QDA and XGBoost Comparison for Cattle Behavior Classification

Joe Champion and Thea Sukianto

6/17/2021

```
library(tidyverse)
```

```
## Warning: package 'tibble' was built under R version 4.0.5
```

```
library(xgboost)
```

```
## Warning: package 'xgboost' was built under R version 4.0.5
```

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 4.0.5
```

```
library(vip)
```

```
## Warning: package 'vip' was built under R version 4.0.5
```

```
library(lubridate)
```

Data Prep

Read in accelerometer data for all cows in November 2017.

```
cow_accel_file <- list.files("rf_2018", pattern="All Cows", full.names = TRUE)
cow_accel_raw <- readxl::read_excel(cow_accel_file)
head(cow_accel_raw)
```

```
## # A tibble: 6 x 15
##   Order Date                Time                Observed      X     Y     Z
##   <dbl> <dttm>              <dttm>              <chr>     <dbl> <dbl> <dbl>
## 1     1 2017-10-30 00:00:00 1899-12-31 09:59:00 G        -0.637 0.635 -2.32
## 2     2 2017-10-30 00:00:00 1899-12-31 09:59:00 G        -0.584 0.669 -2.37
## 3     3 2017-10-30 00:00:00 1899-12-31 09:59:00 G        -0.530 0.718 -2.36
## 4     4 2017-10-30 00:00:00 1899-12-31 10:00:00 G        -0.569 0.657 -2.41
## 5     5 2017-10-30 00:00:00 1899-12-31 10:00:00 G        -0.530 0.648 -2.46
## 6     6 2017-10-30 00:00:00 1899-12-31 10:00:00 G        -0.552 0.682 -2.38
## # ... with 8 more variables: MIMEAN <dbl>, SMAMEAN <dbl>, XYPLZ <dbl>,
## #   XZ <dbl>, XPlYPlZ <dbl>, MIMAX <dbl>, SMAMAX <dbl>, COWID <dbl>
```

Compute additional physics measures from X, Y, and Z acceleration, where t_{i+1} and t_i are successive timestamps:

Column	Formula
XY.Z	$XY + Z$
XZ	$X * Z$
X.Y.Z	$X + Y + Z$
norm_accel	$\sqrt{(X^2 + Y^2 + Z^2)}$

Column	Formula
tiltX	$\arcsin(\frac{X}{\sqrt{(X^2+Y^2+Z^2)}})$
tiltY	$\arcsin(\frac{Y}{\sqrt{(X^2+Y^2+Z^2)}})$
tiltZ	$\arcsin(\frac{Z}{\sqrt{(X^2+Y^2+Z^2)}})$
pitch	$\arctan(\frac{X}{Y^2+Z^2})$
roll	$\arctan(\frac{Y}{X^2+Z^2})$
yaw	$\arctan(\frac{Z}{X^2+Y^2})$
jerkX	$\frac{X_{t_{i+1}}-X_{t_i}}{t_{i+1}-t_i}$
jerkY	$\frac{Y_{t_{i+1}}-Y_{t_i}}{t_{i+1}-t_i}$
jerkZ	$\frac{Z_{t_{i+1}}-Z_{t_i}}{t_{i+1}-t_i}$
tvelX	$(t_{i+1}-t_i)(-X_1+\sum_{k=1}^{i} X_k)$
tvelY	$(t_{i+1}-t_i)(-Y_1+\sum_{k=1}^{i} Y_k)$
tvelZ	$(t_{i+1}-t_i)(-Z_1+\sum_{k=1}^{i} Z_k)$
tdispX	$(t_{i+1}-t_i)(\sum_{k=1}^{i} tvelX)$
tdispY	$(t_{i+1}-t_i)(\sum_{k=1}^{i} tvelY)$
tdispZ	$(t_{i+1}-t_i)(\sum_{k=1}^{i} tvelZ)$

```
clean_accel_data <- function(data){
  # before using, prep the data to format: datetime, X, Y, Z, Behavior
  data %>%
    dplyr::mutate(
      elapsed = lubridate::seconds(datetime - dplyr::lag(datetime)),
      elapsed = as.numeric(gsub("S", "", elapsed)),
      elapsed = replace_na(elapsed, 0),
      XY.Z = X*Y+Z,
      XZ = X*Z,
      X.Y.Z = X+Y+Z,
      norm_accel = sqrt(X^2+Y^2+Z^2),
      tiltX = asin(X/norm_accel),
      tiltY = asin(Y/norm_accel),
      tiltZ = acos(Z/norm_accel),
      pitch = atan(X/(Y^2+Z^2)),
      roll = atan(Y/(X^2+Z^2)),
      yaw = atan(Z /(X^2+Y^2)),
      jerkX = (X-dplyr::lag(X))/elapsed, #meters/sec^3
      jerkY = (Y -dplyr::lag(Y))/elapsed,
      jerkZ = (Z-dplyr::lag(Z))/elapsed,
      tvelX = (cumsum(X) - dplyr::first(X))*elapsed, #meters/sec
      tvelY = (cumsum(Y) - dplyr::first(Y))*elapsed,
      tvelZ = (cumsum(Z) - dplyr::first(Z))*elapsed,
```

