MATH5743M: Statistical Learning - Lecture 9

Dr Seppo Virtanen, s.virtanen@leeds.ac.uk

Semester 2: 21 Mar 2022

Recap

- Collective intelligence in statistical modelling: a combination of different statistical models can do better at making predictions than a single model alone.
- Bootstrap Aggregating (BAGGING): bootstrapping to generate lots of different data sets from which to train many different versions of the same (logistic regression) model.
- We saw that BAGGED models were able to fit better than a single logistic regression model when the underlying reality did not exactly fit the assumptions of the logistic model.

Outline

- A Random Forest (RF) is an ensemble of *decision trees* (many trees make a forest).
- Exploits collective wisdom to generate very accurate predictive models from hundreds or thousands of individually poor models.

Decision tree (recap)

- A fitted decision tree is an easily understood statistical model.
- You start from the top of the tree and progress down, making clearly defined decisions along the way based on the input values, until you reach a leaf node.
- Each leaf specifies either a single class for your output, or the probability that your output belongs to one of many possible classes.
- Fitting these models is also relatively straightforward, as we have seen, and fits well with how humans think about data in real life: spotting dividing lines that separate one type of object from another.
- Simple and interpretable

Decision tree (recap)

- Crucial problem: they are often unstable and produce poor predictions for all but the simplest data sets.
- Small changes to the data can produce different optimal splits in the early parts of the decision tree, leading to completely different trees being produced from very similar data.
- As such it doesn't make sense to see a fitted decision tree as *the* tree to describe a given data set, but one of many that could do so equally well.
- Moreover, splitting the data into two parts at each split is a crude way to identify patterns, meaning that any individual tree is often poor at making predictions.

Decision tree (recap)

- So we are left with the possibility of many equally valid, generally poor classifiers based on a given data set.
- Remember that many poor classifiers can make excellent predictions when used *together* (recall Condorcet's Theorem).
- One of the conditions of collective intelligence was that the different agents or models would be *diverse*, i.e. different from each other.

BAGGING of decision trees: Random Forests

- Ensembles of decision trees, known as Random Forests.
- The term 'forest' tells us that these are based on ensembles of trees.
- The use of the word 'random' is because they make use of random number generation, through the use of bootstrapping (recall that bootstrapping involves taking random samples from the original data).

The Random Forest algorithm

Creating a Random Forest works by applying the ideas we learnt for BAGGING to decision trees.

1. Create N bootstrap samples from the original data set
2. Fit one decision tree model for each bootstrap sample
3. Average over the predictions of all trees when making predictions of new data. Settle ties at random (for classification).

