Practical Machine Learning - Write-Up Project

Carlos Alberto Guevara Díez

November, 2015

The goal of this project is to predict the manner in which people did the exercise. This is the "classe" variable in the training set. I created this report to describe how I built the model, how I used cross validation, what I think the expected out of sample error is. There is also the script for the 20 predictions.

Data Processing and Analysis

The training and testing datasets are avaiable here:

Training dataset: https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv

Testing dataset: https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv

Load all the librarys necesary to run the project.

```
library(caret)
```

```
## Loading required package: lattice
## Loading required package: ggplot2
```

```
library(ggplot2)
library(corrplot)
library(randomForest)
```

```
## randomForest 4.6-12
## Type rfNews() to see new features/changes/bug fixes.
```

assing the data from pml-training.csv and pml-testing.csv files to the variables.

```
setwd("C:/Users/220194/Documents/Data Science Specialization/08 Practical Machine Learning/PracticalMach
training <- read.csv("pml-training.csv",na.strings = c("NA", ""))
test <- read.csv("pml-testing.csv",na.strings = c("NA", ""))
```

We search in the variables those columns that dont contibute to the analysis because they are character or non-numerical.

```
sapply(test,class)
```

```
##                        X                user_name     raw_timestamp_part_1
##                "integer"                 "factor"                "integer"
##     raw_timestamp_part_2           cvtd_timestamp               new_window
##                "integer"                 "factor"                 "factor"
##               num_window                roll_belt               pitch_belt
##                "integer"                "numeric"                "numeric"
##                 yaw_belt         total_accel_belt       kurtosis_roll_belt
##                "numeric"                "integer"                "logical"
##      kurtosis_picth_belt        kurtosis_yaw_belt       skewness_roll_belt
##                "logical"                "logical"                "logical"
##     skewness_roll_belt.1        skewness_yaw_belt            max_roll_belt
##                "logical"                "logical"                "logical"
##           max_picth_belt             max_yaw_belt            min_roll_belt
##                "logical"                "logical"                "logical"
##           min_pitch_belt             min_yaw_belt      amplitude_roll_belt
##                "logical"                "logical"                "logical"
##     amplitude_pitch_belt       amplitude_yaw_belt     var_total_accel_belt
##                "logical"                "logical"                "logical"
##            avg_roll_belt         stddev_roll_belt            var_roll_belt
##                "logical"                "logical"                "logical"
##           avg_pitch_belt        stddev_pitch_belt           var_pitch_belt
##                "logical"                "logical"                "logical"
##             avg_yaw_belt          stddev_yaw_belt             var_yaw_belt
##                "logical"                "logical"                "logical"
##             gyros_belt_x             gyros_belt_y             gyros_belt_z
##                "numeric"                "numeric"                "numeric"
##             accel_belt_x             accel_belt_y             accel_belt_z
##                "integer"                "integer"                "integer"
##            magnet_belt_x            magnet_belt_y            magnet_belt_z
##                "integer"                "integer"                "integer"
##                 roll_arm                pitch_arm                  yaw_arm
##                "numeric"                "numeric"                "numeric"
##          total_accel_arm            var_accel_arm             avg_roll_arm
##                "integer"                "logical"                "logical"
##          stddev_roll_arm             var_roll_arm            avg_pitch_arm
##                "logical"                "logical"                "logical"
##         stddev_pitch_arm            var_pitch_arm              avg_yaw_arm
##                "logical"                "logical"                "logical"
##           stddev_yaw_arm              var_yaw_arm              gyros_arm_x
##                "logical"                "logical"                "numeric"
##              gyros_arm_y              gyros_arm_z              accel_arm_x
##                "numeric"                "numeric"                "integer"
##              accel_arm_y              accel_arm_z             magnet_arm_x
##                "integer"                "integer"                "integer"
##             magnet_arm_y             magnet_arm_z        kurtosis_roll_arm
##                "integer"                "integer"                "logical"
##       kurtosis_picth_arm         kurtosis_yaw_arm        