HomeCredit Modeling

Michael Tom

October 08, 2023

Contents

#Load Packages and Data

#RF 1 Tune Grid, 1.5 DownSample

```r
tic()
set.seed(123)
#Create Random Forest Train Data set
RF1Tune1.5ds <- clean_train

#Split Data 75% Training 25% Test
RF1Tune1.5dssplit <- initial_split(RF1Tune1.5ds, strata = TARGET, prop = 0.75)
RF1Tune1.5dstrain <- training(RF1Tune1.5dssplit)
RF1Tune1.5dstest <- testing(RF1Tune1.5dssplit)

#Create Recipe with 1.5 Downsample
RF1Tune1.5dsrecipe <- recipe(TARGET ~ ., data = RF1Tune1.5dstrain)

#Create Model with Hyperperameter Tuning (Set Peramters based on tuning to save run time)
RF1Tune1.5dsmodel <- rand_forest(
  mtry = (14),
  trees = (1343),
  min_n = (33))%>%
  set_mode("classification") %>%
  set_engine("ranger")

#Set 5 fold CV
RF1Tune1.5dsfolds <- vfold_cv(RF1Tune1.5dstrain, v=5)

doParallel::registerDoParallel(cores = 8)

#Create Workflow
RF1Tune1.5ds_WF <- workflow () %>%
    add_model(RF1Tune1.5dsmodel) %>%
    add_recipe(RF1Tune1.5dsrecipe)

#Tune Model with grid = 1 (very time intensive)
# 50 Grid gives mtry (14), trees (1343), min_n (33)
#RF1Tune1.5ds_tune <- RF1Tune1.5ds_WF %>% tune_grid( resamples = RF1Tune1.5dsfolds, grid = 1)
```

```r
#Create Best model using ROC_AUC
#best_modelRF1Tune1.5ds <- RF1Tune1.5ds_tune %>% select_best ("roc_auc")
#best_modelRF1Tune1.5ds

#Create final WF based on best model
finalWFRF1Tune1.5dsmodel <- RF1Tune1.5ds_WF %>% finalize_workflow(RF1Tune1.5dsmodel)

#Create final fit
final_fitRF1Tune1.5dsmodel <- finalWFRF1Tune1.5dsmodel %>% last_fit(split = RF1Tune1.5dssplit)

#Run Final Fit
final_fitRF1Tune1.5dsmodel %>% collect_metrics()
```

```
## # A tibble: 2 x 4
##   .metric  .estimator .estimate .config
##   <chr>    <chr>          <dbl> <chr>
## 1 accuracy binary         0.921 Preprocessor1_Model1
## 2 roc_auc  binary         0.724 Preprocessor1_Model1
```

```r
#Create Confusion matrix
final_fitRF1Tune1.5dsmodel %>%
  collect_predictions()%>%
  conf_mat(truth = TARGET, estimate = .pred_class)
```

```
##           Truth
## Prediction    No   Yes
##        No  70697  6037
##        Yes    70    73
```

```r
#extract WF
final_wfRF1Tune1.5dsmodel <- final_fitRF1Tune1.5dsmodel %>%
  extract_workflow()
final_wfRF1Tune1.5dsmodel
```

```
## == Workflow [trained] ==========================================================
## Preprocessor: Recipe
## Model: rand_forest()
##
## -- Preprocessor ----------------------------------------------------------------
## 0 Recipe Steps
##
## -- Model -----------------------------------------------------------------------
## Ranger result
##
## Call:
##  ranger::ranger(x = maybe_data_frame(x), y = y, mtry = min_cols(~(14),      x), num.trees = ~(1343),
##
## Type:                             Probability estimation
## Number of trees:                  1343
## Sample size:                      230630
## Number of independent variables:  23
## Mtry:                             14
## Target node size:                33
## Variable importance mode:         none
## Splitrule:                        gini
## OOB prediction error (Brier s.):  0.07029278
```

```
#Predict on test data
RF1Tune1.5dsPred <- predict(final_wfRF1Tune1.5dsmodel, new_data = clean_test, type = 'prob')
#Generate Submission
submission <- data.frame(SK_ID_CURR = as.integer(clean_test$SK_ID_CURR), TARGET = RF1Tune1.5dsPred)
summary(submission)
```

```
