AH_Code_of_Project_STAT_627

Aidan Hennessy

AH Code for STAT 627 Project

```
#Here I load a publicly available data set on 2023 Mecklenburg County Home
#Loans that is required to be disclosed under the Home Mortgage Lending Act
#of 1970.

library(tidyverse)
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.1.4     v readr     2.1.5
v forcats   1.0.0     v stringr   1.5.1
v ggplot2   3.5.1     v tibble    3.2.1
v lubridate 1.9.3     v tidyr     1.3.1
v purrr     1.0.2
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becor
```

```
library(ordinal)
```

```
Attaching package: 'ordinal'

The following object is masked from 'package:dplyr':

    slice
```

```
Mecklenburg_Home_Loans <- read_csv("County_Mecklenburg_Home_Loans.csv",
                                   na = c("", "NA", "Exempt"))
```

```
Warning: One or more parsing issues, call `problems()` on your data frame for details,
e.g.:
  dat <- vroom(...)
  problems(dat)

Rows: 44322 Columns: 99
-- Column specification --------------------------------------------------------
Delimiter: ","
chr (14): lei, state_code, conforming_loan_limit, derived_loan_product_type,...
dbl (75): activity_year, derived_msa-md, county_code, census_tract, action_t...
lgl (10): total_points_and_fees, applicant_ethnicity-3, applicant_ethnicity-...

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.
```

```
library(ISLR2)
library(class)
library(janitor)
```

```
Attaching package: 'janitor'

The following objects are masked from 'package:stats':

    chisq.test, fisher.test
```

```
set.seed(123)

#I filter action_taken down to 5 or less because the sum of the number of
#applications received is the sum of options 1-5 for action_taken. This
#code is only intended to study instances where applications are received.
#It also excludes instances where race is not available, sex is not
#available, ethnicity is not available and debt_to_income_ratio is exempt.
#These filters narrow it down to where action_taken is either options 1, 2
#or 3. I also get rid of columns with just one unique variable. And any "-"
#within the name of each column is replaced with a "_" because the subset
#function that I later use is only okay with me using "-" when it is not
```

```
#included within a column's name.

Mecklenburg_Home_Loans_Rev <- Mecklenburg_Home_Loans |>
  filter(action_taken <= 5) |>
  filter(derived_race != "Race Not Available") |>
  filter(derived_sex != "Sex Not Available") |>
  filter(derived_race != "Free Form Text Only") |>
  filter(derived_ethnicity != "Ethnicity Not Available") |>
  filter(debt_to_income_ratio != "Exempt") |>
  select_if(~length(unique(.)) > 1) |>
  rename_with(~ gsub("-", "_", .x, fixed = TRUE))


#This takes out into account all of the columns that have more than 1,000
#null values.
Mecklenburg_Home_Loans_na_count <- Mecklenburg_Home_Loans_Rev |>
  summarise_all(~ sum(is.na(.)))
Mecklenburg_Home_Loans_na_count_final <-
  data.frame(Mecklenburg_Home_Loans_na_count)
Mecklenburg_Home_Loans_na_count_final_Rev <-
  data.frame(t(Mecklenburg_Home_Loans_na_count_final)) |>
  rename(c_name = t.Mecklenburg_Home_Loans_na_count_final.)
Mecklenburg_Home_Loans_na_count_final_Rev_Sort <-
  arrange(Mecklenburg_Home_Loans_na_count_final_Rev,
          desc(Mecklenburg_Home_Loans_na_count_final_Rev$c_name))
Mecklenburg_Home_Loans_na_count_final_Rev_Sort_Filtered <-
  Mecklenburg_Home_Loans_na_count_final_Rev_Sort |>
  filter(Mecklenburg_Home_Loans_na_count_final_Rev_Sort$c_name > 1000)

#This is where the columns with over 1,000 null values get deleted. Two
#other columns are deleted too. "lei" is also deleted because it has too
#many different character variables for a regression equation to work when
#it is included. "state_code" is also deleted because it has just one
#unique variable. The previous code to get rid of unique variables could
#only get rid of numeric variables that have just one unique variable and
#not a character variable like "state_code" that had that same condition.
#The rows with null values in the remaining columns are omitted as well.
#And our dependent variable, action_taken, becomes a factor variable.
Mecklenburg_Co_Home_Loans_Rev_Two <- subset(Mecklenburg_Home_Loans_Rev,
                                            select = c(-lei, -state_code,
-applicant_ethnicity_3, -denial_reason_4, -applicant_race_5,
-applicant_race_4, -co_applicant_race_3, -aus_5, -aus_4,
```

```
-applicant_race_3, -denial_reason_3, -aus_3, -co_applicant_ethnicity_2,
-co_applicant_race_2, -denial_reason_2, -applicant_ethnicity_2, -aus_2,
-applicant_race_2, -prepayment_penalty_term, -lender_credits,
-discount_points, -co_applicant_age_above_62,
-intro_rate_period, -origination_charges,
-total_loan_costs, -rate_spread, -interest_rate, -loan_term)) |>
  na.omit(Mecklenburg_Home_Loans_Rev) |>
  mutate(action_taken = as.factor(action_taken))
```

```
library(leaps)
library(car)
```

```
Loading required package: carData
```

```
Attaching package: 'car'
```

```
The following object is masked from 'package:dplyr':

    recode
```

```
The following object is masked from 'package:purrr':

    some
```

```
library(caret)
```

```
Loading required package: lattice
```

```
Attaching package: 'caret'
```

```
The following object is masked from 'package:purrr':

    lift
```

```
library(nnet)
library(VGAM)
```

```
Loading required package: stats4

Loading required package: splines


Attaching package: 'VGAM'

The following object is masked from 'package:caret':

    predictors

The following object is masked from 'package:car':

    logit

The following objects are masked from 'package:ordinal':

    dgumbel, dlgamma, pgumbel, plgamma, qgumbel, rgumbel, wine
```

```
library(biotools)
```

```
Loading required package: MASS


Attaching package: 'MASS'

The following object is masked from 'package:ISLR2':

    Boston

The following object is masked from 'package:dplyr':

    select

---
biotools version 4.2
```

```
#Here I use the function polr to examine the relationship between
#action_taken and almost every variable in the subset I created at the
