_M$ are identically and independently distributed with equal variance σ^2.

$$\text{Var}(\bar{q}) = \text{Var}\left(\frac{1}{M}\sum_{i=1}^{M} q_i\right) = \frac{1}{M^2}\sum_{i=1}^{M} \text{Var}(q_i) = \frac{1}{M}\sigma^2$$

As the number of members M increases, the ensemble variance is reduced. Further, one can also see that the interactions between members determine the variance reduction. Assume ensemble members have equal pairwise covariance. Then

$$\rho =_{\text{def}} \frac{\text{Cov}(q_i, q_j)}{\sigma^2} \quad \leftrightarrow \text{Cov}(q_i, q_j) = \rho\sigma^2$$

Further,

$$\text{Var}\left(\bar{q}\right) = \text{Var}\left(\frac{1}{M}\sum_{i=1}^{M} q_i\right) = \frac{1}{M^2}\left(\underbrace{\sum_{i=1}^{M}\text{Var}\left(q_i\right)}_{M\sigma^2} + \underbrace{2\sum_{i<j}^{M}\text{Cov}(q_i, q_j)}_{M(M-1)\rho\sigma^2}\right) = \frac{\sigma^2}{M} + \frac{M-1}{M}\rho\sigma^2$$

One can show that $\rho \geq 0$ [**louppe_UnderstandingRandomForests_2015**]. This illustrates that ensemble variance is smallest if the members are not correlated, i.e. $\rho \approx 0$.

Variance reduction for classification margins For classification, a classical analysis is the bound given by **breiman_RandomForests_2001** in [**breiman_RandomForests_2001**], in which Random Forests are first introduced. The basic idea is to consider variances with respect to the ensemble margin margin (see def. 1.6.2). We open with Chebychev's inequality to bound the generalisation error in terms of the variance of the ensemble margin.

$$\mathbb{E}\left[\ell(Y, \bar{q}(X))\right] = \mathbb{P}\left[\text{mr}(X, Y; D) < 0\right] \quad \leq \quad \frac{\text{Var}\left(\text{mr}(X, Y; D)\right)}{\mathbb{E}_{X,Y}\left[\text{mr}(X, Y; D)\right]^2}$$

We can already see that we have an interaction between the performance of the individual members, as reflected in the ensemble margin $\mathbb{E}_{X,Y}\left[\text{mr}(X, Y; D)\right]$ and what we will shortly show to be indicative of the diversity of the ensemble. The generalisation error is in part determined by the ratio of these two quantities.

$s =_{\text{def}} \mathbb{E}_{X,Y}\left[\text{mr}(X, Y; D)\right]$ is also called the *strength* of the ensemble. s is assumed to be non-negative. For binary classification, this is equivalent to the weak-learner assumption (see 4.6.1).

Note that

$$\text{mr}(X, Y; D) = \mathbb{E}_{\Theta}\left[\underbrace{\mathbb{1}\left[q_i = Y\right] - \mathbb{1}\left[q_i = K\right]}_{=_{\text{def}}\ \text{rmg}(X,Y,\Theta)} \mid (X, Y), D\right]$$

where K is the next-best class and we define the *raw margin function* $\text{rmg}(X, Y, \Theta)$ to be the inner part of that expectation. So, $\text{mr}(X, Y; D) = \mathbb{E}_{\Theta}\left[\text{rmg}(X, Y, \Theta) \mid (X, Y), D\right]$. Notably, mr is the *ensemble* margin measuring the ratio of incorrect members and rmg corresponds to the *classifier* margin, measuring the ratio of incorrectly classfied examples by a member.

Theorem 2.3.1 *([**breiman_RandomForests_2001**]) The variance of the ensemble margin can be expressed in terms of the covariance between the raw member margins of two members parameterised by i.i.d* Θ, Θ'.

$$Var_{X,Y}\left(mr(X, Y; D)\right) = \mathbb{E}_{\Theta,\Theta'}\left[Cov_{(X,Y)}(rmg(\Theta), rmg(\Theta'))\right]$$

Proof. For brevity, we write $Z =_{\text{def}} (X, Y)$, $\text{mr}(Z) =_{\text{def}} \text{mr}(X, Y; D)$ and $\text{rmg}(\Theta) =_{\text{def}}$

$\text{rmg}(X, Y; \Theta)$. By the definition of variance, have

$$\begin{aligned}\text{Var}_Z\left(\text{mr}(Z)\right) &= \mathbb{E}_Z\left[\left(\text{mr}(Z) - \mathbb{E}_Z\left[\text{mr}(Z)\right]\right)^2\right] \\ &= \mathbb{E}_Z\left[\text{mr}(Z)^2\right] - \mathbb{E}_Z\left[\text{mr}(Z)\right]^2\end{aligned}$$

For the left term, by the rule of iterated expectation and the fact that Z and Θ are independent (see lemma 1.3.1), it holds that

$$\mathbb{E}_Z\left[\text{mr}(Z)^2\right] = \mathbb{E}_Z\left[\mathbb{E}_\Theta\left[\text{rmg}(\Theta) \mid Z\right]^2\right] = \mathbb{E}_{Z,\Theta}\left[\text{rmg}(\Theta)^2\right] = \mathbb{E}_\Theta\left[\mathbb{E}_Z\left[\text{rmg}(\Theta)^2\right]\right]$$

For the right-hand-side term, we can make use of the fact that, for a function f, it holds that $\mathbb{E}_\Theta\left[f(\Theta)^2\right] = \mathbb{E}_{\Theta,\Theta'}\left[f(\Theta)f(\Theta')\right]$ where Θ and Θ' are independent and identically distributed. Then we apply the rule of iterated expectation and exploit that Z and Θ are independent.

$$\begin{aligned}\mathbb{E}_Z\left[\text{mr}(Z)\right]^2 &= \mathbb{E}_Z\left[\mathbb{E}_\Theta\left[\text{rmg}(\Theta) \mid Z\right] \cdot \mathbb{E}_{\Theta'}\left[\text{rmg}(\Theta') \mid Z\right]\right] \\ &= \mathbb{E}_Z\left[\mathbb{E}_\Theta\left[\text{rmg}(\Theta)\right]\mathbb{E}_{\Theta'}\left[\text{rmg}(\Theta')\right]\right] \\ &= \mathbb{E}_{\Theta,\Theta'}\left[\mathbb{E}_Z\left[\text{rmg}(\Theta)\right]\mathbb{E}_Z\left[\text{rmg}(\Theta')\right]\right] \\ &= \mathbb{E}_\Theta\left[\mathbb{E}_Z\left[\text{rmg}(\Theta)\right]^2\right]\end{aligned}$$

Plugging these equalities in the decomposition of variance above proves the statement.

$$\begin{aligned}\text{Var}_Z\left(\text{mr}(Z)\right) &= \mathbb{E}_\Theta\left[\mathbb{E}_Z\left[\text{rmg}(\Theta)^2\right]\right] - \mathbb{E}_\Theta\left[\mathbb{E}_Z\left[\text{rmg}(\Theta)\right]^2\right] \\ &= \mathbb{E}_\Theta\left[\mathbb{E}_Z\left[\text{rmg}(\Theta)^2\right] - \mathbb{E}_Z\left[\text{rmg}(\Theta)\right]^2\right] \\ &= \mathbb{E}_\Theta\left[\text{Var}_Z\left(\text{rmg}(\Theta)\right)\right] \\ &= \mathbb{E}_{\Theta,\Theta'}\left[\text{Cov}_Z(\text{rmg}(\Theta), \text{rmg}(\Theta'))\right]\end{aligned}$$

□

This is the expected covariance between the classification margins of two individual members. As we have seen before for the regression case, the generalisation error is lower if individual members are uncorrelated. Unfortunately, this is only an upper bound. In section 2.4, we will be able to make the same observation from an exact decomposition of the generalisation error.

Ensemble bias equals average member bias It can be shown that the bias of a homogeneous ensemble with the arithmetic mean combiner is equal to the average bias of the ensemble members **[louppe_UnderstandingRandomForests_2015]**. We will give an illustrative argument for the arithmetic mean combiner here. We will show this in detail in a more general and inuitive way in section **??**.

Consider individual learner inputs $D = (D_1, \ldots, D_M)$. Each member depends on some D_i and the combiner $\bar{q}$ depends wholly on D. Assuming that $D_1, \ldots, D_M$ are independent and identically distributed, we can write

$$\mathbb{E}_{D,\Theta}\left[\bar{q}\right] = \mathbb{E}_D\left[\frac{1}{M}\sum_{i=1}^{M} q_{D_i}\right] = \frac{1}{M}\sum_{i=1}^{M}\mathbb{E}_{D_i}\left[q_{D_i}\right] = \mathbb{E}_{D'}\left[q_{D'}\right]$$

where D' is distributed as any D_i. We can conclude $\mathbb{E}_D[\bar{q}] = \mathbb{E}_D[q_D]$ (see section 2.1). This implies

$$\bar{q}^\star = \mathbb{E}_D[\bar{q}] = \mathbb{E}_D[q_D] = q^\star$$

and thus the bias of the ensemble is the same as the bias of a member model q:

$$\mathbb{E}_X\left[\ell(y^\star(X), \bar{q}^\star(X))\right] = \mathbb{E}_X\left[\ell(y^\star(X), q^\star(X))\right]$$

This argument depends on the linearity of expectations, the arithmetic mean combiner and the fact that the ensemble members are constructed following the same parameter distribution. In other words, the ensemble is *homogeneous*. We will later show this more directly for any loss function and any combiner (see 4.6.1).

This implies that the ensemble improvement for homogeneous ensembles under the arithmetic mean combiner is solely due to variance reduction.

2.4 Bias, Variance and Covariance

In section 2.3, we have already seen hints that the covariance between members is an essential factor to ensemble performance. Indeed, the notion of covariance and uncorrelatedness has been a guiding thought in the literature [**didaci_DiversityClassifierEnsembles_2013**, **brown_ManagingDiversityRegression_2005**, **buschjager_GeneralizedNegativeCorrelation_2020**]. We will now introduce an exact decomposition of the ensemble error for the squared-error loss that includes the average covariance between member predictions.

Theorem 2.4.1 *(Bias-Variance-Covariance decomposition **??**) It holds that*

$$\mathbb{E}_{(X,Y),D}\left[(y - \bar{q})^2\right] = \overline{bias}^2 + \frac{1}{M}\overline{var} + \left(1 - \frac{1}{M}\right)\overline{covar}$$

$$\text{for } \overline{bias} =_{def} \frac{1}{M}\sum_{i=1}^{M}(\mathbb{E}_D[q_i] - y)$$

$$\overline{var} =_{def} \frac{1}{M}\sum_{i=1}^{M}\mathbb{E}_D\left[(q_i - \mathbb{E}_D[q_i])^2\right]$$

$$\overline{covar} =_{def} \frac{1}{M(M-1)}\sum_{i\neq j}\mathbb{E}_D\left[(q_i - \mathbb{E}_D[q_i])(q_j - \mathbb{E}_D\left[q_j\right])\right]$$

This can be interpreted as a decomposition into characteristics of individual learners (mean bias and variance), plus a quantity describing the interactions between diferent learners. Again, one can see that ensemble performance profits if members are uncorrelated. However, this decomposition is only given for the squared error and the arithmetic mean combiner.

2.5 Applications

...

3 Random Forests

In this chapter, we describe the Random Forest learning algorithm. We first motivate and describe decision trees, which are the basic components of a Random Forest. We then proceed to describe a particular property of decision trees: they are likely to exhibit high variance. While this is undesirable for a learning algorithm, we will see that combining several randomized decision trees into a Random Forest ensemble turns exactly that property into a crucial advantage (see section 4.6.1).

