EAS 345: Phase 01
Systemic racism in America
Aghose, 10/09/2020

Author: Akash Ghose.

Area of research: Social issues surrounding racial tension in America.

Title of project: Does systemic racism exist in America?

Potential clients: People who wish to view the data about social disparities in America

Potential sponsors: People who wish to inform others of the existence (or non-existence) of social disparities in America

Potential data sources:

FBI Crime/Arrest data:

https://ucr.fbi.gov/crime-in-the-u.s/2019/crime-in-the-u.s.-2019/tables/table-49

https://ucr.fbi.gov/crime-in-the-u.s/2018/crime-in-the-u.s.-2018/tables/table-49

**Replace 'YEAR' with actual year, dating back until 1995.

***US Sentencing commission data*:**

https://www.ussc.gov/sites/default/files/pdf/research-and-publications/annual-reports-and-sourcebooks/2019/2019-Annual-Report-and-Sourcebook.pdf

^Of interest here, 56% of federal offenders were Hispanic. Even though Hispanics make up a very small percentage of the US population. How is this possible?

US census bureau population data for the last 10 years:

https://www.census.gov/data/tables/time-series/demo/popest/2010s-national-detail.html

***Consumer Financial Protection Bureau data*:**

https://www.consumerfinance.gov/data-research/hmda/historic-data/

Goal:

In the Oct. 7, 2020 Vice Presidential debate, Mike Pence declared that systematic racism does not exist in America. My goal with this project is to aggregate data to prove or disprove that notion. I will try to do so by looking at the public data available and attempt to compare race vs arrests vs population size, race vs severity of crime vs incarceration rate/time, race vs mortgage loans denied/accepted, race vs income/job opportunities, race vs educational opportunities. In the end, I hope to be able to use this data to paint a very clear and coherent picture about social disparities in America, and aggregate it all in a very clean and concise place for all to view.

Phase 02: Data Collection

Akash Ghose, 10/23/20

Crime Related Data sources

Data source 01:

Name of files: 2019_FBI_arrests_by_race_total.csv
2019_FBI_arrests_by_race_under18.csv
2019_FBI_arrests_by_race_18_and_over.csv

Source: https://ucr.fbi.gov/crime-in-the-u.s/2019/crime-in-the-u.s.-2019/topic-pages/tables/table-49

Details: As the name suggests, this contains data about arrests in 2019. It contains details such as the race of the perpetrators and the type of crime they were arrested for.

**Of note: I currently only downloaded the data provided for 2019, because I am not sure I need more than one year's data. So, for the sake of cleanliness, I have limited the data. However, if in the future, I need/want to get more data, it can be obtained with little to no effort.

Finance Related Data sources

Data source 01:

Name of file: NFWBS_PUF_2016_data_readable.csv

Source: https://www.consumerfinance.gov/data-research/financial-well-being-survey-data/

Details: This is the National Financial Wellbeing Survey data from a survey that was conducted in 2016. This contains details about respondents and respondents' financial well-being, including characteristics like income, age, race, savings, past financial experiences, financial skills, behaviors, attitudes ect.

**Of note: The original file I downloaded was: NFWBS_PUF_2016_data.csv. I used NFWBS_PUF_2016_read_in_R.R to read the file and then write it into the more readable csv.

Data source 02:

Name of file: hmda_2017_nationwide_all-records_labels.csv

Source: https://www.consumerfinance.gov/data-research/hmda/historic-data/?geo=nationwide&records=all-records&field_descriptions=labels

Details: This contains all the mortgage applications filed in 2017. It contains data about the applications and applicants, including details such as applicants' demographics and whether the application was accepted or rejected.

US Population Data source

Data source 01:

Name of file: US_population_est_2010-2019.csv

Source: https://www.census.gov/data/tables/time-series/demo/popest/2010s-national-detail.html#par_textimage_1537638156

Details: Contains US population estimates from 2010-2019. Includes separation by race as well as the totals.

