```
##Make predictions on New Data
library(caret)
library(mlbench)
library(MASS)
#Load dataset
data("PimaIndiansDiabetes")
#Create 80%/20% for training and validation dataset
set.seed(9)
validationIndex=createDataPartition(PimaIndiansDiabetes$diabetes,p=0.8
, list=FALSE)
validation=PimaIndiansDiabetes[-validationIndex,]
training=PimaIndiansDiabetes[validationIndex,]
#Train a model and summarize model
set.seed(9)
trainControl=trainControl(method="cv", number=10)
fit.lda=train(diabetes~., data=training,
method="lda",metric="Accuracy", trControl=trainControl)
print(fit.lda)
print(fit.lda$finalModel)
#Estimate skill on validation dataset
set.seed(9)
predictions=predict(fit.lda, newdata=validation)
confusionMatrix(predictions, validation$diabetes)

##Create a standalone model
library(caret)
library(mlbench)
library(randomForest)
#Load dataset
data("Sonar")
set.seed(7)
#Create 80%/20% for training and validation datasets
validationIndex=createDataPartition(Sonar$Class, p=0.8, list=FALSE)
validation=Sonar[-validationIndex,]
training=Sonar[validationIndex,]
#Train a model and summarize model
set.seed(7)
trainControl=trainControl(method="repeatedcv", number=10, repeats=3)
fit.rf=train(Class~., data = training, method="rf", metric="Accuracy",
trControl=trainControl, ntree=2000)
print(fit.rf)
print(fit.rf$finalModel)
#Create standalone model using all training data
set.seed(7)
finalModel=randomForest(Class~., training, mtry=2, ntree=2000)
#Make a predictions on "new data" using the final model
finalPredictions=predict(finalModel, validation[,1:60])
confusionMatrix(finalPredictions, validation$Class)

##Save and load your model
```

```
#Load packages
library(caret)
library(mlbench)
library(randomForest)
#Load dataset
data("Sonar")
set.seed(7)
#Create 80%/20% for training and validation datasets
validationIndex=createDataPartition(Sonar$Class, p=0.8, list=FALSE)
validation=Sonar[-validationIndex,]
training=Sonar[validationIndex,]
#Create final standalone model using all training data
set.seed(7)
finalModel=randomForest(Class~., training, mtry=2, ntree=2000)
#Save model to disk
saveRDS(finalModel,"./finalModel.rds")
#Load the model
superModel=readRDS("./finalModel.rds")
print(superModel)
#Make a predictions on "new data" using the final model
finalPredictions=predict(superModel, validation[,1:60])
confusionMatrix(finalPredictions, validation$Class)
```