Lab4 - Logisitic Regressions

Colleen McCamy

2023-01-30

Lab 4: Fire and Tree Mortality

The database we'll be working with today includes 36066 observations of individual trees involved in prescribed fires and wildfires occurring over 35 years, from 1981 to 2016. It is a subset of a larger fire and tree mortality database from the US Forest Service (see data description for the full database here: link). Our goal today is to predict the likelihood of tree mortality after a fire.

Data Exploration

Outcome variable: *yr1status* = tree status (0=alive, 1=dead) assessed one year post-fire.

Predictors: *YrFireName, Species, Genus_species, DBH_cm, CVS_percent, BCHM_m, BTL* (Information on these variables available in the database metadata (link)).

```
# reading in the data
trees_dat<- read_csv(file = "https://raw.githubusercontent.com/MaRo406/eds-232-machine-learning/main/da
  select(-"...1") |>
  janitor::clean_names()
```

```
## New names:
## Rows: 36066 Columns: 9
## -- Column specification
## -------------------------------------------------------- Delimiter: "," chr
## (3): YrFireName, Species, Genus_species dbl (6): ...1, yr1status, DBH_cm,
## CVS_percent, BCHM_m, BTL
## i Use `spec()` to retrieve the full column specification for this data. i
## Specify the column types or set `show_col_types = FALSE` to quiet this message.
## * `` -> `...1`
```

Question 1: Recode all the predictors to a zero_based integer form

```
#creating a recipe and encoding the categorical predictors using has_type
trees_rec <- recipe(yr1status ~ ., data = trees_dat) |>
  step_integer(all_predictors(),
    zero_based = TRUE) |>
  prep(trees_dat) |>
  bake(trees_dat)
```

Data Splitting

Question 2: Create trees_training (70%) and trees_test (30%) splits for the modeling

```
set.seed(123) # setting a seed
# splitting data for 70 for training with a stratification of the outcome variable
tree_split <- initial_split(trees_rec,
                            prop = .70,
                            strata = yr1status)

# setting the split data to training and testing
trees_training <- training(tree_split)
trees_test  <- testing(tree_split)
```

Question 3: How many observations are we using for training with this split?

```
# finding the number of observations
train_ob <- nrow(trees_training)

print(paste0("There are ", train_ob, " number of observations used for the training data."))
```

```
## [1] "There are 25246 number of observations used for the training data."
```

RESPONSE: There are 25,246 numbers of observations used for the training data.

Simple Logistic Regression

Let's start our modeling effort with some simple models: one predictor and one outcome each.

Question 4: Choose the three predictors that most highly correlate with our outcome variable for further investigation.

```
# computing the correlation matrix
cormat_trees <- round(cor(trees_rec), 2)

# reshaping correlation matrix
melted_cormat <- melt(cormat_trees)

# creating a correlation heatmap between the variables
cor_plot <- ggplot(data = melted_cormat, aes(x=Var1, y=Var2, fill=value)) +
  geom_tile(col = '#faf7f5') +
  labs(title = "Correlation of Tree Variables",
       caption = "Figure 1",
       x = "Varibales 1 ",
       y = "Variables 2") +
  scale_fill_gradientn(colors = blues9,
                       limits = c(-1,1)) +
  theme(axis.text.x = element_text(angle = 90))

cor_plot
```