UB box that contains my data:

https://buffalo.box.com/s/9231grwf8pw2sjs5jhkvodavc6z21gt3

Phase: 03, Data cleaning and processing

<u>Steps I have taken to clean my data:</u>

- Dropped irrelevant rows.
    - I have dropped rows using both the native "-" operator and also the dplyr slice method.
    - I did this because these rows were unnecessary

```r
#Getting rid of unnecessary rows/rows without data from the FBI arrest datasets
intermediate_FBI_arrest_by_race_under18 <- X2019_FBI_arrests_by_race_under18[-c(1:6,39:43),]
intermediate_FBI_arrest_by_race_total <- X2019_FBI_arrests_by_race_total %>% slice(-c(1:6,39:42),)
```

- Dropped rows with NA values.
    - I did this because these rows were unnecessary

```r
intermediate_FBI_arrest_by_race_18_and_over <- na.omit(X2019_FBI_arrests_by_race_18_and_over) %>% slice(-c(32),)
```

- Dropped irrelevant columns.
    - I dropped ethnicity data because I do not need them. Race data is sufficient for my intents and purposes.

```r
#Removing irrelevant columns (features)
intermediate_FBI_arrest_by_race_total <- select(intermediate_FBI_arrest_by_race_total, -c(14:19))
intermediate_FBI_arrest_by_race_under18 <- select(intermediate_FBI_arrest_by_race_under18, -c(14:19))
intermediate_FBI_arrest_by_race_18_and_over <- select(intermediate_FBI_arrest_by_race_18_and_over, -c(14:19))
```

- Changed column values so I can use them as column names
    - I wanted to assign my first row to be column names (as that is how the data is mean to be read), however, as it stood, R wouldn't let me do so because it wanted the column names to be unique, and the values in the first row were not unique.
    - So, I had to change the values (by adding "%" in front of values that need them) so that it can be read the way it was meant to be read

```r
#Changing column values so I can use them as column names later
indecies <- seq(8,13)
for(i in indecies){
  "For each of the columns 8:13,
  add a '%' sign in front of the values of the first row"
  val <- intermediate_FBI_arrest_by_race_total[1,i]
  intermediate_FBI_arrest_by_race_total[1,i] = paste("%",val)

  val <- intermediate_FBI_arrest_by_race_under18[1,i]
  intermediate_FBI_arrest_by_race_under18[1,i] = paste("%",val)

  val <- intermediate_FBI_arrest_by_race_18_and_over[1,i]
  intermediate_FBI_arrest_by_race_18_and_over[1,i] = paste("%",val)
}
```

- Assigned first row to be column names
  - Instead of column names being just numbers, they are now properly labeled
  - Also, the first row (which contained what are now the column names) is dropped as it becomes redundant here.

```r
#Assigning appropriate column names for ease of readability
names(intermediate_FBI_arrest_by_race_total) <- intermediate_FBI_arrest_by_race_total[1,]
names(intermediate_FBI_arrest_by_race_under18) <- intermediate_FBI_arrest_by_race_under18[1,]
names(intermediate_FBI_arrest_by_race_18_and_over) <- intermediate_FBI_arrest_by_race_18_and_over[1,]

#Dropping the first rows as they are no longer needed
intermediate_FBI_arrest_by_race_total <- intermediate_FBI_arrest_by_race_total[-c(1),]
intermediate_FBI_arrest_by_race_under18 <- intermediate_FBI_arrest_by_race_under18[-c(1),]
intermediate_FBI_arrest_by_race_18_and_over <- intermediate_FBI_arrest_by_race_18_and_over[-c(1),]
```

- Changed data values from character to numeric
- Changed first column from characters to factors
  - Both of the last two changes were done so that I have an easier time analyzing the data in the EDA phase

```r
#Changing the data values from character to numeric
intermediate_FBI_arrest_by_race_total[,2:13] <- lapply(2:13, function(x) as.numeric(intermediate_FBI_arrest_by_race_total[[x]]))
intermediate_FBI_arrest_by_race_under18[,2:13] <- lapply(2:13, function(x) as.numeric(intermediate_FBI_arrest_by_race_under18[[x]]))
intermediate_FBI_arrest_by_race_18_and_over[,2:13] <- lapply(2:13, function(x) as.numeric(intermediate_FBI_arrest_by_race_18_and_over[[x]]

#Changing the first column into factors
intermediate_FBI_arrest_by_race_total[,1] <- lapply(1, function(x) as.factor(intermediate_FBI_arrest_by_race_total[[x]]))
intermediate_FBI_arrest_by_race_under18[,1] <- lapply(1, function(x) as.factor(intermediate_FBI_arrest_by_race_under18[[x]]))
intermediate_FBI_arrest_by_race_18_and_over[,1] <- lapply(1, function(x) as.factor(intermediate_FBI_arrest_by_race_18_and_over[[x]]))
```