##    SK_ID_CURR     TARGET..pred_No  TARGET..pred_Yes
##  Min.   :100001   Min.   :0.2919   Min.   :0.001305
##  1st Qu.:188558   1st Qu.:0.8747   1st Qu.:0.035784
##  Median :277549   Median :0.9313   Median :0.068707
##  Mean   :277797   Mean   :0.9070   Mean   :0.093031
##  3rd Qu.:367556   3rd Qu.:0.9642   3rd Qu.:0.125316
##  Max.   :456250   Max.   :0.9987   Max.   :0.708081
```

```
submission <- submission %>% rename ("TARGET" = "TARGET..pred_Yes") %>% select(c(SK_ID_CURR, TARGET))
write.csv(submission, file = 'RF1Tune1.5dsmodel.csv', row.names = FALSE)

toc()
```

```
## 1346.072 sec elapsed
```

#RF 1 Tune Grid, No DownSample

```
tic()
set.seed(123)
#Create Random Forest Train Data set
RF1TuneNods <- clean_train

#Split Data 75% Training 25% Test
RF1TuneNodssplit <- initial_split(RF1TuneNods, strata = TARGET, prop = 0.75)
RF1TuneNodstrain <- training(RF1TuneNodssplit)
RF1TuneNodstest <- testing(RF1TuneNodssplit)

#Create Recipe with 1.5 Downsample
RF1TuneNodsrecipe <- recipe(TARGET ~ ., data = RF1TuneNodstrain) %>% step_downsample(TARGET, under_rati

#Create Model with Hyperperameter Tuning (Set Perameters based on tuning to save run time)
RF1TuneNodsmodel <- rand_forest(
  mtry = (14),
  trees = (1343),
  min_n = (33))%>%
  set_mode("classification") %>%
  set_engine("ranger")

#Set 5 fold CV
RF1TuneNodsfolds <- vfold_cv(RF1TuneNodstrain, v=5)
```

```r
doParallel::registerDoParallel(cores = 8)

#Create Workflow
RF1TuneNods_WF <- workflow () %>%
    add_model(RF1TuneNodsmodel) %>%
    add_recipe(RF1TuneNodsrecipe)

#Tune Model with grid = 1 (very time intensive)
# 50 Grid gives mtry (14), trees (1343), min_n (33)
#RF1TuneNods_tune <- RF1TuneNods_WF %>% tune_grid( resamples = RF1TuneNodsfolds, grid = 1)

#Create Best model using ROC_AUC
#best_modelRF1TuneNods <- RF1TuneNods_tune %>% select_best ("roc_auc")
#best_modelRF1TuneNods

#Create final WF based on best model
finalWFRF1TuneNodsmodel <- RF1TuneNods_WF %>% finalize_workflow(RF1TuneNods_WF)

#Create final fit
final_fitRF1TuneNodsmodel <- finalWFRF1TuneNodsmodel %>% last_fit(split = RF1TuneNodssplit)

#Run Final Fit
final_fitRF1TuneNodsmodel %>% collect_metrics()
```

```
## # A tibble: 2 x 4
##   .metric  .estimator .estimate .config
##   <chr>    <chr>          <dbl> <chr>
## 1 accuracy binary         0.789 Preprocessor1_Model1
## 2 roc_auc  binary         0.733 Preprocessor1_Model1
```

```r
#Create Confusion matrix
final_fitRF1TuneNodsmodel %>%
  collect_predictions()%>%
  conf_mat(truth = TARGET, estimate = .pred_class)
```

```
##           Truth
## Prediction    No   Yes
##        No  57597  3014
##        Yes 13170  3096
```

```r
#extract WF
final_wfRF1TuneNodsmodel <- final_fitRF1TuneNodsmodel %>%
  extract_workflow()
final_wfRF1TuneNodsmodel
```

```
## == Workflow [trained] ==========================================================
## Preprocessor: Recipe
## Model: rand_forest()
##
## -- Preprocessor ----------------------------------------------------------------
## 1 Recipe Step
##
```

```
## * step_downsample()
##
## -- Model -----------------------------------------------------------------------
## Ranger result
##
## Call:
##  ranger::ranger(x = maybe_data_frame(x), y = y, mtry = min_cols(~(14),      x), num.trees = ~(1343),
##
## Type:                             Probability estimation
## Number of trees:                  1343
## Sample size:                      46787
## Number of independent variables:  23
## Mtry:                             14
## Target node size:                 33
## Variable importance mode:         none
## Splitrule:                        gini
## OOB prediction error (Brier s.):  0.2014148
```