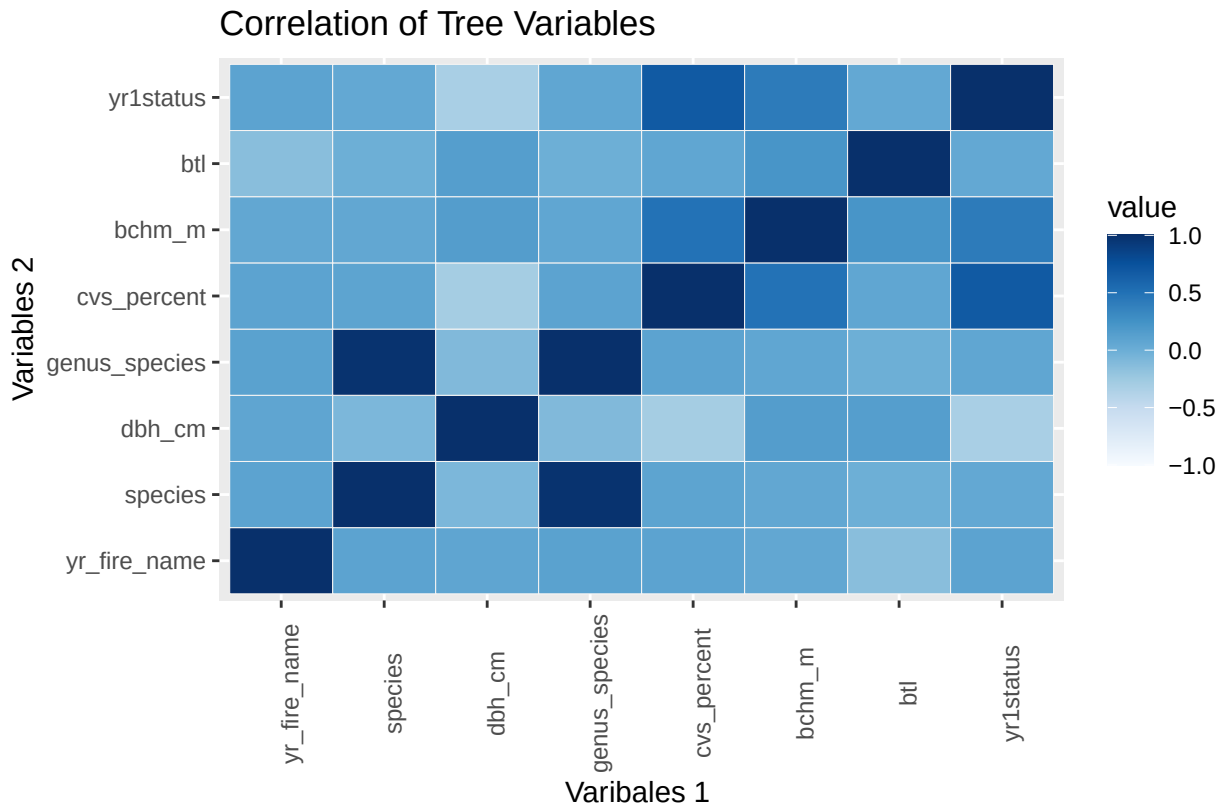

Figure 1

```
# Finding the correlation of year 1 status and
dbh_cor <- cor(trees_rec$yr1status, trees_rec$dbh_cm)

# Finding the correlation of year 1 status and
cvs_cor <- cor(trees_rec$yr1status, trees_rec$cvs_percent)

# Finding the correlation of year 1 status and
bchm_cor <- cor(trees_rec$yr1status, trees_rec$bchm_m)

print(paste0("The three predictors that highly correlate with death after one year are diameter at breas
```

```
## [1] "The three predictors that highly correlate with death after one year are diameter at breast heig
```

RESPONSE: The three predictors that highly correlate with death after one year are diameter at breast height (dbh) with a correlation of -0.32 percent of the pre-fire crown volume that was scorched or consumed by fire (cvs) with a correlation of 0.679 and maximum bark char (bchm) with a correlation of 0.424.

Question 5: Use glm() to fit three simple logistic regression models, one for each of the predictors you identified.

```
# simple logistic regression for dbh_cm predicting status of tree after 1 year
model_dbh <- glm(data = trees_training,
                 yr1status ~ dbh_cm,
                 family = "binomial")
```

```
# simple logistic regression for dbh_cm predicting status of tree after 1 year
model_cvs <- glm(data = trees_training,
                 yr1status ~ cvs_percent,
                 family = "binomial")

# simple logistic regression for bchm predicting status of tree after 1 year
model_bchm <- glm(data = trees_training,
                  yr1status ~ bchm_m,
                  family = "binomial")
```

Interpret the Coefficients

We aren't always interested in or able to interpret the model coefficients in a machine learning task. Often predictive accuracy is all we care about.