In its essence, a Random Forest is a collection of randomized decision trees. A decision tree is a data-driven recursive partitioning scheme, combined with a means to produce a prediction based on the training points in a partition cell. The Random Forest prediction then is an aggregate of the predictions of all individual trees.

3.1 Decision Trees

As described in section 1.4, we are interested in learning algorithms that, given some training data, produce a model that is able to predict a reasonable outcome when queried with a previously unseen example.

One intuitive approach is to consider the examples in the training data that are, in some sense, "close" or "similar" to the query point. Then, one might claim that the outcome for the query point must surely be similar to the outcomes of the similar points – which we already know. Indeed, finding a proper notion of "closeness" is at the heart of many machine learning algorithms such as k-Nearest-Neigbours, k-Means, etc.

Constructing a decision tree means recursively partitioning the input space $\mathcal{X}$, guided by the training data D. Then, given a query X, we check the partition cell that X belongs to and all the training examples that are in it. These are the examples we consider "close" to X. The tree's prediction will be an aggregation of the outcomes of all training points in that cell.

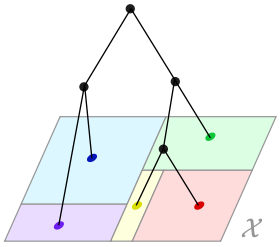

Figure 3.1: Rendering of a decision tree structure. Each inner node corresponds to a partitioning of the parent edge. In standard decision trees, this is a binary partition. In other words, the examples are *split* at a certain value threshold in a certain feature dimension.

Because we are recursively partitioning the input space, we have at hand a tree structure of decision rules. The cells of the resulting partition are the leaves of the tree. The non-leaf nodes are also referred to as *decision nodes*, but there is no inherent difference between leaf and non-leaf nodes. We will use the terms *leaf* and *cell* interchangeably, depending which aspect we want to emphasize.

In summary, there are the following main components to the implementation of a decision tree:

- The *splitting criterion* to apply recursively to subsets of the training data.
- The *stopping criterion* that determines whether a node should be split further. This will determine the depth of the decision tree.
- The *leaf aggregation function* that produces a prediction for a specific cell. When using the constructed tree for prediction, this will be the leaf node that the query point is assigned in.

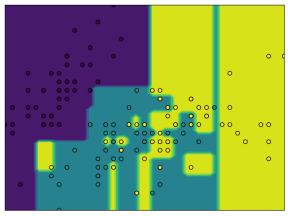

Figure 3.2: Decision boundaries of a tree in two (arbitrary) features constructed on the *iris* dataset. Visualisation based on [_SklearnInspectionDecisionBoundaryDis

3.1.1 Centroids are good leaf combiners

We will consider the leaf aggregation function first. Consider a parent node P that, due to some split, was partitioned into the disjoint union $L \dot\cup R$. Let y_P, y_L and y_R be the output values of the parent and the two new leaf nodes, produced by the leaf aggregation function. Since the leaf output is constant over a single cell, the gain in loss due to a split is the difference between the loss of the parent node and the sum of losses of the two individual child nodes. For brevity, we write $\ell_P(y) =_{\text{def}} \sum_{i \in P} \ell(y, y_i)$.

$$\textit{Loss Gain}: \quad \sum_{i \in P} \ell(y_P, y_i) - \left(\sum_{i \in L} \ell(y_L, y_i) + \sum_{i \in R} \ell(y_R, y_i) \right)$$

$$= \ell_P(y_P) - (\ell_L(y_L) + \ell_R(y_R))$$

In order for a split to yield positive loss gain, the leaf aggregation function needs to be such that the loss does not increase as constraints are removed, i.e. the set of considered examples is reduced. Recall that $\bar z$ is a centroid with respect to a loss function ℓ and a set of outcomes P if and only if $\bar z = \arg\min_z \sum_{i \in P} \ell(z, y_i) = \arg\min_z \ell_P(z)$ (**??**).

Lemma 3.1.1 *For a loss function ℓ, if the leaf aggregator of a node P is $y_P =_{def} \arg\min_z \sum_{i \in P} \ell(z, y_i)$, i.e. the centroid with respect to ℓ, the loss gain is nonnegative.*

Proof. Let $\ell_P(y) =_{\text{def}} \sum_{i \in P} \ell(y, y_i)$. Since $P = L \dot\cup R$, we need to show that $\ell_P(y_P) = \ell_L(y_P) + \ell_R(y_P) \geq \ell_L(y_L) + \ell_R(y_R)$. Assume $\ell_L(y_P) < \ell_L(y_L)$. This contradicts the definition of y_L as the minimizer, and as such $\ell_L(y_P) \geq \ell_L(y_L)$. Likewise, we can conclude that $\ell_R(y_P) \geq \ell_R(y_R)$. Combining the two inequalities yields the statement. □

This rigorously motivates the specific choice of leaf aggregation function. The majority vote is a centroid with respect to the 0/1-loss for classification, while the arithmetic mean is the centroid with respect to the squared error loss for regression.

3.1.2 Splitting criteria greedily minimise loss functions

In the best possible case, all training examples in a given cell correspond to the same (classification) or very similar (regression) outcomes. If a query point then falls within that cell, i.e. it has similar features, one can say with high confidence that the query point should have the same (similar) outcome. In the spirit of greedy optimisation, we aim to split cells such that the resulting child cells are more *pure* with respect to their outcomes.

Note that the notion of local purity is linked to the training error: If a leaf cell is perfectly pure, all training examples in that cell correspond to the same outcome. Hence, the leaf aggregation function, which us usually implemented as some kind of mean, will produce exactly that outcome for any query point that belongs to this cell, in particular any training points. Consequently, for a suitable definition of "error", perfectly pure cells have zero training error.

Consider a split, parameterised by Θ, that partitions a parent node P into the disjoint union $L_\Theta \dot\cup R_\Theta$. Let n, n_L, n_R be the cardinalities of the parent and the two child

The splitting function can be understood as a very simple classifier. Standard decision trees, are restricted to axis-aligned splits that divide the data points by a certain threshold value in a certain feature dimension. Extensions have been proposed to construct oblique (non-axis-aligned) splits or to use more sophisticated classifers instead [**todo**]. In this work, we focus on axis-aligned splits but note that results and insights should transfer to other splitting functions.

nodes. Let H be an impurity measure. We will select the split that yields the lowest impurity.

$$\arg\min_{\Theta} \frac{n_L}{n} H(L_\Theta) + \frac{n_R}{n} H(R_\Theta)$$

The gain in purity is then the difference between impurities before and after the split.

$$\textit{Purity Gain}: \quad H(P) - \left(\frac{n_L}{n} H(L_\Theta) + \frac{n_R}{n} H(R_\Theta)\right) \tag{3.1}$$

Note that this is different from the gain in *loss* achieved due to a split.

We now proceed to define two commonly used splitting criteria. These are the Gini Index for classification and Variance Reduction (also known as CART) for regression [**tibshirani_ElementsStatisticalLearning_2017**]. Obviously, a reasonable impurity measure should also also lead to a positive loss gain. Often, this is intuitively clear but a comprehensive explanation appears to be hard to find in the literature, particularly for the case of Gini impurity. We clarify this in the following.

Variance Reduction

A commonly used impurity measure for regression is the squared-error variance.

$$H_{\text{var}}(P) =_{\text{def}} \frac{1}{n_P} \sum_{i \in P} (y_i - y_P)^2 \qquad \text{for} \quad y_P =_{\text{def}} \frac{1}{n_p} \sum_{i \in P} y_i$$

To motivate this impurity measure, it remains to be shown that a split guided by this impurity measure actually reduces the value of a specific loss function and which one that is. Luckily, it is easy to see that this holds for the squared error loss. Plugging the definition into the purity gain (eq. (3.1)) yields

$$H(P) = \frac{1}{n_P} \sum_{i \in P} (y_i - y_P)^2 = \frac{1}{n_P} \underbrace{\sum_{i \in L} (y_i - y_P)^2}_{\ell_L(y_P)} + \frac{1}{n_P} \underbrace{\sum_{i \in R} (y_i - y_P)^2}_{\ell_R(y_P)}$$

and

$$\frac{n_L}{n_P} H(L) + \frac{n_R}{n_P} H(R) = \frac{n_L}{n_P} \frac{1}{n_L} \underbrace{\sum_{i \in L} (y_i - y_L)^2}_{\ell_L(y_L)} + \frac{n_R}{n_P} \frac{1}{n_R} \underbrace{\sum_{i \in R} (y_i - y_R)^2}_{\ell_R(y_R)}$$

By lemma 3.1.1, we can directly conclude that the loss gain is positive for any split if the arithmetic mean is used as a leaf combiner.

Gini Index

One may suggest a measure of purity as the probability of drawing two different outcomes from the examples in the current cell. Let $p_k = \mathbb{P}_P[k|X]$ be the probability of drawing an example of class k from node P. The probability of drawing one example of class k and one of a different class is $p_k(1 - p_k)$. The probability of drawing two examples of *any* two different classes then is the *Gini index*

$$G =_{\text{def}} \sum_k p_k (1 - p_k)$$

In binary classification, i.e. if outcomes are in $\{0, 1\}$, the squared error reduces to the 0/1-loss. As such, the mean 0/1-loss, i.e. the error rate, trivially also is a measure of variance.

Another impurity measure is the entropy, defined as

$$H_{\text{entr}}(P) =_{\text{def}} -\sum_{i \in P} p_k(x_i) \log(p_k(x_i))$$

With a similar argument as for the case of variance reduction, one can see that the entropy splitting criterion amounts to minimising the the cross-entropy loss given as

$$-\frac{1}{n} \sum_i^n \sum_k \mathbf{1}[y_i = k] \log(p_k(x_i))$$

We will now argue that a reduction in the Gini index in fact pushes values p_k to the extremes of the probability simplex. We perform a slight shift in perspective and consider the classification *margin* (see 1.6.1) instead of the estimated probabilites.

We will argue that the Gini index split criterion, which finds a split such that the Gini index is reduced, in fact maximises the classification margins.

Recap: The classifier margin for class k of an example X is the difference between the model's confidence that X is of class k and the next-best class:

$$m_k(X) =_{\text{def}} \mathbb{P}[k|X] - \max_{j \neq k} \mathbb{P}[j|X]$$

For a pair (X, y) of example and true outcome, the model's prediction is correct iff $m_y(X) > 0$ The vector $m(x) = [m_1(x), \ldots, m_K(x)]^\top$, where K is the total number of classes, is called a *margin vector* iff its components sum to zero.

Lemma 3.1.2 ⋆ *Let p be a probability distribution and u an arbitrary vector Let $G = \sum_k p_k(1-p_k)$ be the Gini index. Then $-G$ is the generator for the Bregman divergence*

$$B_{-G}(p,u) = \sum_k (p_k - u_k)^2$$

Proof. Let $\phi(q) =_{\text{def}} (-1)\sum_k p_k(1-p_k)$. Then, the first equality follows by definition of a Bregman divergence (see 1.7.1) and the second equality by arithmetic.

$$\begin{aligned} B_{-G}(p,u) &= \underbrace{(-1)\sum_k p_k(1-p_k)}_{\phi(p)} - \underbrace{(-1)\sum_k u_k(1-u_k)}_{\phi(u)} - \underbrace{\sum_k (2u_k-1)(p_k-u_k)}_{\langle \nabla\phi(u), p-u\rangle} \\ &= \sum_k (p_k - u_k)^2 \end{aligned}$$

□

Note further that maximising the value of the generator function with respect to one parameter $\phi(p)$ while leaving the other fixed also maximises the divergence $B_\phi(p,u)$.[1] The sign of the generator value and the sign of the divergence are related in that $B_{-\phi}(p,u) = -B_\phi(p,u)$. This means that minimising the Gini index during splitting maximises component-wise sum of squared errors between p and u.

