

Computational Sociology

Machine Learning

Dr. Thomas Davidson

Rutgers University

March 29, 2021

Plan

1. Course updates
2. Introduction to machine learning
3. Model evaluation
4. Classification algorithms
5. Machine learning in R

Course updates

Homework 3

- ▶ Homework 3 has now been released
 - ▶ Due next Monday at 4pm

Roadmap

A month of machine learning

1. Introduction to supervised machine learning
2. Supervised text classification
3. Challenges: Unpredictability and bias
4. Supervised image classification / computer vision

Introduction to machine learning

What is machine learning?

- ▶ Machine learning is a method to “automate discovery from data” (Molina and Garip 2019).
- ▶ It is a method well-suited to working with large datasets and has a range of different applications (Grimmer, Roberts, and Stewart 2021).

Introduction to machine learning

Supervised and unsupervised learning

- ▶ *Supervised machine learning*
 - ▶ We observe an output Y for each input X .
 - ▶ The goal is to learn a function to predict X given Y
 - ▶ Often SML is used for *classification* problems, where the goal is the classify observation into discrete classes.
- ▶ *Unsupervised machine learning*
 - ▶ We only observe X , there is often no “correct” answer Y
 - ▶ e.g. Topic modeling.

Introduction to machine learning

Prediction and explanation in sociology

- ▶ Most sociological analyses are *explanatory*
 - ▶ The objective is to explain a relationship between variables, descriptive or causal.
- ▶ Watts (2014) argues that we must pay more attention to *prediction* and consider the *predictive validity* of sociological theories.
 - ▶ This may allow us to better understand the scope and robustness of many sociological findings.

See Watts, Duncan J. 2014. "Common Sense and Sociological Explanations." *American Journal of Sociology* 120 (2): 313–51. <https://doi.org/10.1086/678271> and the debate with Turco and Zuckerman.

Introduction to machine learning

Prediction and explanation in sociology

- ▶ Consider the following linear model:

$$Y = \hat{\beta}_0 + \hat{\beta}_1 + \epsilon$$

- ▶ Social scientists are typically interested in the $\hat{\beta}$ given Y , whereas computer scientists are more interested models to predict \hat{Y} .
- ▶ What can we learn by constructing models optimized to predict \hat{Y} ?

Introduction to machine learning

Prediction and explanation in sociology

- ▶ Predictive models are specified differently to explanatory ones:
 - ▶ In a regression context, we might use theory to guide the selection of a handful of *variables* to appropriately estimate $\hat{\beta}$.
 - ▶ In a predictive context, we want to find the function $f(X)$, such that $Y = f(X)$. This often involves many more variables and a more complex functional form.
 - ▶ Variables in the ML context are referred to as *features*.

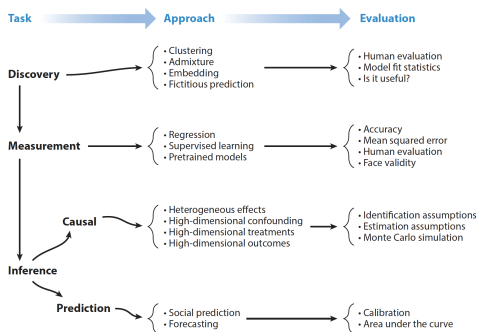
Introduction to machine learning

Why predict?

- ▶ Policy interventions
- ▶ Document classification
- ▶ Causal inference
- ▶ Record linkage and data imputation

Introduction to machine learning

Why predict?



Grimmer, Roberts, and Stewart 2021.

Introduction to machine learning

Data splitting and model training

- ▶ In supervised machine learning we generally split our data into two groups, *training* and *testing*
- ▶ The *training* data is used to train a model or to estimate $Y = f(X)$.
 - ▶ The model uses X to predict a *known* Y .
- ▶ The *testing* data is used to choose and evaluate a model. This is also referred to as *held-out* or *out-of-sample* data.
 - ▶ We take our trained model and predict Y for the test examples.
 - ▶ We then compare \hat{Y} to Y to assess predictive accuracy.
- ▶ Sometimes we reserve a third *validation* set of data that we only use at the end of the process.
 - ▶ Unlike the testing data, we never use it for model selection.

