Introduction to Machine Learning

Dave Kincaid

IDEXX Laboratories, Inc.

11/12/2011

Outline

Topic

1. Overview of Machine Learning
2. Species Classifier Example
3. Species Classifier and Naive Bayes
4. Species Classifier and Artificial Neural Networks
5. Species Classifier and Random Forest
6. Summary of the 3 algorithms and next steps
7. References

Definitions and introduction to our species classifier

Definitions

- Target: the result of running the model

Example (a species classifier)

Name	Sex	Age	Weight	Visits	Sibs	ActSibs	Species
Fluffy	Female	200	12	2	1	3	Feline
Spot	Neuter	1243	50	1	1	1	Canine
Max	Male	50	7	2	3	2	Feline

Definitions and introduction to our species classifier

Definitions

- Target: the result of running the model
- Features: the data elements used to predict

Example (a species classifier)

Name	Sex	Age	Weight	Visits	Sibs	ActSibs	Species
Fluffy	Female	200	12	2	1	3	Feline
Spot	Neuter	1243	50	1	1	1	Canine
Max	Male	50	7	2	3	2	Feline

Definitions and introduction to our species classifier

Definitions

- Target: the result of running the model
- Features: the data elements used to predict
- Training examples: the sets of features and targets used to construct the model

Example (a species classifier)

Name	Sex	Age	Weight	Visits	Sibs	ActSibs	Species
Fluffy	Female	200	12	2	1	3	Feline
Spot	Neuter	1243	50	1	1	1	Canine
Max	Male	50	7	2	3	2	Feline

Definitions and introduction to our species classifier

Definitions

- Target: the result of running the model
- Features: the data elements used to predict
- Training examples: the sets of features and targets used to construct the model
- Machine learning: given the training data learn a mapping function $f(x)$ that can map feature variables to target variables

Example (a species classifier)

Name	Sex	Age	Weight	Visits	Sibs	ActSibs	Species
Fluffy	Female	200	12	2	1	3	Feline
Spot	Neuter	1243	50	1	1	1	Canine
Max	Male	50	7	2	3	2	Feline

Types of machine learning

Supervised learning

Learning using training examples which have both features and the desired target.

Types of machine learning

Supervised learning

Learning using training examples which have both features and the desired target.

Unsupervised learning

Learning using only features. Don't know (or don't provide) the targets

Types of machine learning

Supervised learning

Learning using training examples which have both features and the desired target.

Unsupervised learning

Learning using only features. Don't know (or don't provide) the targets

Reinforcement learning

Computer is only given feedback as to whether the answer is right or wrong.

Types of machine learning

Supervised learning

Learning using training examples which have both features and the desired target.

Unsupervised learning

Learning using only features. Don't know (or don't provide) the targets

Reinforcement learning

Computer is only given feedback as to whether the answer is right or wrong.

Evolutionary learning

Learning where a solution is evolved from some starting population based on a fitness function.

Problem types

Regression

- The target is a continuous number

Problem types

Regression

- The target is a continuous number

Classification

- Target is a discrete set of classes
- Binary or multiclass

Feature representation

- **Continuous features (numerical):** Represented as themselves. Depending on the algorithm may need to be standardized ($N(0, 1)$) or normalized ([0,1])
- **Categorical features (ordinal, text) also known as factors or levels:** can be represented as dummy variables.

Example (Species data)

Name	Sex	Age	Weight	Visits	Sibs	ActSibs	Species
Fluffy	Female	200	12	2	1	3	Feline
Spot	Neuter	1243	50	1	1	1	Canine
Max	Male	50	7	2	3	2	Feline

becomes:

F	M	N	Age	Weight	Visits	Sibs	ActSibs	Fluffy	Spot	Max	Species
1	0	0	200	12	2	1	3	1	0	0	Feline
0	0	1	1243	50	1	1	1	0	1	0	Canine
0	1	0	50	7	2	3	2	0	0	1	Feline

Short List of Algorithms

Supervised learning algorithms

- Naive Bayes
- k-Nearest Neighbors
- Decision trees
- Random forests
- Logistic regression
- Support Vector Machines (SVM)
- Artificial Neural networks
- Stochastic Gradient Descent