Phase: 04, EDA and Data engineering

<u>List of EDA steps I have taken</u>:

- Used
  - Head()
  - Tail()
  - Summary()
  - Colnames()
  - View()
- In various places throughout this phase and throughout the previous data cleaning phase to get a better understanding of the data I'm dealing with and figure out what to do next. For example, head and tail were useful in figuring out quickly whether the top and bottom of the data were similar, whether there were any inconsistencies that needed to be dealt with. Summary() gave me a whole lot of useful information. To start, it would tell me quickly if the data I am dealing with numbers as it seems or characters. With my mortgage data, summary() told me that I have 51 NAs in my "cleaned" mortgage_data$loan_amount_000s. It also told me that the minimum amount of loan requested was $1000 and maximum was $30,000,000, which I thought was interesting. Colnames() was needed because I realized that some of the column names were not what they seemed. For example, in my FBI arrests data, I see a column name as "Black or African American" when I look at it with view(), but the actual column name is "Black or\r\nAfrican\r\nAmerican". View() was used frequently not only to get a wholistic idea of the raw data, but also to see if the changes I was making while cleaning was behaving the way I expected them to.

```r
"Initial EDA of financial_well_being_survey"
summary(financial_well_being_survey)
colnames(financial_well_being_survey)

"Initial EDA of mortgage_data"
colnames(mortgage_data)
head(mortgage_data)
tail(mortgage_data)
summary(mortgage_data)
```

```r
#Initial look at the cleaned data sets
View(FBI_arrest_by_race_18_and_over)
View(FBI_arrest_by_race_under18)
View(FBI_arrest_by_race_total)
head(FBI_arrest_by_race_total)
colnames(FBI_arrest_by_race_total)
```

```
> summary(mortgage_data)
 loan_amount_000s  preapproval_name    preapproval    action_taken_name   action_taken    applicant_race_name_1 applicant_race_1
 Min.   :    1.0   Length:138236      Min.   :1.000   Length:138236      Min.   :1.000   Length:138236         Min.   :1.000
 1st Qu.:  100.0   Class :character   1st Qu.:3.000   Class :character   1st Qu.:1.000   Class :character      1st Qu.:5.000
 Median :  177.0   Mode  :character   Median :3.000   Mode  :character   Median :2.000   Mode  :character      Median :5.000
 Mean   :  221.3                      Mean   :2.786                      Mean   :2.416                         Mean   :4.521
 3rd Qu.:  285.0                      3rd Qu.:3.000                      3rd Qu.:3.000                         3rd Qu.:5.000
 Max.   :30000.0                      Max.   :3.000                      Max.   :8.000                         Max.   :7.000
 NA's   :51
 applicant_race_name_2 applicant_race_2 applicant_race_name_3 applicant_race_3 applicant_race_name_4 applicant_race_4
 Length:138236         Min.   :1.00     Length:138236         Min.   :1.00     Mode:logical          Mode:logical
 Class :character      1st Qu.:4.00     Class :character      1st Qu.:3.00     NA's:138236           NA's:138236
 Mode  :character      Median :5.00     Mode  :character      Median :5.00
                       Mean   :4.31                           Mean   :4.36
                       3rd Qu.:5.00                           3rd Qu.:5.00
                       Max.   :5.00                           Max.   :5.00
                       NA's   :136643                         NA's   :138059
 applicant_race_name_5 applicant_race_5 co_applicant_race_name_1 co_applicant_race_1 co_applicant_race_name_2 co_applicant_race_2