Question 6: That said, take a stab at interpreting our model coefficients now.

```
# understanding the coefficients for dbh
exp(coef(model_dbh))
```

```
## (Intercept)      dbh_cm
##   1.4629638   0.9963086
```

```
# understanding the coefficients for cvs
exp(coef(model_cvs))
```

```
## (Intercept) cvs_percent
## 0.001111539 1.081396629
```

```
# understanding the coefficients for
exp(coef(model_bchm))
```

```
## (Intercept)      bchm_m
##   0.1370805   1.0062756
```

Response: For the bdh model, the odds that a tree will be dead after one year since a fire increase multiplicatively by 0.996 for every one cm increase in diameter at breast height. However the odds that a tree will be dead after one year since a fire increase multiplicatively by 1.081 for every one percentage increase of the pre-fire crown volume that was scorched or consumed by fire. Additionaly, for every one meter increase in maximum bark char the odds that a tree will be dead one year after a fire increases multiplicatively by 1.006.

Question 7: Now let's visualize the results from these models. Plot the fit to the training data of each model.

```
# plotting dbh and the model
plot_dbh <- trees_rec |>
  ggplot(mapping = aes(x = dbh_cm, y =  yr1status)) +
  geom_point(col = "#4cad95",
             size = 1.6,
             alpha = 0.5) +
```

```
  geom_smooth(method = glm,
              method.args=list(family="binomial"),
              formula = y ~ x,
              color = "midnightblue",
              size = 1.2,
              se = FALSE) +
  labs(y = "Status One Year After Fire",
       x = "Diameter at Breast Height (centimeters)",
       title = "Status of A Tree Based After Fire Based on Tree Diameter",
       caption = "Figure 2,
       Key: 1 = Tree Dies, 0 = Tree Survives") +
  theme_minimal()

# calling the plot back
plot_dbh
```