1: TODO explain, at least with intuition. Convexity of $B_\phi(.,.)$ etc

$$\min G \longrightarrow \min B_G(p,u) \longrightarrow \max B_{-G}(p,u) = \sum_k (p_k - u_k)^2$$

If p is a probability distribution and $u =_{\text{def}} [\frac{1}{k}, \ldots, \frac{1}{k}]^\top$, then $p - u$ is a margin vector and the optimisation corresponds to maximising the classification margins as measured by the squared error.

A common approach in training classification models is *margin maximisation* [**schapire_BoostingFoundationsAlgorithms_2012**] in which we aim to maximise the margin of the true label $m_y(X)$. A margin loss function $\ell : \mathbb{R} \to \mathbb{R}$ is a *margin-maximising* loss if $\ell'(m_y(X)) \leq 0$ for all values of m_y [**leistner_SemiSupervisedRandomForests_2009**].

An example for a margin-maximising loss function is the *hinge loss* defined as $\ell(m_y(x)) =_{\text{def}} \max\{0, 1-p\}$. Its subderivative is

$$\frac{\partial \ell}{\partial p} = \begin{cases} -1 & p \leq 1 \\ 0 & \text{else} \end{cases}$$

and hence it is a margin-maximising loss.

A decision tree can also be evaluated based on a margin loss. The empirical error of a decision tree node P with respect to a margin loss ℓ can be written as $L(P) = \frac{1}{|P|}\sum_{i\in P} \ell\left(m_y(x_i)\right)$ where $m_y(x_i)$ is the value of the true margin. Then the following holds [**leistner_SemiSupervisedRandomForests_2009**].

$$\begin{aligned} L(P) &= \frac{1}{|P|}\sum_{i\in P}\sum_k \mathbb{1}[y_i = k] \cdot \ell\left(m_y(x_i)\right) \\ &= \sum_k \frac{1}{|P|}\sum_{i\in P} \mathbb{1}[y_i = k] \cdot \ell(m_k(x_i)) \\ &= \sum_k p_k(x_i)\ell(m_k(x_i)) \end{aligned}$$

Hence we see that the Gini index splitting criterion greedily optimises classification margins and thus margin-maximising losses such as the Hinge loss.

3.1.3 Splitting criteria as 2-means clustering

If indeed a centroid is chosen as leaf combiner, as is the case in standard random forests, common splitting functions can be seen to be equivalent to a generalised k-means for $k = 2$. **[banerjee]** generalise k-means to bregman divergences. The derivation is analogous to the classical case for squared error.

Let I_ϕ be the Bregman information as defined in definition 1.7.2. Let X be a random variable representing data points. Let M be a random variable taking values in $\mathcal{M}$ representing the set of cluster representatives. Then the objective for generalised k-means hard clustering is to minimise the loss in Bregman information due to the quantisation induced by M:

$$\ell_\phi(M) =_{\text{def}} I_\phi(X) - I_\phi(M)$$

One can show **[banerjee]** that

$$\ell_\phi(M) = \mathbb{E}_\pi\left[I_\phi(X_h)\right] \approx \sum_{h=1}^{k} \sum_{x_i \in \mathfrak{X}_h} \upsilon_i B_\phi\left(x_i, \mu_h\right)$$

where k is the number of clusters, $\mathfrak{X}_h$ are the cells of the clustering, υ_i is the distribution of the x_i and μ_h is the right Bregman centroid of $\mathfrak{X}_h$.

Classical k-means is a special case of this for the squared-error divergence. The KL-divergence implies the mutual information as a variance and yields *information-theoretic clustering* **[todo]**. The Ikura-Saito divergence yields the LBG algorithm **[todo]**.

Particularly relevant for decision trees is the following insight: The choice of Bregman divergence implies a measure of variance (see 1.7.2 for examples). Optimising for this measure of variance implies the splitting criterion [2]. This gives a theoretical rationale for choosing splitting function and leaf combiner in decision trees.

2: The leaf combiner, which corresponds to the right Bregman centroid is independent on the chosen divergence, see **??**.

3.1.4 Stopping Criteria & Tree depth

...

3.2 The Random Forest scheme

The deeper a decision tree becomes, the closer the decision regions will fit the training data. This approximation is in fact guided *only* by the training data. In the extreme case, if the tree is fully grown, each partition cell will correspond to a single example and the outcome for that cell will be the outcome of that example. The tree essentially degenerates to a 1-nearest-neighbour scheme with respect to the training dataset. This means that trees constructed with different samples D of training datasets from the original distribution $P(X, Y)$ potentially predict quite different outcomes for testing datapoints. This is captured in the concept of model variance as defined in 1.5. In **??**, one can indeed observe that trees of greater depth have greater variance. At the same time, their fit to the training data improves, which is captured by lower bias.

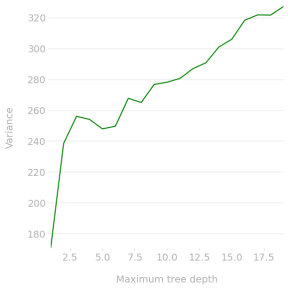

Figure 3.3: Variances of decision trees of increasing depths. Evaluated for squared-error regression on a synthetic dataset.

Mitigating this strong dependence on the training data is one of the main motivations of Random Forests. The basic idea is as follows: If we produce several uncorrelated decision trees and average their predictions, then the predictions should exhibit lower variance [3]. Exactly how we facilitate uncorrelated trees gives rise to the Random Forest scheme.

3: We use this intuition here to motivate the basic components of Random Forests. We will treat effect of variance reduction through combining multiple models thoroughly in **??**.

The basic idea is to introduce randomness into the decision tree construction. This is achieved by two mechanisms:

- **Bootstrapping**: Each tree is constructed not on the entire training dataset but a random subset of it. Usually, this *bootstrap sample* is produced by drawing the same amount of points with replacement.
- **Random feature selection**: When determining where to split a node, not all features are considered but only a random subset of a certain size.

Thus, a random forest additionally requires the following parameters:

- number of trees...

3.2.1 Bagging

A vital ingredient to Random Forests is the *Bagging* procedure, which stands for *bootstrapping and aggregating*. Bagging is an ensemble learning technique not specific to Random Forests. In Bagging, each member is constructed not on the full training dataset but a *bootstrap sample* of it. The bootstrap sample is usually determined by drawing n out of n examples uniformly with replacement [**breiman**, **others**]. We will refer to this as *uniform bootstrapping*. One can also determine the bootstrap sample by drawing n out of n points with replacement according to a probability distribution $\{p_1, ..., p_n\}$, which we call *weighted bootstrapping*. Bootstrapping means that each member will be trained on a different dataset.

In uniform bootstrapping,if we draw n samples from n available points, the probability of an example being selected in a single draw is $\frac{1}{n}$. Conversely, the probability of an example not being selected in a single draw is $1 - \frac{1}{n}$. We draw n times. Hence, the probability of an observation not being selected in any of the draws is $(1 - \frac{1}{n})^n$. The probability of an example indeed being selected in at least one of the draws then is $1 - (1 - \frac{1}{n})^n$. For large n, one can approximate $\lim_{n\to\infty} 1 - (1 - \frac{1}{n})^n = 1 - e^{-1} \approx 0.632$ [**todo**].

3.2.2 Feature & Split selection

3.2.3 Number of trees

3.2.4 Depth of trees

3.2.5 Random Forests converge

As the number of trees increases, for almost surely all sequences $(\Theta_1, \dots)$, the generalisation error $PE^\star$ converges:

$$\mathbb{E}_{(X,Y),D}\left[\ell(Y, q_D(X))\right] \to \mathbb{P}_{\mathbf{X},Y}\left[\mathrm{mr}(\mathbf{X}, Y) < 0\right]$$

3.2.6 Random Forests do not overfit

3.2.7 Random Forests are consistent

3.2.8 Practical advantages

parallelisable, oob estimation, feature importance, proximity, ...

4 Diversity

chapter introduction. We will now introduce a way to understand ensemble diversity.

4.1 Measures of ensemble diversity

Review. Maybe a table with all these measures? Would look nice. But probably not too relevant.

4.2 Ambiguity

As we have seen in 4.1, formally expressing the notion of ensemble diversity is not straightforward.

We take inspiration from 1.5 and derive a measure of ensemble diversity by considering the *effect* of diversity on the ensemble error. If we consider the members to be constructed according to a parameter Θ, a reasonable measure of the member performance is its loss in expectation over the parameter distribution: $\mathbb{E}_\Theta[L(Y, q(X;\Theta))] \approx \frac{1}{M}\sum_{i=1}^{M} L(Y, q(X;\Theta_i))$. What can we hope to gain from using an ensemble $\bar{q}$, which combines the output of individual members $q_1, \ldots, q_M$ over just using a single member model? The *ensemble improvement* for finite ensembles is

$$\frac{1}{M}\sum_{i=1}^{M} L(Y, q_i) - L(Y, \bar{q})$$

Conveniently, this is also the *effect of ensembling* on the error. Due to this, we will refer to this quantity as *ambiguity-effect*.

Theorem 4.2.1 *(Ambiguity-Effect decomposition [**todo**]) For any loss function L, target label Y, ensemble members $q_1, \ldots, q_M$ with combiner $\bar{q}$*

$$L(Y, \bar{q}) = \frac{1}{M}\sum_{i=1}^{M} L(Y, q_i) - \underbrace{\left(\frac{1}{M}\sum_{i=1}^{M} L(Y, q_i) - L(Y, \bar{q})\right)}_{\textit{Ambiguity-Effect / Ensemble Improvement}}$$

Similar to variance, ambiguity and ambiguity-effect are measures of spread. Variance measures the spread of training error across models trained with different draws of the training dataset D around a model that is a centroid with respect to the distribution of D. Similarly, ambiguity measures the spread of individual member model errors around a model that is centroid with respect to the distribution of Θ, namely the combiner $\bar{q}$. In case of squared loss, this is indeed the statistical variance. For other losses, this is a different quantity.

At this point, it is not yet clear whether the ensemble improvement is even nonnegative, i.e. that ensembling does not hurt performance. We will address this in **??**.

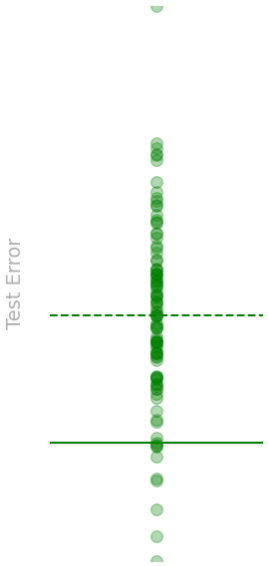

Figure 4.1: The spread of individual tree predictions in a random forest ensemble. Glyphs correspond to test errors of individual trees. The dashed line is the average test error of individual trees $\frac{1}{M}\sum_1^M L(y, q_i)$. The solid line is the test error of the ensemble $L(y, \bar{q})$. The difference between these values is the *ensemble improvement* or *ambiguity-effect*. (TODO resolve double terms)

4.3 The Diversity-Effect decomposition

Note that the first term in the ambiguity-effect decomposition 4.2.1 is the average loss of the ensemble members. One can now decompose the loss of an individual ensemble member into bias- and variance-effect just like in **??**.