Unsupervised learning algorithms

- k-means clustering
- Artificial neural networks
- Self-organizing maps
- Hierarchical clustering
- Mean shift clustering
- Affinity propagation

Languages and libraries

Java

- Apache Mahout
- Weka

Python

- Scikit-learn
- PyBrain
- Natural Language Toolkit (NLTK)
- PyML

C#

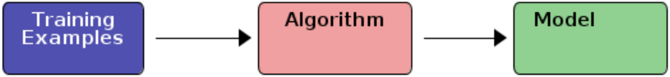
- IKVM & Weka
- AForge.NET & Accord.NET
- Infer.NET

Others

- R stats package w/various add-ons
- libsvm, libFANN (C/C++)
- Incanter (Clojure)

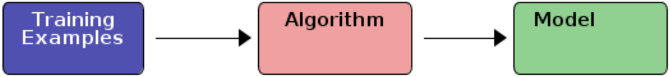
Workflow

- Training the model

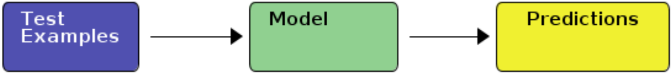


Workflow

- Training the model

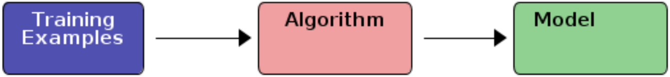


- Testing the model

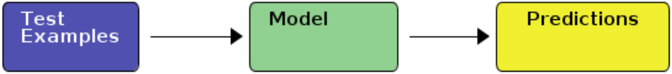


Workflow

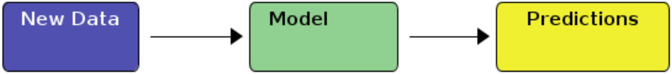
- Training the model



- Testing the model



- Using the model



Topic

Species Classifier

Example (Species Classifier Example)

- Features: name, sex, age, weight, # of visits, # of siblings
- Target: Species

Species Classifier

Example (Species Classifier Example)

- Features: name, sex, age, weight, # of visits, # of siblings
- Target: Species

Algorithms

- Naive Bayes - probabilistic
- Artificial neural network - weighting and combination of features
- Random Forest - based on decision trees

Species Classifier

Example (Species Classifier Example)

- Features: name, sex, age, weight, # of visits, # of siblings
- Target: Species

Algorithms

- Naive Bayes - probabilistic
- Artificial neural network - weighting and combination of features
- Random Forest - based on decision trees

Code used

- R with caret package (and others in a supporting role)

R software and the Caret package

R Software Package

- Open source, free language and environment for statistical computing and graphics.
- Provides a wide variety of statistical (linear and nonlinear modelling, classical statistical tests, time-series analysis, classification, clustering, ...) and graphical techniques, and is highly extensible.

Caret package (Classification and Regression Training)

- Massively streamlines and simplifies the process for creating predictive models.
- Tools for data splitting, pre-processing, model tuning, variable importance estimation

Species Classifier: Sample data

Total number of training examples: 72,696 with 69 features

Name	Sex	Age	Weight	Visits	TotSibs	ActSibs	Species
NIKA	Spayed Female	5215	8.2	0	1	1	Feline
SOPHIE	Spayed Female	1101	8.12	0	4	3	Feline
DIXIE	Spayed Female	4033	35.5	0	4	3	Canine
SAMBO	Neutered Male	6224	7	0	4	3	Feline
BUDDY	Male	3962	1.8	0	2	2	Feline
SHELBY	Spayed Female	5896	34.7	0	2	2	Canine
OTIS	Male	5725	6.3	0	1	1	Canine
HEINIKEN	Male	4435	4.1	0	1	1	Canine
COOKIE JANE	Spayed Female	4150	11	0	1	1	Canine
SERENDIPITY	Spayed Female	3952	12	0	2	2	Feline
Phoebe	Female	5040	3	0	2	1	Feline
Riley	Neutered Male	4985	4.38	0	2	1	Feline
Puck	Neutered Male	5562	29.38	0	2	2	Canine
Puck.Ee	Female	5137	15.38	0	2	2	Canine
Marley	Neutered Male	5466	71.19	0	1	1	Canine
Atlas	Male	4422	18.56	0	3	1	Canine
Cachet	Spayed Female	6249	5.19	0	3	1	Canine
CACHET3	Spayed Female	4422	17.7	0	3	1	Canine
Stanley	Neutered Male	9640	4.38	0	1	0	Feline
Coco	Female	5562	51	0	3	1	Canine

