Seq.Rmd

Jeff B

May 24, 2019

Sequence Question

This question is a sequence classification question. Considering the large number of variables, however, we can start by examining it as a straightforward classification problem. This will allow us some of the benefits of straightforward classification, most importantly variable selection. This part was done in R. The sequence classification portion was done in Python, and can be found in the seq folder as seq.py

Start by importing the necessary libraries and setting the seed for reproducibility

```
## Loading required package: Matrix
```

```
## Loading required package: foreach
```

```
## Loaded glmnet 2.0-16
```

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

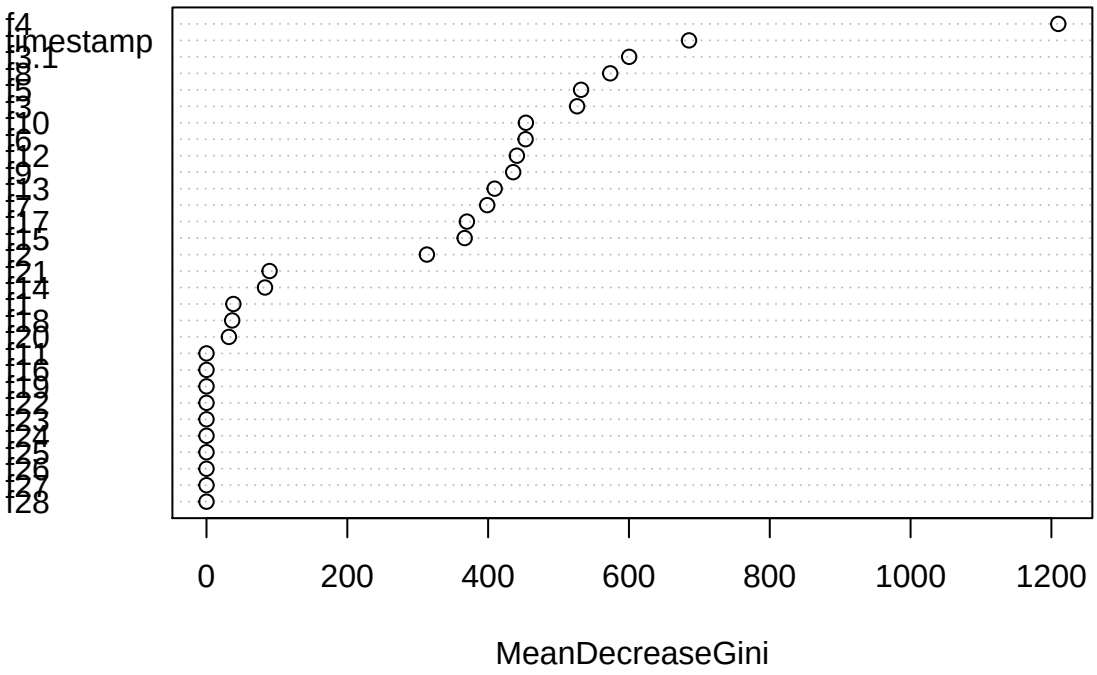
```
## Type rfNews() to see new features/changes/bug fixes.
```

Next, we read in the data, and generate a RandomForest model for this data. RandomForests have built in cross-validation via Out-of-Bag estimates, so we don't need to do any extra cross-validation. We can also look at the variable importance plot here to determine the most important variables. Somewhat arbitrarily, we'll set a cutoff at 200 MeanDecreaseGini. This leaves us with 15 relevant variables.

```
sample <- read.csv("seq/sample.csv")
sample$class = factor(sample$class)

#Use a random forest to generate a model to predict class
rf <- randomForest(factor(class)~., data=data.frame(sample[,-1]))

#Now look at the Variable Importance Plot of the Random Forest to determine relevant variables to the m
varImpPlot(rf)
```



For comparison, we'll run a logistic regression using only variables above the threshold

```
simdat <- sample[,-c(1,4,24,17,21,23,14,19,22,25,26,27,28,29,30,31)]

#now using only those, we can run a logistic regression
simlog <- glm(class ~ ., family="binomial", data=simdat)
```

Now we'll get the confusion matrix for both

```
#First the randomForest
table(predict(rf, newdata=sample[,-1]), sample$class)
```

```
##
##         0     1
##   0 10100   419
##   1     1  7897
```

```
#Then the logistic regression
table(predict(simlog, type="response") > 0.5, sample$class)
```

```
##
##            0    1
##   FALSE 8221 3388
##   TRUE  1880 4928
```

The randomForest model perdorms better for predictions, so given data about the next event, we would use the randomForest model to predict the class. Considering we don't have that information, this process has still been useful, since we will use only those variables we've deemed relevant in our sequence prediction problem.