 Mode:logical          Mode:logical     Length:138236            Min.   :1.000       Length:138236            Min.   :1.00
 NA's:138236           TRUE:3           Class :character         1st Qu.:5.000       Class :character         1st Qu.:4.00
                       NA's:138233      Mode  :character         Median :8.000       Mode  :character         Median :5.00
                                                                 Mean   :6.647                                Mean   :4.45
                                                                 3rd Qu.:8.000                                3rd Qu.:5.00
                                                                 Max.   :8.000                                Max.   :5.00
                                                                                                              NA's   :137732
 co_applicant_race_name_3 co_applicant_race_3 co_applicant_race_name_4 co_applicant_race_4 co_applicant_race_name_5 co_applicant_race_5
 Mode:logical             Mode:logical        Mode:logical             Mode:logical        Mode:logical             Mode:logical
 NA's:138236              TRUE:1              NA's:138236              NA's:138236         NA's:138236              TRUE:1
                          NA's:138235                                                                               NA's:138235



 applicant_sex_name applicant_sex   co_applicant_sex_name co_applicant_sex applicant_income_000s denial_reason_name_1 denial_reason_1
 Length:138236      Min.   :1.000   Length:138236         Min.   :1.000    Min.   :     1.0      Length:138236        Min.   :1.00
 Class :character   1st Qu.:1.000   Class :character      1st Qu.:2.000    1st Qu.:    49.0      Class :character     1st Qu.:1.00
 Mode  :character   Median :1.000   Mode  :character      Median :5.000    Median :    75.0      Mode  :character     Median :3.00
                    Mean   :1.331                         Mean   :3.711    Mean   :   109.9                           Mean   :3.72
                    3rd Qu.:2.000                         3rd Qu.:5.000    3rd Qu.:   118.0                           3rd Qu.:5.00
                    Max.   :2.000                         Max.   :5.000    Max.   :175000.0                           Max.   :9.00
                                                                           NA's   :5992                               NA's   :111388
 denial_reason_name_2 denial_reason_2  denial_reason_name_3 denial_reason_3   population    minority_population hud_median_family_income
 Length:138236        Min.   :1.00     Length:138236        Min.   :1.00     Min.   :    0   Min.   :  0.00      Min.   : 20500
 Class :character     1st Qu.:3.00     Class :character     1st Qu.:3.00     1st Qu.: 3702   1st Qu.: 10.02      1st Qu.: 62600
 Mode  :character     Median :3.00     Mode  :character     Median :5.00     Median : 5009   Median : 23.04      Median : 72400
                      Mean   :4.28                          Mean   :5.55     Mean   : 5751   Mean   : 33.58      Mean   : 70592
                      3rd Qu.:6.00                          3rd Qu.:9.00     3rd Qu.: 6631   3rd Qu.: 51.72      3rd Qu.: 77500
                      Max.   :9.00                          Max.   :9.00     Max.   :53812   Max.   :100.00      Max.   :131500
                      NA's   :132297                        NA's   :137367   NA's   :577     NA's   :577         NA's   :577
 tract_to_msamd_income number_of_owner_occupied_units number_of_1_to_4_family_units
```