skewness_roll_arm
##                "logical"                "logical"                "logical"
##       skewness_pitch_arm         skewness_yaw_arm             max_roll_arm
##                "logical"                "logical"                "logical"
##            max_picth_arm              max_yaw_arm             min_roll_arm
##                "logical"                "logical"                "logical"
##            min_pitch_arm              min_yaw_arm       amplitude_roll_arm
##                "logical"                "logical"                "logical"
##      amplitude_pitch_arm        amplitude_yaw_arm            roll_dumbbell
##                "logical"                "logical"                "numeric"
##           pitch_dumbbell             yaw_dumbbell   kurtosis_roll_dumbbell
##                "numeric"                "numeric"                "logical"
##  kurtosis_picth_dumbbell    kurtosis_yaw_dumbbell   skewness_roll_dumbbell
##                "logical"                "logical"                "logical"
##  skewness_pitch_dumbbell    skewness_yaw_dumbbell        max_roll_dumbbell
##                "logical"                "logical"                "logical"
##       max_picth_dumbbell         max_yaw_dumbbell        min_roll_dumbbell
##                "logical"                "logical"                "logical"
##       min_pitch_dumbbell         min_yaw_dumbbell  amplitude_roll_dumbbell
##                "logical"                "logical"                "logical"
## amplitude_pitch_dumbbell   amplitude_yaw_dumbbell     total_accel_dumbbell
##                "logical"                "logical"                "integer"
##       var_accel_dumbbell        avg_roll_dumbbell     stddev_roll_dumbbell
##                "logical"                "logical"                "logical"
##        var_roll_dumbbell       avg_pitch_dumbbell    stddev_pitch_dumbbell
##                "logical"                "logical"                "logical"
##       var_pitch_dumbbell         avg_yaw_dumbbell      stddev_yaw_dumbbell
##                "logical"                "logical"                "logical"
##         var_yaw_dumbbell         gyros_dumbbell_x         gyros_dumbbell_y
##                "logical"                "numeric"                "numeric"
##         gyros_dumbbell_z         accel_dumbbell_x         accel_dumbbell_y
##                "numeric"                "integer"                "integer"
##         accel_dumbbell_z        magnet_dumbbell_x        magnet_dumbbell_y
##                "integer"                "integer"                "integer"
##        magnet_dumbbell_z             roll_forearm            pitch_forearm
##                "integer"                "numeric"                "numeric"
##              yaw_forearm    kurtosis_roll_forearm   kurtosis_picth_forearm
##                "numeric"                "logical"                "logical"
##     kurtosis_yaw_forearm    skewness_roll_forearm   skewness_pitch_forearm
##                "logical"                "logical"                "logical"
##     skewness_yaw_forearm         max_roll_forearm        max_picth_forearm
##                "logical"                "logical"                "logical"
##          max_yaw_forearm         min_roll_forearm        min_pitch_forearm
##                "logical"                "logical"                "logical"
##          min_yaw_forearm   amplitude_roll_forearm  amplitude_pitch_forearm
##                "logical"                "logical"                "logical"
##    amplitude_yaw_forearm      total_accel_forearm        var_accel_forearm
##                "logical"                "integer"                "logical"
##         avg_roll_forearm      stddev_roll_forearm         var_roll_forearm
##                "logical"                "logical"                "logical"
##        avg_pitch_forearm     stddev_pitch_forearm        var_pitch_forearm
##                "logical"                "logical"                "logical"
##          avg_yaw_forearm       stddev_yaw_forearm          var_yaw_forearm
##                "logical"                "logical"                "logical"
##          gyros_forearm_x          gyros_forearm_y          gyros_forearm_z
##                "numeric"                "numeric"                "numeric"
##          accel_forearm_x          accel_forearm_y          accel_forearm_z
##                "integer"                "integer"                "integer"
##         magnet_forearm_x         magnet_forearm_y         magnet_forearm_z
##                "integer"                "integer"                "integer"
##               problem_id
##                "integer"
```

Using What we saw in the previous part we should remove some of the columns that not make sence to have with the grep function, because thay are not numeric and create a new vector from this operation.

```
training_aux <- training[, -(grep("timestamp|X|user_name|num_window|new_window", names(training)))]
test_aux <- test[, -(grep("timestamp|X|user_name|num_window|new_window", names(test)))]
```