```
      tdispX = cumsum(tvelX)*elapsed, # meters
      tdispY = cumsum(tvelY)*elapsed,
      tdispZ = cumsum(tvelZ)*elapsed
    ) %>%
    mutate(across(starts_with("jerk"), function(x){ replace_na(x,0)}) ) %>% # replace missing values wi
    filter(abs(elapsed) < 60, !is.infinite(jerkX)) # limit dataset to rows with time differences < 60 s
}
```

Assume the sampling rate is 12 Hz and adjust timestamps from HH:MM:00 accordingly.

```
cow_accel <- cow_accel_raw %>%
  dplyr::rename(Behavior = Observed) %>%
  dplyr::mutate(Behavior = as.factor(Behavior),
                Time = format(strptime(Time, format="%Y-%m-%d %H:%M:%S"), format="%H:%M:%S"),
                datetime = as.POSIXct(paste(Date, Time))) %>%
  dplyr::group_by(datetime, COWID) %>%
  dplyr::mutate(Offset = row_number()-n()-1,
                datetime = datetime + minutes(1) + seconds(5*Offset)) %>%
  dplyr::filter(!duplicated(datetime)) %>%
  dplyr::ungroup() %>%
  dplyr::select(COWID, datetime, Behavior, X, Y, Z)

cow_accel <- cow_accel %>% clean_accel_data()
```

Randomly partition half the data for model training and the other half for testing, ignoring autocorrelated variables (those including `elapsed` in the formula).

For all models, the classification task is to predict `Behavior` (levels: G, W, R).

```
set.seed(1812)

cow_accel2 <- cow_accel %>%
  mutate(rowid = 1:nrow(.)) %>%
  dplyr::select(rowid, Behavior, COWID, X, Y, Z, XY.Z, XZ, norm_accel:yaw)

accel_train2 <-  cow_accel2 %>%
  group_by(COWID) %>%
  sample_frac(.5) %>% ungroup()

accel_test2 <- cow_accel2 %>%
  filter(!rowid %in% accel_train2$rowid)
```

Create 5 folds from the training data for cross-validation.

```
train.control <- trainControl(method = "cv", number = 5)
```

Modeling (5-variable set)

Select X, Y, Z, XY.Z, and XZ as predictor variables.

```
cow_accel_sprinkle <- accel_train2 %>%
  select(X,Y,Z,XY.Z, XZ, Behavior)
```

QDA

```
model_qda_sprinkle <- train(Behavior ~ .,
                    data = cow_accel_sprinkle, method = "qda",
                    trControl = train.control)
print(model_qda_sprinkle)
```

```
## Quadratic Discriminant Analysis
##
## 17793 samples
##     5 predictor
##     3 classes: 'G', 'R', 'W'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 14235, 14234, 14234, 14235, 14234
## Resampling results:
##
##   Accuracy   Kappa
##   0.5624687  0.3246782
```

```
confusionMatrix(model_qda_sprinkle)
```

```
## Cross-Validated (5 fold) Confusion Matrix
##
## (entries are percentual average cell counts across resamples)
##
##           Reference
## Prediction    G    R    W
##          G 17.4  3.6  3.6
##          R  8.2  7.8  3.3
##          W 11.0 14.1 31.0
##
##  Accuracy (average) : 0.5625
```

XGBoost

XGBoost/eXtreme Gradient Boosting Chen and Guestrin 2016 is an algorithm that builds tree ensemble models that are scalable to large datasets and less prone to overfitting than Random Forest. In gradient boosting, decision trees are added one-by-one to the overall model by minimizing the loss function (i.e. traversing the parameter space in the direction of the loss function's most negative gradient). XGBoost improves on gradient boosting in that it is "sparsity-aware" - it learns the pattern of missing values in the data and finds the optimal "default" branch split for when the predictor is missing. In addition, to reduce overfitting, XGBoost introduces an additional regularization term, shrinkage (preventing individual trees from dominating), and feature subsampling.

```
gboost_sprinkle_train <- train(Behavior ~ .,
                           data = accel_train2 %>%
                             dplyr::select(-c( COWID, rowid)), method = "xgbTree",
                           trControl = train.control)
```

For the sake of efficiency, load the fitted model. The code used to train the model is listed above.