#end of the last code chunk. I eliminate variables the variables that
#when included, say "fitted probabilities numerically 0 or 1 occurred",
#and/or "Warning: NaNs produced" as well as those that cause this
#regression equation to break because of "Error in svd(X) : infinite or
#missing values in 'x'." The function polr is appropriate for a
#dependent variable that is a factor variable with more than one level.

Home_Loan_Reg_Two <- polr(action_taken ~ conforming_loan_limit +
  derived_dwelling_category + derived_ethnicity + derived_race +
  derived_sex + preapproval + loan_type + loan_purpose + lien_status +
  reverse_mortgage + open_end_line_of_credit +
  business_or_commercial_purpose + loan_to_value_ratio +
  negative_amortization + interest_only_payment + balloon_payment +
  other_nonamortizing_features + occupancy_type +
  manufactured_home_secured_property_type +
  manufactured_home_land_property_interest + total_units + income +
  debt_to_income_ratio + applicant_credit_score_type +
  co_applicant_credit_score_type + applicant_ethnicity_1 +
  co_applicant_ethnicity_1 + applicant_ethnicity_observed +
  co_applicant_ethnicity_observed + applicant_race_1 +
  co_applicant_race_1 + applicant_race_observed +
  co_applicant_race_observed + applicant_sex + co_applicant_sex +
  applicant_sex_observed + co_applicant_sex_observed + applicant_age +
  co_applicant_age + applicant_age_above_62 + submission_of_application +
  initially_payable_to_institution + tract_population +
  tract_minority_population_percent + tract_to_msa_income_percentage +
  tract_median_age_of_housing_units,
  data = Mecklenburg_Co_Home_Loans_Rev_Two, Hess = TRUE)


#The subset here only includes the predictor variables that were not causing
#any problems in the regression equation immediately above.

Mecklenburg_Co_Home_Loans_Rev_Three <-
  subset(Mecklenburg_Co_Home_Loans_Rev_Two,
    select = c(conforming_loan_limit, derived_dwelling_category,
               derived_ethnicity, derived_race, derived_sex, preapproval,
               loan_type, loan_purpose, lien_status, reverse_mortgage,
               open_end_line_of_credit, business_or_commercial_purpose,
               loan_to_value_ratio, negative_amortization,
```

```
                interest_only_payment, balloon_payment,
                other_nonamortizing_features, occupancy_type,
                manufactured_home_secured_property_type,
                manufactured_home_land_property_interest, total_units,
                income, debt_to_income_ratio, applicant_credit_score_type,
                co_applicant_credit_score_type, applicant_ethnicity_1,
                co_applicant_ethnicity_1, applicant_ethnicity_observed,
                co_applicant_ethnicity_observed, applicant_race_1,
                co_applicant_race_1, applicant_race_observed,
                co_applicant_race_observed, applicant_sex,
                co_applicant_sex, applicant_sex_observed,
                co_applicant_sex_observed, applicant_age, co_applicant_age,
                applicant_age_above_62, submission_of_application,
                initially_payable_to_institution, tract_population,
                tract_minority_population_percent,
                tract_to_msa_income_percentage,
                tract_median_age_of_housing_units,
                action_taken))


#One key deficiency of a polr summary is that they do not include p-values.
#The code immediately below displays the regression output from the previous
#polr summary alongside each predictor variable's p-values.

Home_Loan_Reg_Three <- polr(action_taken ~ .,
                            data = Mecklenburg_Co_Home_Loans_Rev_Three,
                            Hess = TRUE)
results_coef_Three <- coef(summary(Home_Loan_Reg_Three))
p_val_Three <- 2*pnorm(abs(results_coef_Three[, "t value"]),
                       lower.tail = FALSE)
results_table_Three <- cbind(results_coef_Three,
                             "p value" = p_val_Three)
```

```
#LDA and KNN do not work well with character variables. So this subset
#eliminates the character variables.

Mecklenburg_Co_Home_Loans_Rev_Four <-
  subset(Mecklenburg_Co_Home_Loans_Rev_Three,
  select = c(-conforming_loan_limit, -derived_dwelling_category,
             -derived_ethnicity, - negative_amortization, -total_units,
             -derived_race, -derived_sex, -debt_to_income_ratio,
             -applicant_age, -co_applicant_age,
```

```
             -applicant_age_above_62)) |>
  mutate(action_taken = as.factor(action_taken))

#This includes the variables that do not give the response "Warning:
#glm.fit: fitted probabilities numerically 0 or 1 occurred" as well as
#"Warning: NaNs produced"

Home_Loan_Reg_Four <- polr(action_taken ~ preapproval + loan_type +
  loan_purpose + lien_status + reverse_mortgage +
  open_end_line_of_credit + business_or_commercial_purpose +
  loan_to_value_ratio + interest_only_payment + balloon_payment +
  other_nonamortizing_features + occupancy_type +
  manufactured_home_secured_property_type +
  manufactured_home_land_property_interest +
  applicant_credit_score_type + co_applicant_credit_score_type +
  applicant_ethnicity_1 + co_applicant_ethnicity_1 +
  applicant_ethnicity_observed + co_applicant_ethnicity_observed +
  applicant_race_1 + co_applicant_race_1 + applicant_race_observed +
  co_applicant_race_observed + applicant_sex + co_applicant_sex +
  applicant_sex_observed + co_applicant_sex_observed +
  submission_of_application + initially_payable_to_institution +
  tract_minority_population_percent + tract_to_msa_income_percentage +
  tract_median_age_of_housing_units,
  data = Mecklenburg_Co_Home_Loans_Rev_Four, Hess = TRUE)

#The code immediately below displays the regression output from the
#previous polr summary along with the predictor variable's #p-values.

results_coef_Four <- coef(summary(Home_Loan_Reg_Four))
p_val_Four <- 2*pnorm(abs(results_coef_Four[, "t value"]),
                      lower.tail = FALSE)
results_table_Four <- cbind(results_coef_Four,
                            "p value" = p_val_Four)
```