Many of the columns we load have NAs `values.  We take out from the variables those data with NA's values The` NAs records make our machine learning algorithm less precise. Finally we update the vector with out any NA's values.

```
NAs <- apply(training_aux, 2, function(x) {
    sum(is.na(x))})

training_aux <- training_aux[, which(NAs == 0)]
test_aux <- test_aux[, which(NAs == 0)]
```

We split the data into two variables. First the training data with the 60 % and the test data with 40%.

```
training.idx <- training_aux[createDataPartition(y = training_aux$classe, p = 0.6, list = FALSE), ]
test.idx <- training_aux[-createDataPartition(y = training_aux$classe, p = 0.6, list = FALSE), ]
```

We take a look how our variables shrink, with out the NA‘s values.

```
dim(training.idx)
```

```
## [1] 11776    53
```

```
head(training.idx)
```

```
##   roll_belt pitch_belt yaw_belt total_accel_belt gyros_belt_x gyros_belt_y
## 1      1.41       8.07    -94.4                3         0.00         0.00
## 3      1.42       8.07    -94.4                3         0.00         0.00
## 5      1.48       8.07    -94.4                3         0.02         0.02
## 6      1.45       8.06    -94.4                3         0.02         0.00
## 7      1.42       8.09    -94.4                3         0.02         0.00
## 9      1.43       8.16    -94.4                3         0.02         0.00
##   gyros_belt_z accel_belt_x accel_belt_y accel_belt_z magnet_belt_x
## 1        -0.02          -21            4           22            -3
## 3        -0.02          -20            5           23            -2
## 5        -0.02          -21            2           24            -6
## 6        -0.02          -21            4           21             0
## 7        -0.02          -22            3           21            -4
## 9        -0.02          -20            2           24             1
##   magnet_belt_y magnet_belt_z roll_arm pitch_arm yaw_arm total_accel_arm
## 1           599          -313     -128      22.5    -161              34
## 3           600          -305     -128      22.5    -161              34
## 5           600          -302     -128      22.1    -161              34
## 6           603          -312     -128      22.0    -161              34
## 7           599          -311     -128      21.9    -161              34
## 9           602          -312     -128      21.7    -161              34
##   gyros_arm_x gyros_arm_y gyros_arm_z accel_arm_x accel_arm_y accel_arm_z
## 1        0.00        0.00       -0.02        -288         109        -123
## 3        0.02       -0.02       -0.02        -289         110        -126
## 5        0.00       -0.03        0.00        -289         111        -123
## 6        0.02       -0.03        0.00        -289         111        -122
## 7        0.00       -0.03        0.00        -289         111        -125
## 9        0.02       -0.03       -0.02        -288         109        -122
##   magnet_arm_x magnet_arm_y magnet_arm_z roll_dumbbell pitch_dumbbell
## 1         -368          337          516      13.05217      -70.49400
## 3         -368          344          513      12.85075      -70.27812
## 5         -374          337          506      13.37872      -70.42856
## 6         -369          342          513      13.38246      -70.81759
## 7         -373          336          509      13.12695      -70.24757
## 9         -369          341          518      13.15463      -70.42520
##   yaw_dumbbell total_accel_dumbbell gyros_dumbbell_x gyros_dumbbell_y
## 1    -84.87394                   37                0            -0.02
## 3    -85.14078                   37                0            -0.02
## 5    -84.85306                   37                0            -0.02
## 6    -84.46500                   37                0            -0.02
## 7    -85.09961                   37                0            -0.02
## 9    -84.91563                   37                0            -0.02
##   gyros_dumbbell_z accel_dumbbell_x accel_dumbbell_y accel_dumbbell_z
## 1                0             -234               47             -271
## 3                0             -232               46             -270
## 5                0             -233               48             -270
## 6                0             -234               48             -269
## 7                0             -232               47             -270
## 9                0             -232               47             -269
##   magnet_dumbbell_x magnet_dumbbell_y magnet_dumbbell_z roll_forearm
## 1              -559               293               -65         28.4
## 3              -561               298               -63         28.3
## 5              -554               292               -68         28.0
## 6              -558               294               -66         27.9
## 7              -551               295               -70         27.9
## 9              -549               292               -65         27.7
##   pitch_forearm yaw_forearm total_accel_forearm gyros_forearm_x
## 1         -63.9        -153                  36            0.03
## 3         -63.9        -152                  36            0.03
## 5         -63.9        -152                  36            0.02
## 6         -63.9        -152                  36            0.02
## 7         -63.9        -152                  36            0.02
## 9         -63.8        -152                  36            0.03
##   gyros_forearm_y gyros_forearm_z accel_forearm_x accel_forearm_y
## 1            0.00           -0.02             192             203
## 3           -0.02            0.00             196             204
## 5            0.00           -0.02             189             206
## 6           -0.02           -0.03             193             203
## 7            0.00           -0.02             195             205
## 9            0.00           -0.02             193             204
##   accel_forearm_z magnet_forearm_x magnet_forearm_y magnet_forearm_z
## 1            -215              -17              654              476
## 3            -213              -18              658              469
## 5            -214              -17              655              473
## 6            -215               -9              660              478
## 7            -215              -18              659              470
## 9            -214              -16              653              476
##   classe
## 1      A
## 3      A
## 5      A
## 6      A
## 7      A
## 9      A
```