```r
#Predict on test data
RF1TuneNodsPred <- predict(final_wfRF1TuneNodsmodel, new_data = clean_test, type = 'prob')
#Generate Submission
submission <- data.frame(SK_ID_CURR = as.integer(clean_test$SK_ID_CURR), TARGET = RF1TuneNodsPred)
summary(submission)
```

```
##    SK_ID_CURR     TARGET..pred_No   TARGET..pred_Yes
##  Min.   :100001   Min.   :0.02915   Min.   :0.0232
##  1st Qu.:188558   1st Qu.:0.53526   1st Qu.:0.2183
##  Median :277549   Median :0.67004   Median :0.3300
##  Mean   :277797   Mean   :0.64830   Mean   :0.3517
##  3rd Qu.:367556   3rd Qu.:0.78169   3rd Qu.:0.4647
##  Max.   :456250   Max.   :0.97680   Max.   :0.9709
```

```r
submission <- submission %>% rename ("TARGET" = "TARGET..pred_Yes") %>% select(c(SK_ID_CURR, TARGET))
write.csv(submission, file = 'RF1TuneNodsmodel.csv', row.names = FALSE)

toc()
```

```
## 147.678 sec elapsed
```

#RF 50 Tune Grid, 1.5 DownSample

```r
tic()
set.seed(123)
#Create Random Forest Train Data set
RF50Tune1.5ds <- clean_train

#Split Data 75% Training 25% Test
RF50Tune1.5dssplit <- initial_split(RF50Tune1.5ds, strata = TARGET, prop = 0.75)
RF50Tune1.5dstrain <- training(RF50Tune1.5dssplit)
RF50Tune1.5dstest <- testing(RF50Tune1.5dssplit)

#Create Recipe with 1.5 Downsample
```

```r
RF50Tune1.5dsrecipe <- recipe(TARGET ~ ., data = RF50Tune1.5dstrain) %>% step_downsample(TARGET, under_

#Create Model with Hyperperameter Tuning (Set Perameters based on tuning to save run time)
RF50Tune1.5dsmodel <- rand_forest(
  mtry = (2),
  trees = (727),
  min_n = (31)) %>%
  set_mode("classification") %>%
  set_engine("ranger")

#Set 5 fold CV
RF50Tune1.5dsfolds <- vfold_cv(RF50Tune1.5dstrain, v=5)

doParallel::registerDoParallel(cores = 8)

#Create Workflow
RF50Tune1.5ds_WF <- workflow () %>%
    add_model(RF50Tune1.5dsmodel) %>%
    add_recipe(RF50Tune1.5dsrecipe)

#Tune Model with grid = 50 (very time intensive)
# 50 Grid gives mtry (2), trees (727), min_n (31)
#RF50Tune1.5ds_tune <- RF50Tune1.5ds_WF %>% tune_grid( resamples = RF50Tune1.5dsfolds, grid = 50)

#Create Best model using ROC_AUC
#best_modelRF50Tune1.5ds <- RF50Tune1.5ds_tune %>% select_best ("roc_auc")
#best_modelRF50Tune1.5ds

#Create final WF based on best model
finalWFRF50Tune1.5dsmodel <- RF50Tune1.5ds_WF %>% finalize_workflow(RF50Tune1.5dsmodel)

#Create final fit
final_fitRF50Tune1.5dsmodel <- finalWFRF50Tune1.5dsmodel %>% last_fit(split = RF50Tune1.5dssplit)

#Run Final Fit
final_fitRF50Tune1.5dsmodel %>% collect_metrics()
```

```
## # A tibble: 2 x 4
##   .metric  .estimator .estimate .config
##   <chr>    <chr>          <dbl> <chr>
## 1 accuracy binary         0.815 Preprocessor1_Model1
## 2 roc_auc  binary         0.743 Preprocessor1_Model1
```

```r
#Create Confusion matrix
final_fitRF50Tune1.5dsmodel %>%
  collect_predictions()%>%
  conf_mat(truth = TARGET, estimate = .pred_class)
```

```
##           Truth
## Prediction    No   Yes
##        No  59817  3278
##        Yes 10950  2832
```

```
#extract WF
final_wfRF50Tune1.5dsmodel <- final_fitRF50Tune1.5dsmodel %>%
  extract_workflow()
final_wfRF50Tune1.5dsmodel
```

```
