Modeling Run Performance

Joe Martin

10/27/2021

Garmin Data Modeling

The two most obvious primary target variables are average speed (avg_spd) in miles per hour, and average pace (avg_pace_sec) in seconds. A higher average speed and a lower average pace are the desired outcome when measuring performance over time. Reviewing the results of the two preliminary linear regression models, the more desirable variable is average pace, as it has stronger relationships with other variables.

```
# Create preliminary model

prelim_spd <- lm(avg_spd ~ ., df)
summary(prelim_spd)
```

```
##
## Call:
## lm(formula = avg_spd ~ ., data = df)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.187959 -0.029389 -0.000816  0.028251  0.219902
##
## Coefficients: (1 not defined because of singularities)
##                             Estimate Std. Error t value Pr(>|t|)
## (Intercept)               -6.402e+00  5.019e-01 -12.755  < 2e-16 ***
## distance                   1.031e-02  7.272e-03   1.417 0.157270
## avg_hr                     4.347e-03  1.026e-03   4.236 2.84e-05 ***
## max_hr                    -1.052e-03  6.662e-04  -1.579 0.115059
## avg_run_cadence            4.733e-02  1.259e-03  37.584  < 2e-16 ***
## max_run_cadence            1.806e-04  2.388e-04   0.756 0.449921
## total_ascent              -5.052e-05  7.741e-05  -0.653 0.514341
## total_decent               6.208e-05  7.328e-05   0.847 0.397386
## avg_stride                 5.142e+00  1.294e-01  39.720  < 2e-16 ***
## min_elevation              3.616e-04  1.058e-04   3.418 0.000697 ***
## max_elevation             -2.596e-05  1.096e-04  -0.237 0.812852
## avg_pace_sec              -1.011e-03  3.465e-04  -2.916 0.003745 **
## best_pace_sec             -9.307e-05  1.039e-04  -0.896 0.370737
## `sweat_loss(ml)`          -4.945e-05  1.336e-04  -0.370 0.711397
## aerobic_TE                -8.122e-02  1.663e-02  -4.885 1.51e-06 ***
## aerobic_fctImpacting      -5.322e-03  9.255e-03  -0.575 0.565576
## aerobic_fctMaintaining     2.091e-02  1.771e-02   1.181 0.238502
## aerobic_fctOverreaching    4.604e-02  1.427e-02   3.225 0.001365 **
## anaerobic_value            1.303e-02  1.097e-02   1.187 0.235838
```

```
## anaerobic_fctMaintaining  -4.067e-03  1.741e-02  -0.234 0.815458
## anaerobic_fctNo Benefit   -4.778e-02  3.440e-02  -1.389 0.165630
## anaerobic_fctSome Benefit -6.501e-02  2.583e-02  -2.517 0.012246 *
## max_spd                   -3.014e-03  2.567e-03  -1.174 0.241070
## short_distanceY            1.336e-02  1.906e-02   0.701 0.483817
## middle_distanceY           1.843e-02  1.414e-02   1.303 0.193186
## long_distanceY                    NA         NA      NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.04525 on 392 degrees of freedom
##   (11 observations deleted due to missingness)
## Multiple R-squared:  0.9978, Adjusted R-squared:  0.9976
## F-statistic:  7263 on 24 and 392 DF,  p-value: < 2.2e-16
```

```
prelim_pace <- lm(avg_pace_sec ~ ., df)
summary(prelim_pace)
```

```
##
## Call:
## lm(formula = avg_pace_sec ~ ., data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -31.013  -3.313  -0.427   3.011  47.420
##
## Coefficients: (1 not defined because of singularities)
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)                1.185e+03  6.190e+01  19.143  < 2e-16 ***
## distance                  -4.443e+00  1.027e+00  -4.326 1.93e-05 ***
## avg_hr                     3.786e-01  1.501e-01   2.522  0.01206 *
## max_hr                    -8.380e-03  9.638e-02  -0.087  0.93075
## avg_run_cadence           -1.787e+00  3.790e-01  -4.715 3.37e-06 ***
## max_run_cadence            5.933e-03  3.445e-02   0.172  0.86337
## total_ascent              -9.885e-03  1.116e-02  -0.886  0.37613
## total_decent               3.999e-03  1.057e-02   0.378  0.70547
## avg_stride                -2.333e+02  4.015e+01  -5.811 1.29e-08 ***
## min_elevation             -2.627e-02  1.542e-02  -1.703  0.08930 .
## max_elevation              2.642e-02  1.575e-02   1.678  0.09417 .
## best_pace_sec              2.566e-02  1.494e-02   1.718  0.08661 .
## `sweat_loss(ml)`           9.787e-02  1.862e-02   5.257 2.41e-07 ***
## aerobic_TE                -8.762e+00  2.429e+00  -3.606  0.00035 ***
## aerobic_fctImpacting      -4.257e+00  1.318e+00  -3.231  0.00134 **
## aerobic_fctMaintaining     5.590e+00  2.542e+00   2.199  0.02849 *
## aerobic_fctOverreaching    5.209e+00  2.069e+00   2.518  0.01220 *
## anaerobic_value           -1.135e+00  1.584e+00  -0.716  0.47435
## anaerobic_fctMaintaining   2.063e+00  2.509e+00   0.822  0.41144
## anaerobic_fctNo Benefit    1.926e-01  4.972e+00   0.039  0.96913
## anaerobic_fctSome Benefit  1.513e+00  3.754e+00   0.403  0.68709
## avg_spd                   -2.101e+01  7.205e+00  -2.916  0.00374 **
## max_spd                    6.393e-02  3.708e-01   0.172  0.86321
## short_distanceY           -5.004e-01  2.750e+00  -0.182  0.85569
## middle_distanceY          -1.782e-01  2.044e+00  -0.087  0.93054
## long_distanceY                    NA         NA      NA       NA
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.525 on 392 degrees of freedom
##   (11 observations deleted due to missingness)
## Multiple R-squared:  0.9901, Adjusted R-squared:  0.9895
## F-statistic:  1641 on 24 and 392 DF,  p-value: < 2.2e-16
```