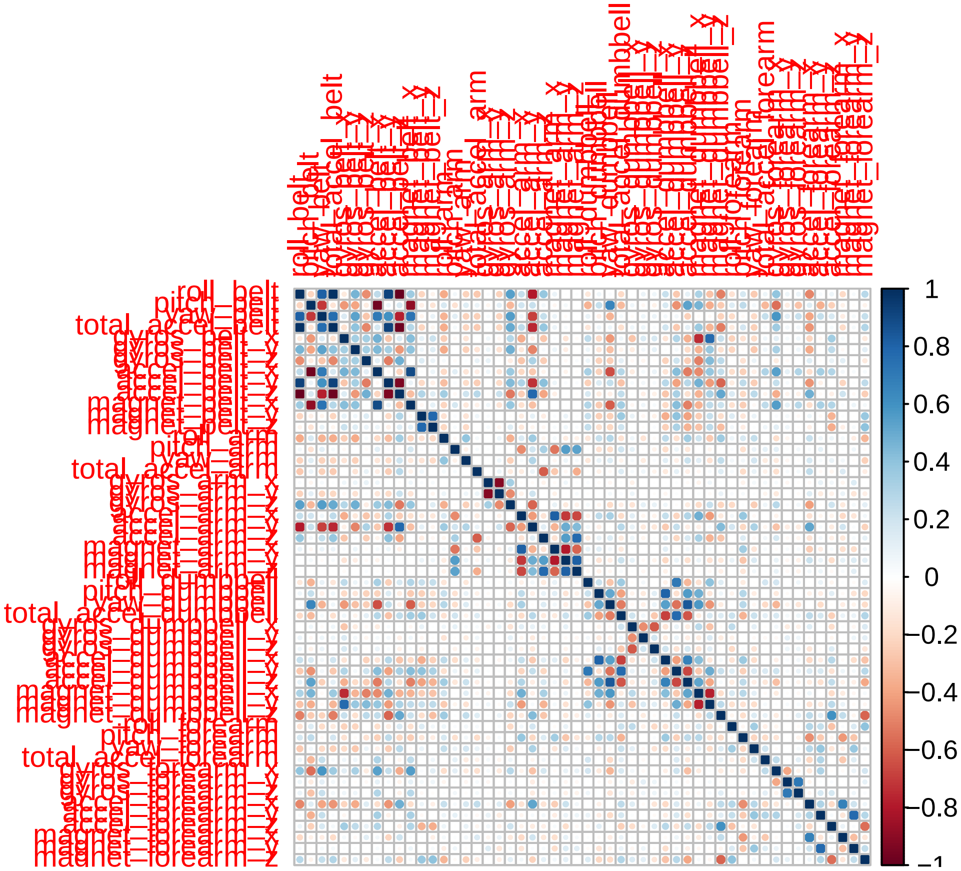
this graph is usefull to look at the correlation with some variables to each others. Note that the dark blue indicated strong correlation and red negative correlation. Our dataset is 11776 X 53 and one of those is Classe, the variable we want to predict. This graph shows us nice information to implement a Random Forest Study.

```
Graph <- training.idx

NAs <- apply(training.idx, 2, function(x) {
    sum(is.na(x))
})
Graph<- training.idx[, which(NAs == 0)]



CorrePlot = cor( Graph[,-c(grep("timestamp|X|user_name|num_window|new_window",names(Graph)), length(Gra
corrplot(CorrePlot, method="circle",tl.cex=0.9)
```