Theorem 4.3.1 *(Bias-Variance-Diversity-Effect decomposition)*

$$\begin{aligned}\mathbb{E}_{D,Y}\left[L(Y,\bar{q})\right] = &\underbrace{\mathbb{E}_y\left[l(y,y^\star)\right]}_{noise} \\ &+ \underbrace{\frac{1}{M}\sum_{i=1}^{M}\mathbb{E}_Y\left[L_{0/1}\left(Y,q_i^*\right) - L_{0/1}\left(Y,Y^*\right)\right]}_{\overline{bias\text{-}effect}} \\ &+ \underbrace{\frac{1}{M}\sum_{i=1}^{M}\mathbb{E}_D\left[\mathbb{E}_Y\left[L_{0/1}\left(Y,q_i\right) - L\left(Y,q_i^*\right)\right]\right]}_{\overline{variance\text{-}effect}} \\ &- \underbrace{\mathbb{E}_D\left[\mathbb{E}_Y\left[\frac{1}{M}\sum_{i=1}^{M}\left[L_{0/1}\left(Y,q_i\right) - L(Y,\bar{q})\right]\right]\right]}_{diversity\text{-}effect}\end{aligned}$$

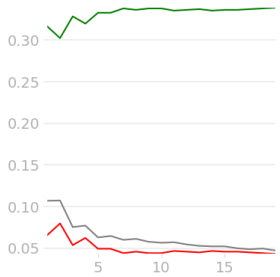

Figure 4.2: foo bar baz, avg bias, variance constant, diversity increases

4.4 Diversity for Bregman Divergences

Note that bias-effect, variance-effect and ambiguity-effect are all of the form $\mathbb{E}\left[L(Y,\circ) - L(Y,\square)\right]$ and depend directly on the target label Y. With variance-effect, we have captured the *effect* of variations between different training datasets) on the prediction error. With ambiguity-effect, we have captured the effect of variations between the different member models on the prediction error. We have already seen that for the squared error, the variance-effect coincides with familiar notion of statistical variance between the predictions. This is convenient, since we can then estimate variance (and its effect) independently of the target label Y. This means that variance can be compared across different datasets. The same applies to ambiguity [1]. Also, labelled data can be scarce in practise, while examples alone may be easier to obtain.

1: Bias will always be indirectly dependent on the target labels via the expected label.

We will now define a class of losses for which the effects reduce to the quantities themselves. This class covers many widely used loss functions and thus allows us to formulate a unified bias-variance-decomposition for all of these. However, for some other losses – in particular the 0-1-loss for classification – such a decomposition is proven to not exist (see [**todo**]). In these cases, we can still consider the more general effect decompositions (see **??**).

Bregman divergences have the one key property that their loss-effect terms collapse. That is, with bregman divergences, we have, for the the proper choice of Z, Z':

$$\mathrm{LE}(Z,Z') = \ell(Z,Z')$$

This is given by the following two lemmas.

Lemma 4.4.1 *([**pfau**], Theorem 0.1 (b)) Let the generator $\phi : \mathcal{S} \to \mathbb{R}$ be a strictly convex, differentiable function. Let Y be a random variable on $\mathcal{S}$. Then, for any $q \in \mathcal{S}$, it holds that*

$$B_\phi\left(y^\star, q\right) = \mathbb{E}\left[B_\phi\left(Y, q\right) - B_\phi\left(Y, y^\star\right)\right]$$

if $y^\star$ is the right Bregman centroid.

This shows that bias-effect collapses to bias for Bregman divergences:

$$B_\phi\left(y^\star, q^\star\right) = \mathbb{E}\left[B_\phi\left(Y, q^\star\right) - B_\phi\left(Y, y^\star\right)\right]$$

Lemma 4.4.2 *(Generalised from [**ref:wood23**]) Let $\mathbf{y}$ be a random vector. Let $\mathbf{q}_Z$ be a random vector dependent on another random variable Z. Then it holds that*

$$B_\phi\left(q^\star, q\right) = \mathbb{E}_Y\left[B_\phi\left(Y, q\right) - B_\phi\left(Y, q^\star\right)\right]$$

if $q^\star$ is the left Bregman centroid.

Proof.

$$\mathbb{E}_{Y,Z}\left[B_\phi\left(y, q\right) - B_\phi\left(y, q^\star\right)\right] = \ldots$$

□

This shows that variance- and ambiguity-effect collapse to variance and ambiguity. The variance (in the bias-variance sense) is the variance of the estimates q_D dependent on D with respect to different realisations of D around its D-centroid.

$$\text{For } q^\star = \mathcal{C}_D\left[q_D\right]\text{:} \quad B_\phi\left(q^\star, q\right) = \mathbb{E}\left[B_\phi\left(Y, q^\star\right) - B_\phi\left(Y, q\right)\right]$$

Ambiguity/Diversity is the variance of the estimates q_Θ dependent on Θ with respect to different realisations of Θ around its centroid.

$$\text{For } \bar{q} = \mathcal{C}_\Theta\left[q_\Theta\right]\text{:} \quad B_\phi\left(\bar{q}, q\right) = \mathbb{E}\left[B_\phi\left(Y, \bar{q}\right) - B_\phi\left(Y, q\right)\right]$$

This yields a generalised bias-variance-diversity decomposition for bregman divergences as a special case of the corresponding effect decomposition **??**.

Theorem 4.4.3 *(Bias-Variance-Diversity decomposition for Bregman divergences)*

$$\begin{aligned}\mathbb{E}_{(X,Y),D}\left[B_\phi\left(Y, q\right)\right] = &\underbrace{\mathbb{E}_{Y|X}\left[B_\phi\left(Y, y^\star\right)\right]}_{\textit{noise}} \\ &+ \underbrace{\frac{1}{M}\sum_{i=1}^{M} B_\phi\left(\overline{\mathbf{Y}}, \mathbf{q}_i^*\right)}_{\overline{\textit{bias}}} \\ &+ \underbrace{\frac{1}{M}\sum_{i=1}^{M}\mathbb{E}_D\left[B_\phi\left(\mathbf{q}_i^*, \mathbf{q}_i\right)\right]}_{\overline{\textit{variance}}} \\ &- \underbrace{\mathbb{E}_D\left[\frac{1}{M}\sum_{i=1}^{M} B_\phi\left(\overline{\mathbf{q}}, \mathbf{q}_i\right)\right]}_{\textit{diversity}}\end{aligned}$$

4.5 Diversity is a measure of model fit

"Sweet spot" of diversity. Review of previous experiments. Will need to go somewhere else.

4.6 Ensemble Improvement

The question that arises immediately when considering ensemble learning is whether combining the outputs of members could possibly *hurt* the generalisation error. If member models are random according to a random variable parameter Θ, we can quantify this by comparing the ensemble loss to the loss of an expected member. This is exactly the ambiguity-effect (**??**.

We have yet to see if and when this quantity is non-negative, i.e. combining individual models in an ensemble does not hurt performance. Further, we want to see how large this improvement actually is and whether on can construct better ensembles by adressing it explicitly.

Ambiguity-effect, sometimes also called *ensemble improvement* is defined as (see thm:ambiguity-effect-decomp)

$$\frac{1}{M}\sum_{i=1}^{M} L(Y, q_i) - L(Y, \bar{q})$$

for ensemble members $q_1, \ldots, q_M$, combiner $\bar{q}$ and loss function L.

4.6.1 Unchanged bias and reduction in variance

In section 2.3 we gave arguments for how in specific cases, the ensemble bias equals the bias of any member. We now have the tools to show this in a more general manner [**wood23**].

For the ensemble bias, application of the ambiguity-effect decomposition (see **??**) to a set of centroid models $q_i^\star$ yields:

$$\underbrace{\mathrm{LE}(y, \bar{q}^\star)}_{\text{ens. bias}} = \underbrace{\frac{1}{M}\sum_{i=1}^{M}\mathrm{LE}(y, q_i^\star)}_{\text{avg. bias}} - \underbrace{\frac{1}{M}\sum_{i=1}^{M}\mathrm{LE}(\bar{q}^\star, q_i^\star)}_{\Delta}$$

For the ensemble variance, application of the diversity-effect decomposition (see **??**) while substituting $y \leftarrow \bar{q}^\star$:

$$\underbrace{\mathbb{E}_D\left[\mathrm{LE}(\bar{q}^\star, \bar{q})\right]}_{\text{ens. var.}} = \underbrace{\frac{1}{M}\sum_{i=1}^{M}\mathrm{LE}(\bar{q}^\star, q_i^\star)}_{\Delta} + \underbrace{\frac{1}{M}\sum_{i=1}^{M}\mathbb{E}_D\left[\mathrm{LE}(q_i^\star, q_i)\right]}_{\text{avg. var.}} - \underbrace{\mathbb{E}_D\left[\frac{1}{M}\sum_{i=1}^{M}\mathrm{LE}(\bar{q}, q_i)\right]}_{\text{diversity}}$$

Due to Lemma (**??**) which states that in homogeneous ensembles $q_i^\star = q_j^\star = \bar{q}^\star$, we can conclude that $\Delta = 0$. This shows two things: First, the ensemble bias-effect is equal to the average member bias-effect:

$$\Delta = 0 \quad \rightarrow \quad \mathrm{LE}(y, \bar{q}^\star) = \frac{1}{M}\sum_{i=1}^{M}\mathrm{LE}(y, q_i^\star)$$

Second, the ensemble variance-effect is exactly the difference between the average variance-effect and the diversity. In other words, *variance reduction is exactly measured by diversity*.

4.6.2 Improvement for Bregman Divergences

In section 2.3, we have used Jensen's inequality to show that the ensemble improvement is non-negative for some cases.

$$\mathbb{E}_\Theta\left[\ell(q_\Theta(X), Y)\right] - \ell(\mathbb{E}_\Theta\left[q_\Theta(X)\right], Y) \geq 0$$

It is evident, that the Jensen gap is but a special case of ambiguity-effect (**??**) for $\bar{q} =_{\text{def}} \frac{1}{M}\sum_{i=1}^{M} q_i$ and convex loss functions. This shows that the ensemble loss is always smaller-equal than the expected member loss, but *only* if the ensemble output is actually produced by an arithmetic mean.

However, it can not be assumed from the outset that the arithmetic mean is the best ensemble combiner. Indeed, for the cross-entropy loss, **abe** proceed to note that the Jensen gap corresponds to a form that is "not immediately recognizable". Although they do find an interpretation of it, it is still necessarily dependent on the outcome Y . As illustrated in 4.4 on Bregman divergences, it seems reasonable to define the ensemble combiner in accordance to the Bregman divergence, i.e. to be the *dual* expectation $\mathscr{C}_\Theta\left[q_\Theta\right]$. Non-negativity is then easily shown since in that case ambiguity-effect reduces to ambiguity (see **??**)

In fact, for the case of cross-entropy, wood23 show that the ambiguity term is still nonnegative, i.e. that the arithmetic mean combiner does not hurt performance.

$$B_\phi\left(\bar{q}, q\right) = \mathbb{E}\left[B_\phi\left(Y, \bar{q}\right) - B_\phi\left(Y, q\right)\right] \quad \text{for } \bar{q} = \mathscr{C}_\Theta\left[q_\Theta\right]$$

and the value of any Bregman divergence is always non-negative. Further, ambiguity is now independent of the outcome.

This shows that for any Bregman divergence, ensembling using the combiner implied by the divergence can not hurt performance. Second, we obtain an intuitive measure of ensemble improvement. Third, this ensemble improvement appears in an exact decomposition of the ensemble generalisation error.