- Used dplyr techniques such as:
  - Select()
    - Select was used in a few different places primarily to drop columns that were unnecessary or those that became obsolete
  - Slice()
    - Slice() was used a couple of times to get rid of unwanted rows

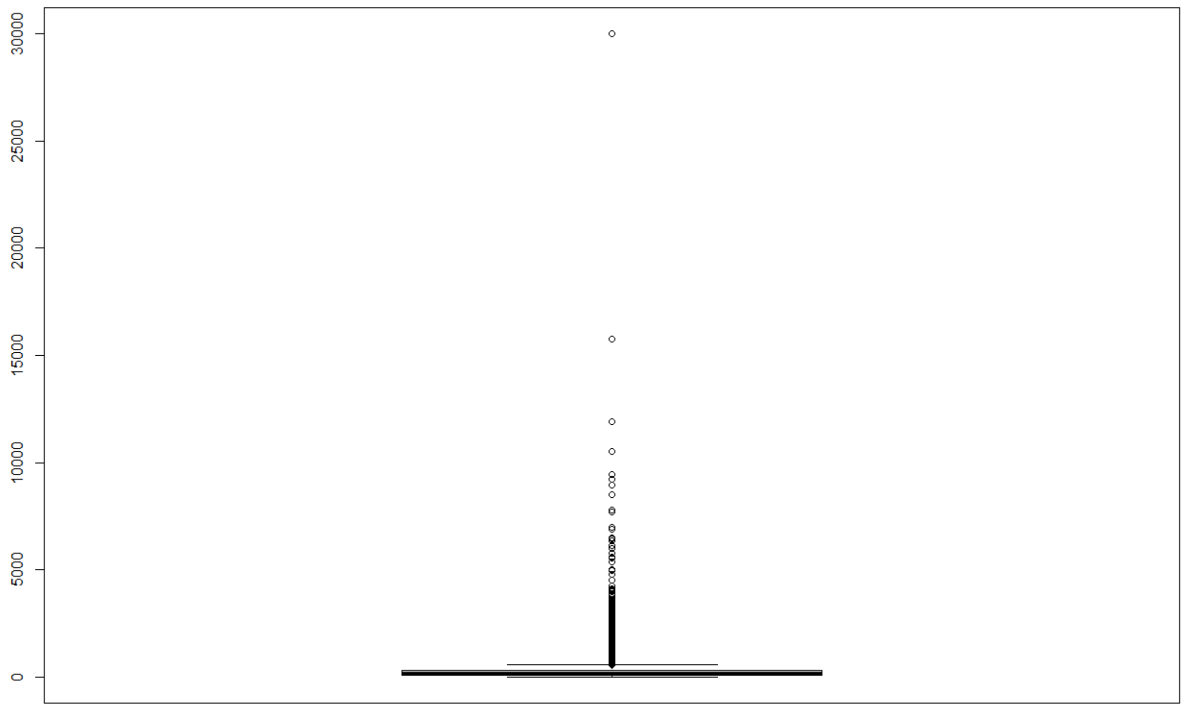
- %>%
  - The pipe operator was used extensively throughout the last two phases for a multitude of reasons, including for simplicity and sake of readability
- Mutate()
  - Was used to add my own column with information about all non-white races in my FBI arrest datasets
- Relocate()
  - Was used to re-arrange the placement of my recently added column for readability purposes.

