ENSEMBLE METHODS

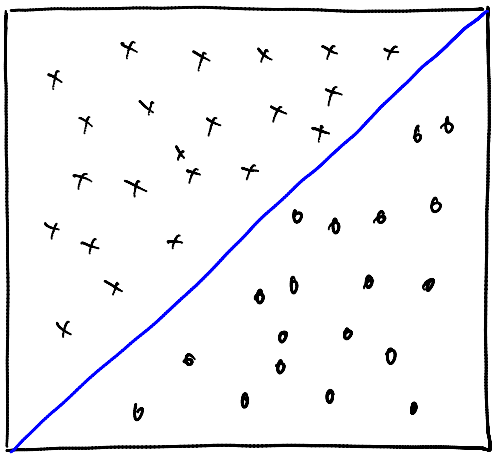
Motivation

The idea behind ensemble methods is to generate several classifiers $f_1, \ldots, f_T$ using a variety of methods, and combine them into a single classifier that outperforms any individual classifier.

To motivate this idea, let's look at a simple example.

Averaged Shifted Histograms

Suppose the feature space is $[0,1]^2$ and the data looks like



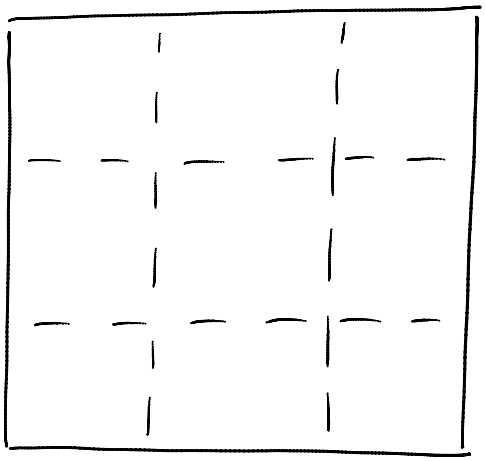
Bayes risk = 0

Marginal of X: uniform

Marginal of Y (class prior): uniform

Further suppose that we are using histogram classifiers, in particular, classifiers based on a regular partition

in particular, classifiers based on a regular partition of $[0,1]$ into nine squares:

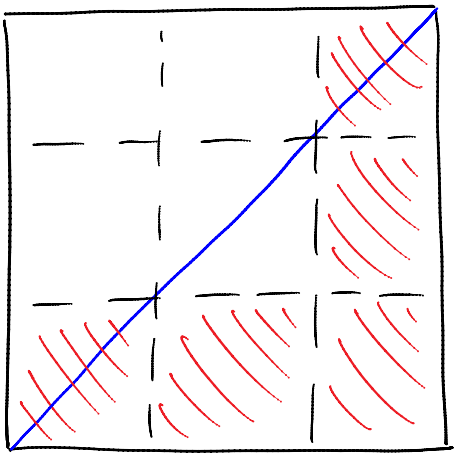


Label of each cell determined by majority vote

As you can imagine, this classifier will not perform very well for the given distribution (or most other distributions).

$$\text{Risk} = \frac{1}{3} \cdot \frac{1}{2} = \frac{1}{6}$$

A third of the time, the classifier has a 50/50 chance of being wrong.

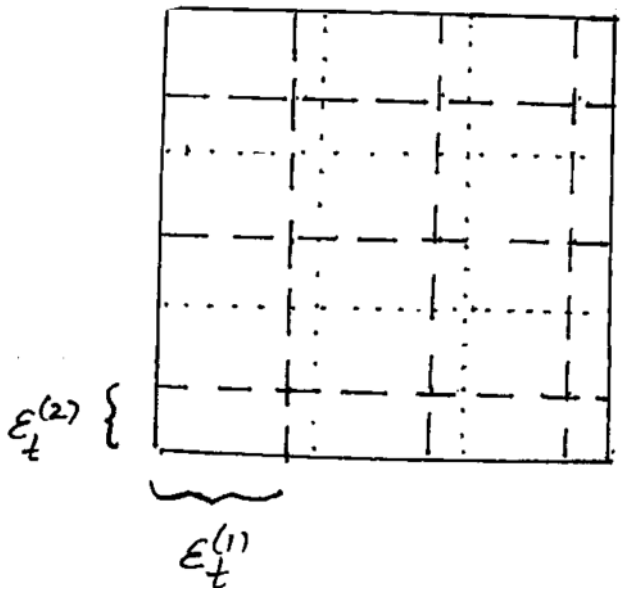


However, with an appropriate ensemble method, the histogram classifier can be much more effective. Consider the following procedure:

For $t = 1, \ldots, T$

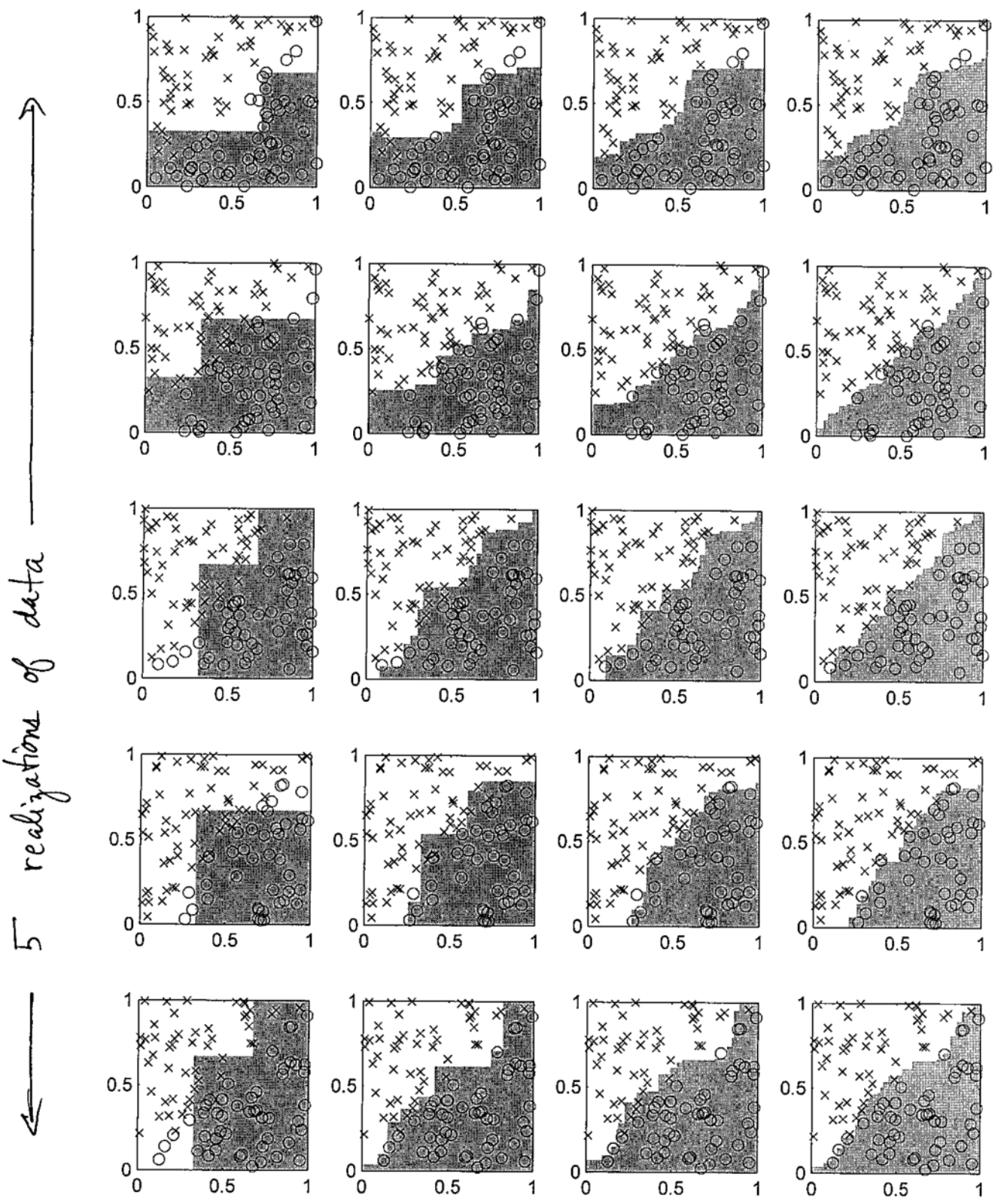
- generate $\epsilon_1^{(t)}, \epsilon_2^{(t)} \in [0, \frac{1}{3})$ uniformly at random
- shift the partition by $[\epsilon_1^{(t)}, \epsilon_2^{(t)}]^T$ and construct f_t based on the shifted partition