Hello

```
library(pROC)
```

```
Type 'citation("pROC")' for a citation.


Attaching package: 'pROC'
```

```
The following objects are masked from 'package:stats':

    cov, smooth, var
```

```
library(caret)
library(reshape2)
```

```


Attaching package: 'reshape2'


The following object is masked from 'package:tidyr':

    smiths
```

```
library(MASS)
set.seed(123)


#This reduces the number of variables down to just the variables that
#LDA does not regard as multicollinear. LDA gives a warning message
#whenever multicollinearity exists. This subset ensures that when
#we use LDA, warning messages do not pop up.

Mecklenburg_Co_Home_Loans_Rev_Six <-
  subset(Mecklenburg_Co_Home_Loans_Rev_Four,
  select = c(preapproval, loan_type, loan_purpose, lien_status,
             reverse_mortgage, open_end_line_of_credit,
             business_or_commercial_purpose, loan_to_value_ratio,
             interest_only_payment, balloon_payment,
             other_nonamortizing_features, occupancy_type,
             manufactured_home_secured_property_type,
             manufactured_home_land_property_interest, income,
             applicant_credit_score_type,
             co_applicant_credit_score_type, applicant_ethnicity_1,
             co_applicant_ethnicity_1, applicant_ethnicity_observed,
             co_applicant_ethnicity_observed, applicant_race_1,
             co_applicant_race_1, applicant_race_observed,
             applicant_sex, co_applicant_sex, applicant_sex_observed,
             co_applicant_sex_observed, submission_of_application,
             initially_payable_to_institution, tract_population,
             tract_minority_population_percent,
```

```
             tract_to_msa_income_percentage,
             tract_median_age_of_housing_units, action_taken))


#This ensures that action_taken is the factor that it is. This chunk produces
#the LDA accuracy rate of our data.

Mecklenburg_Home_Loans_Rev_Seven <-  Mecklenburg_Co_Home_Loans_Rev_Six |>
  mutate(action_taken = as.factor(action_taken))

#The chunk of data we have is split in half here. Half of the data goes into
#the training dataset and the other half goes into the testing dataset.

Project_split <- floor(0.5*nrow(Mecklenburg_Home_Loans_Rev_Seven))
Project_sample <- sample(seq_len(nrow(Mecklenburg_Home_Loans_Rev_Seven)),
                         size = Project_split)
Project_training_original <-
  Mecklenburg_Home_Loans_Rev_Seven[Project_sample,]
Project_testing_original <-
  Mecklenburg_Home_Loans_Rev_Seven[-Project_sample,]
Project_training_data <-
  Mecklenburg_Home_Loans_Rev_Seven[Project_sample, -35]
Project_test_data <-
  Mecklenburg_Home_Loans_Rev_Seven[-Project_sample, -35]
Project_training_lab <-
  Project_training_original$action_taken
Project_test_lab <-
  Project_testing_original$action_taken


Project_LDA <- lda(action_taken ~ ., data = Project_training_original)
summary(Project_LDA)
```

```
        Length Class  Mode
prior     3    -none- numeric
counts    3    -none- numeric
means   102    -none- numeric
scaling  68    -none- numeric
lev       3    -none- character
svd       2    -none- numeric
N         1    -none- numeric
call      3    -none- call
```

```
terms     3    terms  call
xlevels   0    -none- list
```

```
Project_LDA_Prediction <-
  predict(Project_LDA, Project_testing_original)$class

Project_LDA_Matrix <-
  table(Project_testing_original$action_taken, Project_LDA_Prediction)
Project_LDA_Matrix
```

```
   Project_LDA_Prediction
       1    2    3
  1 9594   22  511
  2  305   75   33
  3 2028  164  393
```

```
Project_LDA_Accuracy_Rate <-
  round(mean(Project_testing_original$action_taken ==
               Project_LDA_Prediction), 5)
Project_LDA_Accuracy_Rate
```

```
[1] 0.76663
```

The LDA method accuracy rate here is 0.7663 or 76.63 percent.

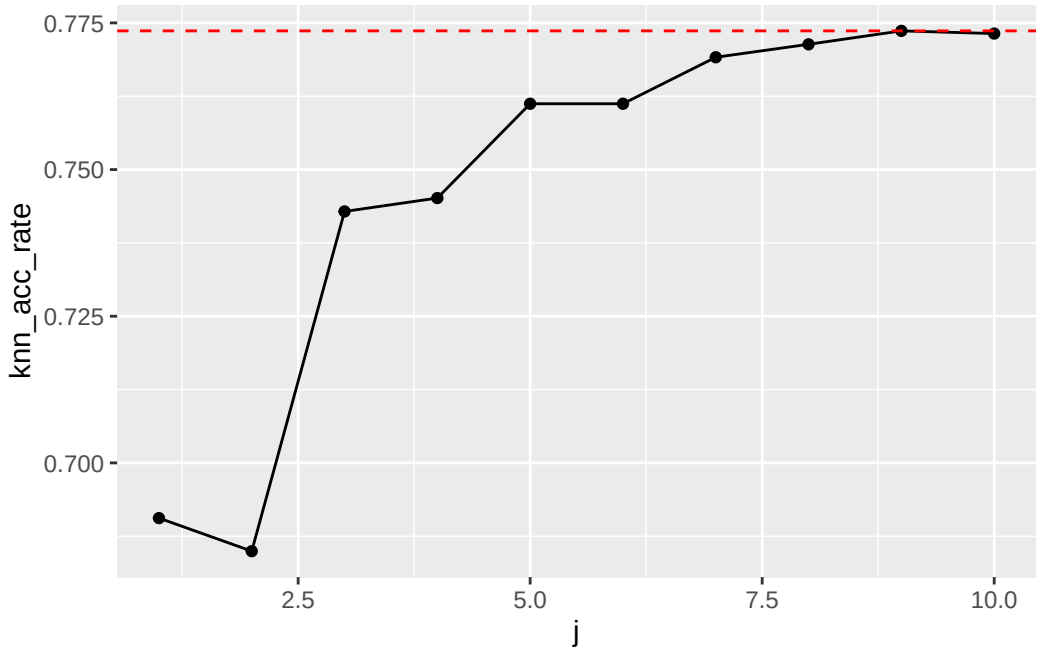
```
#This produces the highest KNN accuracy rate when we are producing a
#replication of the data up to 10. It returns the highest accuracy
#when j = 9.

set.seed(123)

knn_acc_rate <- rep(0, 10)

for (j in 1:10) {
  knn_pred <- knn(Project_training_data, Project_test_data,
                  Project_training_lab, j)
  knn_acc_rate[j] <- mean(Project_test_lab == knn_pred)
}

which.max(knn_acc_rate)
```

```
[1] 9
```

```
knn_c_acc_rate <- round(knn_acc_rate[which.max(knn_acc_rate)], 5)
knn_c_acc_rate
```

```
[1] 0.77364
```

```
#This produces the KNN plot from the data in the chunk immediately above.
ggplot(tibble(knn_acc_rate, j = 1:10), aes(x = j, y = knn_acc_rate)) +
  geom_line() +
  geom_point() +
  geom_hline(yintercept = max(knn_acc_rate), color = "red", lty = 2)
```