## == Workflow [trained] ==========================================================
## Preprocessor: Recipe
## Model: rand_forest()
##
## -- Preprocessor ----------------------------------------------------------------
## 1 Recipe Step
##
## * step_downsample()
##
## -- Model -----------------------------------------------------------------------
## Ranger result
##
## Call:
##  ranger::ranger(x = maybe_data_frame(x), y = y, mtry = min_cols(~(2),      x), num.trees = ~(727), mi
##
## Type:                             Probability estimation
## Number of trees:                  727
## Sample size:                      46787
## Number of independent variables:  23
## Mtry:                             2
## Target node size:                 31
## Variable importance mode:         none
## Splitrule:                        gini
## OOB prediction error (Brier s.):  0.2012943
```

```
#Predict on test data
RF50Tune1.5dsPred <- predict(final_wfRF50Tune1.5dsmodel, new_data = clean_test, type = 'prob')
#Generate Submission
submission <- data.frame(SK_ID_CURR = as.integer(clean_test$SK_ID_CURR), TARGET = RF50Tune1.5dsPred)
summary(submission)
```

```
##    SK_ID_CURR     TARGET..pred_No  TARGET..pred_Yes
##  Min.   :100001   Min.   :0.1875   Min.   :0.0790  
##  1st Qu.:188558   1st Qu.:0.5439   1st Qu.:0.2511  
##  Median :277549   Median :0.6575   Median :0.3425  
##  Mean   :277797   Mean   :0.6396   Mean   :0.3604  
##  3rd Qu.:367556   3rd Qu.:0.7489   3rd Qu.:0.4561  
##  Max.   :456250   Max.   :0.9210   Max.   :0.8125
```

```
submission <- submission %>% rename ("TARGET" = "TARGET..pred_Yes") %>% select(c(SK_ID_CURR, TARGET))
write.csv(submission, file = 'RF50Tune1.5dsmodel.csv', row.names = FALSE)

toc()
```

```
## 30.458 sec elapsed
```

#RF 50 Tune Grid, No DownSample

```r
tic()
set.seed(123)
#Create Random Forest Train Data set
RF50TuneNods <- clean_train

#Split Data 75% Training 25% Test
RF50TuneNodssplit <- initial_split(RF50TuneNods, strata = TARGET, prop = 0.75)
RF50TuneNodstrain <- training(RF50TuneNodssplit)
RF50TuneNodstest <- testing(RF50TuneNodssplit)

#Create Recipe with 1.5 Downsample
RF50TuneNodsrecipe <- recipe(TARGET ~ ., data = RF50TuneNodstrain)

#Create Model with Hyperperameter Tuning (Set Perameters based on tuning to save run time)
RF50TuneNodsmodel <- rand_forest(
  mtry = (2),
  trees = (727),
  min_n = (31)) %>%
  set_mode("classification") %>%
  set_engine("ranger")

#Set 5 fold CV
RF50TuneNodsfolds <- vfold_cv(RF50TuneNodstrain, v=5)

doParallel::registerDoParallel(cores = 8)

#Create Workflow
RF50TuneNods_WF <- workflow () %>%
    add_model(RF50TuneNodsmodel) %>%
    add_recipe(RF50TuneNodsrecipe)

#Tune Model with grid = 50 (very time intensive)
# 50 Grid gives mtry (2), trees (727), min_n (31)
#RF50TuneNods_tune <- RF50TuneNods_WF %>% tune_grid( resamples = RF50TuneNodsfolds, grid = 50)

#Create Best model using ROC_AUC
#best_modelRF50TuneNods <- RF50TuneNods_tune %>% select_best ("roc_auc")
#best_modelRF50TuneNods

#Create final WF based on best model
finalWFRF50TuneNodsmodel <- RF50TuneNods_WF %>% finalize_workflow(RF50TuneNodsmodel)

#Create final fit
final_fitRF50TuneNodsmodel <- finalWFRF50TuneNodsmodel %>% last_fit(split = RF50TuneNodssplit)

#Run Final Fit
final_fitRF50TuneNodsmodel %>% collect_metrics()
```

```
## # A tibble: 2 x 4
##   .metric  .estimator .estimate .config
##   <chr>    <chr>          <dbl> <chr>
## 1 accuracy binary         0.921 Preprocessor1_Model1
## 2 roc_auc  binary         0.738 Preprocessor1_Model1
```

```
#Create Confusion matrix
final_fitRF50TuneNodsmodel %>%
  collect_predictions()%>%
  conf_mat(truth = TARGET, estimate = .pred_class)
```

```
##           Truth
## Prediction    No   Yes
##         No  70767  6108
##         Yes     0     2
```

```
#extract WF