The ultimate goal of this model is to utilize data leading up to a performance event. As many races take place on Sunday and the typical long-distance run in this data set takes place on Sunday, the final linear regression model will begin with predicting Sunday performance.

To being predicting run performance, an initial linear regression model will be built below using all available data. Based on the preliminary linear regression above, an aerobic training effect that has a high impact (value between 4 and 4.9) is strongly related to average pace. This variable will be the target variable in the logistic regression that follows.

```r
set.seed(456)
# Split data into training and testing sets
df_split <- initial_split(df, prop = 3/4)

train_df <- training(df_split)
test_df <- testing(df_split)

# Create recipe
pace_rec <- recipe(avg_pace_sec ~ ., data = train_df)

summary(pace_rec)
```

```
## # A tibble: 22 x 4
##    variable        type    role      source
##    <chr>           <chr>   <chr>     <chr>
##  1 distance        numeric predictor original
##  2 avg_hr          numeric predictor original
##  3 max_hr          numeric predictor original
##  4 avg_run_cadence numeric predictor original
##  5 max_run_cadence numeric predictor original
##  6 total_ascent    numeric predictor original
##  7 total_decent    numeric predictor original
##  8 avg_stride      numeric predictor original
##  9 min_elevation   numeric predictor original
## 10 max_elevation   numeric predictor original
## # ... with 12 more rows
```

```r
lm_pace <- linear_reg() %>%
  set_engine("lm")

pace_wflow <- workflow()%>%
  add_model(lm_pace) %>%
  add_recipe(pace_rec)

pace_fit <- pace_wflow %>%
  fit(data = train_df)
```

```r
tidy(pace_fit)
```

```
## # A tibble: 26 x 5
##    term              estimate std.error statistic  p.value
##    <chr>                <dbl>     <dbl>     <dbl>    <dbl>
##  1 (Intercept)     1166.        73.6       15.8   4.97e-41
##  2 distance          -3.81       1.22      -3.12  2.00e- 3
##  3 avg_hr             0.400      0.181      2.21  2.80e- 2
##  4 max_hr            -0.104      0.119     -0.880 3.80e- 1
##  5 avg_run_cadence   -1.70       0.456     -3.73  2.27e- 4
##  6 max_run_cadence    0.0245     0.0436     0.562 5.75e- 1
##  7 total_ascent      -0.00996    0.0139    -0.714 4.76e- 1
##  8 total_decent       0.00380    0.0130     0.293 7.70e- 1
##  9 avg_stride      -233.        48.9       -4.76  3.09e- 6
## 10 min_elevation     -0.0261     0.0193    -1.35  1.77e- 1
## # ... with 16 more rows
```

```r
predict(pace_fit, test_df)
```

```
## Warning in predict.lm(object = object$fit, newdata = new_data, type =
## "response"): prediction from a rank-deficient fit may be misleading
```

```
## # A tibble: 107 x 1
##    .pred
##    <dbl>
##  1  450.
##  2  448.
##  3  436.
##  4  440.
##  5  397.
##  6  414.
##  7  427.
##  8  435.
##  9  433.
## 10  448.
## # ... with 97 more rows
```

```r
pace_aug <- augment(pace_fit, test_df)
```

```
## Warning in predict.lm(object = object$fit, newdata = new_data, type =
## "response"): prediction from a rank-deficient fit may be misleading
```

```r
pace_aug %>% select(avg_pace_sec, .pred)
```

```
## # A tibble: 107 x 2
##    avg_pace_sec .pred
##           <dbl> <dbl>
##  1          447  450.
##  2          449  448.
##  3          432  436.
```

```
## 4          438  440.
## 5          391  397.
## 6          414  414.
## 7          419  427.
## 8          432  435.
## 9          430  433.
## 10         444  448.
## # ... with 97 more rows
```

