Feedback — Quiz 4 - **Please Note: No Grace Period**

Help Center

Thank you. Your submission for this quiz was received.

You submitted this quiz on **Tue 26 May 2015 9:25 PM PDT**. You got a score of **12.00** out of **15.00**. You can attempt again, if you'd like.

Question 1

Load the vowel.train and vowel.test data sets:

```
library(ElemStatLearn)
data(vowel.train)
data(vowel.test)
```

Set the variable y to be a factor variable in both the training and test set. Then set the seed to 33833. Fit (1) a random forest predictor relating the factor variable y to the remaining variables and (2) a boosted predictor using the "gbm" method. Fit these both with the train() command in the caret package.

What are the accuracies for the two approaches on the test data set? What is the accuracy among the test set samples where the two methods agree?

Your Answer		Score	Explanation
(•) RF Accuracy = 0.6061 GBM Accuracy = 0.5325 Agreement Accuracy = 0.6518	✔	3.00	
( ) RF Accuracy = 0.6061 GBM Accuracy = 0.6518 Agreement Accuracy = 0.5325			
( ) RF Accuracy = 0.9881 GBM Accuracy = 0.5325 Agreement Accuracy = 0.9973			

○ RF Accuracy = 0.6518 GBM Accuracy = 0.5325 Agreement Accuracy = 0.5325	
Total	3.00 / 3.00

Question 2

Load the Alzheimer's data using the following commands

```
library(caret)
library(gbm)
set.seed(3433)
library(AppliedPredictiveModeling)
data(AlzheimerDisease)
adData = data.frame(diagnosis,predictors)
inTrain = createDataPartition(adData$diagnosis, p = 3/4)[[1]]
training = adData[ inTrain,]
testing = adData[-inTrain,]
```

Set the seed to 62433 and predict diagnosis with all the other variables using a random forest ("rf"), boosted trees ("gbm") and linear discriminant analysis ("lda") model. Stack the predictions together using random forests ("rf"). What is the resulting accuracy on the test set? Is it better or worse than each of the individual predictions?

Your Answer		Score	Explanation
◉ Stacked Accuracy: 0.76 is better than random forests and boosting, but not lda.	✖	0.00	
○ Stacked Accuracy: 0.79 is better than random forests and lda and the same as boosting.			
○ Stacked Accuracy: 0.88 is better than all three other methods			
○ Stacked Accuracy: 0.79 is worse than all the other methods.			
Total		0.00 / 3.00	

Question 3

Load the concrete data with the commands:

```
set.seed(3523)
library(AppliedPredictiveModeling)
data(concrete)
inTrain = createDataPartition(concrete$CompressiveStrength, p = 3/4)[[1]]
training = concrete[ inTrain,]
testing = concrete[-inTrain,]
```

Set the seed to 233 and fit a lasso model to predict Compressive Strength. Which variable is the last coefficient to be set to zero as the penalty increases? (Hint: it may be useful to look up ? plot.enet).

Your Answer		Score	Explanation
○ Water			
○ CoarseAggregate			
◉ Cement	✔	3.00	
○ BlastFurnaceSlag			
Total		3.00 / 3.00	

Question 4

Load the data on the number of visitors to the instructors blog from here:

https://d396qusza40orc.cloudfront.net/predmachlearn/gaData.csv

Using the commands:

```
library(lubridate)  # For year() function below
dat = read.csv("~/Desktop/gaData.csv")
training = dat[year(dat$date) < 2012,]
testing = dat[(year(dat$date)) > 2011,]
tstrain = ts(training$visitsTumblr)
```

Fit a model using the bats() function in the forecast package to the training time series. Then forecast this model for the remaining time points. For how many of the testing points is the true value within the 95% prediction interval bounds?

Your Answer		Score	Explanation
(•) 96%	✔	3.00	
( ) 93%			
( ) 92%			
( ) 94%			
Total		3.00 / 3.00	

Question 5

Load the concrete data with the commands:

```
set.seed(3523)
library(AppliedPredictiveModeling)
data(concrete)
inTrain = createDataPartition(concrete$CompressiveStrength, p = 3/4)[[1]]
training = concrete[ inTrain,]
testing = concrete[-inTrain,]
```

Set the seed to 325 and fit a support vector machine using the e1071 package to predict Compressive Strength using the default settings. Predict on the testing set. What is the RMSE?

Your Answer		Score	Explanation
( ) 107.44			
( ) 11543.39			
( ) 45.09			
(•) 6.72	✔	3.00	
Total		3.00 / 3.00	