final_wfRF50TuneNodsmodel <- final_fitRF50TuneNodsmodel %>%
  extract_workflow()
final_wfRF50TuneNodsmodel
```

```
## == Workflow [trained] ==========================================================
## Preprocessor: Recipe
## Model: rand_forest()
##
## -- Preprocessor ----------------------------------------------------------------
## 0 Recipe Steps
##
## -- Model -----------------------------------------------------------------------
## Ranger result
##
## Call:
##  ranger::ranger(x = maybe_data_frame(x), y = y, mtry = min_cols(~(2),      x), num.trees = ~(727), mi
##
## Type:                             Probability estimation
## Number of trees:                  727
## Sample size:                      230630
## Number of independent variables:  23
## Mtry:                             2
## Target node size:                 31
## Variable importance mode:         none
## Splitrule:                        gini
## OOB prediction error (Brier s.):  0.06934983
```

```
#Predict on test data
RF50TuneNodsPred <- predict(final_wfRF50TuneNodsmodel, new_data = clean_test, type = 'prob')
#Generate Submission
submission <- data.frame(SK_ID_CURR = as.integer(clean_test$SK_ID_CURR), TARGET = RF50TuneNodsPred)
summary(submission)
```

```
##    SK_ID_CURR     TARGET..pred_No  TARGET..pred_Yes
##  Min.   :100001   Min.   :0.5099   Min.   :0.007218
##  1st Qu.:188558   1st Qu.:0.8924   1st Qu.:0.042783
##  Median :277549   Median :0.9328   Median :0.067150
##  Mean   :277797   Mean   :0.9160   Mean   :0.083999
##  3rd Qu.:367556   3rd Qu.:0.9572   3rd Qu.:0.107639
##  Max.   :456250   Max.   :0.9928   Max.   :0.490143
```

```
submission <- submission %>% rename ("TARGET" = "TARGET..pred_Yes") %>% select(c(SK_ID_CURR, TARGET))
write.csv(submission, file = 'RF50TuneNodsmodel.csv', row.names = FALSE)

toc()
```

```
## 141.964 sec elapsed
```

#RF 100 Tune Grid, 1.5 DownSample

```
tic()
set.seed(123)
#Create Random Forest Train Data set
RF100Tune1.5ds <- clean_train

#Split Data 75% Training 25% Test
RF100Tune1.5dssplit <- initial_split(RF100Tune1.5ds, strata = TARGET, prop = 0.75)
RF100Tune1.5dstrain <- training(RF100Tune1.5dssplit)
RF100Tune1.5dstest <- testing(RF100Tune1.5dssplit)

#Create Recipe with 1.5 Downsample
RF100Tune1.5dsrecipe <- recipe(TARGET ~ ., data = RF100Tune1.5dstrain) %>% step_downsample(TARGET, unde

#Create Model with Hyperperameter Tuning (Set Perameters based on tuning to save run time)
RF100Tune1.5dsmodel <- rand_forest(
  mtry = (2),
  trees = (1003),
  min_n = (33)) %>%
  set_mode("classification") %>%
  set_engine("ranger")

#Set 5 fold CV
RF100Tune1.5dsfolds <- vfold_cv(RF100Tune1.5dstrain, v=5)

doParallel::registerDoParallel(cores = 8)

#Create Workflow
RF100Tune1.5ds_WF <- workflow () %>%
    add_model(RF100Tune1.5dsmodel) %>%
    add_recipe(RF100Tune1.5dsrecipe)

#Tune Model with grid = 50 (very time intensive)
# 100 Grid gives mtry (2) trees (1003) min_n (33)
#RF100Tune1.5ds_tune <- RF100Tune1.5ds_WF %>% tune_grid( resamples = RF100Tune1.5dsfolds, grid = 100)

#Create Best model using ROC_AUC
#best_modelRF100Tune1.5ds <- RF100Tune1.5ds_tune %>% select_best ("roc_auc")
#best_modelRF100Tune1.5ds

#Create final WF based on best model
finalWFRF100Tune1.5dsmodel <- RF100Tune1.5ds_WF %>% finalize_workflow(RF100Tune1.5dsmodel)

#Create final fit
final_fitRF100Tune1.5dsmodel <- finalWFRF100Tune1.5dsmodel %>% last_fit(split = RF100Tune1.5dssplit)
```

```r
#Run Final Fit
final_fitRF100Tune1.5dsmodel %>% collect_metrics()
```

```