At this point we create our model using Random Forest and the set of data originally given but reduced. We train our model over the dataset using the "boot" method inside trainControl.

```
set.seed(10)

MyModel.Forest<- train(training.idx$classe ~ ., data = training.idx, method = "rf",
    prof = TRUE, trControl = trainControl(method = "boot", number = 5, allowParallel = TRUE))

summary(MyModel.Forest)
```

```
##                 Length Class      Mode
## call                5  -none-     call
## type                1  -none-     character
## predicted       11776  factor     numeric
## err.rate         3000  -none-     numeric
## confusion          30  -none-     numeric
## votes           58880  matrix     numeric
## oob.times       11776  -none-     numeric
## classes             5  -none-     character
## importance         52  -none-     numeric
## importanceSD        0  -none-     NULL
## localImportance     0  -none-     NULL
## proximity           0  -none-     NULL
## ntree               1  -none-     numeric
## mtry                1  -none-     numeric
## forest             14  -none-     list
## y               11776  factor     numeric
## test                0  -none-     NULL
## inbag               0  -none-     NULL
## xNames             52  -none-     character
## problemType         1  -none-     character
## tuneValue           1  data.frame list
## obsLevels           5  -none-     character
```

```
MyModel.Forest
```

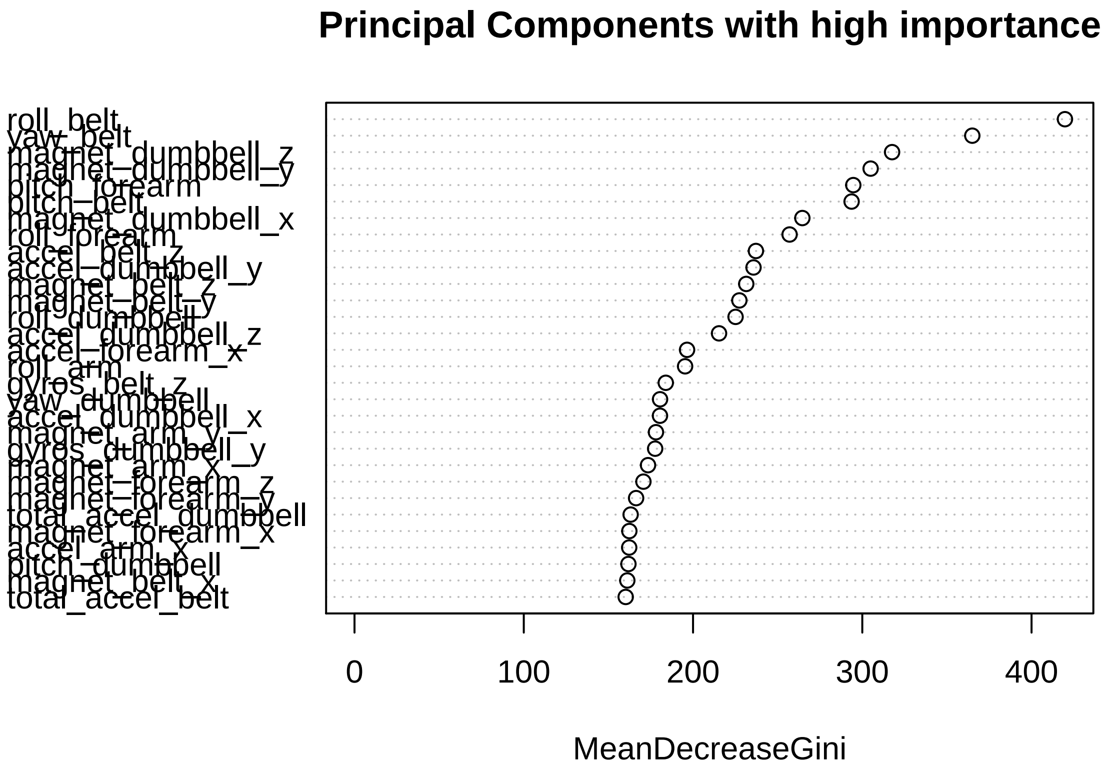
```
## Random Forest
##
## 11776 samples
##    52 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Bootstrapped (5 reps)
## Summary of sample sizes: 11776, 11776, 11776, 11776, 11776
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa      Accuracy SD  Kappa SD
##    2    0.9863446  0.9827060  0.001599790  0.002023190
##   27    0.9862003  0.9825210  0.002381382  0.003031075
##   52    0.9788633  0.9732262  0.003715593  0.004725817
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 2.
```

```
MyModel.Forest.Final<- MyModel.Forest$results
round(max(MyModel.Forest.Final$Accuracy), 3) * 100
```

```
## [1] 98.6
```