Introduction to machine learning

Vignette

- ▶ Let's say we want to predict a college-attendance given information about their early childhood.
- ▶ The explanatory approach would be to construct some regression model to predict

$$Y_{college} = \hat{\beta}_0 + \hat{\beta}_{1:K} X_{1:K} + \epsilon$$

using K predictor variables.

- ▶ Assuming the model is appropriately specified (e.g. We have addressed missingness, multicollinearity, heteroskedasticity), we would then analyze the statistical significance of the coefficients to develop an explanation.
 - ▶ e.g. "Mother's level of education is a positive predictor of college attendance ($p < 0.05$)."

Introduction to machine learning

Vignette

- ▶ Now consider a predictive version of this model. Here the goal is to develop the best possible prediction of $Y_{college}$.
- ▶ We will include M predictors, such that $M \gg K$, to better approximate $f(X)$.
- ▶ We estimate a model using our training data,

$$Y_{college_{train}} = \hat{\beta}_0 + \beta_{1:M} X_{1:M} + \epsilon$$

- ▶ We then use this model to predict $\hat{Y}_{college_{test}}$ and compare the predictions to the known values, $Y_{college_{test}}$.
- ▶ Finally, we make a statement about the accuracy of our model.
 - ▶ e.g. "The model predicted out-of-sample college attendance with 75% accuracy."

Introduction to machine learning

Explanatory models \neq predictive models

- ▶ Mullainathan and Spiess (2017) evaluate the relationship between predictive and explanatory models. In an ideal world, we might want to have a model optimized for predicting \hat{Y} and $\hat{\beta}$'s.
- ▶ Explanatory models often have low-predictive power. But can predictive models be explanatory?

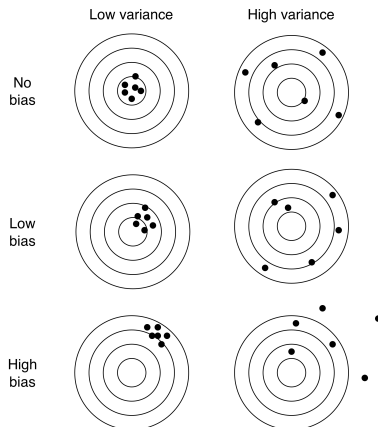
Introduction to machine learning

Explanatory models \neq predictive models

- ▶ “The very appeal of these algorithms is that they can fit many different functions. But this creates an Achilles’ heel: more functions mean a greater chance that two functions with very different coefficients can produce similar prediction quality” (Mullainathan and Spiess 2017: 97–98).
- ▶ In short, there might be many different subsets of a dataset that produce equally good predictions. This makes it hard to develop a coherent explanation based on a predictive model.

Model evaluation

Bias-variance trade-offs



Salganik 2017. See [this website](#) for a visualization of the bias-variance trade-off.

Model evaluation

Underfitting and overfitting

- ▶ *Underfitting* occurs when a model poorly fits the data.
 - ▶ e.g. A linear model may not capture non-linear relationships between variables.
- ▶ *Overfitting* occurs when a model fits random noise in the training data.
 - ▶ If a model has overfit then it does not generalize well to unseen data.
 - ▶ This tends to be a more serious problem in machine-learning than underfitting since we often have richly parameterized models.
 - ▶ *Out-of-sample* validation allows us to directly measure overfitting, something generally ignored in explanatory approaches (Watts 2014).

Model evaluation

Cross-validation

- ▶ Train-test splits reduce the amount of data available to us, increasing risk of underfitting and potentially making results sensitive to the particular split.
- ▶ *Cross-validation* is an approach to split the data into small test-train subsets.
- ▶ A popular approach is *k-fold* cross-validation where we split the data into k subsets.
 - ▶ We successively train a model on each $k - 1$ folds and test it on the k^{th} fold.
 - ▶ The results are then averaged across all k folds.

Model evaluation

Cross-validation



Source: Wikipedia.

Model evaluation

Cross-validation

- ▶ The extreme is called *leave-one-out* or *LOO* cross-validation.
 - ▶ Given N observations, we train N models, each time using $N - 1$ data points.
 - ▶ However this is rarely used because it is very computationally expensive.

Note how these approaches are similar to *bootstrap* and *jackknife* resampling.