## # A tibble: 2 x 4
##   .metric  .estimator .estimate .config
##   <chr>    <chr>          <dbl> <chr>
## 1 accuracy binary         0.815 Preprocessor1_Model1
## 2 roc_auc  binary         0.742 Preprocessor1_Model1
```

```r
#Create Confusion matrix
final_fitRF100Tune1.5dsmodel %>%
  collect_predictions()%>%
  conf_mat(truth = TARGET, estimate = .pred_class)
```

```
##           Truth
## Prediction    No   Yes
##        No  59833  3275
##        Yes 10934  2835
```

```r
#extract WF
final_wfRF100Tune1.5dsmodel <- final_fitRF100Tune1.5dsmodel %>%
  extract_workflow()
final_wfRF100Tune1.5dsmodel
```

```
## == Workflow [trained] ==========================================================
## Preprocessor: Recipe
## Model: rand_forest()
##
## -- Preprocessor ----------------------------------------------------------------
## 1 Recipe Step
##
## * step_downsample()
##
## -- Model -----------------------------------------------------------------------
## Ranger result
##
## Call:
##  ranger::ranger(x = maybe_data_frame(x), y = y, mtry = min_cols(~(2),      x), num.trees = ~(1003),
##
## Type:                             Probability estimation
## Number of trees:                  1003
## Sample size:                      46787
## Number of independent variables:  23
## Mtry:                             2
## Target node size:                 33
## Variable importance mode:         none
## Splitrule:                        gini
## OOB prediction error (Brier s.):  0.2011443
```

```r
#Predict on test data
RF100Tune1.5dsPred <- predict(final_wfRF100Tune1.5dsmodel, new_data = clean_test, type = 'prob')
#Generate Submission
submission <- data.frame(SK_ID_CURR = as.integer(clean_test$SK_ID_CURR), TARGET = RF100Tune1.5dsPred)
summary(submission)
```

```
##    SK_ID_CURR     TARGET..pred_No  TARGET..pred_Yes
##  Min.   :100001   Min.   :0.1924   Min.   :0.07991
##  1st Qu.:188558   1st Qu.:0.5440   1st Qu.:0.25161
##  Median :277549   Median :0.6575   Median :0.34245
##  Mean   :277797   Mean   :0.6395   Mean   :0.36052
##  3rd Qu.:367556   3rd Qu.:0.7484   3rd Qu.:0.45602
##  Max.   :456250   Max.   :0.9201   Max.   :0.80765
```

```r
submission <- submission %>% rename ("TARGET" = "TARGET..pred_Yes") %>% select(c(SK_ID_CURR, TARGET))
write.csv(submission, file = 'RF100Tune1.5dsmodel.csv', row.names = FALSE)

toc()
```

```
## 41.723 sec elapsed
```

#RF 100 Tune Grid, No DownSample

```r
tic()
set.seed(123)
#Create Random Forest Train Data set
RF100TuneNods <- clean_train

#Split Data 75% Training 25% Test
RF100TuneNodssplit <- initial_split(RF100TuneNods, strata = TARGET, prop = 0.75)
RF100TuneNodstrain <- training(RF100TuneNodssplit)
RF100TuneNodstest <- testing(RF100TuneNodssplit)

#Create Recipe with 1.5 Downsample
RF100TuneNodsrecipe <- recipe(TARGET ~ ., data = RF100TuneNodstrain)

#Create Model with Hyperperameter Tuning (Set Perameters based on tuning to save run time)
RF100TuneNodsmodel <- rand_forest(
  mtry = (2),
  trees = (1003),
  min_n = (33)) %>%
  set_mode("classification") %>%
  set_engine("ranger")

#Set 5 fold CV
RF100TuneNodsfolds <- vfold_cv(RF100TuneNodstrain, v=5)

doParallel::registerDoParallel(cores = 8)

#Create Workflow
RF100TuneNods_WF <- workflow () %>%
    add_model(RF100TuneNodsmodel) %>%
    add_recipe(RF100TuneNodsrecipe)

#Tune Model with grid = 50 (very time intensive)
# 100 Grid gives mtry (2) trees (1003) min_n (33)
#RF100TuneNods_tune <- RF100TuneNods_WF %>% tune_grid( resamples = RF100TuneNodsfolds, grid = 100)

