Module 5 Assignment

In this assignment you will complete a variety of tasks related to binary classification stock market data. A full description of the dataset is available here: https://www.kaggle.com/cnic92/200-financial-indicators-of-us-stocks-20142018.
Deliverable: All of your work for this assignment should be done in an R Markdown document. Knit your document into a Word file and submit the Word file as the deliverable for this assignment.

Libraries: For this assignment you will need the following libraries: tidyverse, caret, nnet, rpart, ranger, caretEnsemble, and xgboost.

Read the "2018Fin.csv" data file into a data frame called "fin". Ignore the warning about missing column names.

Examine the structure and summary of the data. There are 225 columns, so PLEASE comment out your structure and summary commands before knitting. **I will be mad and take points off if you leave the str and summary commands in :)**

This is a fun dataset to play around with due to the large number of interesting columns. However, to simplify things a bit, we are only going to look at a subset of the columns. Select the following columns: Class, Revenue Growth, EPS Diluted, EBITDA Margin, priceBookValueRatio, debtEquityRatio, debtRatio, PE ratio, Sector, 5Y Revenue Growth (per Share), returnOnAssets, returnOnEquity, returnOnCapitalEmployed, and quickRatio. Be aware that reducing the number of variables will likely have an adverse effect on our model quality.

Remember that when you use the select function to select these columns you will need to put variable names that have spaces inside backticks. An example would be `Revenue Growth`. These are NOT apostrophes. These are backticks (located on the same key on your keyboard as the tilde ~).

Don't know what these terms mean? They are commonly used in finance, so it's easy to find defintions via Google.

Convert the Class variable (this will be your response variable in our models) and Sector variable to factors. The data is from 2018 and the Class variable indicates if the stock increased in 2019.

Recode the Class factor to convert 0 to No and 1 to Yes.

Let's do an aggressive cleaning of missing data via row-wise deletion (using drop_na). Verify that your drop_na was successful before proceeding as most of our methods DO NOT work correctly or at in the presence of missingness.

You should 2,360 rows of data remaining after the drop_na().

We should probably look at outliers in this dataset before we begin our model building. To save us time, I did this part for you (use the code below). You could certainly choose other values that would make sense, but let's go with my approach.

```
fin = fin %>% filter(`Revenue Growth` <= 1)
fin = fin %>% filter(`EPS Diluted` >= -10, `EPS Diluted` <= 10)
fin = fin %>% filter(`EBITDA Margin` >= -5, `EBITDA Margin` <= 5)
fin = fin %>% filter(priceBookValueRatio >= 0, priceBookValueRatio <= 5)
fin = fin %>% filter(debtEquityRatio >= -1, debtEquityRatio <= 2)
fin = fin %>% filter(debtRatio <= 1)
fin = fin %>% filter(`PE ratio` <= 100)
fin = fin %>% filter(returnOnAssets >= -5, returnOnAssets <= 5)
fin = fin %>% filter(returnOnEquity >= -5, returnOnEquity <= 5)
fin = fin %>% filter(returnOnCapitalEmployed >= -2, returnOnCapitalEmployed <= 2)
fin = fin %>% filter(quickRatio <= 20)
```

You should have 1,362 rows of data after this outlier elimination.

Task 1: Split the data into training and testing sets. Your training set should have 70% of the data. Use a random number (set.seed) of 12345. Recall that your predictor variable is Class. **Hint: Use dplyr::slice to avoid conflicts with xgboost package**

Task 2: Create a neural network to predict stock performance (given by the Class variable). Use a grid to search sizes 1 through an appropriate maximum (recall that you can get the "rule of thumb" size by counting the number of predictor variables, including factor levels) and decay rates of 0.5, 0.1, 1e-2, 1e-3, 1e-4, 1e-5, 1e-6, 1e-7.

Use caret to implement 10-fold k-fold cross-validation.

Use a random number seed of 1234.

To suppress all of the text describing model convergence, add the command: trace = FALSE after verbose = FALSE in your train function.

Note: This model may take some time to run! Be patient, particularly if you are using an older computer. It took about 8 minutes to run on a higher-end machine.

Task 3: Use your model from Task 2 to develop predictions on the training set. Use caret's confusionMatrix function to evaluate the model quality. Comment on the model quality.

Task 4: Use your model from Task 3 to develop predictions on the testing set. Use the confusionMatrix command to assess and comment on the quality of the model.

Task 5: Let's build an ensemble model. The ensemble model will contain the following models: a logistic regression model (use all variables as predictors), a classification tree, a random forest (with the range package), and a new neural network.

Use 5 folds to save time (rather than 10). Your random number seed should be set to 111. Maximizing AUC will be our objective.

Do not forget to include the index =... line as you define your control object.

For your tuneList (inside your caretList function), use the code below. This will perform a pretty thorough parameter tuning for tne neural network and ranger models. NOTE: You will still need a methodList that includes your logistic regression and classification tree models (this list should be before your tuneList in the code).

Use the code below to set up the tuneList part of your caretList:

```
tuneList=list(
      ranger = caretModelSpec(method="ranger", max.depth = 5, tuneGrid =
                        expand.grid(mtry = 1:13,
                                    splitrule = c("gini","extratrees","hellinger"),
                                    min.node.size=1:5)),
      nn = caretModelSpec(method="nnet", tuneGrid =
                        expand.grid(size = 1:23,
                        decay = c(0.5, 0.1, 1e-2, 1e-3, 1e-4, 1e-5, 1e-6, 1e-7)),trace=FALSE)))
```

Note: On a high-end machine, this model took about 11 minutes to run.

Task 6: Discuss the correlation that exists (or does not exist) between the models in the ensemble.

Task 7: Build the ensemble model (using the caretEnsemble function) and comment on the results. How does the model perform on the training and testing sets?

Task 8: Build a stacked model and use the stacked model to make predictions on the training and testing set. Remember to use a new trainControl object. Use 10 fold k-fold cross-validation in this object.

Task 9: Build an xgboost model. First you will need to create new training and tests sets in which the categorical variables are one-hot encoded to dummy variables. Use select to exclude the NEGATIVE level of the Class variable from your training and testing sets.

For your xgboost model tune as follows (same as we did in the lecture):
* nrounds = 100
* max_depth = 1,2,3,4 * eta = 0.01, 0.1, 0.2, 0.3
* gamma = 0
* colsample_bytree = 0.6, 0.8, 1
* min_child_weight = 1
* subsample = 0.8, 1

Use 5 fold k-fold cross-validation with a random number seed of 999. **Note: This model will may take some time to run. On a reasonably modern machine, it took about 25 seconds to run**

Develop predictions on the training and testing sets.