```
load("supervised/gboost_sprinkle_train2.rda")
confusionMatrix(gboost_sprinkle_train2)
```

```
## Cross-Validated (5 fold) Confusion Matrix
##
```

```
## (entries are percentual average cell counts across resamples)
##
##           Reference
## Prediction    G    R    W
##          G 29.5  1.5  4.1
##          R  2.1 22.9  2.1
##          W  4.3  2.0 31.5
##
##  Accuracy (average) : 0.8384
```

Modeling (Expanded variable set)

QDA

```
model_qda_expanded <- train(Behavior ~ .,
                            data = accel_train2 %>% dplyr::select(-c( COWID, rowid, XY.Z, XZ)), method =
                            trControl = train.control)
print(model_qda_expanded)
```

```
## Quadratic Discriminant Analysis
##
## 17793 samples
##    10 predictor
##     3 classes: 'G', 'R', 'W'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 14235, 14234, 14235, 14234, 14234
## Resampling results:
##
##   Accuracy   Kappa
##   0.6152426  0.4032459
```

Expanding the variable set increases QDA performance on the training data by about 6%.

```
confusionMatrix(model_qda_expanded)
```

```
## Cross-Validated (5 fold) Confusion Matrix
##
## (entries are percentual average cell counts across resamples)
##
##           Reference
## Prediction    G    R    W
##          G 27.9  8.0  8.3
##          R  1.7  8.5  4.5
##          W  7.1  8.9 25.1
##
##  Accuracy (average) : 0.6152
```

XGBoost

```
gboost_all_train <- train(Behavior ~ .,
                          data = accel_train2 %>%
                            dplyr::select(-c( COWID, rowid, XY.Z, XZ)), method = "xgbTree",
                          trControl = train.control)
```

Load the fitted model:

```
load("supervised/gboost_all_train3.rda")
confusionMatrix(gboost_all_train3)
```