4.6.3 Improvement for the $0/1$-Loss

Bregman divergences are certainly useful for regression tasks. Further, divergences such as the KL-divergence are useful for estimating class *probabilities* and indeed,

classification tasks can be approached by selecting the class with the highest overall probability. In other settings, however, we are mainly concerned with whether the correct class is assigned or not. This gives rise to the 0/1-loss $\ell(y', y) =_{\text{def}} \mathbb{1}[y' \neq y]$. The expected loss of a model q is $\ell(q) =_{\text{def}} \mathbb{E}_D\left[\ell_{0/1}(q(X), Y)\right]$. Let $\bar{q}$ be the majority vote combiner. In the following, we will review results which characterise the ensemble improvement in this setting.

Given ensemble members q_Θ parameterised by a random variable Θ, denote the proportion of erroneous ("wrong") classifiers for an example-outcome pair (X, Y) as $W_\Theta =_{\text{def}} W_\Theta(X, Y) =_{\text{def}} \mathbb{E}_\Theta\left[\ell_{0/1}(q_\Theta(X), Y)\right]$. This expectation is approximated by the average member loss $\frac{1}{M}\sum_{i=1}^{M}\ell_{0/1}(q_i(X), Y)$.

Using Markov's inequality, we can readily upper-bound the error of the ensemble in terms of expected errors of the members **[theisen]** [2] .

$$0 \;\leq\; L(\bar{q}) \;\leq\; \mathbb{P}\left[W_\Theta \geq \frac{1}{2}\right] \;\leq\; 2\mathbb{E}\left[W_\Theta\right] = 2\mathbb{E}_\Theta\left[L(q_\Theta)\right]$$

While there indeed exist examples for which this upper bound is tight **??**, it is reasonable to suspect that the ensemble being worse by a factor of two is only a pathological case and not relevant for practise. The way forward is to impose assumptions on the performance of the member models.

2: Markov's inequality states that for a nonnegative random variable X and $a > 0$

$$\mathbb{P}[X \geq a] \leq \frac{\mathbb{E}[X]}{a}.$$

The final equality is due to that one can swap the order of expectations in $\mathbb{E}_D[W_\Theta] = \mathbb{E}_D[\mathbb{E}_\Theta[\mathbb{1}[q_\Theta(X) \neq Y]]] = \mathbb{E}_\Theta[L(q_\Theta)]$.

Definition 4.6.1 *A model q_Θ is a weak learner iff it performs better than as if randomly guessing, i.e. $\mathbb{E}_D[W_\Theta] \geq \frac{1}{2}$.*

Theorem 4.6.1 *([**wood23**]) In ensembles of weak learners, diversity-effect is non-negative:*

$$\mathbb{E}_D\left[\mathbb{E}_Y\left[\frac{1}{M}\sum_{i=1}^{M}\ell_{0/1}(Y, q_i) - \ell_{0/1}(Y, \bar{q})\right]\right] \;\geq\; 0$$

In particular, this means that the ensemble improvement is non-negative.

[theisen] further restrict the weak learner assumption to a condition called *competence*.

Definition 4.6.2 *(Competence, **[theisen]**) An ensemble is competent iff*

$$\forall t \in \left[0, \frac{1}{2}\right] : \mathbb{P}_{(X,Y)\sim D}\left[W_\Theta(X, Y) \in \left[t, \frac{1}{2}\right]\right] \;\geq\; \mathbb{P}_{(X,Y)\sim D}\left[W_\Theta(X, Y) \in \left[\frac{1}{2}, 1 - t\right]\right]$$

The weak learner assumption is a special case of this for $t = \frac{1}{2}$. Ensembles of weak learners are competent and as such, this provides a more general proof for **??**. The competence assumption, however, allows to exclude pathological cases and give tighter bounds for the ensemble improvement – here in terms of disagreements between members. [3]

3: Because we focus mainly on binary classification, we give this result in its simplified form for $k = 2$ classes but note that their actual result gives bounds for arbitrary k.

Theorem 4.6.2 *([**theisen**], for binary classification) Let $D(q_\Theta, q_{\Theta'}) =_{def} \mathbb{E}_D\left[\ell_{0/1}(q_\Theta, q_{\Theta'})\right]$ be the disagreement rate between two members. In competent ensembles it holds that*

$$\mathbb{E}_{\Theta,\Theta'}\left[D(q_\Theta, q_{\Theta'})\right] \;\geq\; \mathbb{E}_\Theta\left[L(q_\Theta) - L(\bar{q})\right] \;\geq\; \mathbb{E}_{\Theta,\Theta'}\left[D(q_\Theta, q_{\Theta'})\right] - \mathbb{E}_\Theta\left[L(q_\Theta)\right]$$

5 Growth Strategies

Some general intuition on the idea behind encouraging diversity (always tradeoff with avg member error)

5.1 Review

Maybe here also review on existing methods, or otherwise relate and motivate where needed.

5.2 Binary classification and Dynamic Random Forests

Diversity-effect not necessarily always being nonnegative is actually useful here in that it allows us an "easy" insight in how these point influence the ensemble (idea of "dragging" combiner towards/away from correct pred)

The setting of binary classification under 0/1-loss allows us to clearly distinguish two cases. An example may be correctly classified, in which case it does not contribute to the overall error. Otherwise, it is classified incorrectly and contributes exactly 1. [1]

1: Note that while overall diversity-effect (in expectation over X) is shown to be nonnegative for weak learners (ref wood23 thm 14 above), there may still be different points that contribute either positively or negatively.

Lemma 5.2.1 *For $y, \bar{q} \in \{-1, 1\}$ it holds that*

$$\frac{1}{M}\sum_{i=1}^{M}\left[\ell_{0/1}(y, q_i) - \ell_{0/1}(y, \bar{q})\right] = (y \cdot \bar{q})\frac{1}{M}\sum_{i=1}^{M}\ell_{0/1}(\bar{q}, q_i) \tag{5.1}$$

Proof. Let $y, \bar{q} \in \{-1, 1\}$.

- Assume the ensemble is correct, i.e. $y = \bar{q}$. Then $\ell_{0/1}(y, \bar{q}) = 0$ and the left-hand-side equals $\frac{1}{M}\sum_{i=1}^{M}\ell_{0/1}(y, q_i) = \frac{1}{M}\sum_{i=1}^{M}\ell_{0/1}(\bar{q}, q_i)$. Further, $y \cdot \bar{q} = 1$.
- Assume the ensemble is incorrect, i.e. $y \neq \bar{q}$. Then $y \cdot \bar{q} = -1$ and, for the left-hand-side, we can write

$$\frac{1}{M}\sum_{i=1}^{M}\left[\ell_{0/1}(y, q_i)\right] - 1 = -\left(1 - \frac{1}{M}\sum_{i=1}^{M}\ell_{0/1}(y, q_i)\right) = -\left(\frac{1}{M}\sum_{i=1}^{M}\ell_{0/1}(\bar{q}, q_i)\right)$$

□

This shows that, for binary classification under 0/1-loss, diversity-effect can be decomposed exactly between points that contribute positively or negatively to the overall loss. Starting from the ambiguity-effect decomposition **??**:

$$\begin{aligned}\mathbb{E}_X\left[L(Y, \bar{q})\right] &= \mathbb{E}_X\left[\frac{1}{M}\sum_{i=1}^{M}\ell_{0/1}(Y, q_i)\right] - \mathbb{E}_X\left[\frac{1}{M}\sum_{i=1}^{M}L(Y, q_i) - L(Y, \bar{q})\right] \\ &= \mathbb{E}_X\left[\frac{1}{M}\sum_{i=1}^{M}\ell_{0/1}(Y, q_i)\right] - \mathbb{E}_X\left[(y \cdot \bar{q})\frac{1}{M}\sum_{i=1}^{M}\ell_{0/1}(\bar{q}, q_i)\right]\end{aligned}$$

An intuition of this is also that of "wasted votes": Under the majority vote combiner, for the ensemble to be correct, we require only at least half of the members to be correct. Any higher ratio of correct ensemble members does not improve the ensemble performance on this point and these can be seen as "wasted". Likewise, the ensemble is incorrect if not more than half of the members are correct. Any positive votes do not influence the ensemble improvement and can be considered "wasted".

We can divide the expectation over X into cases. X_+ where the ensemble is correct. Ambiguity on these points has a decreasing effect on the overall ensemble error. X_- corresponds to the points for which the ensemble is incorrect. Ambiguity on these points has an increasing effect on the overall ensemble error. This yields a decomposition into "good" and "bad" diversity.

Theorem 5.2.2 *([**kuncheva**])*

$$\mathbb{E}_X\left[L(Y,\bar{q})\right] = \mathbb{E}_X\left[\frac{1}{M}\sum_{i=1}^{M}\ell_{0/1}\left(Y,q_i\right)\right] - \underbrace{\mathbb{E}_{X_+}\left[\frac{1}{M}\sum_{i=1}^{M}\ell_{0/1}\left(\bar{q},q_i\right)\right]}_{\text{"good" diversity}} + \underbrace{\mathbb{E}_{X_-}\left[\frac{1}{M}\sum_{i=1}^{M}\ell_{0/1}\left(\bar{q},q_i\right)\right]}_{\text{"bad" diversity}}$$

This leads to a key insight. It is that diversity/ambiguity is only beneficial *on points at which the model can actually afford to be diverse*. It has been noted that, for diversity to be effective, it has to mitigate the average member error **[todo]** (as can be seen from the ambiguity decomposition **??**). However, to the best of our knowledge, it has not yet been exploited that diversity is measured *per point*.

Further, we can express both good and bad diversity in terms of the average member error. Let $\neg y$ be .... [2]

2: This also gives a nice illustration of the Jury theorem (**??**). If ensemble members are weak learners, i.e. they predict the correct class with probability greater than $\frac{1}{2}$, adding an additional learner is more likely to increase the good diversity term than the bad diversity term.

$$\begin{aligned}
\mathbb{E}_{X_+}\left[\frac{1}{M}\sum_{i=1}^{M}\ell_{0/1}\left(q_i,\bar{q}\right)\right] &= \mathbb{E}_{X_+}\left[\frac{1}{M}\sum_{i=1}^{M}\ell_{0/1}\left(q_i,y\right)\right] \\
\mathbb{E}_{X_-}\left[\frac{1}{M}\sum_{i=1}^{M}\ell_{0/1}\left(q_i,\bar{q}\right)\right] &= \mathbb{E}_{X_-}\left[\frac{1}{M}\sum_{i=1}^{M}\ell_{0/1}\left(q_i,\neg y\right)\right] \\
&= \mathbb{E}_{X_-}\left[1-\frac{1}{M}\sum_{i=1}^{M}\ell_{0/1}\left(q_i,y\right)\right]
\end{aligned}$$

5.2.1 Guiding ensemble construction with example weights

(introduction ensemble construction, tree-by-tree with adaptive weights)

Now that we know exactly how correct and incorrect classifications affect the ensemble error, we will work towards leveraging this insight to find an ensemble construction scheme ...

On points for which the ensemble is correct, diversity is beneficial and corresponds directly to the average member error. This means that we might expect to see an increase in ensemble performance if disagreement on correct points is encouraged – as long as it does not cross the majority vote threshold. Indeed, in the perfect case, all examples would be correctly classified with a member error just below the majority vote threshold. This would result in an ensemble with large average member error but also with high diversity which mitigates the member error.

On points for whch the ensemble is incorrect, diversity hurts the ensemble performance. One could argue that, to minimise bad diversity, the average member error should be large, i.e. all members (instead of only some) should be driven to *mis*-classify the example. However, we conjecture that this would cause the ensemble construction procedure to "give up" on misclassified examples. We will consider a different strategy first. Instead of giving up, the ensemble construction scheme should put more emphasis on these points, in the hope of eventually pushing it over the majority vote threshold

towards a correct classification. This means that we are effectively *increasing* bad diversity, in the hope that it eventually turns into good diversity.