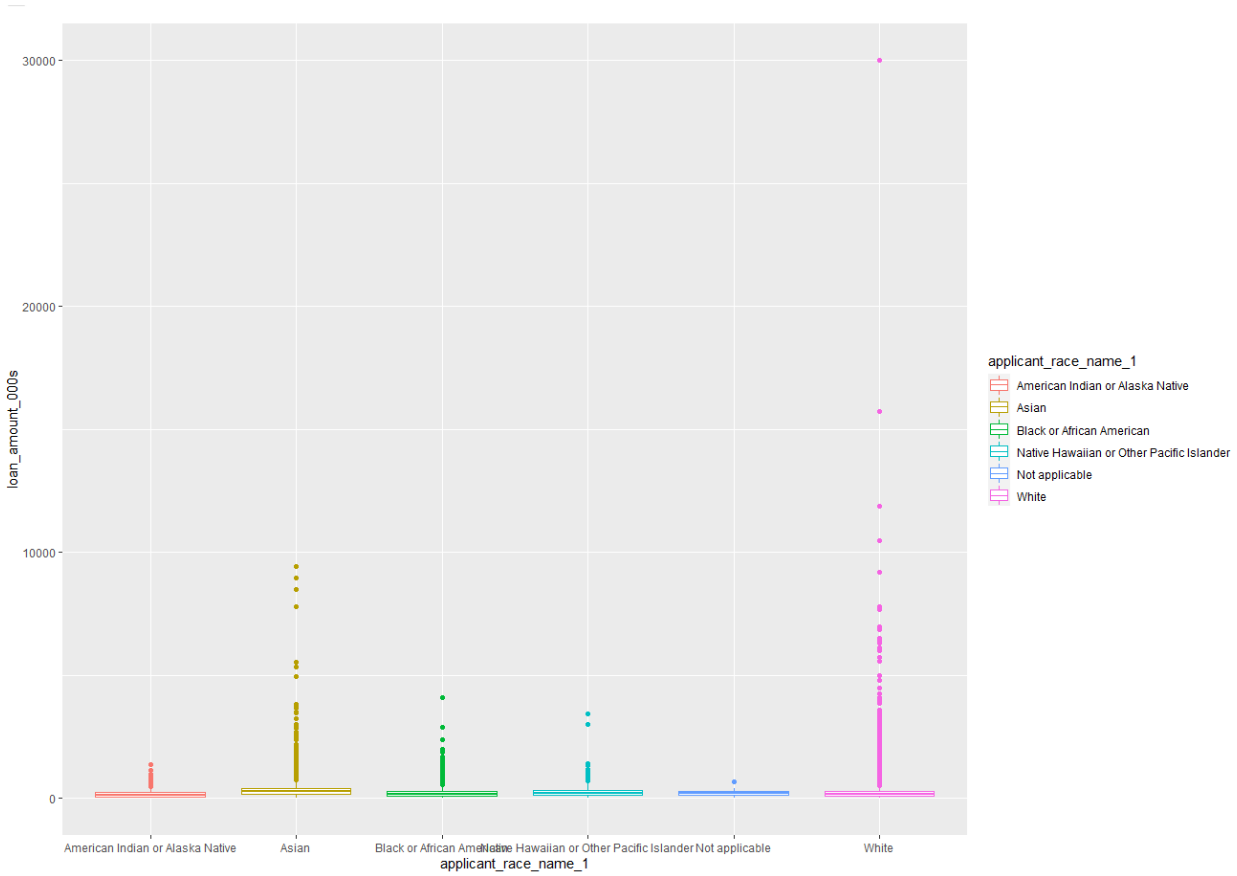
```
#Combine non-white races into a single column
#Combine %non-whites into a single column
#Drop first column because it is useless
#relocate the new rows to a better position for readability
alt_FBI_arrest_by_race_total <-FBI_arrest_by_race_total %>%
  mutate(Non_white = .[[5]]+.[[6]]+.[[7]]+.[[8]]) %>%
  mutate("% Non_white" = .[[11]]+.[[12]]+.[[13]]+.[[14]]) %>%
  select(-c(1)) %>%
  relocate(Non_white, .after = "White") %>%
  relocate("% Non_white", .after = "% White")
alt_FBI_arrest_by_race_under18 <-FBI_arrest_by_race_under18 %>%
  mutate(Non_white = .[[5]]+.[[6]]+.[[7]]+.[[8]]) %>%
  mutate("% Non_white" = .[[11]]+.[[12]]+.[[13]]+.[[14]]) %>%
  select(-c(1)) %>%
  relocate(Non_white, .after = "White") %>%
  relocate("% Non_white", .after = "% White")
alt_FBI_arrest_by_race_18_and_over <-FBI_arrest_by_race_18_and_over %>%
  mutate(Non_white = .[[5]]+.[[6]]+.[[7]]+.[[8]]) %>%
  mutate("% Non_white" = .[[11]]+.[[12]]+.[[13]]+.[[14]]) %>%
  select(-c(1)) %>%
  relocate(Non_white, .after = "White") %>%
  relocate("% Non_white", .after = "% White")
```