Species Classifier: Load the data

```
species.full = read.table
            ("../data/speciesprocesses.csv",
                            header=T, sep=",")
namefreq = as.data.frame(with(species.features,
                                   table(name)))
excludename = as.character(namefreq[namefreq$Freq < 10
badnames = as.integer(rownames(species.features[
                          species.features$name %in% excl
levels(species.features$name) = c(levels(species.featu
species.features[badnames,]$name = "Other"
species.features$name = species.features$name[drop=T]
```

Species Classifier: Reformat and split the data

```
species.features = subset(species.full,
                          select=c("age","weight", ...))
species.targets = subset(species.full,select="species"

library(caret)
set1index = createDataPartition(species.targets,
                          p=.2, list=FALSE, times=1)
species.targets.test = species.targets[set1index]
species.features.test = species.features[set1index,]
species.targets.train = species.targets[-set1index]
species.features.train = species.features[-set1index]
```

Topic

Algorithms: Naive Bayes - Overview

Rooted in probability theory and based on Bayes Theorem. The **Naive** part comes from the simplifying assumption that the features are independent.

Notation:

$$X = \text{vector of features} \quad C_j = \text{targets}$$

$$P(X) = \text{probability of obtaining the features } X$$

$$P(X|C_j) = \text{probability of obtaining } X \text{ given a value of } C_j$$

$$P(X, C_j) = \text{joint probability of } X \text{ and } C_j \text{ happening together}$$

Bayes Theorem

$$P(C_j|X) = \frac{P(X|C_j)P(C_j)}{P(X)}$$

Bayes Theorem

$$P(C_j|X) = \frac{P(X|C_j)P(C_j)}{P(X)}$$

$$posterior = \frac{likelihood \times prior}{evidence}$$

Algorithms: Naive Bayes - Small Example

Species	Weight	Sex
Canine	35	Male
Feline	8	Female
Feline	15	Female
Feline	10	Male
Canine	75	Female

The goal is to calculate the probabilities of each species given a weight and a sex.

$$P(Canine|W = a, S = b) = \frac{P(W = a, S = b|Canine)P(Canine)}{P(W = a, S = b)}$$

$$P(Feline|W = a, S = b) = \frac{P(W = a, S = b|Feline)P(Feline)}{P(W = a, S = b)}$$

Algorithms: Naive Bayes - Small Example

Species	Weight	Sex
Canine	35	Male
Feline	8	Female
Feline	15	Female
Feline	10	Male
Canine	75	Female

Training the model consists of calculating all the terms on the right hand side:

$$P(Canine|W = a, S = b) = P(W = a, S = b|Canine)P(Canine)$$

$$P(Feline|W = a, S = b) = P(W = a, S = b|Feline)P(Feline)$$

Algorithms: Naive Bayes - Small Example

Species	Weight	Sex
Canine	35	Male
Feline	8	Female
Feline	15	Female
Feline	10	Male
Canine	75	Female

Training the model consists of calculating all the terms on the right hand side:

$$P(Canine|W = a, S = b) = P(W = a, S = b|Canine)P(Canine)$$

$$P(Feline|W = a, S = b) = P(W = a, S = b|Feline)P(Feline)$$

Simplifying assumption:

$$P(W = a, S = b|Canine) = P(W = a|Canine)P(S = b|Canine)$$

$$P(W = a, S = b|Feline) = P(W = a|Feline)P(S = b|Feline)$$

Algorithms: Naive Bayes - Small Example

Species	Weight	Sex
Canine	35	Male
Feline	8	Female
Feline	15	Female
Feline	10	Male
Canine	75	Female

Training the model consists of calculating all the terms on the right hand side:

$$\text{Priors: } P(Canine) = \frac{2}{5} = 0.4 \quad P(Feline) = \frac{3}{5} = 0.6$$