In standard Random Forests, each next tree is constructed independently of the ensemble constructed so far. Instead, we may try to construct the next tree in a coordinated manner such that it more optimally complements the ensemble constructed so far. Both good and bad diversity can be expressed in terms of the average member error. Each member that is added to the ensemble contributes to it. Consequently, we might be able to steer the development of ensemble performance by encouraging the next member to either correctly or inclassify a point, given how the ensemble constructed so far performs on this point.

We now proceed to define some weighting functions informed by diversity (see **??**).

Definition 5.2.1 *(DRF weighting scheme [**bernard-drf**]) Let $\bar{q}$ be the ensemble constructed so far. For a pair $(X, Y) \in D$, define the Dynamic Random Forest weighting scheme as*

$$w_{DRF}(X) =_{def} \frac{1}{M} \sum_{i=1}^{M} \ell_{0/1}(Y, q_i(X))$$

This will have the effect that correctly classified examples are assigned lower weight and incorrectly classified examples are assigned higher weight.

This weighting scheme was first proposed in [**bernard-drf**]. However, they only give a heuristic, intuitive justification in that if a high number of trees misclassifies an example, the next tree should put more emphasis on it, similar to boosting strategies (**??**). We derive and motivate this weighting scheme from a perspective of diversity and give insight into how exactly it works – namely, that it does *not* flat-out increase the performance (as in boosting schemes) but instead encourages diversity.

XuChen re-iterate on the DRF weighting scheme and propose an alternative scheme.

$$w_{\text{XuChen}}(X) =_{\text{def}} \begin{cases} \varepsilon^2 & \varepsilon \leq \frac{1}{2} \\ \sqrt{\varepsilon} & \varepsilon > \frac{1}{2} \end{cases} \qquad \text{for } \varepsilon =_{\text{def}} \frac{1}{M} \sum_{i=1}^{M} \ell_{0/1}(Y, q_i(X))$$

Again, the authors provide only a heuristic motivation, which is that, compared to w_{DRF}, their method has a more drastic effect of up- and downweighting. We can now give a more informed interpretation. Inspecting w_{DRF}, which is continuous around the majority vote threshold, one can see that very similar weights are assigned to examples which are classified just barely correctly (resulting in a 0/1-loss of 0) and examples which are classified just barely incorrectly (resulting in a 0/1-loss of 1). This may mean suboptimal guidance in ensemble construction since both cases have very similar weights, but their effect on the ensemble loss is actually dramatically different. One disadvantage is that we take a heuristic step away from theory.

Now we just have to figure out how to actually influence the training of the next member such that it's performance on some points is (likely) increased or decreased. One way that is suited well for Random Forests is to assign *weights* $w : X \to [0, 1]$ to examples and consider them during member training. In section **??**, we will argue in detail how example weights influence the ensemble.

For random forests, these weights can possibly come into effect via two mechanisms:

- *Weighted bootstrapping*: Instead of drawing the bootstrap samples uniformly, draw a sample with probability according to its weight. If the bootstrap sample is large, examples with higher weight are more likely to be oversampled and thus appear multiple times in the bootstrap sample.
- *Weighted tree construction*: We have seen in **??** that tree construction according to some impurity measure greedily optimises a loss function. Likewise, weighting examples during computation of the impurity measure optimises a weighted loss.

5.2.2 Experimental Evaluation of the DRF weighting schemes

We compare different variants of the weighting scheme introduced in section 5.2.1. We look at the overall ensemble generalisation error, as well as the components of the error as given by the bias-variance decomposition **??** and the diversity decomposition **??**. Further, we analyse the ensemble margins (definition 1.6.2).

Experiment Setup

Compared learners In summary, we compare the following learning algorithms. For a learner, any configuration is similar to the one mentioned before unless otherwise specified. Since for the weighted variants the construction of the next tree depends on the performance of the ensemble so far, trees are constructed in sequence.

standard-rf-classifier: Standard Random Forest implementation based on *sklearn*. The tree hyperparameters are such that trees are grown until each leaf contains one data point. The number of randomly sampled candidate dimensions to search for the best split is set to $\sqrt{d}$ where d is the total number of dimensions. The impurity measure is the Gini impurity as defined in section 3.1.2. Each tree is grown on a bootstrap sample determined by sampling n out of n data points uniformly, with replacement.

drf-weighted-bootstrap-classifier: Each tree is grown on a bootstrap sample determined by sampling n out of n points according to the DRF weighting scheme (see 5.2.1). To yield a valid probability distribution, the weights are normalised via $w'(x_i) \leftarrow \frac{w(x_i)}{\sum_{j=1}^{n} w(x_j)}$.

drf-weighted-fit-classifier: Each tree is grown on a uniform bootstrap sample. For tree construction, namely measuring impurity, each example is weighted according to w_{DRF} (again, normalised).

drf-weighted-fit-oob-classifier: The example weights for a point x_i are determined based only on *out-of-bag* trees for x_i. These are those trees whose bootstrap sample has not included x_i.

Compared datasets We give a brief motivation for the classification datasets we have selected for evaluation. A detailed summary of each dataset can be found in **??**. *cover* is a dataset with a relatively high number of examples and low feature dimensionality. *mnist-subset* is a dataset with a moderate number of examples and high dimensionality. *diabetes* is a dataset with relatively high error rates. *bioresponse* is a small dataset with a very high number of features ($d \approx \frac{1}{2}n$). *qsar-biodeg* is a small dataset used for quick testing. Further, [**bernard**, **xuChen**] evaluated on *mnist* (although not just a subset of it), *spambase-openml* , *digits* and *diabetes* .

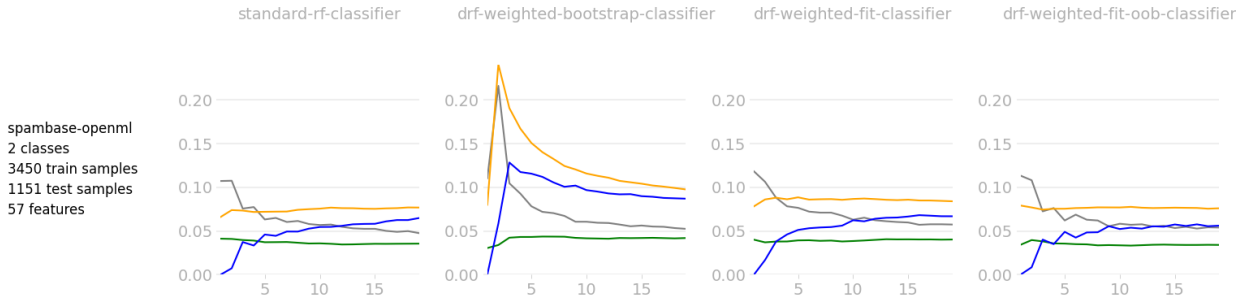

Figure 5.1: Comparison of components of the ensemble generalisation error on the *spambase-openml* dataset. Visualised are ensemble generalisation error, average member bias, average member variance and diversity.

Approximating statistical quantities For a dataset with n examples, $n_{\text{train}} =_{\text{def}} \frac{3}{4}n$ examples were assigned to be part of the *training split*, the other n_{test} for the *testing split*. Examples in the testing split are used for evaluation only and were never used in training a model. If X is a random variable taking values in the space of examples $\mathcal{X}$, expectations over X are approximated as the arithmetic mean over given examples in the testing split, i.e. for a function g:

$$\mathbb{E}_X [g(X)] \approx \sum_{i=1}^{n_{\text{test}}} g(x_i)$$

If D is a random variable corresponding to the input to a learner, for instance the training dataset or randomness in the learning algorithm, an expectation over D is approximated by an arithmetic mean over results of a fixed number of trials. In our case, we performed 3 trials.

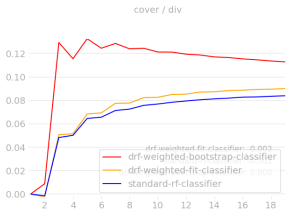

Figure 5.2: Diversity-effect by number of trees for different learners on the *cover* dataset. Weighted bootstrapping amounts to a sharp increase in diversity for the first couple of trees. Weighted tree construction only causes a slight increase in diversity as compared to a standard Random Forest.

Results

We describe and analyse the results here and provide several plots. The full results plotted for comparison across learners and datasets are given in section B.2.

Generalisation error and diversity We can see that weighted bootstrapping and weighted tree construction (see **??**) behave quite differently. The case for the *spambase-openml* dataset is given in figure **??**.

It is striking that, for every dataset, weighted bootstrapping initially brings a sharp increase in average bias and, consequently, generalisation error. The average variance stays mostly constant after an initial slight increase. As the number of trees grows, the generalisation error and the average bias diminish. A sharp initial increase in diversity mitigates the increase in average bias. The average bias then continously decreases to a similar or slightly higher level as the other learners. The average member variance seems to be very similar to that of other learners.

Weighted bootstrapping seems to be able to achieve similar or, often, better generalisation error than any other learner. It also consistently produces higher diversity ensembles than the other learners. It is interesting to note that on *spambase-openml* , where the initial increase in average bias and diversity appear most pronounced, diversity actually *decreases* as more trees are added to the ensemble.

For weighted tree construction, average bias and average variance are mostly constant and the decrease in generalisation error with a growing number of trees is solely due to increasing diversity. This is the same behaviour we also observe and motivate

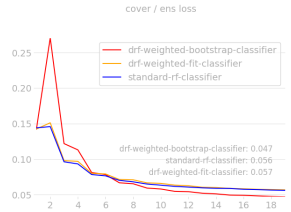

Figure 5.3: Expected generalisation error for different learners on the *cover* dataset. Weighted tree construction and standard Random Forests behave almost identically. For weighted bootstrapping, an initial increase in error is followed by a consistently lower error rate.

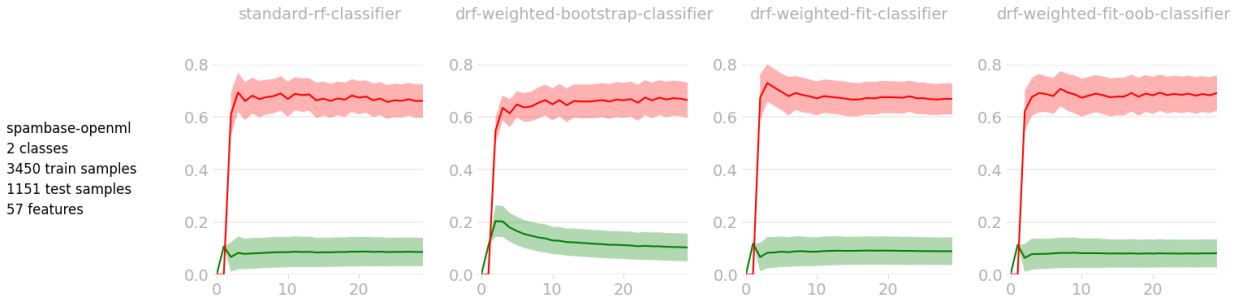

Figure 5.5: Ratio of incorrect trees per number of trees in the ensemble, plotted separately for correctly (green) and incorrectly (red) classified examples.

theoretically for standard Random Forests. Weighted tree construction performans as good as or slightly worse than standard Random Forests in terms of generalisation error. This is somewhat surprising since the intuitive motivation was that weighted tree construction will influence the splitting criteria.

Weighted tree contruction with out-of-bag weights does not appear to bring any advantage.

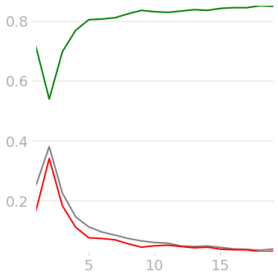

Figure 5.4: Ensemble generalisation error, ensemble bias and ensemble variance of weighted bootstrapping on the *digits* dataset.

