Background:

Using devices such as *Jawbone Up*, *Nike FuelBand*, and *Fitbit* it is now possible to collect a large amount of data about personal activity relatively inexpensively. These types of devices are part of the quantified self-movement – a group of enthusiasts who take measurements about themselves regularly to improve their health, to find patterns in their behavior, or because they are tech geeks. One thing that people regularly do is quantify how *much* of a particular activity they do, but they rarely quantify *how well they do it*. In this project, your goal will be to use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants. They were asked to perform barbell lifts correctly and incorrectly in 5 different ways. More information is available from the website here: http://web.archive.org/web/20161224072740/http:/groupware.les.inf.puc-rio.br/har (see the section on the Weight Lifting Exercise Dataset).

Goal: To build a model to predict the way the exercise was performed which is given in the outcome variable/column "classe" in the data set

High Level Approach:

Partition & sub-split the training data (pml-training.csv) into training and testing to build a model, cross-validate using the testing data (part of training dataset), identify a better classifier to build model, evaluate the need to stack or aggregate multiple models to improve the accuracy and finally, predict using the model with better accuracy factor / with less sample errors over the testing sample (remember we are still in the training data set).

Then use the model to validate against the out-of-sample data created from (pml-testing.csv). We will name with a variable name - the former *training* and latter *validation.*

Detailed Steps:

1. Download, load and partition the training data

```
trainD <- read.csv(pml-training.csv,header = TRUE)
validationD <- read.csv(pml-testing.csv, header = TRUE)

set.seed(12345)
inTrain <- createDataPartition(y=trainD$classe, p=0.7, list = FALSE)
training <- trainD[inTrain,]
testing <- trainD[-inTrain,]
```

2. Data cleanup

With 160 fields in the dataset, a) let us eliminate that are non-predictors, b) that may have nothing but NAs c) have not many unique values. NA cleanup step is not needed if the predictability of the out-of-sample data will be impacted, but for this project we will create a cleaner data set.

```
trainNZV <- nearZeroVar(training);

training <- training[,-trainNZV] ; testing <- testing[,-trainNZV]; validation <-
validation[,trainNZV]

naColumns <- sapply(names(training), function(x) all(is.na(training[,x]) == TRUE))
training <- training[, naColumns==FALSE]

testing <- testing[, naColumns==FALSE]
validation <- validation[, naColumns==FALSE]
```

3. Build Model

Using the training data pruned, let us build a model with classe as the outcome and all other attributes as predictors. We are choosing tree classifier, random forests and gradient boosting to either stack of compare the model accuracies.

```
modelFit_rpart <- train(classe ~., method = "rpart", data = na.exclude(training))

modelFit_rf <- train(classe ~., method = "rf", data = na.exclude(training),
prox=TRUE)

modelFit_gbm <- train(classe ~., method = "gbm", data = na.exclude(training),
verbose= FALSE)
```

4. Predict & Identify the model with best accuracy

```
pred_rpart <- predict(modelFit_rpart,newdata = na.exclude(testing))

pred_rf <- predict(modelFit_rf,newdata = na.exclude(testing))

pred_gbm <- predict(modelFit_gbm,newdata = na.exclude(testing))
```

To identify which model gives the best accuracy given the sample data, use the accuracy results of each prediction.

```
> mean(modelFit_gbm$results$Accuracy)
[1] 0.9909355

> mean(modelFit_rpart$results$Accuracy)
[1] 0.5414623

> mean(modelFit_rf$results$Accuracy)
[1] 0.9211933
```

Based on the accuracy results, gbm based modeling yields a better accuracy results. Hence, I would choose gbm model to take it up for out-of-sample validation

5. Validate with out-of-sample

Using the gbm based model let us evaluate the validation data (out of sample). And create a table to cross-check the prediction accuracy.

```
pred_oos <- predict(modelFit_gbm,newdata = na.exclude(validation))
```

Following values get displayed when printing pred_oos

- 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

Conclusion: This concludes the modeling and prediction of the data sets given. I tried to do modeling with and without cleaning NA values. Because validation data columns have NAs for many columns, doing the modeling on training data w/o NA removal gave a better accuracy results but was not significant.