Machine Learning Prediction Project

NK

Thursday, August 20, 2015

Summary

The following report describes the procedure to create a machine learning algorithm that predicts the effectivness of excersises from raw accelemrometer data. The report covers (i) data importing, cleaning and pre-processing, (ii) feature selection, (iii) model selection and training, and (vi) prediction. The classification algorithm was a bagged tree model. A k-fold cross-validation with k=10 folds was used during training to avoid overfitting. The resulting accuracies for the training, validation, and test set were 0.971, 0.978, and 0.977, respectively. A pdf file was also submitted along with the HTML file report for easier reading.

Data Cleaning and Pre-Processing

First, we load the data and import the training data. Also set the seed so that the results become reproducible.

```
set.seed(12345)
library(caret)
df=read.csv("pml-training.csv")
str(df) use this command to see all the columns
```

The str() command shows us that there are a lot of NA values. We have to get rid of those columns. Also there are a lot of blank values (""), so we get rid of those columns too.

```
#Remove NA columns
nadims=is.na(df) #Find na values in df
outvar=colSums(nadims)>19000 #find columsn those with larger than 19000 na entries
df=df[,!outvar]
sum(is.na(df)) #verify absence of NA values

#Remove "" columns, since they have a low of empty values
outvar=colSums(df=="")>19000
df=df[,!outvar]
```

We have a training and a testing data-set file. The testing data set does not include any annotations and is for the course submission. We will have to take the training file data and divide it into a training set, a validation set, and a test set with proportions of 60%, 20%, and, 20% respectively. Training will occur on the training set. Initial testing will occur on the validation set to compare multiple models that might have been fit. The test set will be used as a final say of the algorithms' prediction accuracy.

```
#Create the following partition: train:60%,test=20%, valid=20%
inTrain=createDataPartition(y=df$classe, p=0.2, list=FALSE,times=2)
dftrain=df[-inTrain,]
dftest=df[inTrain[,1],]
dfvalid=df[inTrain[,2],]
```

Feature Selection

The data set has 59 variables and one output variable. Not all of the variables will contribute to the output and extra variables will slow down the traning. We will remove highly cross-correlated variables. If two variables are cross-correlated, then the information cotained within them is very similar. One of those variables will be redundant. We will use a cut-off value of 0.8 to quantify if a variable should be removed.

```
#Remove highly correlated variables
correlationMatrix <- cor(dftrain[,7:59]) #find coorelation matrix of numeric variables
highlyCorrelated <- findCorrelation(correlationMatrix, cutoff=0.8)
dftrain=dftrain[,-(highlyCorrelated+6)] #remove correlated variables from training set
```

Model Training with Cross Validation

First, we try to use bagged trees as a learning algorithm for classification. Bagged trees are fairly accurate and we will try them first. We will use the “B=5”" training parameter, which means that there should be 5 bagging iterations. We will only use the numerical variables and the “user_name”-variable" for prediction. We will also use the “allowParallel=TRUE”" option to take advantage of multi-core processors. To avoid over-fitting to the training data we also use k-fold cross-validation with k-10 folds.

```
#Train model with the "treebag" method"
modFit2=train(classe~., data=dftrain[,8:47], method="treebag", B=5,
              train_control = trainControl(method="cv", number=10),
              prox=TRUE,allowParallel=TRUE)
```

The actual training time was about 1 hour on a quadcore 2.3 GHz computer. The training set error was 0.971 according to the modFit object. That accuracy is a balanced accuracy of the 5 classes that the model predicts.

Validation and Test Set Accuracy

First, let’s predict the validation set error and view the accuracy

```
pred=predict(modFit,newdata=dfvalid) #predict the values from test set
confusionMatrix(data = pred, dfvalid$classe) #show accuracy
```

The validation set accuracy is 0.978 on 3927 entries. If we had multiple algorithms that modeled the data that we wanted to compare, we could use the validation set to compare them and possibly tune a few parameters and re-train. The training for the bagged tree method took too long, so only one learning algorithm was considered in this report.

Now, we can see the test set error with the following commands.

```
pred=predict(modFit,newdata=dftest) #predict the values from test set
confusionMatrix(data = pred, dftest$classe) #show accuracy
```

Te accuracy for the test set is 0.977, which is comparable to the validation set. Based on the test and validation set error, it is save to assume that the assignment should get either all 20 correct or make one misclassification.

Load and Process the Submission Data

Finally, we can make a prediction on the submission data:

```
#Make submission data
dft=read.csv("pml-testing.csv")
pred=predict(modFit, newdata=dft)
pml_write_files(pred)
```

The submission got 19 out of 20 classification correct, which is roughly in line with the test and validation set accuracies.