Model evaluation

Regularization

- ▶ *Regularization* is another approach we can use to prevent overfitting.
 - ▶ We constrain the parameter space to try to prevent fitting noise.
- ▶ For example, the *least absolute shrinkage and selection operator (LASSO)* imposes a penalty on regression coefficients,
 - ▶ In addition to minimizing the squared error, we also want to minimize $\sum_{j=1}^k |\beta_j|$, the sum of the absolute values of the coefficients.
 - ▶ In practice, this forces many coefficients to equal zero, $\beta_j = 0$, reducing the complexity of the parameter space.

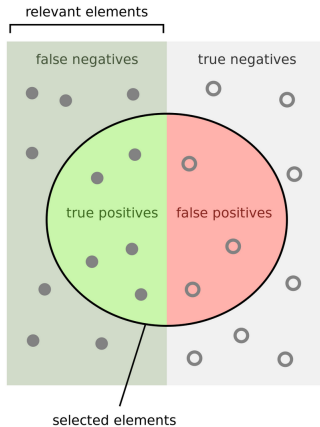
Model evaluation

Metrics: Binary classification

- ▶ The following metrics apply to binary classification problems, although many can be generalized to multi-class or continuous outcomes.
- ▶ A binary classifier learns a function $f(X)$ to predict Y , where $Y = 1$ or $Y = 0$.
 - ▶ Many algorithms return a predicted probability $P(Y|X)$, but some only return a discrete value (1 or 0).

Model evaluation

Metrics: TP, FP, TN, FN




Source: Wikipedia.

Model evaluation

Metrics: Precision

How many selected items are relevant?


$$\text{Precision} = \frac{\text{Relevant}}{\text{Selected}}$$


Source: Wikipedia.

Model evaluation

Metrics: Recall

How many relevant items are selected?

$$\text{Recall} = \frac{\text{Number of relevant items selected}}{\text{Total number of relevant items}}$$


Source: Wikipedia.

Model evaluation

Metrics: F1

The F_1 score is the *harmonic mean* of precision and recall and is often used as an overall description of predictive performance for a classifier.

$$F_1 = 2 \frac{\text{precision} * \text{recall}}{\text{precision} + \text{recall}} = \frac{TP}{TP + \frac{1}{2}(FP + FN)}$$

Model evaluation

Metrics: The confusion matrix

| True categories | Hate | Offensive | Neither | |
|-----------------|------|----------------------|-----------|---------|
| | 0.61 | 0.31 | 0.09 | |
| | 0.05 | 0.91 | 0.04 | |
| | 0.02 | 0.03 | 0.95 | |
| | | Hate | Offensive | Neither |
| | | Predicted categories | | |

Davidson, Thomas, Dana Warmlesley, Michael Macy, and Ingmar Weber. 2017. "Automated Hate Speech Detection and the Problem of Offensive Language." In Proceedings of the 11th International Conference on Web and Social Media (ICWSM), 512–15.

Model evaluation

Metrics: Receiver Operating Characteristic (ROC) curve

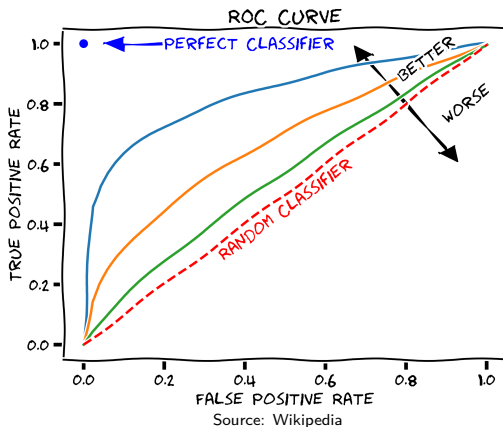
- ▶ If an algorithm returns a predicted probability then we must identify a *threshold*, such that

$$\text{Class}(Y) = \begin{cases} 1, & P(\hat{Y}|X) \geq \text{threshold} \\ 0, & P(\hat{Y}|X) < \text{threshold} \end{cases}$$

- ▶ Plot the true positive rate ($TPR = TP / TP + FN$) against false positive rate ($FPR = FP / FP + TN$) for different predicted probability thresholds to identify the optimal value. This is known as the *ROC* curve.
- ▶ The area under the ROC curve (*AUC*) provides an overall measure of classifier performance.

