Prediction Assignment Writeup

Libraries

```
library("caret")
library("rpart")
library("tree")
library(randomForest)
```

Load Data

```
> train<- read.csv("E://pml-training.csv",na.strings=c("NA","","#DIV/0!"))
> dim(train)
[1] 19622     160
> test<- read.csv("E://pml-testing.csv",na.strings=c("NA","","#DIV/0!"))
> dim(test)
[1]    20 160
```

Tidy the Data

Remove variables which have an excess number of NA values.

```
> extraneous = c('X', 'user_name', 'raw_timestamp_part_1', 'raw_timestamp_part_2', 'cvtd_timestamp', 'new_window', 'num_window')
> training_ext <- training[, -which(names(training) %in% extraneous)]
> dim(training_ext)
[1] 19622       53
```

Explore and Preprocess Data

Check low variance variables.

```
> near_ZV= nearZeroVar(training_ext[sapply(training_ext, is.numeric)], saveMetrics = TRUE)
> train_nzv = training_ext[,near_ZV[, 'nzv']==0]
> dim(train_nzv)
[1] 19622       53
```

Remove variables with high correlation

```
> cor_matrix <- cor(na.omit(train_nzv[sapply(train_nzv, is.numeric)]))
> remove_corr<-findCorrelation(cor_matrix, cutoff = .90, verbose = TRUE)
Compare row 10    and column    1 with corr    0.992
   Means:    0.27 vs 0.168 so flagging column 10
```

```
Compare row 1  and column  9 with corr  0.925
  Means:  0.25 vs 0.164 so flagging column 1
Compare row 9  and column  4 with corr  0.928
  Means:  0.233 vs 0.161 so flagging column 9
Compare row 8  and column  2 with corr  0.966
  Means:  0.245 vs 0.157 so flagging column 8
Compare row 19  and column  18 with corr  0.918
  Means:  0.091 vs 0.158 so flagging column 18
Compare row 46  and column  31 with corr  0.914
  Means:  0.101 vs 0.161 so flagging column 31
Compare row 46  and column  33 with corr  0.933
  Means:  0.083 vs 0.164 so flagging column 33
All correlations <= 0.9
> training_corr<-training_nzv[,-remove_corr]
> dim(training_corr)
[1] 19622    46
```

Cross Validation

```
> training_dataPart <- createDataPartition(y=training_corr$classe, p=0.7, list=FALSE)
> training <- training_corr[training_dataPart,]; testing <- training_corr[-training_dataPart,]
> dim(training)
[1] 13737    46
> dim(testing)
[1] 5885   46
> set.seed(5555555)
> tree_training = tree(classe~.,data=training)
> tree_prune <- train(classe ~ .,method="rpart",data=training)
> tree_predict = predict(tree_prune,testing)
> pred_matrix = with(testing,table(tree_predict,classe))
> sum(diag(pred_matrix))/sum(as.vector(pred_matrix))
[1] 0.4927782
> set.seed(555555)
> training_ranForest = randomForest(classe~.,data=training,ntree=100, importance=TRUE)
> training_ranForest
Call:
 randomForest(formula = classe ~ ., data = training, ntree = 100,          importance = TRUE)
                Type of random forest: classification
                      Number of trees: 100
No. of variables tried at each split: 6
        OOB estimate of  error rate: 0.6%
Confusion matrix:
     A    B    C    D    E class.error
```

```
A 3901        4        0        0        1 0.001280082
B      14 2637        7        0        0 0.007900677
C        1      13 2379        3        0 0.007095159
D        0        0      31 2219        2 0.014653641
E        0        0        1        6 2518 0.002772277
> tree_predict = predict(training_ranForest,testing,type="class")
> pred_matrix = with(testing,table(tree_predict,classe))
> sum(diag(pred_matrix))/sum(as.vector(pred_matrix))
[1] 0.9937128
```

conclusion

```
> predict <- predict(training_ranForest, test)
> predict
 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
 B  A  B  A  A  E  D  B  A  A  B  C  B  A  E  E  A  B  B  B
Levels: A B C D E
```

Inserting this data in the Quiz, it provides a 100% scoring