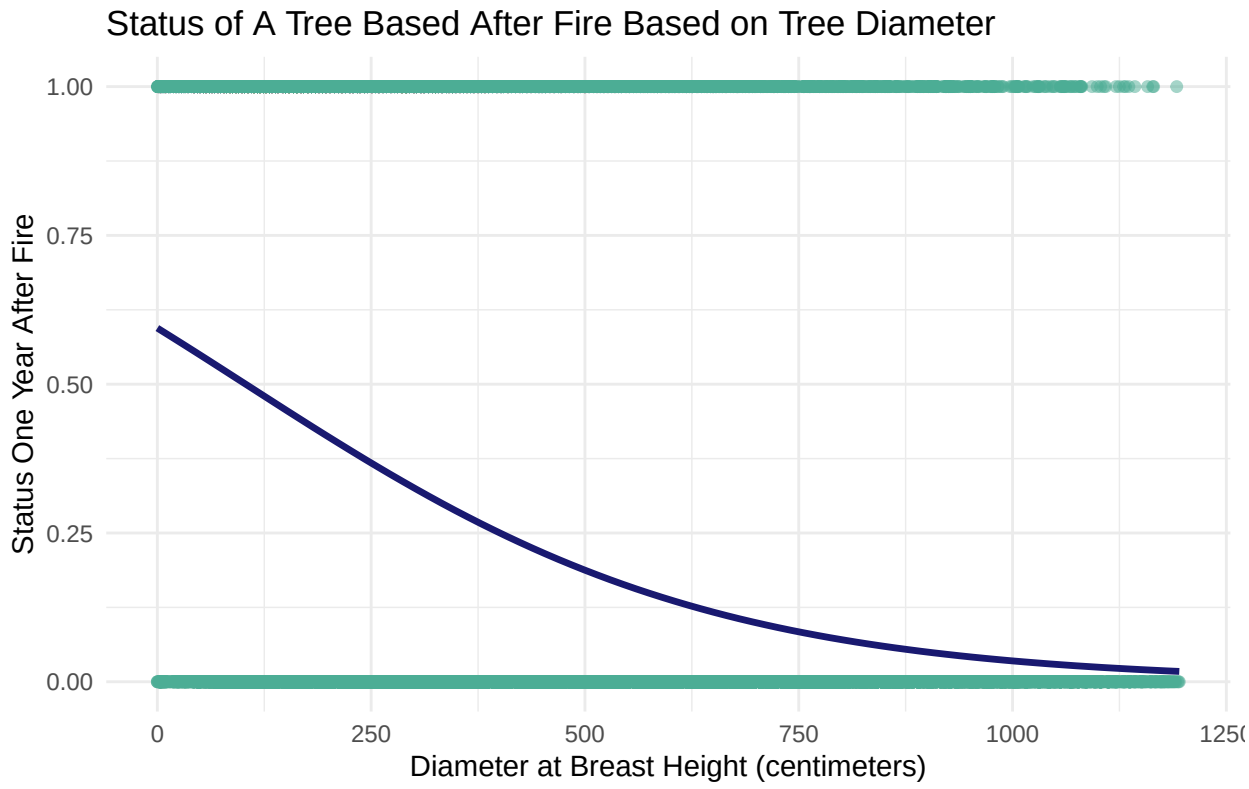

Figure 2,
Key: 1 = Tree Dies, 0 = Tree Survives

```
# plotting cvs percentage and the model
plot_cvs <- trees_training |>
  ggplot(mapping = aes(x = cvs_percent, y =  yr1status)) +
  geom_point(col = "#bc90de",
             size = 1.6,
             alpha = 0.5) +
  geom_smooth(method = glm,
              method.args=list(family="binomial"),
              formula = y ~ x,
```

```
              color = "midnightblue",
              size = 1.2,
              se = FALSE) +
  labs(y = "Status One Year After Fire",
       x = "Percent of the Pre-fire Crown Volume that was Scorched or Consumed by Fire",
       title = "Status of A Tree Based After Fire Based on Percentage Scorched Crown Volume",
       caption = "Figure 3,
       Key: 1 = Tree Dies, 0 = Tree Survives") +
  theme_minimal() +
  theme(plot.title = element_text(size = 11))

# calling the plot back
plot_cvs
```