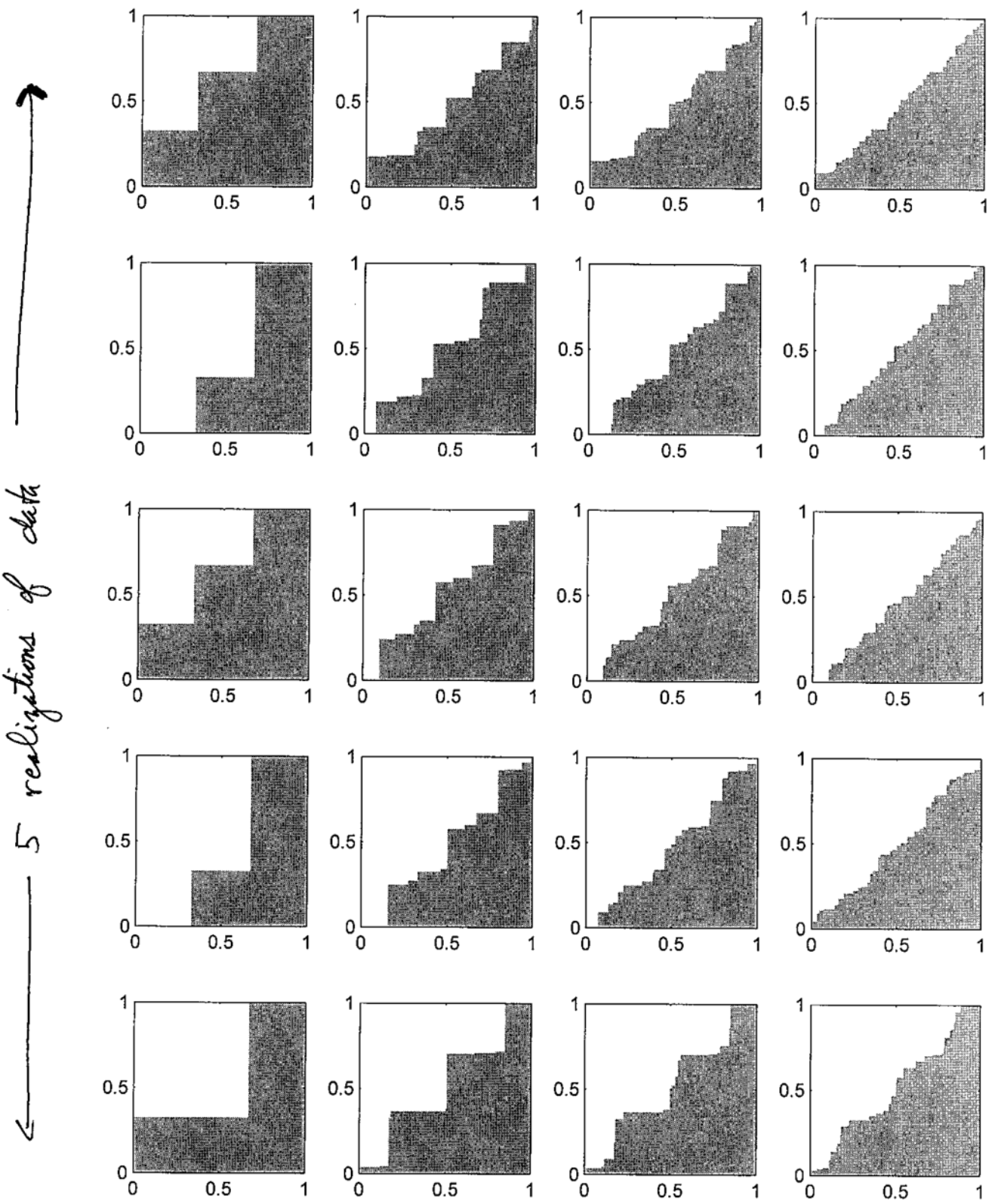
Output $f(x) =$ majority vote over $f_1(x), \ldots, f_T(x)$



A single randomly shifted histogram classifier

Although each individual classifier will perform poorly, the ensemble can perform remarkably well.


# of votes = 1
5
11
21
5 realizations of data
n = 100 points


# of votes = 1
5
11
21
5 realizations of data
n = 1000 points

Not only is the ensemble method better performing, it is also more stable. A machine learning method is stable if a small change in the input to the algorithm leads to a small change in the output, e.g., the learned classifier

Decision trees are a primary example of an unstable classifier, and benefit considerably from ensemble methods.

Like the example above, most ensemble methods result from introducing some form of randomization into the learning algorithm.

Bagging

Bagging stands for bootstrap aggregation. It was developed by Leo Breiman.

Let $B \geq 1$, an integer. Given a training sample of size n, let I_b, $b = 1, \ldots, B$, be a list of size n obtained by sampling from $\{1, 2, \ldots, n\}$ with replacement. I_b is called a bootstrap sample.

Now suppose we have a fixed learning strategy, e.g., decision trees. Let f_b be a classifier obtained by applying this strategy to $\{(x_i, y_i)\}_{i \in I_b}$. The bagging classifier is

$$f(x) = \text{majority vote over } f_1(x), \ldots, f_B(x)$$

Bagging has been shown to improve the performance and stability of decision trees.

Random Forests

A random forest is an ensemble of decision trees where each decision tree is randomized in an iid fashion.

Bagging with decision trees is therefore one example of a random forest.

Another common way to generate a random forest is to base each decision tree on a randomly selected subset of features. This idea can be combined with bagging.

One reason for training with random feature subsets is

the following:

Bootstrap samples are highly correlated. Therefore, in bagging, the different decision trees tend to select the same informative features, leading to highly correlated predictions. The introduction of random features leads to less correlated predictions, which translates to decreased variance of the ensemble prediction.

A rule of thumb for the number of randomly selected features is $\sqrt{d}$, where d is the total number of features.

Random forests are possibly the best "off-the-shelf" method for classification. The idea also extends to regression.