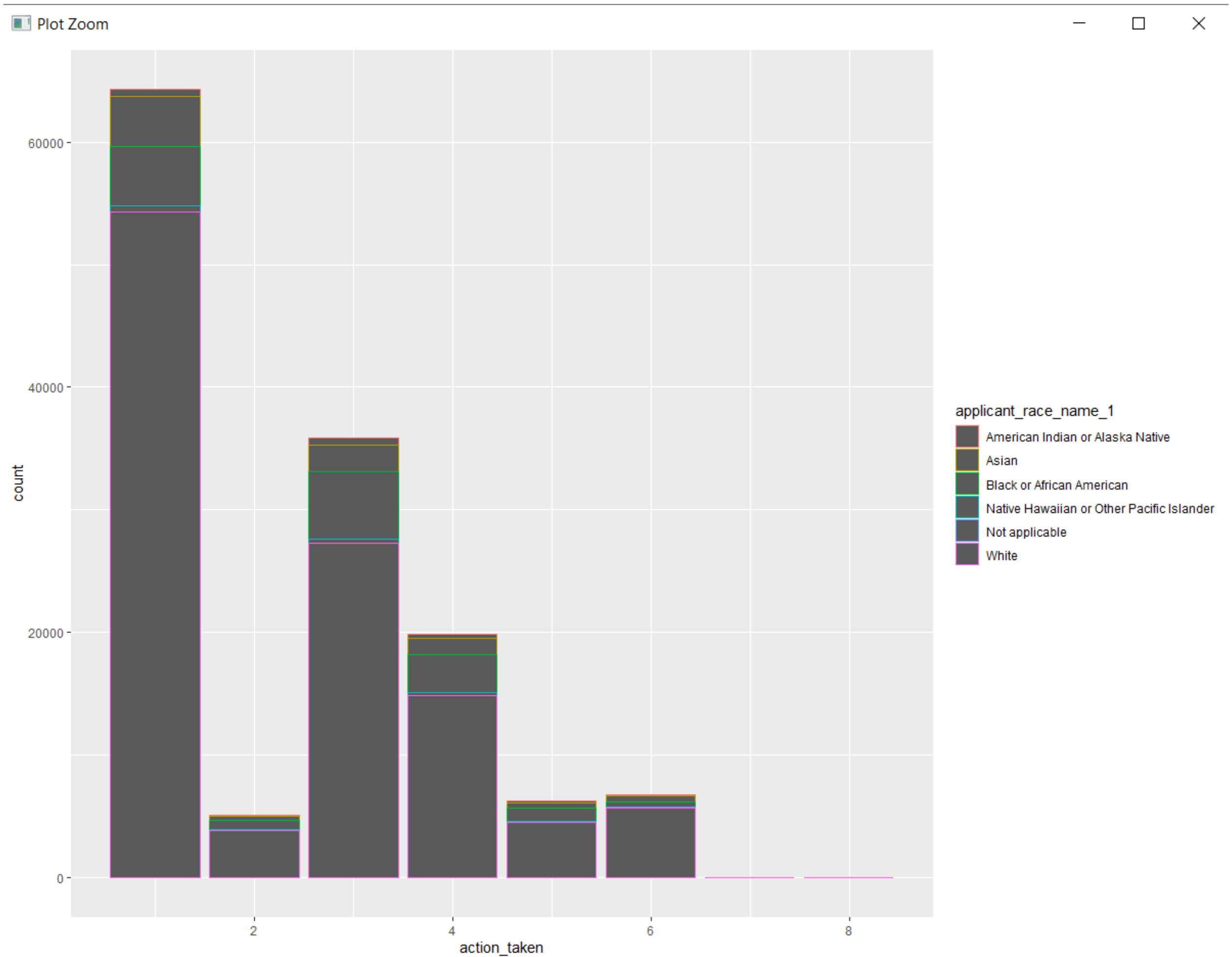
- Used the following techniques to create graphs:
  - Boxplot()
    - I initially created the boxplot in hopes of learning some useful information about the loan amounts that were requested. I found none, so I moved on to geom_boxplot
  - Geom_boxplot()
    - After tinkering with this a little bit, I was able to graph something that actually showed me useful information
  - Geom_bar()

- The goem barplot was used to draw the number of loans that were accepted and denied and with colors to show much of it belonged to each race

```r
mortgage_data <- select(mortgage_data, -c(1))

boxplot(mortgage_data$loan_amount_000s)

ggplot(mortgage_data, aes(y=loan_amount_000s, x=applicant_race_name_1,
                          color=applicant_race_name_1)) + geom_boxplot()

ggplot(mortgage_data, aes(x=action_taken_name)) + geom_bar()

barplot <- ggplot(mortgage_data, aes(x=action_taken,
                                     color=applicant_race_name_1)) + geom_bar()
barplot

```




30000
20000
10000
0
loan_amount_000s
American Indian or Alaska Native
Asian
Black or African American
Native Hawaiian or Other Pacific Islander
Not applicable
White
applicant_race_name_1
applicant_race_name_1
American Indian or Alaska Native
Asian
Black or African American
Native Hawaiian or Other Pacific Islander
Not applicable
White


Plot Zoom
count
60000
40000
20000
0
2
4
6
8
action_taken
applicant_race_name_1
American Indian or Alaska Native
Asian
Black or African American
Native Hawaiian or Other Pacific Islander
Not applicable
White