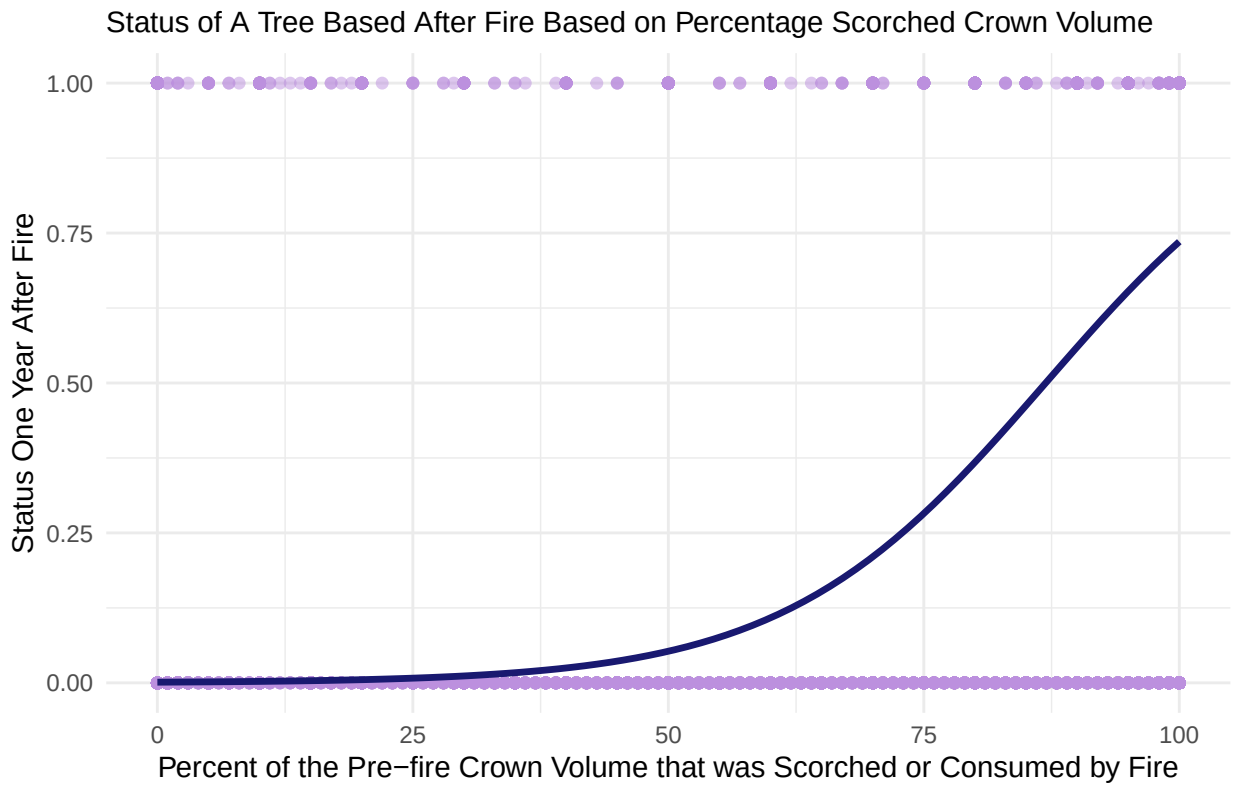

Figure 3,
Key: 1 = Tree Dies, 0 = Tree Survives

```
# plotting bchm and the model
plot_bchm <- trees_training |>
  ggplot(mapping = aes(x = bchm_m, y = yr1status)) +
  geom_point(col = "#88bf94",
             size = 1.6,
             alpha = 0.5) +
  geom_smooth(method = glm,
              method.args=list(family="binomial"),
              formula = y ~ x,
              color = "midnightblue",
              size = 1.2,
```

```
              se = FALSE) +
  labs(y = "Status One Year After Fire", x = "Maximum Bark Char (meters)",
       title = "Status of A Tree Based After Fire Based on Maximum Bark Char",
       caption = "Figure 4,
       Key: 1 = Tree Dies, 0 = Tree Survives") +
  theme_minimal()


# calling the plot back
plot_bchm
```

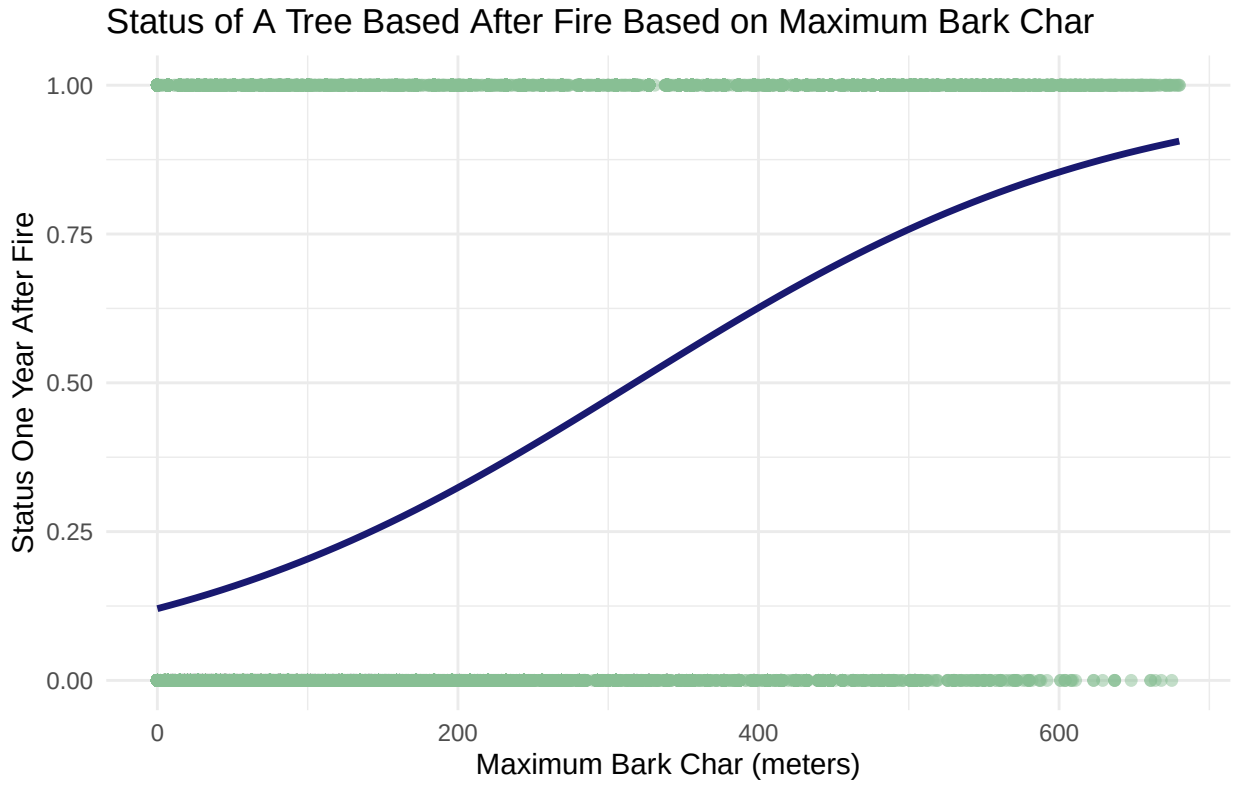

Figure 4,
Key: 1 = Tree Dies, 0 = Tree Survives

Multiple Logistic Regression

Let's not limit ourselves to a single-predictor model. More predictors might lead to better model performance.