The R Mean-Squared Error for this model is 5.41. In other words, this model can predict average pace within 5.41 seconds.

```
pace_error <- pace_aug %>%
  rmse(truth = avg_pace_sec, .pred)

pace_error
```

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 rmse    standard        5.41
```

The most significant variables (based on p-value) are average heart rate, average cadence, average stride, and aerobic training effect. The binary variable aerobic__fct__Impacting had a good p-value, as well, but that value is related to aerobic training effect, so it is left out of this analysis. As an attempt to improve the quality of the model, only the variables with the highest p-values will be included in this analysis.

```
set.seed(456)
# Split data into training and testing sets
df_split <- initial_split(df, prop = 3/4)

train_df <- training(df_split)
test_df <- testing(df_split)

# Create recipe
pace_rec_2 <- recipe(avg_pace_sec ~ avg_hr + avg_run_cadence + avg_stride + aerobic_TE, data = train_df

summary(pace_rec_2)
```

```
## # A tibble: 5 x 4
##   variable        type    role      source
##   <chr>           <chr>   <chr>     <chr>
## 1 avg_hr          numeric predictor original
## 2 avg_run_cadence numeric predictor original
## 3 avg_stride      numeric predictor original
## 4 aerobic_TE      numeric predictor original
## 5 avg_pace_sec    numeric outcome   original
```

```
lreg <- linear_reg() %>%
  set_engine("lm")

pace_wflow_2 <- workflow()%>%
```

```r
  add_model(lreg) %>%
  add_recipe(pace_rec_2)

pace_fit_2 <- pace_wflow_2 %>%
  fit(data = train_df)

tidy(pace_fit_2)
```

```
## # A tibble: 5 x 5
##   term            estimate std.error statistic   p.value
##   <chr>              <dbl>     <dbl>     <dbl>     <dbl>
## 1 (Intercept)     1460.       20.4       71.6  2.32e-197
## 2 avg_hr             0.215     0.141      1.53 1.27e-  1
## 3 avg_run_cadence   -3.15      0.147    -21.5  1.01e- 63
## 4 avg_stride      -379.        7.16     -53.0  2.49e-159
## 5 aerobic_TE        -7.35      0.992     -7.41 1.16e- 12
```

```r
predict(pace_fit_2, test_df)
```

```
## # A tibble: 107 x 1
##    .pred
##    <dbl>
##  1  452.
##  2  450.
##  3  444.
##  4  441.
##  5  388.
##  6  414.
##  7  430.
##  8  430.
##  9  442.
## 10  447.
## # ... with 97 more rows
```

```r
pace_aug_2 <- augment(pace_fit_2, test_df)

pace_aug_2 %>% select(avg_pace_sec, .pred)
```

```
## # A tibble: 107 x 2
##    avg_pace_sec .pred
##           <dbl> <dbl>
##  1          447  452.
##  2          449  450.
##  3          432  444.
##  4          438  441.
##  5          391  388.
##  6          414  414.
##  7          419  430.
##  8          432  430.
##  9          430  442.
## 10          444  447.
## # ... with 97 more rows
```

Reviewing the results, the quality of the model decreased slightly. However, it seems that average pace will be a good target variable in exploring performance improvements.