#Create Best model using ROC_AUC
#best_modelRF100TuneNods <- RF100TuneNods_tune %>% select_best ("roc_auc")
#best_modelRF100TuneNods

#Create final WF based on best model
finalWFRF100TuneNodsmodel <- RF100TuneNods_WF %>% finalize_workflow(RF100TuneNodsmodel)

#Create final fit
final_fitRF100TuneNodsmodel <- finalWFRF100TuneNodsmodel %>% last_fit(split = RF100TuneNodssplit)

#Run Final Fit
final_fitRF100TuneNodsmodel %>% collect_metrics()
```

```
## # A tibble: 2 x 4
##   .metric  .estimator .estimate .config
##   <chr>    <chr>          <dbl> <chr>
## 1 accuracy binary         0.921 Preprocessor1_Model1
## 2 roc_auc  binary         0.738 Preprocessor1_Model1
```

```
#Create Confusion matrix
final_fitRF100TuneNodsmodel %>%
  collect_predictions()%>%
  conf_mat(truth = TARGET, estimate = .pred_class)
```

```
##           Truth
## Prediction    No   Yes
##        No  70767  6108
##        Yes     0     2
```

```
#extract WF
final_wfRF100TuneNodsmodel <- final_fitRF100TuneNodsmodel %>%
  extract_workflow()
final_wfRF100TuneNodsmodel
```

```
## == Workflow [trained] ==========================================================
## Preprocessor: Recipe
## Model: rand_forest()
##
## -- Preprocessor ----------------------------------------------------------------
## 0 Recipe Steps
##
## -- Model -----------------------------------------------------------------------
## Ranger result
##
## Call:
##  ranger::ranger(x = maybe_data_frame(x), y = y, mtry = min_cols(~(2),      x), num.trees = ~(1003),
##
## Type:                             Probability estimation
## Number of trees:                  1003
## Sample size:                      230630
## Number of independent variables:  23
## Mtry:                             2
## Target node size:                 33
## Variable importance mode:         none
## Splitrule:                        gini
## OOB prediction error (Brier s.):  0.06933137
```

```
#Predict on test data
RF100TuneNodsPred <- predict(final_wfRF100TuneNodsmodel, new_data = clean_test, type = 'prob')
#Generate Submission
submission <- data.frame(SK_ID_CURR = as.integer(clean_test$SK_ID_CURR), TARGET = RF100TuneNodsPred)
summary(submission)
```

```
##    SK_ID_CURR     TARGET..pred_No  TARGET..pred_Yes
##  Min.   :100001   Min.   :0.5116   Min.   :0.00888
##  1st Qu.:188558   1st Qu.:0.8923   1st Qu.:0.04284
##  Median :277549   Median :0.9327   Median :0.06727
##  Mean   :277797   Mean   :0.9160   Mean   :0.08396
##  3rd Qu.:367556   3rd Qu.:0.9572   3rd Qu.:0.10774
##  Max.   :456250   Max.   :0.9911   Max.   :0.48841
```

```
submission <- submission %>% rename ("TARGET" = "TARGET..pred_Yes") %>% select(c(SK_ID_CURR, TARGET))
write.csv(submission, file = 'RF100TuneNodsmodel.csv', row.names = FALSE)

toc()
```

```
## 199.827 sec elapsed
```

#Summary When running the Random Forest Models we attempted 2 different modeling variations, different size tuning grids and with or with out downsampling. With using different size tuning grids we found that 50 was the spot where we achieved the highest AUC. It is also should be noted that increasing the Tuning grid number added significant computational time. In all of our models we also found that DownSampeling was preferred over not. With out downsampling we found the model did not return nearly any predictions of default. This gave our best model from the tests to be a tuning grid of 50 with downsampling of 1.5, which gave us an AUC of .743 and an Accuracy of .815.