The value j = 9 produces the highest accuracy rate when the j value can only be a maximum of 10. Accuracy rate is only slightly higher than when j = 10. This is true because the red y-intercept dashes go through the center of j = 9 and at the top of where j = 10. The red dashes are meant to go through the center of the j value that has the highest accuracy rate.

The highest KNN accuracy rate when the j value can only be a maximum of 10 is 0.7736 or 77.36 percent. 77.36 percent is the accuracy rate for when j = 9.

```r
#Here we test our data using the Tree method and find out what it's
#accuracy rate is under the Tree method.

library(tree)
set.seed(123)
project_tree_test <- tree(action_taken ~ .,
                          data = Project_training_original)
summary(project_tree_test)
```

```
Classification tree:
tree(formula = action_taken ~ ., data = Project_training_original)
Variables actually used in tree construction:
[1] "loan_purpose"                      "initially_payable_to_institution"
[3] "income"                            "loan_to_value_ratio"
Number of terminal nodes:  6
Residual mean deviance:  1.072 = 14060 / 13120
Misclassification error rate: 0.2039 = 2676 / 13124
```

```r
tree_test_pred <- predict(project_tree_test,
                          newdata = Project_testing_original,
                          type = "class")
table(tree_test_pred, Project_testing_original$action_taken)
```

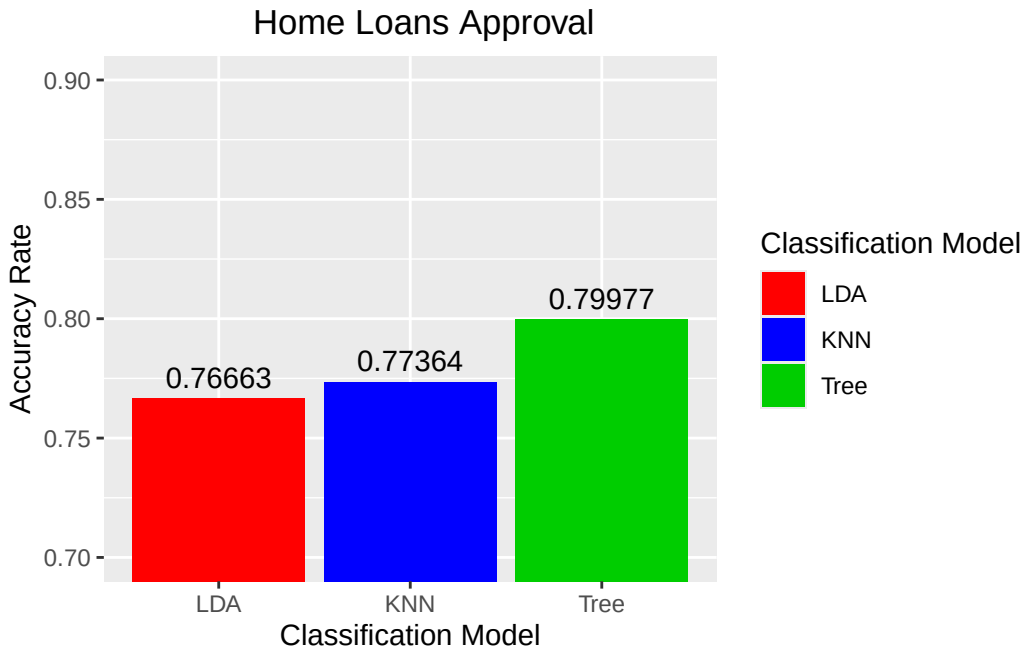
```
tree_test_pred    1    2    3
             1 9716  317 1805
             2    0   66   65
             3  411   30  715
```

```r
Final_Tree <- round(mean(Project_testing_original$action_taken ==
                           tree_test_pred), 5)
Final_Tree
```

```
[1] 0.79977
```

The Tree method accuracy rate here is 0.7998 or 79.98 percent.

```r
#This creates a bar chart of our accuracy rate results in the three code
#chunks immediately above.

Classification_Models_Race <- data.frame("Classification_Model" = c("LDA",
         "KNN", "Tree"),
       "Rate" = c(Project_LDA_Accuracy_Rate, knn_c_acc_rate, Final_Tree)) |>
  arrange(desc(`Rate`))

ggplot(Classification_Models_Race,
       aes(x = reorder(Classification_Model, Rate),
           y = Rate, fill = reorder(Classification_Model, Rate))) +
  geom_bar(stat = "identity") +
  xlab("Classification Model") +
  ylab("Accuracy Rate") +
  coord_cartesian(ylim = c(0.7, 0.9)) +
  geom_text(aes(label = Rate), vjust = -0.5) +
  labs(fill = "Classification Model") +
  ggtitle(("Home Loans Approval")) +
  theme(plot.title = element_text(hjust = 0.55)) +
  scale_fill_manual(values = c("red", "blue", "green3"))
```