Ensemble bias and variance The initial increase in average bias is also reflected in an initial increase in ensemble bias. At the same time, ensemble variance is decreased. Note how the average variance stays almost constant, while the ensemble variance varies greatly. This is the "outside view" on how diversity is a component of ensemble variance.

As the number of trees grows, for some datasets, ensemble variance is higher. For others, it is similar to standard Random Forests.

Ensemble margins The weighting schemes are essentially thought of to influence the ratio of incorrect members and thus bias, variance and diversity. Under the weighting schemes defined in **??**, on points for which the ensemble prediction is correct, the ratio of incorrect trees should *increase* (more diversity) and on points for which the ensemble prediction is incorrect, the ratio of incorrect trees should *decrease*. We can directly observe the ratio of incorrect members. We plot the average ratio of incorrect trees $\frac{1}{M}\sum_{i=1}^{M}\ell_{0/1}\left(\left(,y\right),q_i\right)$ separately for correctly and incorrectly classified examples. For weighted bootstrapping, across all datasets, we can indeed observe the expected effect. This is another view on the sharply increasing diversity as seen in e.g. figure 5.1. Again, the effect diminishes as the number of trees grows.

Further, we can look at the distribution of ensemble margin per example. We plot a histogram of examples with respect to the number of trees incorrectly classifying that example. In binary classification, more than $\frac{1}{2}M$ trees being incorrect leads to an incorrect ensemble prediction. For instance for the *cover* dataset, we can observe that the distribution of ensemble margins indeed seems to be skewed slightly to the right (i.e. in direction of more disagreement) for positive examples (see figure 5.6)

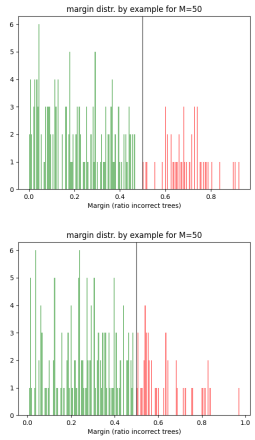

Figure 5.6: Histograms of ensemble margins per example for the standard Random Forest (top) and weighted bootstrapping (bottom) learners grown each of $M = 50$ trees on the *diabetes* dataset. For weighted bootstrapping, the distribution of the margins appears slightly skewed to the center, reflecting that the weighting schemes encourages disagreement on points where many ensemble members are correct and agreement on points where many ensemble members are incorrect.

Comparing weighted bootstrapping and weighted tree construction It is worth discussing why weighted bootstrapping and weighted tree construction produce such different behaviour. In practise, any bootstrap sample is finite. The bootstrap sample is drawn with replacement, thus a bootstrap sample does not necessarily include all examples from the training dataset. Under uniform bootstrapping, each example has

equal chance to be included in the bootstrap sample (see 3.2.1). It is then considered during tree construction according to its weight. On the other hand, under weighted bootstrapping, examples with high weight are more likely and examples with low weight are less likely to appear in the bootstrap sample. In particular, this means that examples with low weight are more likely to not be included at all in the bootstrap sample and consequently not be considered at all during tree construction. This might be an intuitive explanation why weighted bootstrapping shows a stronger effect than weighted tree construction in our experiments. A thorough discussion has to be left for future work.

Comparison to [bernard-drf] **bernard-drf** evaluated only the following approach: They would perform both weighted bootstrapping and weighted tree construction. Further, weights would be determined only on out-of-bag-trees (see section 5.2.2). Additionally, unrelated to the question at hand, they also employed a different way to determine candidate split features. Unlike in standard random forests, where a fixed number of candidate split features is sampled from all available features, the number of sampled candidate features was left fully random here. It was left unanswered which of these compenents actually affect the ensemble to what extent. Further, they did not provide any explanation or empirical analysis in terms of diversity. Looking at our results, the following points seem likely:

- An improvement in generalisation error is still obtained with standard candidate split feature sampling.
- The improvement can be explained using the notions of diversity.
- Weighted tree construction alone appears to have only very little effect as compared to a standard Random Forest.
- Weighted bootstrapping seems to provide the main effect.
- Determination of weights using out-of-bag-trees only does not improve performance for weighted tree construction.

Ambiguity-effect decomp plots ...

5.2.3 "Boosting" rationale for example weighting in classifier ensembles

re. spike: not super surprising, if there are only 1, 2, 3 trees, weights as ratio of incorrect trees are super extreme – motivation to dampen

would be good to put here to explain "spike". Can such a spike be observed in boosting methods?

initial increase in bias also observed for classical boosting models? (wood23 p26)

5.2.4 Outlook: Generalising to other losses

We can reach a similar insight for the general case of any loss function. For Bregman divergences, the main difference is that the ensemble improvement / ambiguity-effect is always non-negative. By the ambiguity decomposition **??** we can see that the ensemble error is always expressed by the gap between average member error and average ambiguity.

$$B_\phi(y, \bar{q}) = \frac{1}{M}\sum_{i=1}^{M} B_\phi(y, q_i) - B_\phi(\bar{q}, q_i) \geq 0$$

As such, there are no points with "bad diversity", which contribute negatively to the ambiguity(-effect) term, as is the case for the 0/1-loss (**??**).

However, we may still consider the above equation and how choosing the next member q_{M+1} affects it. Suppose the average member loss $\frac{1}{M}\sum_{i=1}^{M} B_\phi(y, q_i)$ is large. Then this is either mitigated by the diversity $\frac{1}{M}\sum_{i=1}^{M} B_\phi(\bar{q}, q_i)$, in which case the ensemble error is still low; or it is not, in which case the ensemble error too is large. However, if the average member loss $\frac{1}{M}\sum_{i=1}^{M} B_\phi(y, q_i)$ itself is small, there cannot be much diversity since, it is bounded by the average member error. Because of this, we need to consider the difference relative to the average member loss (which, interestingly, is the *ensemble improvement rate* of [**theisen**]).

$$\frac{\frac{1}{M}\sum_{i=1}^{M} B_\phi(y, q_i) - B_\phi(\bar{q}, q_i)}{\frac{1}{M}\sum_{i=1}^{M} B_\phi(y, q_i)}$$

If this quantity is low (min 0), then there is high member error but also high diversity. If it is high (max. 1), then the diversity relative to the average member error is low.

5.2.5 Outlook: Alternatives to example weights

Leaf refinement Another common approach to influence decision tree outcomes is re-adjusting the leaf outcomes once the tree is constructed [**others**]. This can be done e.g. via gradient descent [**buschj-negative-correlation-forests**]. While this is a valid approach, ours has the benefit that *(a)* it does not require an extra stage of optimisation, *(b)* it has the ability to (actually, it works solely through) influencing the tree structure. It would certainly be thinkable, however, to adapt an approach like [**buschj-ncl-forests**] to the ideas presented here.

Splitting functions Another possible approach would be to re-define the splitting criterion to not only optimise local purity but also (the right kind of) diversity (impurity) across the entire ensemble. (computationall much cheaper?)

5.3 Generalized Negative Correlation Learning

Consider the bias-variance-covariance decomposition for the squared error loss **??**:

$$\begin{aligned}\mathbb{E}_D\left[(y-\bar{q}(x))^2\right] = (\mathbb{E}_D[\bar{q}(x)] - y)^2 \\ + \mathbb{E}_D\left[\frac{1}{M^2}\sum_{i=1}^{M}(q_i(x) - \mathbb{E}_D[q_i(x)])^2\right] \\ + \mathbb{E}_D\left[\frac{1}{M^2}\sum_{i\neq j}^{M}(q_i(x) - \mathbb{E}_D[q_i(x)])(q_j(x) - \mathbb{E}_D\left[q_j(x)\right])\right]\end{aligned}$$

The covariance term contributes positively (resp. negatively) to the ensemble error if the outputs of the members are positively (resp. negatively) correlated.

Training neural network models involves optimising the weights of the neural network with respect to a given loss function. A common practise is to add *regularisation terms* to this loss function in order to bias the model towards e.g. sparse predictions [**todo**].

[**LiuYao**] suggest training an ensemble of neural networks simultaneously. The i-th member network is trained using a loss function ℓ_i that contains a regularisation

term p_i, which is intended to influence the training of the i-th member such that its predictions are negatively correlated with the other members. This is known as *Negative Correlation Learning* (NCL). The hyperparameter $\lambda \in \mathbb{R}$ determines how much weight is put on either of the two components.

$$e_i =_{\text{def}} \sum_{k=1}^{n} (q_i(x_k) - y_k)^2 \quad + \quad \lambda \, p_i$$

where the first term is the individual error of the i-th member and the regularisation term is

$$p_i =_{\text{def}} \sum_{k=1}^{n} \left((q_i(x_k) - \bar{q}(x_k))^2 \sum_{i \neq j}^{M} (q_i(x_k) - \bar{q}(x_k)) \right)$$

If p_i is small, then the i-th member is negatively correlated with the rest of the ensemble. Note that the penalty term corresponds to contribution of the i-th member to the ambiguity (**??**).

$$p_i = -\left(\sum_{k}^{n} q_i(x) - \bar{q}(x) \right)^2$$

Indeed, as [**brown2005**] noted, this approach can also be motivated via the ambiguity decomposition. The error of the full ensemble is [3]

$$\frac{1}{2}(\bar{q}(x) - y)^2 = \frac{1}{M} \sum_{i=1}^{M} \frac{1}{2}(q_i(x) - y)^2 \quad - \quad \frac{1}{M} \sum_{i=1}^{M} \frac{1}{2}(q_i - \bar{q})^2$$

3: We additionally use a factor of $\frac{1}{2}$

Due to this, they propose a diversity-encouraging loss-function for the i-th member of the form

Definition 5.3.1 *(NCL neural network objective for squared error [**brown2005**])*

$$\ell_i(x, y) =_{def} \frac{1}{2} \frac{1}{M} \sum_{i=1}^{M} \frac{1}{2}(q_i - y)^2 \quad - \lambda \frac{1}{M} \sum_{i=1}^{M} \frac{1}{2}(q_i - \bar{q})^2$$

Then the penalty coeffient λ smoothly distinguishes between training q_i to either maximise its individual performance or the ensemble performance, since

$$\frac{\partial \ell_i}{\partial q_i} = \frac{1}{M} \left((q_i - y) - \lambda (q_i - \bar{q}) \right)$$

and

$$\lambda = 0 \rightarrow \frac{\partial \ell_i}{\partial q_i} = \frac{1}{M}(q_i - y) = \frac{1}{M} \frac{\partial \ell_i}{\partial q_i}$$
$$\lambda = 1 \rightarrow \frac{\partial \ell_i}{\partial q_i} = \frac{1}{M}(\bar{q} - y) = \frac{\partial \bar{q}}{\partial q_i}$$

Several authors have attempted to transfer or generalise Negative Correlation Learning to other loss functions. For instance, [**webb21**] directly proved an ambiguity decomposition for the KL-divergence K and proposed the objective

$$\ell_i(x, y) =_{\text{def}} \frac{1}{M} \sum_{i=1}^{M} K(y \mid\mid q_i) \quad - \quad \lambda \, \frac{1}{M} \sum_{i=1}^{M} K(\bar{q} \mid\mid q_i)$$

where $\bar{q}$ is the geometric mean combiner (see **??**). The proof is not trivial and does not give insight into whether such a decomposition might also exist for other loss functions.