Algorithms: Naive Bayes - Small Example

Species	Weight	Sex
Canine	35	Male
Feline	8	Female
Feline	15	Female
Feline	10	Male
Canine	75	Female

Training the model consists of calculating all the terms on the right hand side:

$$\text{Priors: } P(Canine) = \frac{2}{5} = 0.4 \quad P(Feline) = \frac{3}{5} = 0.6$$

$$\text{Likelihood: } P(S = female|Canine) = \frac{3}{5} = 0.6 \quad P(S = male|Canine) = \frac{2}{5}$$

Algorithms: Naive Bayes - Small Example

Species	Weight	Sex
Canine	35	Male
Feline	8	Female
Feline	15	Female
Feline	10	Male
Canine	75	Female

Training the model consists of calculating all the terms on the right hand side:

$$\text{Priors: } P(Canine) = \frac{2}{5} = 0.4 \quad P(Feline) = \frac{3}{5} = 0.6$$

$$\text{Likelihood: } P(S = female|Canine) = \frac{3}{5} = 0.6 \quad P(S = male|Canine) = \frac{2}{5}$$

$$P(W = a|Canine) = \frac{1}{\sqrt{2\pi\sigma^2_{canine}}} e^{-(a-\mu_{canine})^2/(2\sigma^2_{canine})}$$

Algorithms: Naive Bayes - Bayes Theorem

Bayes Theorem

$$P(C_j|X) = \frac{P(X|C_j)P(C_j)}{P(X)}$$

Now that we know all the terms on the right hand side, given a weight and a sex we can calculate the probabilities on the left for each class (Canine and Feline) and compare.
Suppose we want to classify an animal with weight 25 and sex male:

$$P(Canine|W = 25, S = Male) = P(Canine)P(W = 25, S = Male|Canine)$$

$$P(Feline|W = 25, S = Male) = P(Feline)P(W = 25, S = Male|Feline)$$

Species Classifier: Naive Bayes: Train, Test, Measure

Train the model

```
nbmodel = train(species.features.train,
                species.targets.train,"nb")
```

Species Classifier: Naive Bayes: Train, Test, Measure

Train the model

```
nbmodel = train(species.features.train,
                species.targets.train,"nb")
```

Test the model

```
speciesPredictions = extractPrediction(list(nbmodel),
              testX=species.features.test,
              testY=species.targets.test)
speciesPredictions = speciesPredictions[
            speciesPredictions$dataType == "Test",]
```

Species Classifier: Naive Bayes: Train, Test, Measure

Train the model

```
nbmodel = train(species.features.train,
                species.targets.train,"nb")
```

Test the model

```
speciesPredictions = extractPrediction(list(nbmodel),
              testX=species.features.test,
              testY=species.targets.test)
speciesPredictions = speciesPredictions[
            speciesPredictions$dataType == "Test",]
```

Measure the accuracy

```
confusionMatrix(speciesPredictions$pred,
                speciesPredictions$obs)
```

Species Classifier: Naive Bayes: Results

```
41502 samples
    2 predictors

Pre-processing:
Resampling: Bootstrap (25 reps)

Summary of sample sizes: 41502, 41502, 41502, 41502, 41502, 41502, ...

Resampling results across tuning parameters:

  usekernel  Accuracy  Kappa  Accuracy SD  Kappa SD
  FALSE      0.756     0.396  0.015        0.0835
  TRUE       0.779     0.463  0.00411      0.00745

Accuracy was used to select the optimal model using  the largest value.
The final value used for the model was usekernel = TRUE.
```

Species Classifier: Naive Bayes: Confusion Matrix

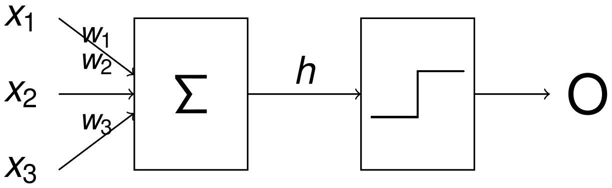
```
Confusion Matrix and Statistics

          Reference
Prediction Canine Feline
    Canine   8237   1296
    Feline   1923   3084

           Accuracy : 0.7786
             95% CI : (0.7718, 0.7853)
No Information Rate : 0.6988
P-Value [Acc > NIR] : < 2.2e-16
```