Based on the results of the KNN, LDA and Tree methods, I would definitely recommend using the Tree method.

```
#This splits up action_taken into two categories. The Binomial family can
#only take two categories. So I split up action_taken into one category that
#consists of all the loans approved and another category that consists of all
#the loans denied. The loans approved are where action_taken equals 1 and 2
#and the loans denied are where action_taken equals 3.

Mecklenburg_Home_Loans_TF <- Mecklenburg_Co_Home_Loans_Rev_Two |>
  mutate(action_taken = as.numeric(action_taken)) |>
  mutate(approved = ifelse(action_taken <= 2, 1, 0))
```

```
#This includes just the predictor variables that do not give an error
#or a warning message.

Mecklenburg_Co_Home_Loans_TF_Reg <- glm(as.factor(approved) ~ census_tract +
  conforming_loan_limit + derived_loan_product_type + derived_ethnicity +
  derived_race + derived_sex + preapproval + loan_type + loan_purpose +
  lien_status + reverse_mortgage + open_end_line_of_credit +
  business_or_commercial_purpose + loan_amount + loan_to_value_ratio +
  interest_only_payment + balloon_payment + other_nonamortizing_features +
  property_value + construction_method + occupancy_type +
  manufactured_home_secured_property_type +
  manufactured_home_land_property_interest + total_units + income +
  debt_to_income_ratio + applicant_credit_score_type +
  co_applicant_credit_score_type + applicant_ethnicity_1 +
  co_applicant_ethnicity_1 + applicant_ethnicity_observed +
  co_applicant_ethnicity_observed + applicant_race_1 + co_applicant_race_1 +
  applicant_race_observed + co_applicant_race_observed + applicant_sex +
  co_applicant_sex + applicant_sex_observed + co_applicant_sex_observed +
  applicant_age + co_applicant_age + applicant_age_above_62 +
  submission_of_application + initially_payable_to_institution +
  tract_population + tract_minority_population_percent +
  tract_to_msa_income_percentage + tract_owner_occupied_units +
  tract_one_to_four_family_homes + tract_median_age_of_housing_units,
  data = Mecklenburg_Home_Loans_TF, family = "binomial")
```

```
#This subsets the data to just include the predictor variables included
#in the previous regression equation.

Mecklenburg_Home_Loans_TF_Two <- subset(Mecklenburg_Home_Loans_TF,
  select = c(census_tract, conforming_loan_limit, derived_loan_product_type,
             derived_ethnicity, derived_race, derived_sex, preapproval,
             loan_purpose, reverse_mortgage, open_end_line_of_credit,
```

```
            business_or_commercial_purpose, loan_amount, loan_to_value_ratio,
            interest_only_payment, balloon_payment,
            other_nonamortizing_features, property_value,
            construction_method, occupancy_type,
            manufactured_home_secured_property_type,
            manufactured_home_land_property_interest, total_units, income,
            debt_to_income_ratio, applicant_credit_score_type,
            co_applicant_credit_score_type, applicant_ethnicity_1,
            co_applicant_ethnicity_1, applicant_ethnicity_observed,
            co_applicant_ethnicity_observed, applicant_race_1,
            co_applicant_race_1, applicant_race_observed,
            co_applicant_race_observed, applicant_sex, co_applicant_sex,
            applicant_sex_observed, co_applicant_sex_observed,
            applicant_age, co_applicant_age, applicant_age_above_62,
            submission_of_application, initially_payable_to_institution,
            tract_population, tract_minority_population_percent,
            tract_to_msa_income_percentage, tract_owner_occupied_units,
            tract_one_to_four_family_homes,
            tract_median_age_of_housing_units, approved))

Mecklenburg_Home_Loans_TF_Two_Reg <- glm(as.factor(approved) ~ .,
                                         data = Mecklenburg_Home_Loans_TF_Two,
                                         family = "binomial")
```

```
#This removes all the variables associated with race and ethnicity.

Mecklenburg_Home_Loans_TF_Three_Reg <- glm(as.factor(approved) ~ .
  -derived_race -derived_ethnicity -applicant_race_1 -co_applicant_race_1
  -applicant_ethnicity_1 -co_applicant_ethnicity_1
  -applicant_ethnicity_observed -co_applicant_ethnicity_observed
  -applicant_race_observed -co_applicant_race_observed
  -tract_minority_population_percent,
  data = Mecklenburg_Home_Loans_TF_Two, family = "binomial")

anova(Mecklenburg_Home_Loans_TF_Three_Reg, Mecklenburg_Home_Loans_TF_Two_Reg)
```

```
Analysis of Deviance Table

Model 1: as.factor(approved) ~ (census_tract + conforming_loan_limit +
    derived_loan_product_type + derived_ethnicity + derived_race +
    derived_sex + preapproval + loan_purpose + reverse_mortgage +
    open_end_line_of_credit + business_or_commercial_purpose +
```

```
    loan_amount + loan_to_value_ratio + interest_only_payment +
    balloon_payment + other_nonamortizing_features + property_value +
    construction_method + occupancy_type + manufactured_home_secured_property_type +
    manufactured_home_land_property_interest + total_units +
    income + debt_to_income_ratio + applicant_credit_score_type +
    co_applicant_credit_score_type + applicant_ethnicity_1 +
    co_applicant_ethnicity_1 + applicant_ethnicity_observed +
    co_applicant_ethnicity_observed + applicant_race_1 + co_applicant_race_1 +
    applicant_race_observed + co_applicant_race_observed + applicant_sex +
    co_applicant_sex + applicant_sex_observed + co_applicant_sex_observed +
    applicant_age + co_applicant_age + applicant_age_above_62 +
    submission_of_application + initially_payable_to_institution +
    tract_population + tract_minority_population_percent + tract_to_msa_income_percentage +
    tract_owner_occupied_units + tract_one_to_four_family_homes +
    tract_median_age_of_housing_units) - derived_race - derived_ethnicity -
    applicant_race_1 - co_applicant_race_1 - applicant_ethnicity_1 -
    co_applicant_ethnicity_1 - applicant_ethnicity_observed -
    co_applicant_ethnicity_observed - applicant_race_observed -
    co_applicant_race_observed - tract_minority_population_percent
Model 2: as.factor(approved) ~ census_tract + conforming_loan_limit +
    derived_loan_product_type + derived_ethnicity + derived_race +
    derived_sex + preapproval + loan_purpose + reverse_mortgage +
    open_end_line_of_credit + business_or_commercial_purpose +
    loan_amount + loan_to_value_ratio + interest_only_payment +
    balloon_payment + other_nonamortizing_features + property_value +
    construction_method + occupancy_type + manufactured_home_secured_property_type +
    manufactured_home_land_property_interest + total_units +
    income + debt_to_income_ratio + applicant_credit_score_type +
    co_applicant_credit_score_type + applicant_ethnicity_1 +
    co_applicant_ethnicity_1 + applicant_ethnicity_observed +
    co_applicant_ethnicity_observed + applicant_race_1 + co_applicant_race_1 +
    applicant_race_observed + co_applicant_race_observed + applicant_sex +
    co_applicant_sex + applicant_sex_observed + co_applicant_sex_observed +
    applicant_age + co_applicant_age + applicant_age_above_62 +
    submission_of_application + initially_payable_to_institution +
    tract_population + tract_minority_population_percent + tract_to_msa_income_percentage +
    tract_owner_occupied_units + tract_one_to_four_family_homes +
    tract_median_age_of_housing_units
  Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
1     26175      17301
2     26158      16970 17   330.19 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The anova results immediately above show that removing race and ethnicity variables makes it harder to predict the outcome of whether or not a loan is approved.