Random Forests in R

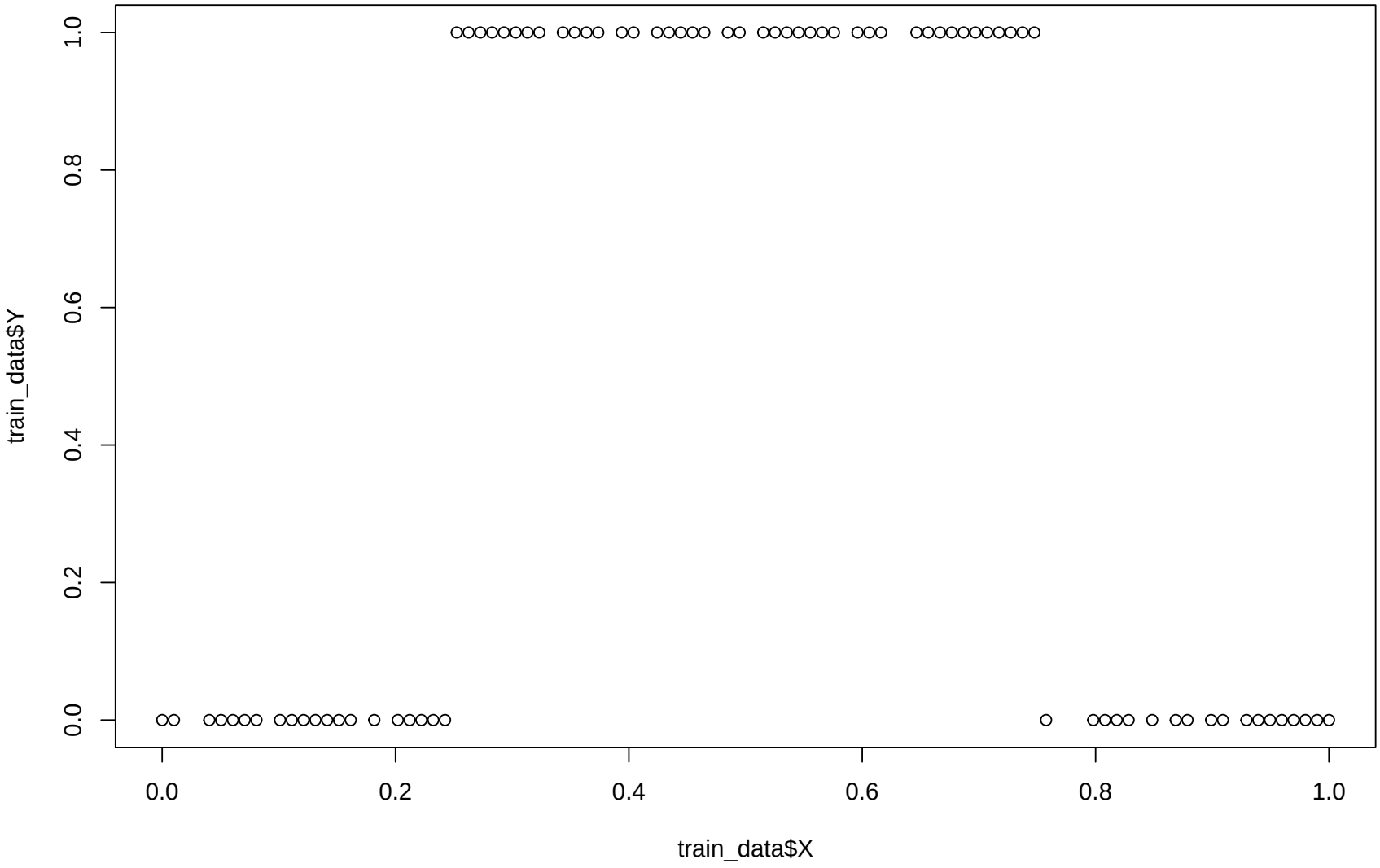
The randomForest package in R provides a very complete set of tools for implementing and using Random Forest models. To install it simply use the following command:

```
install.packages("randomForest")
```

Use of the Random Forest package follows a very similar pattern to our previous use of glm. There is a function **randomForest** that is the equivalent of **glm**, for specifying and fitting a model, and a **predict** method that operates on a fitted model to make predictions for new data.

Example

Let's follow on from our previous look at BAGGING of logistic models by considering data of the same form:



Example

```
library(randomForest)
```

```
## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
train_data$Y = as.factor(train_data$Y)
myForest = randomForest(Y ~ X, data=train_data)
prediction = predict(myForest,
          newdata=test_data, type="prob")
prediction = prediction[, 2]
```

Prediction with RF

The way the Random Forest package predicts the probability for a class is to send the input data through each decision tree in the forest, and count how many of those trees predict each possible class for the output.

- Problems when all the trees agree (i.e. they all predict the same class).
- Class probabilities may be either zero or one. This is unrealistic - we can never be 100% sure in our predictions! And it will cause problems for us if the output in the test data set is different from the prediction, as the predictive log-likelihood will be -INF!

Prediction with RF

- a simple 'trick'. We imagine that we create two more trees, one of which predicts class '1', the other of which predicts class '0'.

```
number_of_trees = myForest$ntree
prediction = (number_of_trees*prediction + 1)/
  (number_of_trees + 2)
```