```
pace_error_2 <- pace_aug_2 %>%
  rmse(truth = avg_pace_sec, .pred)

pace_error_2
```

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 rmse    standard        6.50
```

Logistic Regression

All Variables

This first logistic regression is meant to predict whether an activity highly impacts aerobic training. This variable is relevant because it is the highest measure for aerobic conditioning without being over-reaching. In this analysis, calories and other variables related to aerobic training effect were removed.

```
set.seed(456)

df2 <- df %>% mutate(high_impact = ifelse(aerobic_fct == "Highly Impacting", 1,0))
df2$high_impact <- factor(df2$high_impact)

# Split data into training and testing sets
df2_split <- initial_split(df2, prop = 3/4)

train_df2 <- training(df2_split)
test_df2 <- testing(df2_split)

# Create recipe. Use all variables except aerobic_TE and related
aerobic_rec <- recipe(high_impact ~ short_distance + middle_distance + long_distance +
                        max_spd + avg_spd + anaerobic_value + `sweat_loss(ml)` + best_pace_sec +
                        avg_pace_sec + max_elevation + min_elevation + avg_stride +
                        total_decent + total_ascent + max_run_cadence + avg_run_cadence +
                        max_hr + avg_hr + distance, data = train_df2)

summary(aerobic_rec)
```

```
## # A tibble: 20 x 4
##    variable        type    role      source
##    <chr>           <chr>   <chr>     <chr>
##  1 short_distance  nominal predictor original
##  2 middle_distance nominal predictor original
##  3 long_distance   nominal predictor original
##  4 max_spd         numeric predictor original
##  5 avg_spd         numeric predictor original
##  6 anaerobic_value numeric predictor original
##  7 sweat_loss(ml)  numeric predictor original
##  8 best_pace_sec   numeric predictor original
```

```
##  9 avg_pace_sec    numeric predictor original
## 10 max_elevation   numeric predictor original
## 11 min_elevation   numeric predictor original
## 12 avg_stride      numeric predictor original
## 13 total_decent    numeric predictor original
## 14 total_ascent    numeric predictor original
## 15 max_run_cadence numeric predictor original
## 16 avg_run_cadence numeric predictor original
## 17 max_hr          numeric predictor original
## 18 avg_hr          numeric predictor original
## 19 distance        numeric predictor original
## 20 high_impact     nominal outcome   original
```

```
log_reg <- logistic_reg() %>%
  set_engine("glm")

aero_wkfl <- workflow()%>%
  add_model(log_reg) %>%
  add_recipe(aerobic_rec)

aero_fit <- aero_wkfl %>%
  fit(data = train_df2)

tidy(aero_fit)
```

```
## # A tibble: 20 x 5
##    term               estimate std.error statistic  p.value
##    <chr>                 <dbl>     <dbl>     <dbl>    <dbl>
##  1 (Intercept)      -111.      107.         -1.04    0.297
##  2 short_distanceY     0.114     1.34        0.0851  0.932
##  3 middle_distanceY    1.89      0.900       2.09    0.0363
##  4 long_distanceY     NA        NA          NA      NA
##  5 max_spd            -0.811     1.05       -0.775   0.439
##  6 avg_spd           -17.1       5.36       -3.19    0.00144
##  7 anaerobic_value    -0.670     0.425      -1.58    0.115
##  8 `sweat_loss(ml)`   -0.00296   0.0227     -0.130   0.896
##  9 best_pace_sec      -0.0278    0.0347     -0.800   0.423
## 10 avg_pace_sec       -0.0252    0.0851     -0.297   0.767
## 11 max_elevation      -0.0283    0.0118     -2.40    0.0165
## 12 min_elevation       0.0288    0.0102      2.81    0.00495
## 13 avg_stride         88.9      32.8         2.71    0.00673
## 14 total_decent        0.00536   0.00665     0.806   0.420
## 15 total_ascent        0.00340   0.00672     0.506   0.613
## 16 max_run_cadence     0.00974   0.0226      0.432   0.666
## 17 avg_run_cadence     0.747     0.298       2.50    0.0123
## 18 max_hr              0.0856    0.0742      1.15    0.249
## 19 avg_hr              0.115     0.0643      1.79    0.0734
## 20 distance            0.279     1.11        0.252   0.801
```

```
predict(aero_fit, test_df2)
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from a rank-deficient fit may be misleading
```

```
## # A tibble: 107 x 1
##    .pred_class
##    <fct>
##  1 0
##  2 0
##  3 0
##  4 0
##  5 0
##  6 0
##  7 0
##  8 1
##  9 0
## 10 1
## # ... with 97 more rows
```