#Results Summary

```
#1 Tune Grid 1.5 DS Results
final_fitRF1Tune1.5dsmodel %>% collect_metrics()
```

```
## # A tibble: 2 x 4
##   .metric  .estimator .estimate .config
##   <chr>    <chr>          <dbl> <chr>
## 1 accuracy binary         0.921 Preprocessor1_Model1
## 2 roc_auc  binary         0.724 Preprocessor1_Model1
```

```r
final_fitRF1Tune1.5dsmodel %>%
  collect_predictions()%>%
  conf_mat(truth = TARGET, estimate = .pred_class)
```

```
##           Truth
## Prediction    No   Yes
##        No  70697  6037
##        Yes    70    73
```

```r
#1 Tune Grid No DS Results
final_fitRF1TuneNodsmodel %>% collect_metrics()
```

```
## # A tibble: 2 x 4
##   .metric  .estimator .estimate .config
##   <chr>    <chr>          <dbl> <chr>
## 1 accuracy binary         0.789 Preprocessor1_Model1
## 2 roc_auc  binary         0.733 Preprocessor1_Model1
```

```r
final_fitRF1TuneNodsmodel %>%
  collect_predictions()%>%
  conf_mat(truth = TARGET, estimate = .pred_class)
```

```
##           Truth
## Prediction    No   Yes
##        No  57597  3014
##        Yes 13170  3096
```

```r
#50 Tune Grid 1.5 DS Results
final_fitRF50Tune1.5dsmodel %>% collect_metrics()
```

```
## # A tibble: 2 x 4
##   .metric  .estimator .estimate .config
##   <chr>    <chr>          <dbl> <chr>
## 1 accuracy binary         0.815 Preprocessor1_Model1
## 2 roc_auc  binary         0.743 Preprocessor1_Model1
```

```r
final_fitRF50Tune1.5dsmodel %>%
  collect_predictions()%>%
  conf_mat(truth = TARGET, estimate = .pred_class)
```

```
##           Truth
## Prediction    No   Yes
##        No  59817  3278
##        Yes 10950  2832
```

```r
#50 Tune Grid No DS Results
final_fitRF50TuneNodsmodel %>% collect_metrics()
```

```
## # A tibble: 2 x 4
##   .metric  .estimator .estimate .config
##   <chr>    <chr>          <dbl> <chr>
## 1 accuracy binary         0.921 Preprocessor1_Model1
## 2 roc_auc  binary         0.738 Preprocessor1_Model1
```

```
final_fitRF50TuneNodsmodel %>%
  collect_predictions()%>%
  conf_mat(truth = TARGET, estimate = .pred_class)
```

```
##           Truth
## Prediction    No   Yes
##        No  70767  6108
##        Yes     0     2
```

```
#100 Tune Grid 1.5 DS Results
final_fitRF100Tune1.5dsmodel %>% collect_metrics()
```

```
## # A tibble: 2 x 4
##   .metric  .estimator .estimate .config
##   <chr>    <chr>          <dbl> <chr>
## 1 accuracy binary         0.815 Preprocessor1_Model1
## 2 roc_auc  binary         0.742 Preprocessor1_Model1
```

```
final_fitRF100Tune1.5dsmodel %>%
  collect_predictions()%>%
  conf_mat(truth = TARGET, estimate = .pred_class)
```

```
##           Truth
## Prediction    No   Yes
##        No  59833  3275
##        Yes 10934  2835
```

```
#100 Tune Grid No DS Results
final_fitRF100TuneNodsmodel %>% collect_metrics()
```

```
## # A tibble: 2 x 4
##   .metric  .estimator .estimate .config
##   <chr>    <chr>          <dbl> <chr>
## 1 accuracy binary         0.921 Preprocessor1_Model1
## 2 roc_auc  binary         0.738 Preprocessor1_Model1
```

```
final_fitRF100TuneNodsmodel %>%
  collect_predictions()%>%
  conf_mat(truth = TARGET, estimate = .pred_class)
```

```
##           Truth
## Prediction    No   Yes
##        No  70767  6108
##        Yes     0     2
```