```
#This removes all the character variables as well as the variable "income"
#that was otherwise causing a "Warning: glm.fit: fitted probabilities
#numerically 0 or 1 occurred"

Mecklenburg_Home_Loans_TF_Four <- subset(Mecklenburg_Home_Loans_TF_Two,
  select = c(-conforming_loan_limit, -derived_ethnicity,
             -derived_loan_product_type, -total_units, -derived_race,
             -derived_sex, -debt_to_income_ratio, -income, -applicant_age,
             -co_applicant_age, -applicant_age_above_62)) |>
  mutate(approved = as.numeric(approved))

Mecklenburg_Home_Loans_TF_Four_Reg <- glm(as.factor(approved) ~ .,
  data = Mecklenburg_Home_Loans_TF_Four, family = "binomial")
```

```
#Here we remove the remaining variables related to race and ethnicity.

Mecklenburg_Home_Loans_TF_Five_Reg <- glm(as.factor(approved) ~ .
  -applicant_race_1 -co_applicant_race_1 -applicant_ethnicity_1
  -co_applicant_ethnicity_1 -applicant_ethnicity_observed
  -co_applicant_ethnicity_observed -applicant_race_observed
  -co_applicant_race_observed -tract_minority_population_percent,
  data = Mecklenburg_Home_Loans_TF_Four, family = "binomial")

anova(Mecklenburg_Home_Loans_TF_Five_Reg, Mecklenburg_Home_Loans_TF_Four_Reg)
```

```
Analysis of Deviance Table

Model 1: as.factor(approved) ~ (census_tract + preapproval + loan_purpose +
    reverse_mortgage + open_end_line_of_credit + business_or_commercial_purpose +
    loan_amount + loan_to_value_ratio + interest_only_payment +
    balloon_payment + other_nonamortizing_features + property_value +
    construction_method + occupancy_type + manufactured_home_secured_property_type +
    manufactured_home_land_property_interest + applicant_credit_score_type +
    co_applicant_credit_score_type + applicant_ethnicity_1 +
    co_applicant_ethnicity_1 + applicant_ethnicity_observed +
    co_applicant_ethnicity_observed + applicant_race_1 + co_applicant_race_1 +
    applicant_race_observed + co_applicant_race_observed + applicant_sex +
    co_applicant_sex + applicant_sex_observed + co_applicant_sex_observed +
    submission_of_application + initially_payable_to_institution +
```