Question 8: Use glm() to fit a multiple logistic regression called "logistic_full", with all three of the predictors included. Which of these are significant in the resulting model?

```
# creating a logistic regression with all three top predictors
logisitic_full <- glm(data = trees_training,
                      yr1status ~ dbh_cm + cvs_percent + bchm_m,
                      family = "binomial")

# checking out the significance for the predictors
summary(logisitic_full)
```

```
##
## Call:
## glm(formula = yr1status ~ dbh_cm + cvs_percent + bchm_m, family = "binomial",
##     data = trees_training)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.3237  -0.2433  -0.0629   0.4464   4.1095
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -5.2892427  0.1180370  -44.81   <2e-16 ***
## dbh_cm      -0.0035734  0.0001175  -30.42   <2e-16 ***
## cvs_percent  0.0639961  0.0012302   52.02   <2e-16 ***
## bchm_m       0.0045870  0.0001613   28.43   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 30115  on 25245  degrees of freedom
## Residual deviance: 13364  on 25242  degrees of freedom
## AIC: 13372
##
## Number of Fisher Scoring iterations: 7
```

Response: All three predictors are significant at the 0.001 significance level. This indicates that it would be beneficial to include these predictors in the model.

Estimate Model Accuracy

Now we want to estimate our model's generalizability using resampling.

Question 9: Use cross validation to assess model accuracy. Use caret::train() to fit four 10-fold cross-validated models (cv_model1, cv_model2, cv_model3, cv_model4) that correspond to each of the four models we've fit so far: three simple logistic regression models corresponding to each of the three key predictors (CVS_percent, DBH_cm, BCHM_m) and a multiple logistic regression model that combines all three predictors.

```r
# setting seed for reproducibility
set.seed(123)

# cross validating the model for the dbh predictor
cv_model1 <-
  caret::train(
  data = trees_training,
  as.factor(yr1status) ~ dbh_cm,
  method = "glm",
  family= "binomial",
  trControl = trainControl(method = "cv", number = 10)
  )

# cross validating the model for the cvs predictor
cv_model2 <-
```

```
  caret::train(
  data = trees_training,
  as.factor(yr1status) ~ cvs_percent,
  method = "glm",
  family = "binomial",
  trControl = trainControl(method = "cv", number = 10)
  )

# cross validating the model for the bchm predictor
cv_model3 <-
  caret::train(
  data = trees_training,
  as.factor(yr1status) ~ bchm_m,
  method = "glm",
  family = "binomial",
  trControl = trainControl(method = "cv", number = 10)
  )

# cross validating the model for all predictors
cv_model4 <-
  caret::train(
  data = trees_training,
  as.factor(yr1status) ~ dbh_cm + cvs_percent + bchm_m,
  method = "glm",
  family = "binomial",
  trControl = trainControl(method = "cv", number = 10)
  )
```

Question 10: Use caret::resamples() to extract then compare the classification accuracy for each model. (Hint: resamples() wont give you what you need unless you convert the outcome variable to factor form). Which model has the highest accuracy?

```
# using resamples to identify which model has the best fit, turning the outcome as a factor with list
summary(
  resamples(
    list(
      model1 = cv_model1,
      model2 = cv_model2,
      model3 = cv_model3,
      model4 = cv_model4
    )
  )
  )$statistics$Accuracy
```

```
##             Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## model1 0.7480190 0.7494802 0.7510888 0.7514851 0.7528724 0.7577197    0
## model2 0.8906498 0.8937203 0.8968519 0.8974883 0.9009901 0.9061014    0
## model3 0.7592079 0.7667558 0.7708003 0.7715680 0.7779319 0.7837624    0
## model4 0.8883168 0.8997721 0.9025550 0.9024011 0.9044176 0.9167987    0
```

Let's move forward with this single most accurate model.