Model evaluation

Metrics: Receiver Operating Characteristic (ROC) curve



Model evaluation

Metrics

- ▶ The choice of metric depends on the outcome of interest and what you want to optimize for. Often we might want to use a metric like ROC or F1 to find a compromise.
- ▶ Consider a carbon monoxide alarm:

| | Alarm | No alarm |
|------------|-------|----------|
| CO present | TP | FN |
| CO absent | FP | TN |
- ▶ False negatives are really bad and should be avoided at all costs.
- ▶ Too many false positives will also be bad, as it may lead people to remove the batteries from the alarm, but a low level will be tolerated.

Classification algorithms*

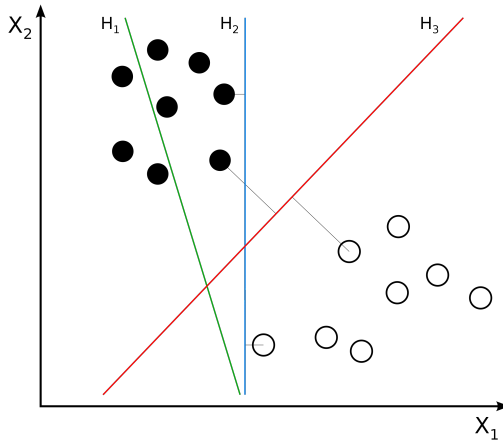
Logistic regression

- ▶ Logistic regression is a regression model for binary outcomes (although there is some debate about when we should estimate a standard linear probability model using OLS).
- ▶ Uses a logit function to estimate the log-odds of an event ($Y = 1$) given predictors X .
- ▶ Multinomial logistic regression can be used if you have a multi-class outcome.
 - ▶ e.g. A model predicting level of education.

*Many of these algorithms can also be used for regression problems where the outcome is continuous.

Classification algorithms

Support Vector Machines

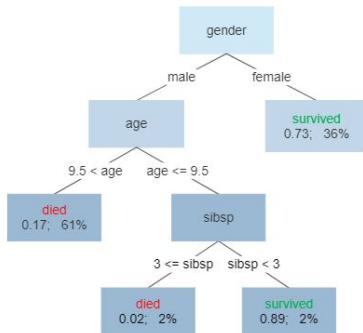


Source: Wikipedia

Classification algorithms

Decision Trees

Survival of passengers on the Titanic



Source: Wikipedia. See this website for an excellent visual introduction to decision trees.

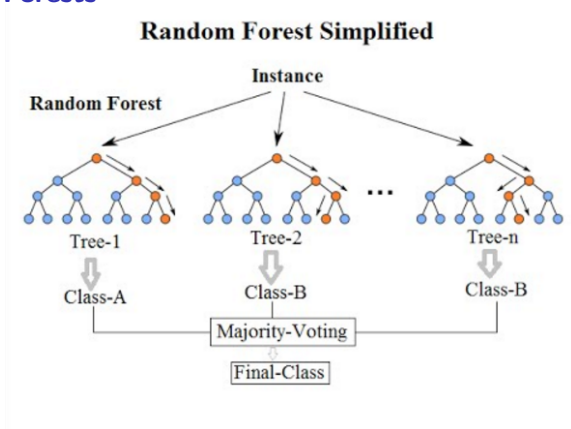
Classification algorithms

Random Forests

- ▶ Decision trees tend to overfit the training data
- ▶ Solution: Grow lots of trees and average over them
 - ▶ Using a procedure called *bootstrap aggregating* or *bagging* for short we can sample from our data and generate a *forest* consisting of many decision trees. This is known as an *ensemble* method because it involves more than one model.
 - ▶ The approach is effective because the algorithm randomly splits the data into leaf nodes based on different features, hence it is a *random* forest.
 - ▶ The final classification is an average across the different decision trees.

