Executive Summary:

The objective of this machine learning exercise is to create a prediction model for the 'Classe' variable using a combination of the other variables.
Analysis was performed using two types of algorithms – Decision Tree and Random Forests. The Random Forests approach provided better results with almost perfect sensitivity, specificity, positive and negative predictive value.
Thus, the Random Forests model is used as the model of preference for this classification exercise.

Details:
The following pre-steps were performed towards this analysis.
1. The training and testing data sets were retrieved from the given URL.
2. Data cleansing
a. The identification was removed to nullify it's impact on the model
b. Variables with no variation and too many NA'swere removed

Thereafter, the training data was used to create two models,
1. Decision Tree based Model
2. Random Forest based Model

The models were trained using the training data sets.
Thereafter the models were tested for the following metrics using a confusionMatrix.

- Sensitivity
- Specificity
- Positive predictive value
- Negative predictive value

Based on the values of the above metrics, the 'Random Forest' model is deemed more appropriate and accurate for prediction.

Procedure:
Getting in the training set –

The source files "pml-training.csv" and "pml-testing.csv" were massaged per the steps above to create a refined training data set 'Training.csv' and a refined testing set 'Testing.csv'
The modified data sets were then imported into R.

```
> Train=read.csv("Training.csv",header=TRUE)
> Test=read.csv("Testing.csv",header=TRUE)
```

Method 1: Decision Tree

The following step was done to create a decision tree based prediction model.

```
modelDT<-rpart(classe~.,Train)
> modelDT
n= 19622

node), split, n, loss, yval, (yprob)
      * denotes terminal node

   1) root 19622 14042 A (0.28 0.19 0.17 0.16 0.18)
     2) roll_belt< 130.5 17977 12411 A (0.31 0.21 0.19 0.18 0.11)
       4) pitch_forearm< -33.95 1578    10 A (0.99 0.0063 0 0 0) *
       5) pitch_forearm>=-33.95 16399 12401 A (0.24 0.23 0.21 0.2 0.12)
        10) magnet_dumbbell_y< 439.5 13870  9953 A (0.28 0.18 0.24 0.19 0.11)
          20) roll_forearm< 123.5 8643  5131 A (0.41 0.18 0.18 0.17 0.061)
            40) magnet_dumbbell_z< -27.5 2913   969 A (0.67 0.21 0.013 0.077 0.03)
              80) roll_forearm>=-136.5 2429   537 A (0.78 0.17 0.014 0.027 0.0062) *
              81) roll_forearm< -136.5 484   290 B (0.11 0.4 0.01 0.33 0.15) *
            41) magnet_dumbbell_z>=-27.5 5730  4162 A (0.27 0.17 0.27 0.21 0.076)
              82) num_window< 241.5 1332   220 A (0.83 0.00075 0 0.062 0.1) *
              83) num_window>=241.5 4398  2849 C (0.1 0.22 0.35 0.26 0.068)
               166) accel_dumbbell_y>=-40.5 3749  2628 D (0.12 0.25 0.25 0.3 0.08)
                 332) roll_belt>=125.5 898   368 C (0 0.37 0.59 0.038 0.0045)
                   664) pitch_belt< -42.65 347    29 B (0 0.92 0.0029 0.069 0.012) *
                   665) pitch_belt>=-42.65 551    22 C (0 0.022 0.96 0.018 0) *
                 333) roll_belt< 125.5 2851  1764 D (0.16 0.21 0.15 0.38 0.1)
                   666) num_window< 278.5 323   156 C (0 0.48 0.52 0 0)
                    1332) num_window< 260 156     0 B (0 1 0 0 0) *
                    1333) num_window>=260 167     0 C (0 0 1 0 0) *
                   667) num_window>=278.5 2528  1441 D (0.18 0.17 0.1 0.43 0.12)
                    1334) pitch_belt< -42.45 537   328 A (0.39 0.34 0.18 0.08 0.011) *
                    1335) pitch_belt>=-42.45 1991   947 D (0.12 0.13 0.08 0.52 0.15) *
               167) accel_dumbbell_y< -40.5 649    51 C (0 0.046 0.92 0.032 0) *
          21) roll_forearm>=123.5 5227  3500 C (0.077 0.18 0.33 0.23 0.18)
            42) magnet_dumbbell_y< 290.5 3047  1569 C (0.093 0.13 0.49 0.15 0.14)
              84) magnet_forearm_z< -251 238    49 A (0.79 0.071 0 0.046 0.088) *
              85) magnet_forearm_z>=-251 2809  1331 C (0.033 0.14 0.53 0.16 0.15)
               170) num_window< 88.5 344   156 B (0.099 0.55 0 0 0.35) *
               171) num_window>=88.5 2465   987 C (0.024 0.082 0.6 0.18 0.12) *
            43) magnet_dumbbell_y>=290.5 2180  1430 D (0.056 0.24 0.11 0.34 0.25)
              86) accel_forearm_x>=-101.5 1398   923 E (0.051 0.3 0.16 0.15 0.34)
               172) roll_dumbbell< 40.19426 273    63 B (0.051 0.77 0.011 0.062 0.11) *
               173) roll_dumbbell>=40.19426 1125   679 E (0.051 0.19 0.19 0.17 0.4) *
              87) accel_forearm_x< -101.5 782   237 D (0.066 0.12 0.036 0.7 0.077) *
        11) magnet_dumbbell_y>=439.5 2529  1243 B (0.032 0.51 0.043 0.22 0.19)
          22) num_window>=258.5 1928   642 B (0.042 0.67 0.056 0.14 0.097)
            44) roll_belt>=-0.58 1781   495 B (0.045 0.72 0.061 0.15 0.022) *
            45) roll_belt< -0.58 147     0 E (0 0 0 0 1) *
          23) num_window< 258.5 601   299 D (0 0 0 0.5 0.5)
            46) pitch_belt>=13.95 313    11 D (0 0 0 0.96 0.035) *
            47) pitch_belt< 13.95 288     0 E (0 0 0 0 1) *
     3) roll_belt>=130.5 1645    14 E (0.0085 0 0 0 0.99) *
```