Response: The model that includes all predictors (model 4) is the most accurate model with average accuracy rate of 90.2%, while the model solely looking at diameter at breast height

has a 75.1% accuracy, percent of the pre-fire crown volume that was scorched or consumed by fire has an 89.7% accuracy, and maximum bark char has a 77.5% accuracy.

Question 11: Compute the confusion matrix and overall fraction of correct predictions by the model.

```
# predicting the class for a caret model
pred_class <- predict(cv_model4, trees_training)

# creating a confusion matrix
caret::confusionMatrix(
  data = pred_class,
  reference = as.factor(trees_training$yr1status))
```

```
## Confusion Matrix and Statistics
##
##            Reference
## Prediction     0     1
##          0 16449   823
##          1  1634  6340
##
##                Accuracy : 0.9027
##                  95% CI : (0.899, 0.9063)
##     No Information Rate : 0.7163
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.7685
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9096
##             Specificity : 0.8851
##          Pos Pred Value : 0.9524
##          Neg Pred Value : 0.7951
##              Prevalence : 0.7163
##          Detection Rate : 0.6515
##    Detection Prevalence : 0.6841
##       Balanced Accuracy : 0.8974
##
##        'Positive' Class : 0
##
```

Question 12: Explain what the confusion matrix is telling you about the types of mistakes made by logistic regression.

RESPONSE: The confusion matrix highlights true positive, false positive, true negative and false negative predictions conducted by the model. The output of running the confusion matrix function outlines that there were 16,449 trees that were identified as living after 1 year of a fire both in the model and in actuality. In addition, there were 1,634 trees that were living one year after a fire however, the model predicted these trees to be dead. This is an example of a false negative.

Consequently,there were 823 trees that were actually dead one year after a fire that were predicted to be living one year after fire from the model. This is an example of false positive. Lastly, there were 6,340 trees that were both dead and predicted to be dead from the model one year after a fire - a true negative.

Overall, the confusion matrix informs us that there is a higher true positive rate of 0.910 than a true negative rate of 0.885. This could be because the majority class in the data is that the trees will survive after one year after a fire.

Question 13: What is the overall accuracy of the model? How is this calculated?

RESPONSE:The accuracy of the model is 90.3%. This accuracy rate is calculated by adding the true positive and the true negatives divided by the sum of the true positives, true negatives, false positives and false negatives.

The accuracy rates is also higher than the no information rate of 71.6%, which is the rate if we would simply predict that the tree would survive one year after a fire. This indicates that the model performed better than random guessing.

In addition, the balanced accuracy is 89.7%. We can look at this evaluation metric as it appears that the there is imbalanced data with trees surviving after one year from a fire more frequently than trees dying. This balanced accuracy is calculated by taking the average of sensitivity (true positive rate) and specificity (true negative rate). This balanced accuracy is still higher than the no information rate.

Test Final Model

Alright, now we'll take our most accurate model and make predictions on some unseen data (the test data).

Question 14: Now that we have identified our best model, evaluate it by running a prediction on the test data, trees_test.

```
# running the prediction on the test data
pred_test <- predict(cv_model4, trees_test)

# creating a confusion matrix for the test data
caret::confusionMatrix(
  data = pred_test,
  reference = as.factor(trees_test$yr1status))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 7074  356
##          1  676 2714
##
##                Accuracy : 0.9046
##                  95% CI : (0.8989, 0.9101)
##     No Information Rate : 0.7163
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.7725
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9128
##             Specificity : 0.8840
##          Pos Pred Value : 0.9521
##          Neg Pred Value : 0.8006
```

```
##              Prevalence : 0.7163
##          Detection Rate : 0.6538
##    Detection Prevalence : 0.6867
##       Balanced Accuracy : 0.8984
##
##        'Positive' Class : 0
##
```

Question 15: How does the accuracy of this final model on the test data compare to its cross validation accuracy? Do you find this to be surprising? Why or why not?

RESPONSE: The model accuracy is consistent with the cross validation accuracy at 90.4% and the balanced accuracy rate is 89.8%. I find this to be a little bit surprising since there are much less data points in the testing data. I also would think that if all predictors are included it could run the risk of overfitting the model. However, given the consistenties with the accuracies of the training and testing data it appears that this model would be useful in predicting tree deaths one year after a fire given data is present on the predictors.