```
## Cross-Validated (5 fold) Confusion Matrix
##
## (entries are percentual average cell counts across resamples)
##
##           Reference
## Prediction    G    R    W
##          G 30.7  1.1  4.3
##          R  1.3 23.7  1.6
##          W  3.9  1.6 31.7
##
##  Accuracy (average) : 0.8609
```

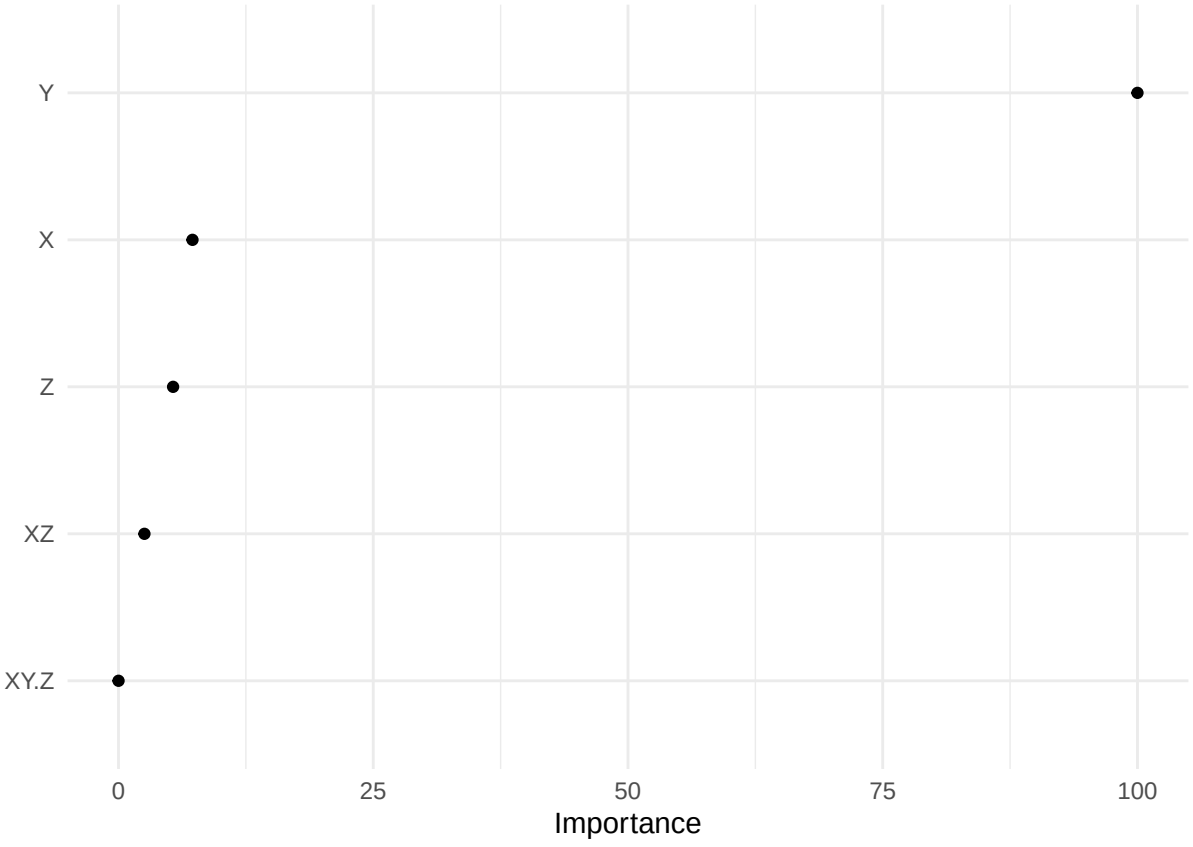
XGBoost performance is improved by around 2% on the training data.

Variable Importance

XGBoost on 5-variable set

Y acceleration is by far the most influential predictor variable.

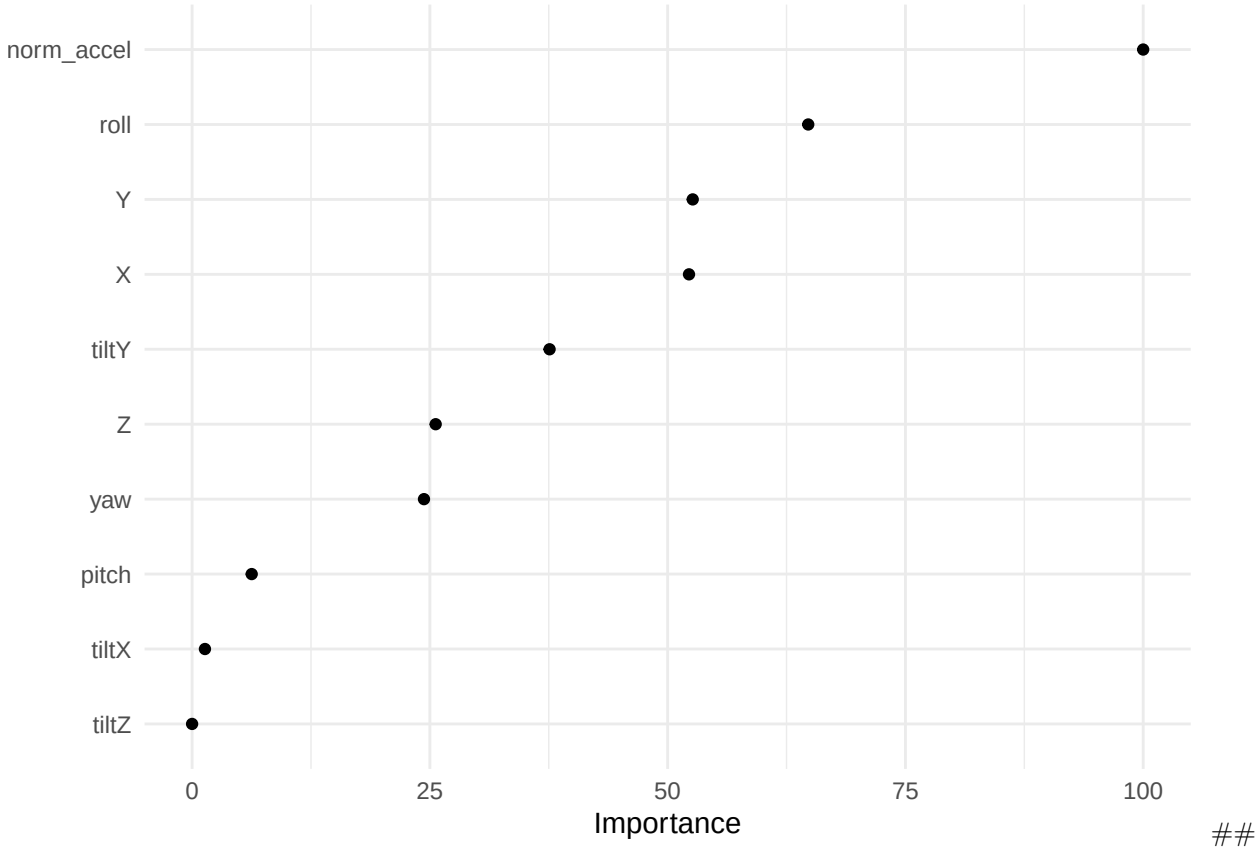
```
vip(gboost_sprinkle_train2, geom="point") + theme_minimal()
```



XGBoost on expanded variable set

When the time-dependent variables are removed, the top 5 most influential predictor variables are the norm of all acceleration directions, roll, Y acceleration, X acceleration, and Y tilt.