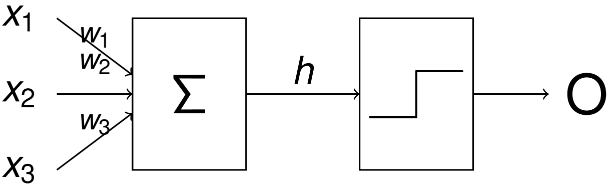
Topic

McCulloch and Pitt's Neuron Model



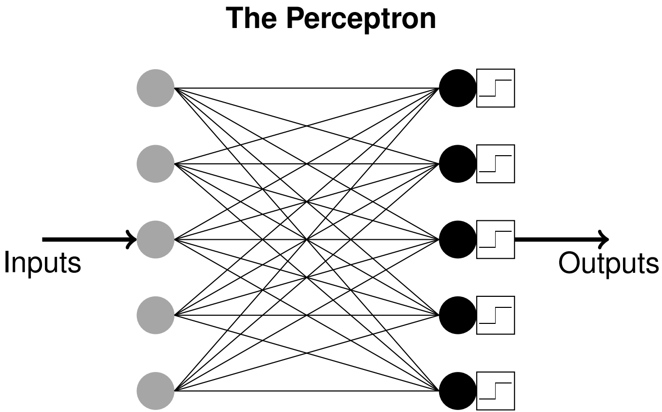
Algorithms: Artificial Neural Network - Neuron Model

McCulloch and Pitt's Neuron Model



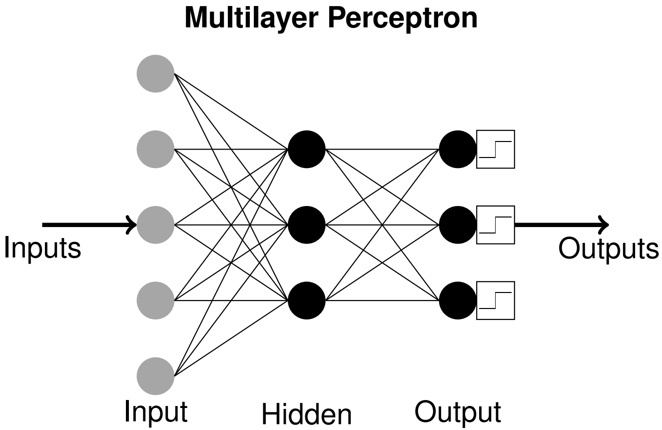
$$h = \sum_{i=1}^{n} w_i x_i, \quad O = g(h) = \begin{cases} 0 & h < \theta \\ 1 & h > \theta \end{cases}$$

Algorithms: ANN - Perceptron



One input for each feature and one output for each class in the target

Algorithms: ANN - Multilayer Perceptron



Again, one input for each feature, one output for each class in the target. There can be any number of neurons in each hidden layer and any number of hidden layers.

Species Classifier: ANN: Train, Test, Measure

Train the model

```
annmodel = train(species.features.train,
                 species.targets.train,"nnet")
```

Species Classifier: ANN: Train, Test, Measure

Train the model

```
annmodel = train(species.features.train,
                 species.targets.train,"nnet")
```

Test the model

```
speciesPredictions = extractPrediction(list(annmodel),
              testX=species.features.test,
              testY=species.targets.test)
speciesPredictions = speciesPredictions[
            speciesPredictions$dataType == "Test",]
```

Species Classifier: ANN: Train, Test, Measure

Train the model

```
annmodel = train(species.features.train,
                 species.targets.train,"nnet")
```

Test the model

```
speciesPredictions = extractPrediction(list(annmodel),
              testX=species.features.test,
              testY=species.targets.test)
speciesPredictions = speciesPredictions[
            speciesPredictions$dataType == "Test",]
```

Measure the accuracy

```
confusionMatrix(speciesPredictions$pred,
                speciesPredictions$obs)
```