```
    tract_population + tract_minority_population_percent + tract_to_msa_income_percentage +
    tract_owner_occupied_units + tract_one_to_four_family_homes +
    tract_median_age_of_housing_units) - applicant_race_1 - co_applicant_race_1 -
    applicant_ethnicity_1 - co_applicant_ethnicity_1 - applicant_ethnicity_observed -
    co_applicant_ethnicity_observed - applicant_race_observed -
    co_applicant_race_observed - tract_minority_population_percent
Model 2: as.factor(approved) ~ census_tract + preapproval + loan_purpose +
    reverse_mortgage + open_end_line_of_credit + business_or_commercial_purpose +
    loan_amount + loan_to_value_ratio + interest_only_payment +
    balloon_payment + other_nonamortizing_features + property_value +
    construction_method + occupancy_type + manufactured_home_secured_property_type +
    manufactured_home_land_property_interest + applicant_credit_score_type +
    co_applicant_credit_score_type + applicant_ethnicity_1 +
    co_applicant_ethnicity_1 + applicant_ethnicity_observed +
    co_applicant_ethnicity_observed + applicant_race_1 + co_applicant_race_1 +
    applicant_race_observed + co_applicant_race_observed + applicant_sex +
    co_applicant_sex + applicant_sex_observed + co_applicant_sex_observed +
    submission_of_application + initially_payable_to_institution +
    tract_population + tract_minority_population_percent + tract_to_msa_income_percentage +
    tract_owner_occupied_units + tract_one_to_four_family_homes +
    tract_median_age_of_housing_units
  Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
1     26219      22710
2     26210      22570  9   140.46 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The anova results immediately above show us that even with character variables removed, the removal of any remaining variables on race and ethnicity still make it harder to predict whether a loan was approved or denied.

```
#This data only covers loans that are accepted. The differences between
#action_taken being equal to 1 and action_taken being equal to 2 are that
#when action_taken equals 1, the approved loan is accepted by the borrower
#while when action_taken equals 2 the approved loan is not accepted by the
#borrower.

Mecklenburg_Home_Loans_App <- Mecklenburg_Co_Home_Loans_Rev_Two |>
  mutate(action_taken = as.numeric(action_taken)) |>
  filter(action_taken <= 2) |>
  mutate(accepted = ifelse(action_taken == 1, 1, 0))
```

```
#This only shows the values where there is "Warning: glm.fit:
#fitted probabilities numerically 0 or 1 occurred."

Mecklenburg_Co_Home_Loans_App_Reg <- glm(as.factor(accepted) ~
  conforming_loan_limit + derived_loan_product_type +
  derived_ethnicity + derived_race + derived_sex + preapproval +
  loan_type + loan_purpose + lien_status + reverse_mortgage +
  open_end_line_of_credit + business_or_commercial_purpose +
  loan_amount + loan_to_value_ratio + interest_only_payment +
  balloon_payment + other_nonamortizing_features + property_value +
  construction_method + occupancy_type +
  manufactured_home_secured_property_type +
  manufactured_home_land_property_interest + total_units +
  debt_to_income_ratio + applicant_credit_score_type +
  co_applicant_credit_score_type + applicant_ethnicity_1 +
  co_applicant_ethnicity_1 + applicant_ethnicity_observed +
  co_applicant_ethnicity_observed + applicant_race_1 +
  co_applicant_race_1 + applicant_race_observed +
  co_applicant_race_observed + applicant_sex +
  co_applicant_sex + applicant_sex_observed +
  co_applicant_sex_observed + applicant_age +
  co_applicant_age + applicant_age_above_62 +
  submission_of_application + initially_payable_to_institution +
  tract_population + tract_minority_population_percent +
  tract_to_msa_income_percentage + tract_owner_occupied_units +
  tract_one_to_four_family_homes +
  tract_median_age_of_housing_units,
  data = Mecklenburg_Home_Loans_App, family = "binomial")
```

```
#This creates a subset that just includes the response variable and all the
#predictor variables in the equation immediately above.

Mecklenburg_Home_Loans_App_Two <- subset(Mecklenburg_Home_Loans_App,
  select = c(conforming_loan_limit, derived_loan_product_type,
             derived_ethnicity, derived_race, derived_sex, preapproval,
             loan_purpose, open_end_line_of_credit,
             business_or_commercial_purpose, loan_amount,
             loan_to_value_ratio, interest_only_payment, balloon_payment,
             other_nonamortizing_features, property_value,
             construction_method, occupancy_type,
             manufactured_home_secured_property_type,
             manufactured_home_land_property_interest, total_units,
```

```
              debt_to_income_ratio, applicant_credit_score_type,
              co_applicant_credit_score_type, applicant_ethnicity_1,
              co_applicant_ethnicity_1, applicant_ethnicity_observed,
              co_applicant_ethnicity_observed, applicant_race_1,
              co_applicant_race_1, applicant_race_observed,
              co_applicant_race_observed, applicant_sex,
              co_applicant_sex, applicant_sex_observed,
              co_applicant_sex_observed, applicant_age, co_applicant_age,
              applicant_age_above_62, submission_of_application,
              initially_payable_to_institution, tract_population,
              tract_minority_population_percent,
              tract_to_msa_income_percentage, tract_owner_occupied_units,
              tract_one_to_four_family_homes,
              tract_median_age_of_housing_units,
              accepted))

Mecklenburg_Co_Home_Loans_App_Two_Reg <-
  glm(as.factor(accepted) ~ .,
      data = Mecklenburg_Home_Loans_App_Two,
      family = "binomial")
```

```
#This eliminates the variables that take race or ethnicity into consideration.

Mecklenburg_Co_Home_Loans_App_Three_Reg <- glm(as.factor(accepted) ~ .
  -derived_race -derived_ethnicity -applicant_race_1 -co_applicant_race_1
  -applicant_ethnicity_1 -co_applicant_ethnicity_1
  -applicant_ethnicity_observed -co_applicant_ethnicity_observed
  -applicant_race_observed -co_applicant_race_observed
  -tract_minority_population_percent,
  data = Mecklenburg_Home_Loans_App_Two, family = "binomial")

anova(Mecklenburg_Co_Home_Loans_App_Three_Reg,
      Mecklenburg_Co_Home_Loans_App_Two_Reg)
```

```
Analysis of Deviance Table

Model 1: as.factor(accepted) ~ (conforming_loan_limit + derived_loan_product_type +
    derived_ethnicity + derived_race + derived_sex + preapproval +
    loan_purpose + open_end_line_of_credit + business_or_commercial_purpose +
    loan_amount + loan_to_value_ratio + interest_only_payment +
    balloon_payment + other_nonamortizing_features + property_value +
    construction_method + occupancy_type + manufactured_home_secured_property_type +
```

```
    manufactured_home_land_property_interest + total_units +
    debt_to_income_ratio + applicant_credit_score_type + co_applicant_credit_score_type +
    applicant_ethnicity_1 + co_applicant_ethnicity_1 + applicant_ethnicity_observed +
    co_applicant_ethnicity_observed + applicant_race_1 + co_applicant_race_1 +
    applicant_race_observed + co_applicant_race_observed + applicant_sex +
    co_applicant_sex + applicant_sex_observed + co_applicant_sex_observed +
    applicant_age + co_applicant_age + applicant_age_above_62 +
    submission_of_application + initially_payable_to_institution +
    tract_population + tract_minority_population_percent + tract_to_msa_income_percentage +
    tract_owner_occupied_units + tract_one_to_four_family_homes +
    tract_median_age_of_housing_units) - derived_race - derived_ethnicity -
    applicant_race_1 - co_applicant_race_1 - applicant_ethnicity_1 -
    co_applicant_ethnicity_1 - applicant_ethnicity_observed -
    co_applicant_ethnicity_observed - applicant_race_observed -
    co_applicant_race_observed - tract_minority_population_percent
Model 2: as.factor(accepted) ~ conforming_loan_limit + derived_loan_product_type +
    derived_ethnicity + derived_race + derived_sex + preapproval +
    loan_purpose + open_end_line_of_credit + business_or_commercial_purpose +
    loan_amount + loan_to_value_ratio + interest_only_payment +
    balloon_payment + other_nonamortizing_features + property_value +
    construction_method + occupancy_type + manufactured_home_secured_property_type +
    manufactured_home_land_property_interest + total_units +
    debt_to_income_ratio + applicant_credit_score_type + co_applicant_credit_score_type +
    applicant_ethnicity_1 + co_applicant_ethnicity_1 + applicant_ethnicity_observed +
    co_applicant_ethnicity_observed + applicant_race_1 + co_applicant_race_1 +
    applicant_race_observed + co_applicant_race_observed + applicant_sex +
    co_applicant_sex + applicant_sex_observed + co_applicant_sex_observed +
    applicant_age + co_applicant_age + applicant_age_above_62 +
    submission_of_application + initially_payable_to_institution +
    tract_population + tract_minority_population_percent + tract_to_msa_income_percentage +
    tract_owner_occupied_units + tract_one_to_four_family_homes +
    tract_median_age_of_housing_units
  Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
1     20985     6157.1
2     20968     6111.1 17   46.083 0.0001677 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The anova results immediately above show that within the approved loans, it is harder to predict whether or not they are accepted by the borrower when the variables regarding race and ethnicity are removed.