```
vip(gboost_all_train3, geom="point", num_features = 20) + theme_minimal()
```



##

Test Performance

QDA on 5-variable set

The performance remains about the same between the training and test datasets.

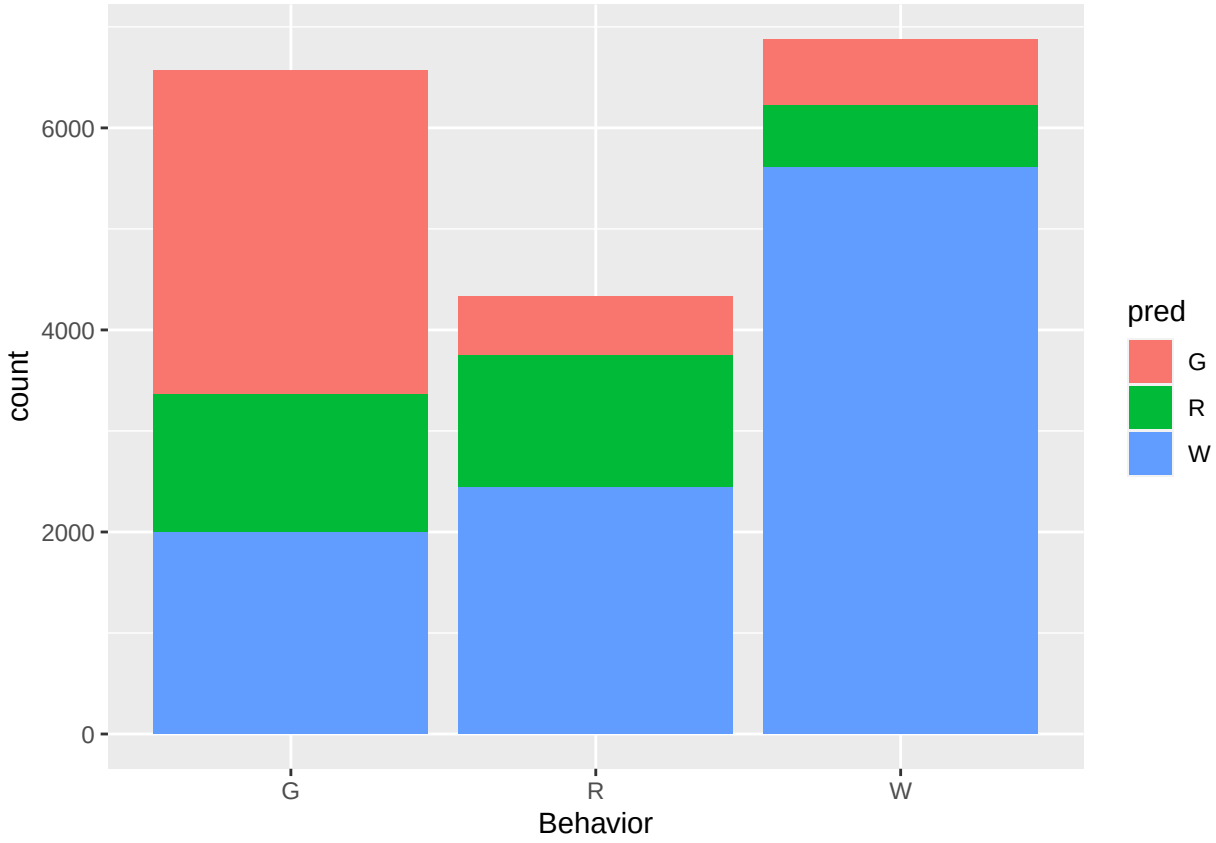
```
accel_sprinkle_test <- accel_test2 %>% modelr::add_predictions(model_qda_sprinkle)
confusionMatrix(accel_sprinkle_test$pred, accel_sprinkle_test$Behavior)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    G    R    W
##          G 3212  594  656
##          R 1371 1305  614
##          W 1992 2442 5608
##
## Overall Statistics
##
##                Accuracy : 0.569
##                  95% CI : (0.5617, 0.5763)
##     No Information Rate : 0.3865
##     P-Value [Acc > NIR] : < 2.2e-16
```

```
##
##                   Kappa : 0.3309
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: G Class: R Class: W
## Sensitivity            0.4885  0.30062   0.8154
## Specificity            0.8886  0.85245   0.5938
## Pos Pred Value         0.7199  0.39666   0.5585
## Neg Pred Value         0.7477  0.79068   0.8362
## Prevalence             0.3695  0.24396   0.3865
## Detection Rate         0.1805  0.07334   0.3152
## Detection Prevalence   0.2508  0.18489   0.5643
## Balanced Accuracy      0.6885  0.57654   0.7046
```

Furthermore, it seems that QDA classifies walking the best.