The same was then used to perform prediction.

```
> predict(modelDT,Test)
               A            B           C             D                E
1   0.09883721 0.5465116279 0.00000000 0.00000000 0.354651163
2   0.77892137 0.1745574310 0.01358584 0.02675998 0.006175381
3   0.05066667 0.1920000000 0.19377778 0.16711111 0.396444444
4   0.83483483 0.0007507508 0.00000000 0.06156156 0.102852853
5   0.83483483 0.0007507508 0.00000000 0.06156156 0.102852853
6   0.02393509 0.0819472617 0.59959432 0.17768763 0.116835700
7   0.06649616 0.1240409207 0.03580563 0.69693095 0.076726343
8   0.12405826 0.1265695630 0.07985937 0.52435962 0.145153189
9   0.99366286 0.0063371356 0.00000000 0.00000000 0.000000000
```

```
10 0.77892137 0.1745574310 0.01358584 0.02675998 0.006175381
11 0.02393509 0.0819472617 0.59959432 0.17768763 0.116835700
12 0.05066667 0.1920000000 0.19377778 0.16711111 0.396444444
13 0.02393509 0.0819472617 0.59959432 0.17768763 0.116835700
14 0.99366286 0.0063371356 0.00000000 0.00000000 0.000000000
15 0.05066667 0.1920000000 0.19377778 0.16711111 0.396444444
16 0.12405826 0.1265695630 0.07985937 0.52435962 0.145153189
17 0.83483483 0.0007507508 0.00000000 0.06156156 0.102852853
18 0.10743802 0.4008264463 0.01033058 0.33057851 0.150826446
19 0.10743802 0.4008264463 0.01033058 0.33057851 0.150826446
20 0.05128205 0.7692307692 0.01098901 0.06227106 0.106227106
```

Method 2: Random Forest

The following step was done to create a random forest based prediction model.

```
> modelRF<-randomForest(classe~.,Train)
> modelRF

Call:
 randomForest(formula = classe ~ ., data = Train)
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 6

        OOB estimate of  error rate: 0.18%
Confusion matrix:
     A    B    C    D    E  class.error
A 5579    1    0    0    0 0.0001792115
B    3 3792    2    0    0 0.0013168291
C    0    7 3414    1    0 0.0023378141
D    0    0   14 3201    1 0.0046641791
E    0    0    0    7 3600 0.0019406709
```

The same was then used to perform prediction.

```
> predict(modelRF,Test)
 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
```

Sample error:

The error rate for the random forest approach is minimal with a maximum of 0.46% error across the classes.

Thus there is a less than 1% chance of expecting an incorrect classification with an out of sample observation with this method.

Result:

The **random forest** approach yielded much better prediction results and would be the preferred model for predicting ‘classe’.