[**buschjaeger**] approached the problem by attempting to derive a generalised decomposition of the ensemble error. The basic idea is to consider a Taylor approximation around a notion of expected model $q^\star =_{\text{def}} \mathbb{E}_\Theta [q_\Theta]$.

$$\begin{aligned}\mathbb{E}[\ell(y,q)] = \mathbb{E}\left[\ell(q^\star)\right] + \mathbb{E}\left[(q-q^\star)^\top \nabla_{q^\star}(\ell(q^\star))\right] \\ + \mathbb{E}_\Theta\left[\frac{1}{2}(q-q^\star)^\top \nabla^2_{q^\star}(\ell(q^\star))(q-q^\star)\right] + \mathbb{E}[R_3]\end{aligned}$$

where R_3 is the remainder of the approximation, which vanishes if the third derivative of ℓ is zero. By definition of $q^\star$, $\mathbb{E}_\Theta\left[\nabla_{q^\star}(\ell(q^\star))\right] = \nabla_{q^\star}(\ell(q^\star))$ and $\mathbb{E}_\Theta[q-q^\star] = 0$ and thus the second term vanishes. The expectations can be approximated as follows:

$$q^\star = \mathbb{E}_\Theta[q_\Theta] \approx f =_{\text{def}} \frac{1}{M}\sum_{i=1}^M q_i$$

$$\mathbb{E}_\Theta\left[\frac{1}{2}(q-q^\star)^\top \nabla^2_{q^\star}(\ell(q^\star))(q-q^\star)\right] \approx \frac{1}{2}\frac{1}{M}\sum_{i=1}^M d_i^\top D d_i$$
$$\text{for } D =_{\text{def}} \nabla^2_{f(x)}(\ell(f(x)),y) \text{ and } d_i =_{\text{def}} (q_i(x) - f(x))$$

$$\mathbb{E}_\Theta[R_3(x)] \approx \tilde{R}$$

Note that here they *assume* the expected model $q^\star$ to also be the ensemble combiner, i.e. $q^\star \approx \frac{1}{M}\sum_{i=1}^M q_i$. Based on this, they propose the following training objective.

Definition 5.3.2 *Generalised NCL objective as proposed by [**buschj**]*

$$\ell(f) =_{def} \frac{1}{M}\sum_{i=1}^M \ell(q_i) - \frac{1}{2}\frac{1}{M}\sum_{i=1}^M d_i^\top D d_i$$

where $D =_{def} \nabla^2_{f(x)}(\ell(f(x)),y)$ *and* $d_i =_{def} (q_i(x) - f(x))$

This is based on the assumption that the remainder to the Taylor approximation R_3 is negligibly small. Let us consider some examples of commonly used loss functions.

- For the squared error, the third derivative vanishes and thus the decomposition is exact.
- For the negatively log-likelihood $\ell(z,y) =_{\text{def}} -\sum_i^k y_i \log(z_i)$, the third derivate is not bounded and thus this decomposition can not be used.
- For the cross-entropy loss $\ell(z,y) =_{\text{def}} -\sum_i^k y_i \log \frac{e^{z_i}}{\sum_j e^{z_j}}$, the third derivative is bounded and while the decomposition is not exact, it can be used for approximation.
- For any other loss function, this would have to be checked.

Thus, this approach makes several assumptions and is not applicable to some widely used loss functions.

It is evident however, that we already have a fully general ambiguity decomposition at hand, namely the ambiguity-effect decomposition (**??**). This decomposition is exact and holds for *any* loss function (including the 0/1-loss). For Bregman divergences, this reduces to the ambiguity decomposition **??**. We claim that the adequate generalisation of the NCL objective follows this structure.

Definition 5.3.3 *We propose the following generalisation of the NCL neural network objective. For general loss functions L:*

$$\ell_i =_{def} \frac{1}{M}\sum_{i=1}^{M} L(y, q_i) \quad - \lambda \left(\frac{1}{M}\sum_{i=1}^{M} L(y, q_i) - L(y, \bar{q}) \right)$$

And for Bregman divergences:

$$\ell_i =_{def} \frac{1}{M}\sum_{i=1}^{M} B_\phi\,(y, q_i) \quad - \lambda \left(\frac{1}{M}\sum_{i=1}^{M} B_\phi\,(\bar{q}, q_i) \right)$$

NCL for the squared error loss as introduced by [**LiuYao**] and [**brown2005**], as well as for the KL-divergence as given by [**webb**] can directly seen to be special cases of this.

This gives a general framework for Negative Correlation Learning with arbitrary loss functions. Because it is founded on the exact and intuitive bias-variance-diversity decomposition, this also yields a natural and intuitive means for understanding and analysing Negative Correlation Learning and its effects.

Note that the Dynamic Random Forest approach **??** is very similar in spirit. The construction procedure of individual decision trees greedily optimises its own performance (**??**), which corresponds to the first term in the NCL neural network objective. The introduction of example weights based on the ensemble diversity influences the decision tree construction to improve the ensemble loss, which corresponds to the second term.

Although the term *Negative Correlation Learning* in the literature refers specifically to neural networks, we can now see that it is rather a style of training ensemble members with respect to the ambiguity decomposition. To the best of our knowledge, only one other algorithm has been published that realises this: [**negative-correlation-forests**] proposes to refine the leaf predictions in a given Random Forest using using gradient descent according to the objective defined in 5.3.2.

5.3.1 Experiments

As a proof of concept, we perform Negative Correlation Learning with small neural networks based on the cross-entropy loss (refthm:cross-entropy-decomp).

The case of KL-divergence was already investigated empirically in detail in [**Webb**].

Experiment setup, NN architecture? (also see margintable command/environment)

5.4 Outlook: Dynamic Negative Correlation Learning

Wait, the canonical definition *is* already point-wise!

because why not. With insight from good/bad diversity could actually argue that tradeoff lambda should be dynamic per example?

This really gets close to boosting...

$\mathbb{K}$

$\mathbb{S}$

$\mathbb{B}$

6 Conclusion

...

Appendix

A Additional notes

The cross-entropy loss is a special case of the KL-divergence.

Lemma A.0.1 *([**wood23**], theorem 5) Theorem* **5** *Let* $\mathbf{y}$ *be a one-hot class vector of length* k*, and* $\mathbf{q} \in \mathbb{R}^k$ *be a model's prediction of the class distribution. Define a set of such models* $\{\mathbf{q}_i\}_{i=1}^M$*, and their combination* $\overline{\mathbf{q}}$ *as their normalised geometric mean. The following decomposition holds.*

$$\underbrace{-\mathbb{E}_D[\mathbf{y} \cdot \ln \overline{\mathbf{q}}]}_{\substack{\textit{expected} \\ \textit{cross-entropy}}} = \underbrace{-\frac{1}{M}\sum_{i=1}^{M} \mathbf{y} \cdot \ln \mathbf{q}_i^*}_{\textit{average bias}} + \underbrace{\frac{1}{M}\sum_{i=1}^{M} \mathbb{E}_D\left[K\left(\mathbf{q}_i^* \| \mathbf{q}_i\right)\right]}_{\textit{average variance}} - \underbrace{\mathbb{E}_D\left[\frac{1}{M}\sum_{i=1}^{M} K\left(\overline{\mathbf{q}} \| \mathbf{q}_i\right)\right]}_{\textit{diversity}},$$

A.1 Flower theorem?

can estimate diversity based purely on tree structure?

B Experiments

B.1 Implementation

what technologies we used, what other projects we built upon, where one can find the code, ...

B.2 Binary classification and Dynamic Random Forests

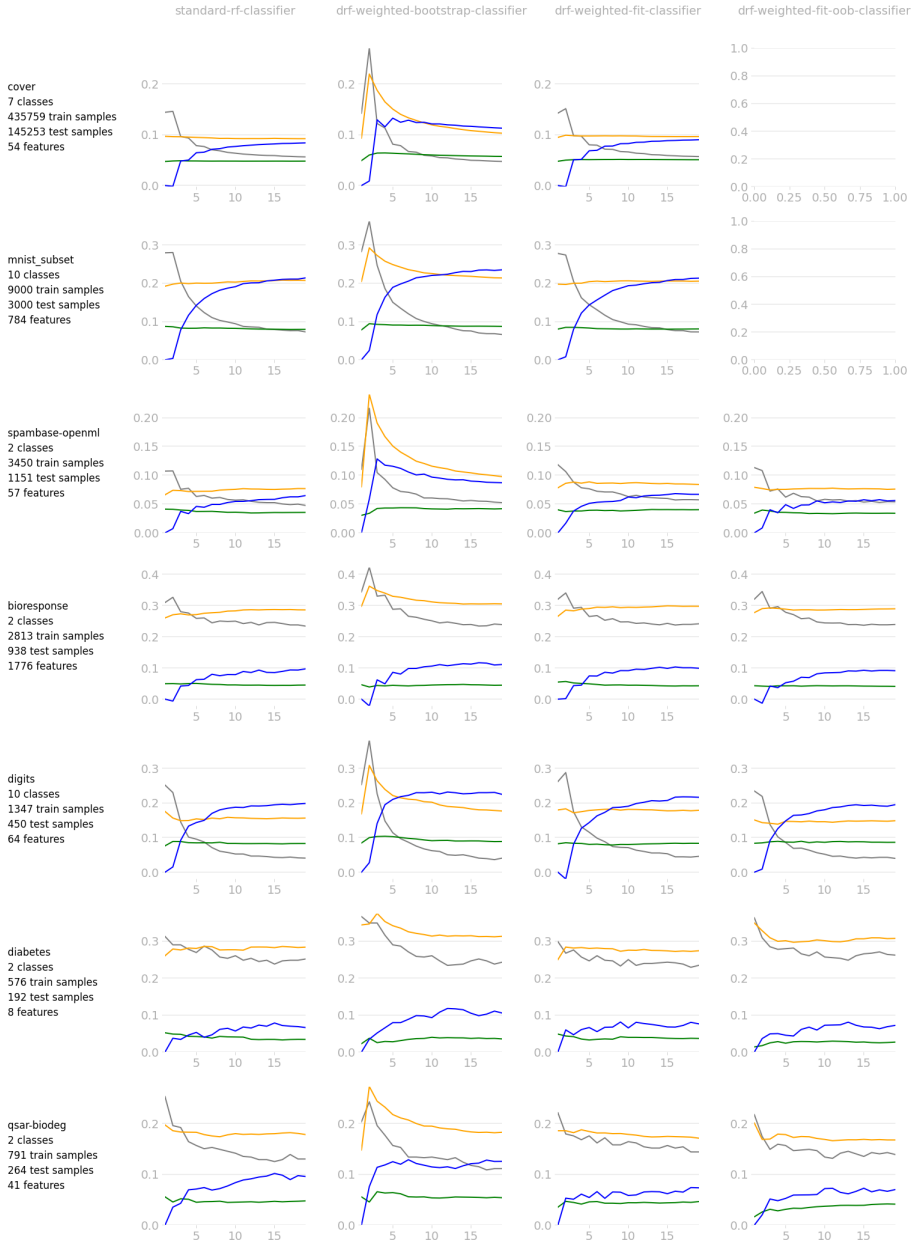

Figure B.1: Full empirical results for 0/1-classification. Rows correspond to different datasets. Columns correspond to different learner variants. The plots show the components of the Bias-Variance-Diversity decomposition of the ensemble error **??**.

Figure B.2: Full empirical results for 0/1-classification. Rows correspond to different datasets. Columns correspond to different learner variants. The plots show the components of the decomposition of the ensemble error in ensemble bias and ensemble variance **??**.

B.3 Boosted Random Forests

just drop derivation and results in here...

advantages of boosting but without the risk of overfitting?

C Outlook

Fuctional Bregman divergences ...

... relationship pairwise comparisons and comparison to centroid – covariance approach vs mean dists approach

properties of metric losses

generalised k-means as splitting function – i'm sure someone tried this with standard k-means already. would then also need to relate to spaeh.

...