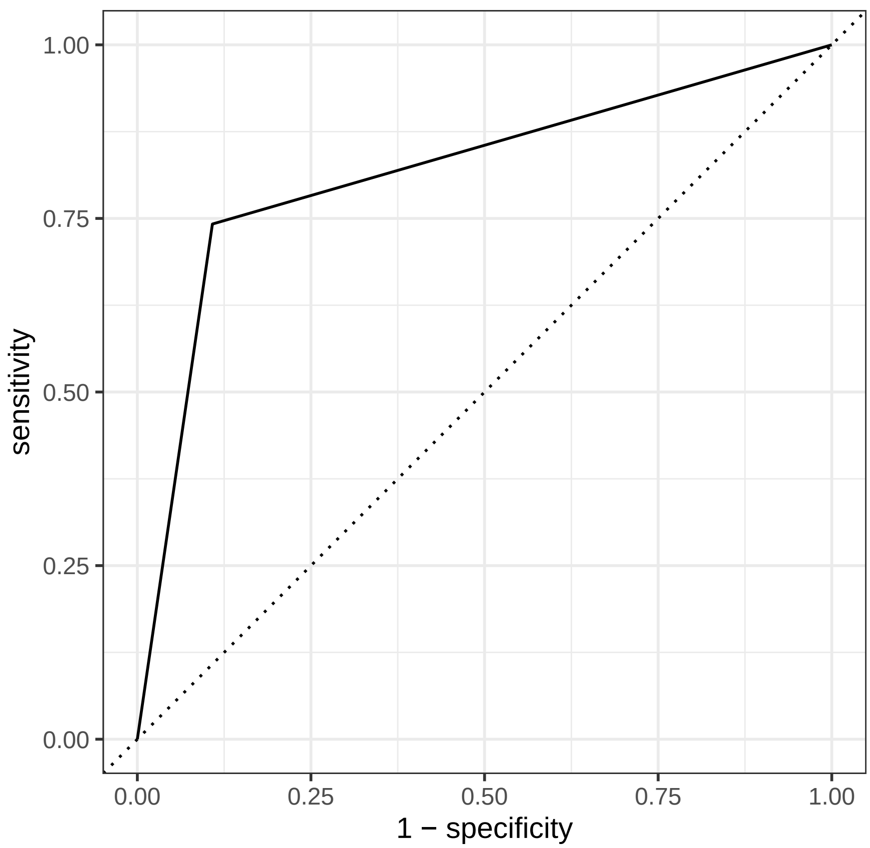
```
aero_aug <- augment(aero_fit, test_df2)
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from a rank-deficient fit may be misleading

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from a rank-deficient fit may be misleading
```

```
aero_aug %>% select(high_impact, .pred_class)
```

```
## # A tibble: 107 x 2
##    high_impact .pred_class
##    <fct>       <fct>
##  1 0           0
##  2 0           0
##  3 0           0
##  4 0           0
##  5 0           0
##  6 0           0
##  7 0           0
##  8 1           1
##  9 0           0
## 10 1           1
## # ... with 97 more rows
```

```
aero_aug$.pred_class <- as.character(aero_aug$.pred_class)
aero_aug$.pred_class <- as.numeric(aero_aug$.pred_class)

aero_aug %>%
  roc_curve(truth = high_impact, .pred_class, event_level="second") %>%
  autoplot()
```




```
aero_aug %>%
  roc_auc(truth = high_impact, .pred_class, event_level="second")
```

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.817
```

The first logistic regression is a good predictive model. The next step is to select fewer variables to see those increase the reliability of the model.

```
# Create recipe. Use all variables except aerobic_TE and related
aerobic_rec2 <- recipe(high_impact ~ middle_distance + avg_spd + min_elevation + avg_stride +
                       avg_run_cadence + avg_hr, data = train_df2)

summary(aerobic_rec2)
```

```
## # A tibble: 7 x 4
##   variable        type    role      source
##   <chr>           <chr>   <chr>     <chr>
## 1 middle_distance nominal predictor original
## 2 avg_spd         numeric predictor original
## 3 min_elevation   numeric predictor original
## 4 avg_stride      numeric predictor original
```

```
## 5 avg_run_cadence numeric predictor original
## 6 avg_hr          numeric predictor original
## 7 high_impact     nominal outcome   original
```