Classification algorithms

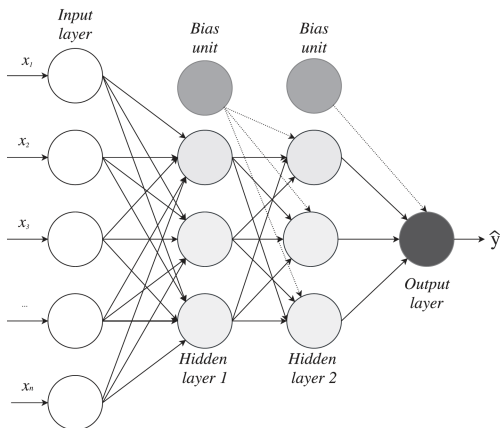
Random Forests



Source: Wikipedia

Classification algorithms

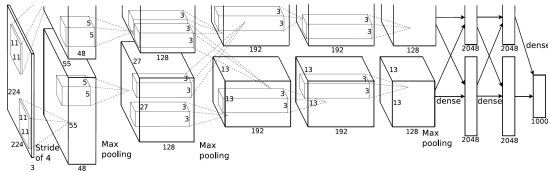
Neural networks



Davidson 2019.

Classification algorithms

Neural networks



Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. 2012. "Imagenet Classification with Deep Convolutional Neural Networks." In *Advances in Neural Information Processing Systems*, 1097–1105.

Classification algorithms

Hyperparameters

- ▶ Each algorithm has hyperparameters that can adjust how it works.
 - ▶ e.g. Regularization type for logistic regression and SVM.
 - ▶ e.g. Number of trees, tree depth, and splitting criterion for random forest.
 - ▶ e.g. Number of layers, activation function, and optimization routine for neural networks.
- ▶ Often we want to find the algorithm that best fits the data so we conduct a search over different hyperparameters and compare many different models.
 - ▶ In many cases we also want to test the effect of different pre-processing or feature construction steps.

Classification algorithms

Hyperparameter search and computational complexity.

- ▶ Davidson (2019) uses neural network models to predict high school GPA.
 - ▶ Three model hyperparameters with 40 different combinations
 - ▶ Number of hidden layers (depth)
 - ▶ Number of neurons per hidden layer (breadth)
 - ▶ Activation function
 - ▶ Each model is trained using 5-fold cross-validation, resulting in 200 different model fits
- ▶ These models took over 12 hours to estimate on a high-end Macbook Pro.

Python code and output is available [here](#).

Classification algorithms

Black-box models and interpretability

- ▶ In contrast to standard explanatory models, which are considered to be interpretable, many of these algorithms are described as “black boxes,” meaning that we are unable to observe their workings.
- ▶ There is a trade-off between model complexity (often associated with better predictions) and human interpretability
 - ▶ Watts (2014) argues that it may be worth sacrificing some interpretability in the interest of better predictions.
- ▶ But there are lots of developments in the field of ML interpretability
 - ▶ See Chrisoph Molar’s open-source book *Interpretable Machine Learning for an overview.

Classification algorithms

Black-box models

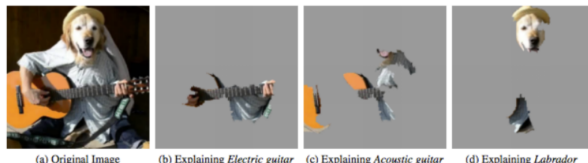
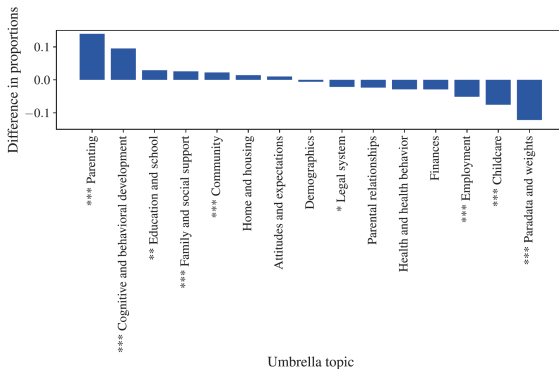


Figure 4: Explaining an image classification prediction made by Google's Inception network, highlighting positive pixels. The top 3 classes predicted are "Electric Guitar" ($p = 0.32$), "Acoustic guitar" ($p = 0.24$) and "Labrador" ($p = 0.21$)

Ribeiro, Marco Tulio, Sameer Singh, and Carlos Guestrin. 2016. " 'Why Should I Trust You?': Explaining the Predictions of Any Classifier." In Proceedings of the 22nd ACM SIGKDD, 1135–44. ACM.
<https://doi.org/10.1145/2939672.2939778>.