Species Classifier: ANN: Results

```
58155 samples
   69 predictors

Pre-processing:
Resampling: Bootstrap (25 reps)

Summary of sample sizes: 58155, 58155, 58155, 58155, 58155, 58155, ...

Resampling results across tuning parameters:
  size  decay  Accuracy  Kappa   Accuracy SD  Kappa SD
  1     0      0.703     0.0234  0.0244       0.117
  1     1e-04  0.704     0.0426  0.0178       0.147
  1     0.1    0.787     0.444   0.0514       0.255
  3     0      0.722     0.116   0.0487       0.237
  3     1e-04  0.738     0.208   0.0533       0.283
  3     0.1    0.81      0.54    0.034        0.163
  5     0      0.726     0.159   0.0463       0.261
  5     1e-04  0.74      0.21    0.0571       0.286
  5     0.1    0.816     0.564   0.0252       0.118

Accuracy was used to select the optimal model using  the largest value.
The final values used for the model were size = 5 and decay = 0.1.
```

Species Classifier: ANN: Confusion matrix

```
Confusion Matrix and Statistics

                 Reference
          Prediction Canine Feline
             Canine   8916   1366
             Feline   1244   3014

               Accuracy : 0.8205
                 95% CI : (0.8142, 0.8267)
    No Information Rate : 0.6988
    P-Value [Acc > NIR] : < 2e-16

Final model: a 69-5-1 network with 356 weights
```

Topic

Algorithms: Random Forest - Overview

The Random Forest algorithm uses random sets of examples and features to create Decision Trees. These Decision Trees are then combined to give a predicted result.

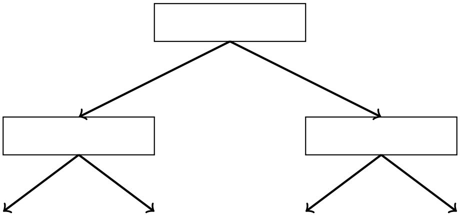
Algorithms: Random Forest - Overview

The Random Forest algorithm uses random sets of examples and features to create Decision Trees. These Decision Trees are then combined to give a predicted result.

What is a Decision Tree?

Algorithms: Random Forest - Decision Tree Overview

A Decision Tree breaks down the classification into individual decisions about each feature one by one. The classification starts from the *root* node and progresses through a set of decisions to arrive at a *leaf* node where the decision is given.



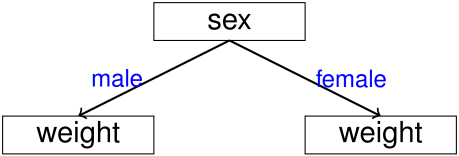
Algorithms: Random Forest - Decision Tree Example

Species	Weight	Sex
Canine	35	Male
Feline	8	Female
Feline	15	Female
Feline	10	Male
Canine	75	Female

$\boxed{\text{sex}}$

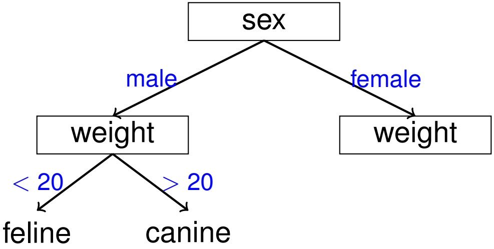
Algorithms: Random Forest - Decision Tree Example

Species	Weight	Sex
Canine	35	Male
Feline	8	Female
Feline	15	Female
Feline	10	Male
Canine	75	Female



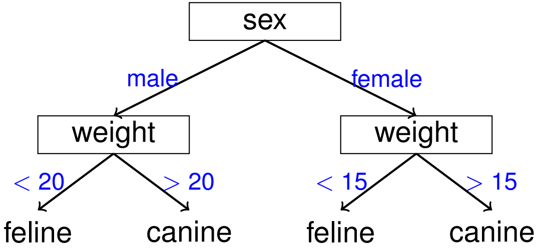
Algorithms: Random Forest - Decision Tree Example

Species	Weight	Sex
Canine	35	Male
Feline	8	Female
Feline	15	Female
Feline	10	Male
Canine	75	Female



Algorithms: Random Forest - Decision Tree Example

Species	Weight	Sex
Canine	35	Male
Feline	8	Female
Feline	15	Female
Feline	10	Male
Canine	75	Female