```
accel_sprinkle_test %>%
  ggplot(aes(x = Behavior, fill = pred)) +
  geom_bar(position = "stack")
```



XGBoost on 5-variable set

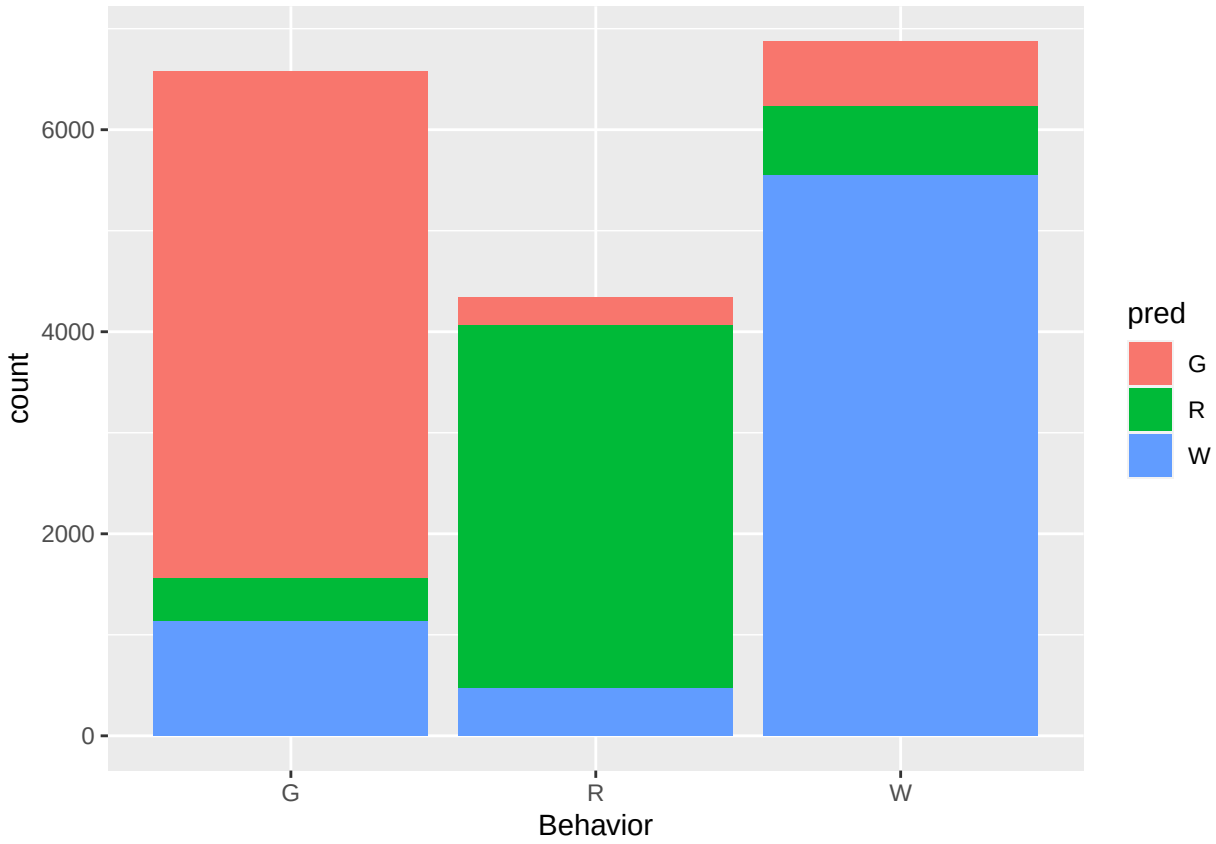
The accuracy of the XGBoost model decreases by approximately 4% on the test data.

```
gboost_sprinkle_test <- accel_test2 %>% modelr::add_predictions(gboost_sprinkle_train2)
confusionMatrix(gboost_sprinkle_test$pred, gboost_sprinkle_test$Behavior)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    G    R    W
##          G 5013  273  648
##          R  432 3601  683
##          W 1130  467 5547
##
## Overall Statistics
##
##                Accuracy : 0.7958
##                  95% CI : (0.7898, 0.8017)
##     No Information Rate : 0.3865
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6892
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: G Class: R Class: W
## Sensitivity            0.7624   0.8295   0.8065
## Specificity            0.9179   0.9171   0.8537
## Pos Pred Value         0.8448   0.7636   0.7765
## Neg Pred Value         0.8683   0.9434   0.8750
## Prevalence             0.3695   0.2440   0.3865
## Detection Rate         0.2817   0.2024   0.3117
## Detection Prevalence   0.3335   0.2650   0.4015
## Balanced Accuracy      0.8402   0.8733   0.8301
```

Compared to QDA, the XGBoost classification accuracy is more balanced with no single class having a visibly greater accuracy.

