Q35

Rahul Atre

2023-11-17

Q35 [Ensemble Learning]

Apply boosting, bagging, and random forests to the Weekly dataset. We build models to try to predict whether the stock market will be up or down based on various lag indicators (we will not be using the Year as it's already been shown to be strongly correlated with the Volume, nor the stock market value the-day-of). Be sure to fit the models on a training set and to evaluate their performance on a test set. How accurate are the results compared to simple methods like linear or logistic regression? Which of these approaches yields the best performance?

Boosting:

```
set.seed(144) #Setting seed for reproducibility

weekly_df = read.csv("weekly.csv")
weekly_df = na.omit(weekly_df)

#Converting the values of Direction to numeric (so that gbm can classify correctly)
weekly_df$Direction <- as.factor(mapvalues(weekly_df$Direction, c("Up", "Down"), c("1","0")))
weekly_df$Direction <- as.character(weekly_df$Direction)

#Let us do a 80/20 split on the training/test data
split = sample.split(weekly_df$Direction, SplitRatio = 0.8)
training_set = weekly_df[split, ]
test_set = weekly_df[!split, ]

#Apply Gradient Boosting to Weekly dataset

# Fit a boosting model to the training data
weekly_boosting_model = gbm(Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume, data = training_set,

weekly_prediction = round(predict(weekly_boosting_model, test_set))
```

```
## Using 100 trees...
```

```
weekly_accuracy = sum(weekly_prediction == test_set$Direction) / nrow(test_set)
weekly_accuracy
```

```
## [1] 0.5458716
```

The accuracy rate using Boosting is 54.58%.

Bagging:

```r
set.seed(144) #Setting seed for reproducibility

weekly_df = read.csv("weekly.csv")
weekly_df = na.omit(weekly_df)
weekly_df$Direction = as.factor(weekly_df$Direction) #Convert to a factor

n = nrow(weekly_df) #number of rows

#Let us do a 80/20 split on the training/test data

training_set = weekly_df[sample(n, 0.8 * n), ]
test_set = weekly_df[setdiff(1:n, rownames(training_set)), ]

weekly_bagging_model = bagging(Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume, data = training_s

weekly_prediction = predict(weekly_bagging_model, test_set)

weekly_accuracy = sum(weekly_prediction == test_set$Direction) / nrow(test_set)
weekly_accuracy
```

```
## [1] 0.5
```

The accuracy rate using Bagging is 50%.

Random Forest:

```r
set.seed(144) #Setting seed for reproducibility

weekly_df = read.csv("weekly.csv")
weekly_df$Direction = as.factor(weekly_df$Direction) #Convert to a factor

n = nrow(weekly_df) #number of rows

#Let us do a 80/20 split on the training/test data

training_set = weekly_df[sample(n, 0.8 * n), ]
test_set = weekly_df[setdiff(1:n, rownames(training_set)), ]

weekly_randomForest_model = randomForest(Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume, data =

weekly_prediction = predict(weekly_randomForest_model, test_set)

weekly_accuracy = sum(weekly_prediction == test_set$Direction) / nrow(test_set)
weekly_accuracy
```

```
## [1] 0.4862385
```

The accuracy rate using Random Forest is 48.62%.

Now, let us fit a linear model and decide which one is the best for the weekly.csv dataset:

```r
set.seed(144) #Setting seed for reproducibility

weekly_df = read.csv("weekly.csv")
weekly_df = na.omit(weekly_df)

#Converting the values of Direction to numeric (so that gbm can classify correctly)
weekly_df$Direction <- as.factor(mapvalues(weekly_df$Direction, c("Up", "Down"), c("1","0")))
weekly_df$Direction <- as.character(weekly_df$Direction)

n = nrow(weekly_df) #number of rows

#Let us do a 80/20 split on the training/test data

training_set = weekly_df[sample(n, 0.8 * n), ]
test_set = weekly_df[setdiff(1:n, rownames(training_set)), ]

weekly_lin_model = lm(Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume, data = training_set)

weekly_prediction = round(predict(weekly_lin_model, test_set))

weekly_accuracy = sum(weekly_prediction == test_set$Direction) / nrow(test_set)
weekly_accuracy
```

```
## [1] 0.5275229
```

The accuracy rate with linear modeling is 52.75%.

Overall, all four methods (boosting, bagging, random forests, linear) seem to have similar accuracy rates at roughly 50%. The most accurate and best performing model was using the boosting approach at 54.58%, and the least accurate was the random forest model, with an accuracy rate of 48.62%.