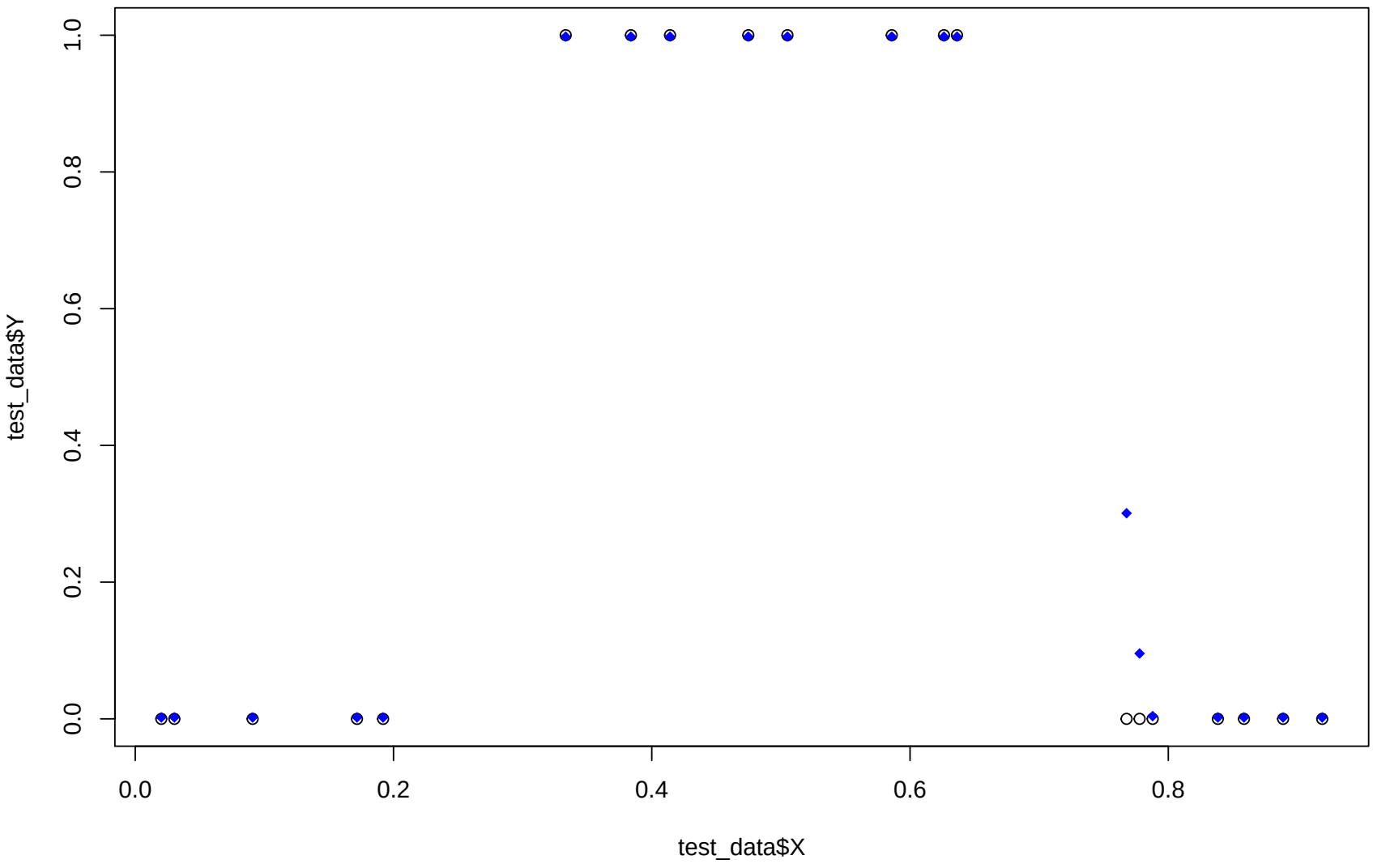
Example

Now, armed with our adjusted prediction, we can calculate the predictive log-likelihood of the test data to evaluate how good our prediction actually is:

```
testLL = sum(log(prediction[which(test_data$Y==1)])) +
  sum(log(1-prediction[which(test_data$Y==0)]))
print(testLL)
```

```
## [1] -0.4962072
```

Example



Discussion

- Remember when we did linear and logistic regression, we could get summaries of the model telling us exactly how important each input was (the regression coefficients), what direction of effect these inputs produced (the sign of the coefficient) and how uncertain these effects were (confidence intervals).
- By contrast, with a Random Forest we have a model that does very well numerically but provides no easily read information about the model's internal function.

Discussion

- If you see data with a very clear monotonic trend you should use linear or logistic regression (for regression or classification problems).
- If you see something that looks a bit more complicated you might use polynomial regression.
- For very complex problems, or ones where prediction accuracy matters more than interpretation, you can consider a Random Forest.