```
gboost_sprinkle_test %>%
  ggplot(aes(x = Behavior, fill = pred)) +
  geom_bar(position = "stack")
```



QDA on expanded variable set

QDA performance with the expanded predictor variable set also remains about the same between train and test.

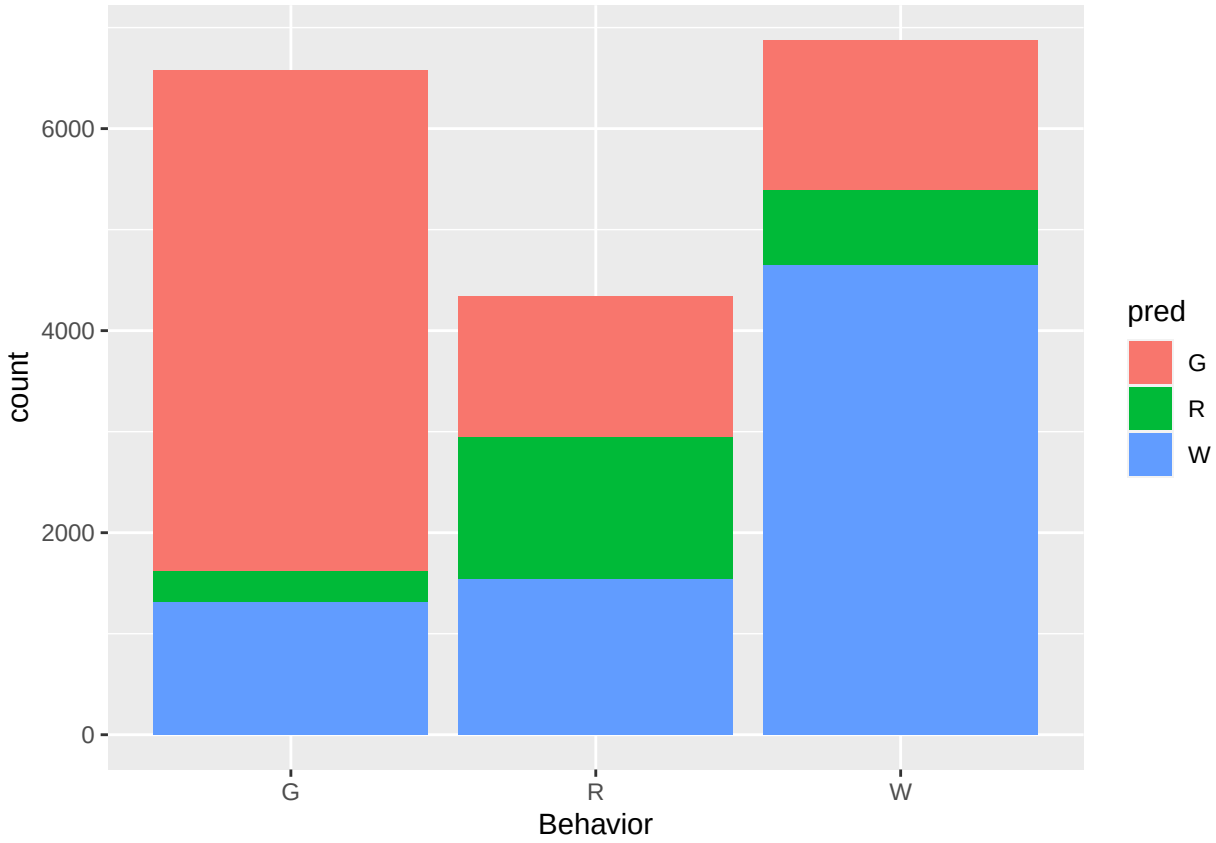
```
accel_expanded_test <- accel_test2 %>% modelr::add_predictions(model_qda_expanded)
confusionMatrix(accel_expanded_test$pred, accel_expanded_test$Behavior)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    G    R    W
##          G 4960 1399 1490
##          R  302 1401  738
##          W 1313 1541 4650
##
## Overall Statistics
##
##                Accuracy : 0.6188
##                  95% CI : (0.6116, 0.6259)
##     No Information Rate : 0.3865
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.4049
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
```

```
##                      Class: G Class: R Class: W
## Sensitivity            0.7544  0.32274   0.6761
## Specificity            0.7425  0.92269   0.7385
## Pos Pred Value         0.6319  0.57395   0.6197
## Neg Pred Value         0.8376  0.80851   0.7835
## Prevalence             0.3695  0.24396   0.3865
## Detection Rate         0.2787  0.07873   0.2613
## Detection Prevalence   0.4411  0.13718   0.4217
## Balanced Accuracy      0.7484  0.62272   0.7073
```

Both grazing and walking are classified well, but the performance on resting is decreased compared to QDA with the 5-variable set.