We got a very nice result of the occuracy at the first attemp: 98.6 %. That‘s ok.

```r
varImpPlot(MyModel.Forest$finalModel,
           main = "Principal Components with high importance")
```



The cross validation study of our model.

```r
test.idx$predRight <- (predict(MyModel.Forest, test.idx))== test.idx$classe
table(predict(MyModel.Forest, test.idx), test.idx$classe)
```

```
##
##        A    B    C    D    E
##   A 2231    9    0    0    0
##   B    1 1505    6    0    0
##   C    0    4 1361   16    2
##   D    0    0    1 1270    4
##   E    0    0    0    0 1436
```

```r
CrossValidated<- postResample((predict(MyModel.Forest, test.idx)), test.idx$classe)
CrossValidated
```

```
##  Accuracy     Kappa
## 0.9945195 0.9930670
```

We got a nice high level of accuracy.

We try out with the ConfussionMatrix

```
set.seed(10)
CrossValidatedError <- confusionMatrix((predict(MyModel.Forest, test.idx)), test.idx$classe)
CrossValidatedError
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction    A    B    C    D    E
##          A 2231    9    0    0    0
##          B    1 1505    6    0    0
##          C    0    4 1361   16    2
##          D    0    0    1 1270    4
##          E    0    0    0    0 1436
## 
## Overall Statistics
## 
##                Accuracy : 0.9945
##                  95% CI : (0.9926, 0.996)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
## 
##                   Kappa : 0.9931
##  Mcnemar's Test P-Value : NA
## 
## Statistics by Class:
## 
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9996   0.9914   0.9949   0.9876   0.9958
## Specificity            0.9984   0.9989   0.9966   0.9992   1.0000
## Pos Pred Value         0.9960   0.9954   0.9841   0.9961   1.0000
## Neg Pred Value         0.9998   0.9979   0.9989   0.9976   0.9991
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2843   0.1918   0.1735   0.1619   0.1830
## Detection Prevalence   0.2855   0.1927   0.1763   0.1625   0.1830
## Balanced Accuracy      0.9990   0.9952   0.9957   0.9934   0.9979
```

```
postResample((predict(MyModel.Forest, test.idx)), test.idx$classe)[[1]]
```

```
## [1] 0.9945195
```

```
1- postResample((predict(MyModel.Forest, test.idx)), test.idx$classe)[[1]]
```

```
## [1] 0.0054805
```

we can see that our calculus were very close with the result of the matrix. The accurance is 99.6%

The 20 cases to predict

Prepare the function to write the files that we are going to submit to Coursera

```
pml_write_files = function(x, directory="solutionfiles"){
  dir.create (directory)

  n = length(x)
  for(i in 1:n){
    filename = paste0("problem_id_",i,".txt")
    filename=file.path(directory, filename)
    write.table(x[i],file=filename,quote=FALSE,row.names=FALSE,col.names=FALSE)
  }
}
```

```
Model.Prediction <- predict(MyModel.Forest, test)
Model.Prediction
```

```
##  [1] 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
```

```
pml_write_files(Model.Prediction)
```

```
## Warning in dir.create(directory): 'solutionfiles' already exists
```