```
#This removes all the character variables.

Mecklenburg_Home_Loans_App_Four <-
  subset(Mecklenburg_Home_Loans_App_Two,
  select = c(-conforming_loan_limit, -derived_ethnicity,
             -derived_loan_product_type, -total_units,
             -derived_race, -derived_sex, -debt_to_income_ratio,
             -applicant_age, -co_applicant_age,
             -applicant_age_above_62)) |>
  mutate(accepted = as.numeric(accepted))

Mecklenburg_Home_Loans_App_Four_Reg <- glm(as.factor(accepted) ~ .,
  data = Mecklenburg_Home_Loans_App_Four, family = "binomial")
```

```
#This removes the remaining predictor variables that take race or ethnicity
#into consideration.

Mecklenburg_Home_Loans_App_Five_Reg <- glm(as.factor(accepted) ~ .
  -applicant_race_1 -co_applicant_race_1 -applicant_ethnicity_1
  -co_applicant_ethnicity_1 -applicant_ethnicity_observed
  -co_applicant_ethnicity_observed -applicant_race_observed
  -co_applicant_race_observed -tract_minority_population_percent,
  data = Mecklenburg_Home_Loans_App_Four, family = "binomial")

anova(Mecklenburg_Home_Loans_TF_Five_Reg, Mecklenburg_Home_Loans_TF_Four_Reg)
```

```
Analysis of Deviance Table

Model 1: as.factor(approved) ~ (census_tract + preapproval + loan_purpose +
    reverse_mortgage + open_end_line_of_credit + business_or_commercial_purpose +
    loan_amount + loan_to_value_ratio + interest_only_payment +
    balloon_payment + other_nonamortizing_features + property_value +
    construction_method + occupancy_type + manufactured_home_secured_property_type +
    manufactured_home_land_property_interest + applicant_credit_score_type +
    co_applicant_credit_score_type + applicant_ethnicity_1 +
    co_applicant_ethnicity_1 + applicant_ethnicity_observed +
    co_applicant_ethnicity_observed + applicant_race_1 + co_applicant_race_1 +
    applicant_race_observed + co_applicant_race_observed + applicant_sex +
    co_applicant_sex + applicant_sex_observed + co_applicant_sex_observed +
    submission_of_application + initially_payable_to_institution +
    tract_population + tract_minority_population_percent + tract_to_msa_income_percentage +
    tract_owner_occupied_units + tract_one_to_four_family_homes +
```

```
    tract_median_age_of_housing_units) - applicant_race_1 - co_applicant_race_1 -
    applicant_ethnicity_1 - co_applicant_ethnicity_1 - applicant_ethnicity_observed -
    co_applicant_ethnicity_observed - applicant_race_observed -
    co_applicant_race_observed - tract_minority_population_percent
Model 2: as.factor(approved) ~ census_tract + preapproval + loan_purpose +
    reverse_mortgage + open_end_line_of_credit + business_or_commercial_purpose +
    loan_amount + loan_to_value_ratio + interest_only_payment +
    balloon_payment + other_nonamortizing_features + property_value +
    construction_method + occupancy_type + manufactured_home_secured_property_type +
    manufactured_home_land_property_interest + applicant_credit_score_type +
    co_applicant_credit_score_type + applicant_ethnicity_1 +
    co_applicant_ethnicity_1 + applicant_ethnicity_observed +
    co_applicant_ethnicity_observed + applicant_race_1 + co_applicant_race_1 +
    applicant_race_observed + co_applicant_race_observed + applicant_sex +
    co_applicant_sex + applicant_sex_observed + co_applicant_sex_observed +
    submission_of_application + initially_payable_to_institution +
    tract_population + tract_minority_population_percent + tract_to_msa_income_percentage +
    tract_owner_occupied_units + tract_one_to_four_family_homes +
    tract_median_age_of_housing_units
  Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
1     26219      22710
2     26210      22570  9   140.46 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The anova results immediately above show once again, it is harder to predict whether or not the approved loans are accepted when race and ethnicity are not taken into consideration. This goes to show that even within just the approved loans, race and ethnicity still matter when trying to examine whether or not approved loans are accepted.