```
accel_expanded_test %>%
  ggplot(aes(x = Behavior, fill = pred)) +
  geom_bar(position = "stack")
```



XGBoost on expanded variable set

XGBoost performance only decreases by about 3.5% on the test set.

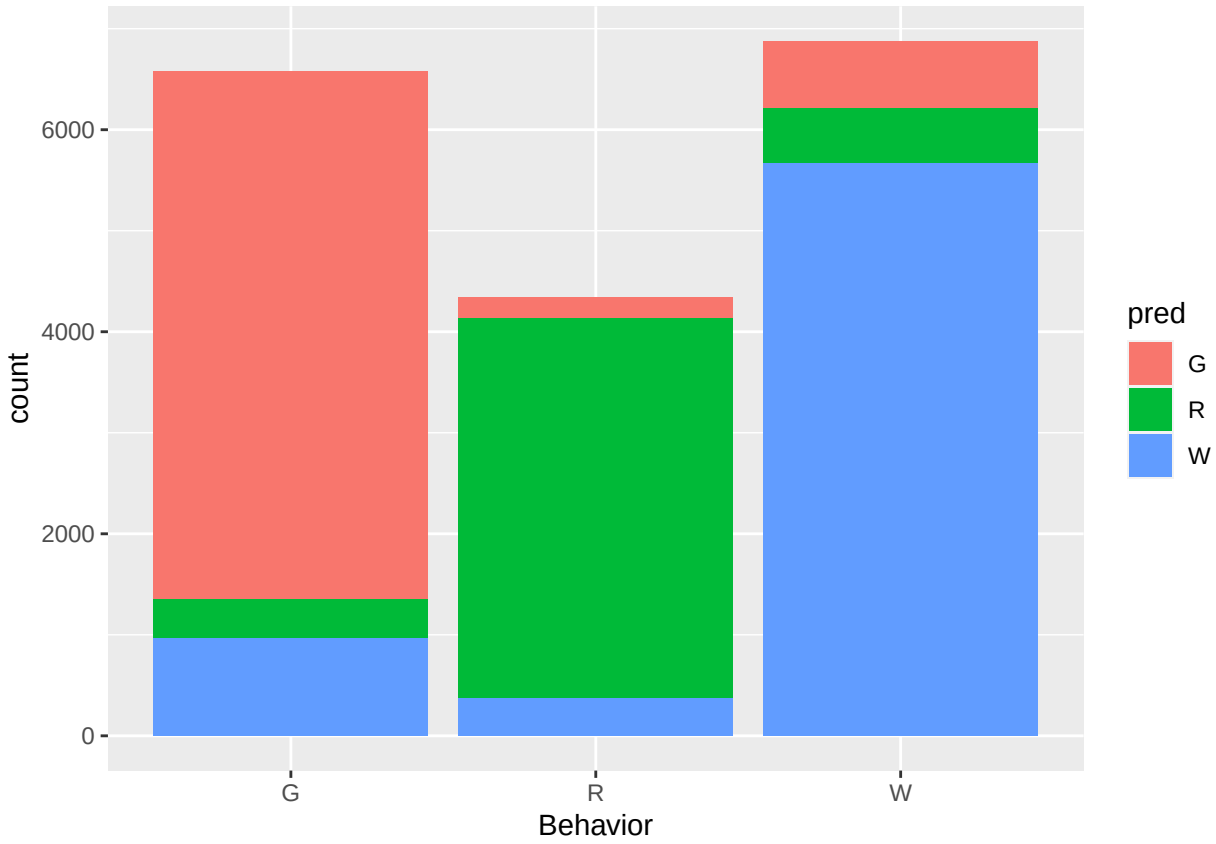
```
gboost_expanded_test <- accel_test2 %>% modelr::add_predictions(gboost_all_train3)
confusionMatrix(gboost_expanded_test$pred, gboost_expanded_test$Behavior)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    G    R    W
##          G 5218  204  669
##          R  396 3761  541
```

```
##          W  961  376 5668
##
## Overall Statistics
##
##                Accuracy : 0.8231
##                  95% CI : (0.8175, 0.8287)
##     No Information Rate : 0.3865
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.7308
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: G Class: R Class: W
## Sensitivity            0.7936   0.8664   0.8241
## Specificity            0.9222   0.9304   0.8775
## Pos Pred Value         0.8567   0.8006   0.8091
## Neg Pred Value         0.8840   0.9557   0.8878
## Prevalence             0.3695   0.2440   0.3865
## Detection Rate         0.2932   0.2114   0.3185
## Detection Prevalence   0.3423   0.2640   0.3937
## Balanced Accuracy      0.8579   0.8984   0.8508
```

Similar to XGBoost performance with the 5-variable set, all three behaviors are classified well.

```
gboost_expanded_test %>%
  ggplot(aes(x = Behavior, fill = pred)) +
  geom_bar(position = "stack")
```


6000
4000
2000
0
count
G
R
W
Behavior
pred
G
R
W