Classification algorithms

Black-box models



Davidson 2019.

Machine learning in R

Tidymodels

- ▶ tidymodels is a set of packages designed to use tidy principles to conduct machine-learning.
 - ▶ See <https://www.tidymodels.org/packages/> for a list of packages.

Pre-Process → Train → Validate



Source: tidymodels tutorial.

Machine learning in R

Loading tidymodels

The tidymodels package loads all of the sub-packages, as well as the tidyverse packages. We're going to be using the iris dataset for today's analysis. This is a simple dataset containing data on 150 irises. There are three species, "setosa", "versicolor", and "virginica" and four variables sepal.Length, sepal.Width, Petal.Length, and Petal.Width. The goal is to predict the species given the sepal and petal information.

```
library(tidymodels)
head(iris)
```

| ## | Sepal.Length | Sepal.Width | Petal.Length | Petal.Width | Species |
|------|--------------|-------------|--------------|-------------|---------|
| ## 1 | 5.1 | 3.5 | 1.4 | 0.2 | setosa |
| ## 2 | 4.9 | 3.0 | 1.4 | 0.2 | setosa |
| ## 3 | 4.7 | 3.2 | 1.3 | 0.2 | setosa |
| ## 4 | 4.6 | 3.1 | 1.5 | 0.2 | setosa |
| ## 5 | 5.0 | 3.6 | 1.4 | 0.2 | setosa |
| ## 6 | 5.4 | 3.9 | 1.7 | 0.4 | setosa |

Machine learning in R

Loading and splitting data

We can use the `initial_split` command to create a train-test split, where 20% of the data are held-out for testing.

```
iris_split <- initial_split(iris, prop = 0.8)
print(iris_split)
```

```
## <Analysis/Assess/Total>
```

```
## <121/29/150>
```

Machine learning in R

Viewing the data

```
iris_split %>% training() %>% head()
```

| ## | Sepal.Length | Sepal.Width | Petal.Length | Petal.Width | Species |
|------|--------------|-------------|--------------|-------------|---------|
| ## 1 | 5.1 | 3.5 | 1.4 | 0.2 | setosa |
| ## 2 | 4.9 | 3.0 | 1.4 | 0.2 | setosa |
| ## 3 | 4.7 | 3.2 | 1.3 | 0.2 | setosa |
| ## 4 | 4.6 | 3.1 | 1.5 | 0.2 | setosa |
| ## 5 | 5.0 | 3.6 | 1.4 | 0.2 | setosa |
| ## 6 | 5.4 | 3.9 | 1.7 | 0.4 | setosa |

Machine learning in R

Pre-processing

We will use the recipes package to pre-process the data.

```
iris_recipe <- training(iris_split) %>%  
  recipe(Species ~.) %>%  
  step_corr(all_predictors()) %>%  
  step_center(all_predictors(), -all_outcomes()) %>%  
  step_scale(all_predictors(), -all_outcomes()) %>%  
  prep()  
  
iris_recipe # Not petal length removed due to correlation  
  
## Data Recipe  
##  
## Inputs:  
##  
##      role #variables  
## outcome      1  
## predictor      4  
##
```


Machine learning in R

Pre-processing the test data

The previous chunk only applied these transformations to the training data. We want to also modify the test data so that they are the same dimensions. We can apply the `recipe` to the new data using the `bake` command. We also want to load the training data into a variable using the `juice` command. This extracts the data directly from the recipe.

```
iris_testing <- iris_recipe %>%  
  bake(testing(iris_split))  
  
iris_training <- juice(iris_recipe)
```

Machine learning in R

Specifying a model

The `parsnip` command allows us to specify a model. ML models in R exist across a range of different packages and `parsnip` gives them a standardized syntax. We define the model, choose the package (in this case `randomForest`), then use `fit` to train the model.

```
library(randomForest)
rf <- rand_forest(trees = 100, mode = "classification") %>%
  set_engine("randomForest") %>%
  fit(Species ~ ., data = iris_training)
```