```
log_reg <- logistic_reg() %>%
  set_engine("glm")

aero_wkfl2 <- workflow()%>%
  add_model(log_reg) %>%
  add_recipe(aerobic_rec2)

aero_fit2 <- aero_wkfl2 %>%
  fit(data = train_df2)

tidy(aero_fit2)
```

```
## # A tibble: 7 x 5
##   term              estimate std.error statistic  p.value
##   <chr>                <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)      -133.      34.0         -3.93 8.63e- 5
## 2 middle_distanceY    2.72     0.358        7.60 3.07e-14
## 3 avg_spd           -14.7      4.15        -3.55 3.85e- 4
## 4 min_elevation      -0.0101   0.00576     -1.75 7.94e- 2
## 5 avg_stride         77.6     23.3          3.33 8.57e- 4
## 6 avg_run_cadence     0.764    0.224        3.40 6.63e- 4
## 7 avg_hr              0.185    0.0444       4.17 2.99e- 5
```

```
predict(aero_fit2, test_df2)
```

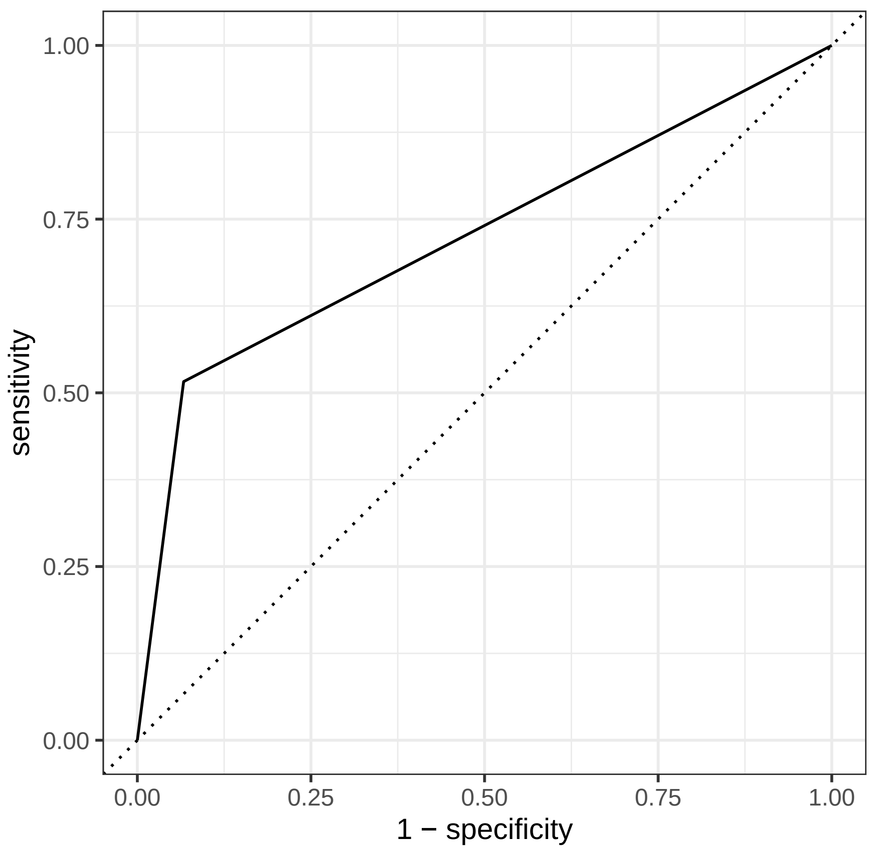
```
## # A tibble: 107 x 1
##    .pred_class
##    <fct>
##  1 0
##  2 0
##  3 0
##  4 0
##  5 0
##  6 0
##  7 0
##  8 1
##  9 0
## 10 1
## # ... with 97 more rows
```

```
aero_aug2 <- augment(aero_fit2, test_df2)

aero_aug2 %>% select(high_impact, .pred_class)
```

```
## # A tibble: 107 x 2
##    high_impact .pred_class
##    <fct>       <fct>
##  1 0           0
```

```
## 2 0 0
## 3 0 0
## 4 0 0
## 5 0 0
## 6 0 0
## 7 0 0
## 8 1 1
## 9 0 0
## 10 1 1
## # ... with 97 more rows
```

```
aero_aug2$.pred_class <- as.character(aero_aug2$.pred_class)
aero_aug2$.pred_class <- as.numeric(aero_aug2$.pred_class)

aero_aug2 %>%
  roc_curve(truth = high_impact, .pred_class, event_level = "second") %>%
  autoplot()
```




```
aero_aug2 %>%
  roc_auc(truth = high_impact, .pred_class, event_level = "second")
```

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.725
```

Both of these models are acceptable for predicting whether a run highly impacts performance. These analyses provide a good starting point for building a more complex model that can predict good performance. The possible next step is to use k-fold cross validation to predict when good performance will happen given a series of events.