Algorithms: Random Forest

Training the model

1. Choose a random set of features and a random set of examples
2. Construct a decision tree using the selected subset of features and examples
3. Repeat some large number of times (randomForest in R defaults to 500)

Algorithms: Random Forest

Training the model

1. Choose a random set of features and a random set of examples
2. Construct a decision tree using the selected subset of features and examples
3. Repeat some large number of times (randomForest in R defaults to 500)

Using the model

1. Run the features through all of the decision trees produced above
2. Combine the outputs of the decision trees to produce a prediction

Species Classifier: Random Forest: Train, Test, Measure

Train the model

```
rfmodel = train(species.features.train,
                species.targets.train,"rf")
```

Species Classifier: Random Forest: Train, Test, Measure

Train the model

```
rfmodel = train(species.features.train,
                species.targets.train,"rf")
```

Test the model

```
speciesPredictions = extractPrediction(list(rfmodel),
              testX=species.features.test,
              testY=species.targets.test)
speciesPredictions = speciesPredictions[
            speciesPredictions$dataType == "Test",]
```

Species Classifier: Random Forest: Train, Test, Measure

Train the model

```
rfmodel = train(species.features.train,
                species.targets.train,"rf")
```

Test the model

```
speciesPredictions = extractPrediction(list(rfmodel),
              testX=species.features.test,
              testY=species.targets.test)
speciesPredictions = speciesPredictions[
            speciesPredictions$dataType == "Test",]
```

Measure the accuracy

```
confusionMatrix(speciesPredictions$pred,
                speciesPredictions$obs)
```

Species Classifier: Random Forest: Results

```
58155 samples
   69 predictors

Pre-processing:
Resampling: Bootstrap (25 reps)

Summary of sample sizes: 58155, 58155, 58155, 58155, 58155, 58155, ...

Resampling results across tuning parameters:

  mtry  Accuracy  Kappa  Accuracy SD  Kappa SD
  2     0.725     0.123  0.0056       0.026
  35    0.837     0.615  0.00235      0.00588
  69    0.83      0.599  0.00252      0.00601

Accuracy was used to select the optimal model using  the largest value.
The final value used for the model was mtry = 35.
```

Species Classifier: Random Forest: Confusion Matrix

```
Confusion Matrix and Statistics

          Reference
Prediction Canine Feline
    Canine   8986   1166
    Feline   1174   3214

               Accuracy : 0.8391
                 95% CI : (0.833, 0.845)
    No Information Rate : 0.6988
    P-Value [Acc > NIR] : <2e-16

Final model:
Number of trees: 500
No. of variables tried at each split: 35
```

Topic

Summary Comparison of the Models

Algorithm	Time To Train (min)	Time to Predict (sec)	Accuracy
Naive Bayes	10+ hours	86.542	0.7786
ANN	115.08	3.221	0.8205
Random Forest		18.539	0.8391

Measurements were taken using R running on an Amazon EC2 Large instance (7.5 GB of memory, 4 EC2 Compute Units (2 virtual cores with 2 EC2 Compute Units each), 850 GB of local instance storage, 64-bit platform)

Next steps for the Species Classifier

- Get more data
- Look for other features
- Try other algorithms and validation methods
- Utilize the species labels from the data under prediction

Topic

Links

- Code and slides for this talk: http://bit.ly/f8ce6f
- My machine learning bookmarks: http://bit.ly/ebRPT1
- R stats software package: http://www.r-project.org
- RStudio GUI: http://www.rstudio.org
- Caret R package: http://caret.r-forge.r-project.org
- Machine Learning competitions: http://www.kaggle.com
- Iain Murray's "Introduction to Machine Learning Videos": http://bit.ly/fSg4rG
- Andrew Ng's Stanford Machine Learning course: http://bit.ly/fvafuI

Recommended reading

- "Machine Learning. An Algorithmic Perspective", Stephan Marsland
- "Programming Collective Intelligence", Toby Segaran
- "Data Analysis with Open Source Tools", Philipp Janert
- "Elements of Statistical Learning", Hastie, et. al. (http://bit.ly/eq74Ct)
- "Machine Learning", Tom Mitchell
- "Pattern Matching and Machine Learning", Chris Bishop