Machine learning in R

Making predictions

```
preds <- predict(rf, iris_testing)
bind_cols(iris_testing, preds)
```

```
## # A tibble: 29 x 5
##   Sepal.Length Sepal.Width Petal.Width Species .pred_class
##         <dbl>         <dbl>         <dbl> <fct>    <fct>
## 1      -1.80        -0.314        -1.33 setosa  setosa
## 2      -1.18         0.147        -1.46 setosa  setosa
## 3      -0.552         1.99        -1.06 setosa  setosa
## 4      -0.176         1.76        -1.20 setosa  setosa
## 5      -1.05        -0.0838       -1.33 setosa  setosa
## 6      -1.43         0.377        -1.33 setosa  setosa
## 7      -1.30         0.147        -1.33 setosa  setosa
## 8      -0.928         0.838        -1.33 setosa  setosa
## 9      -1.05         1.07        -1.20 setosa  setosa
## 10     -0.928         1.76        -1.06 setosa  setosa
## # ... with 19 more rows
```

Machine learning in R

Calculating metrics

```
precision <- bind_cols(iris_testing, preds) %>% precision(truth=Species  
recall <- bind_cols(iris_testing, preds) %>% recall(truth=Species, esti  
print(bind_rows(precision, recall))
```

```
## # A tibble: 2 x 3  
##   .metric .estimator .estimate  
##   <chr>    <chr>      <dbl>  
## 1 precision macro      0.925  
## 2 recall   macro      0.925
```

Machine learning in R

Calculating metrics

We can also extract the predicted probabilities by adding an argument to the predict function.

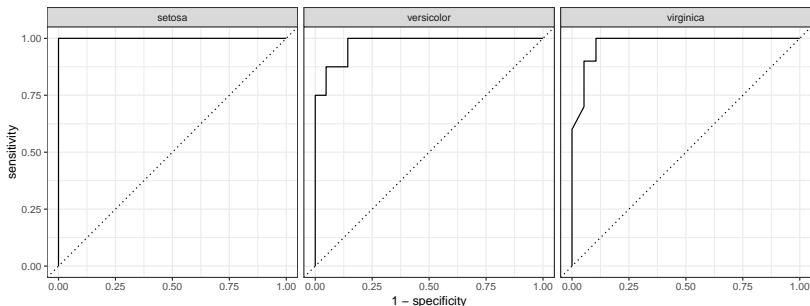
```
probs <- rf %>%  
  predict(iris_testing, type = "prob") %>%  
  bind_cols(iris_testing)  
head(probs)
```

```
## # A tibble: 6 x 7  
##   .pred_setosa .pred_versicolor .pred_virginica Sepal.Length Sepal.W  
##           <dbl>           <dbl>           <dbl>           <dbl>           <  
## 1         0.93           0.07           0           -1.80          -0.  
## 2         0.97           0.03           0           -1.18           0.  
## 3         1           0           0           -0.552          1.  
## 4         0.95           0.05           0           -0.176          1.  
## 5         0.91           0.09           0           -1.05          -0.  
## 6         1           0           0           -1.43           0.  
## # ... with 2 more variables: Petal.Width <dbl>, Species <fct>
```

Machine learning in R

Calculating metrics

```
probs %>% roc_curve(Species, .pred_setosa:.pred_virginica ) %>% autoplot
```



Machine learning in R

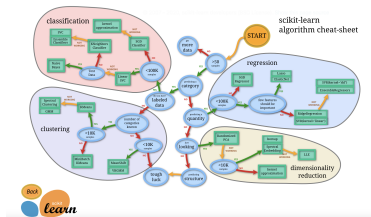
Next week

- ▶ We will go into more depth using `tidymodels` to implement cross-validation and a hyperparameter search and will evaluate multiple different models.
- ▶ Supervised machine learning to perform text classification.
- ▶ Discuss sampling and data annotation procedures.

Machine learning in R

Alternatives

- ▶ Python has a more developed ML ecosystem than R.
 - ▶ `scikit-learn` provides a suite of tools for most machine-learning tasks except deep-learning, which requires specialized libraries.



Source: scikit-learn documentation. See this tutorial for how to run scikit-learn using R.

Summary

- ▶ Machine learning techniques allow us to “automate discovery from data”
- ▶ Supervised and unsupervised ML techniques
- ▶ Prediction vs. explanation
- ▶ Over and underfitting
- ▶ Regularization
- ▶ Algorithms