Phase 05: Modeling and analysis

Modeling algorithms used:

Linear regression:

- The intent while using this algorithm was to try and predict either the loan amount that would be requested or the applicant's income, given the available information I have about the applicant
- I made 4 different models.
- The first one tried to model loan vs the applicant's income and applicant's race, where the amount of loan requested was the output/what the model would be predicting, and income and race were the inputs.
  - The p-value from this model was very good, less than 2.2e-16, which tells me that my variables had a very high degree of relatability.
  - However, the R-squared and Adjusted R-squared values were not that high, less than 0.3, which meant that my predictability power of my model was not that good.
- I tried to increase the R-squared by adding more variables for my second and third model, i.e. co-applicant's presence and co-applicant's race.
  - Neither model crossed the 0.3 threshold for the R-squared, but I had the highest R-squared and Adjusted R-squared of all the models when using just income, race, and presence of co-applicant as my inputs.
- For my fourth lm model, I used loan amount requested, applicant's race and presence of co-applicant to try and predict the applicant's income.
  - Again, I found that I had very low (good) p-values, but my R-squared did not cross 0.3, unfortunately.
- I think the reason why my lm model does not have good predicting power is because, while there is significant correlation between all the variables (i.e. white people are more likely to request for larger loans than say black people, or people with higher incomes would ask for higher sums etc.), *most* people still

just never ask for more than a few thousand dollars; they all cluster near the bottom.

```

"LM models"

#LM - loan vs applicant_income and applicant_race
lm_model_00 <- lm(loan_amount_000s ~ applicant_income_000s+
                    applicant_race_1,
                  data = mortgage_data_subset0)
lm_model_00
summary(lm_model_00)
plot(lm_model_00)

#Going to add more variables to perhaps make the lm_model more accurate as
#LM - loan vs income level, applicant_income, applicant_race and presence
lm_model_01 <- lm(loan_amount_000s ~ applicant_income_000s+
                    applicant_race_1+
                    co_applicant,
                  data = mortgage_data_subset0)
lm_model_01
summary(lm_model_01)
plot(lm_model_01)
#ggplot(lm_model_01) + aes(col=applicant_race_1)

#LM - loan vs income level, applicant_income, applicant_race and co_applic
lm_model_02 <- lm(loan_amount_000s ~ applicant_income_000s+
                    applicant_race_1+
                    co_applicant_race_1,
                  data = mortgage_data_subset0)
lm_model_02
summary(lm_model_02)
plot(lm_model_02)

#Going to try using the LM algorithm to try to predict income instead
lm_model_03 <- lm(applicant_income_000s ~ loan_amount_000s+
                    applicant_race_1+
                    co_applicant,
                  data = mortgage_data_subset0)
lm_model_03
summary(lm_model_03)
plot(lm_model_03)

```

```
> summary(lm_model_00)

Call:
lm(formula = loan_amount_000s ~ applicant_income_000s + applicant_race_1,
    data = mortgage_data_subset0)

Residuals:
     Min       1Q   Median       3Q      Max 
-15306.2    -97.2    -28.8     60.6  19947.3 

Coefficients:
                        Estimate Std. Error t value Pr(>|t|)    
(Intercept)           217.149833   2.828705   76.77   <2e-16 ***
applicant_income_000s   0.600336   0.003109  193.09   <2e-16 ***
applicant_race_1      -13.999386   0.602541  -23.23   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 210.1 on 127723 degrees of freedom
Multiple R-squared:  0.2284,	Adjusted R-squared:  0.2284 
F-statistic: 1.891e+04 on 2 and 127723 DF,  p-value: < 2.2e-16
```

```
> summary(lm_model_01)

Call:
lm(formula = loan_amount_000s ~ applicant_income_000s + applicant_race_1 +
    co_applicant, data = mortgage_data_subset0)

Residuals:
     Min       1Q   Median       3Q      Max 
-15092.8    -96.0    -28.0     59.7  20070.6 

Coefficients:
                        Estimate Std. Error t value Pr(>|t|)    
(Intercept)           210.711042   2.832448   74.39   <2e-16 ***
applicant_income_000s   0.591744   0.003119  189.72   <2e-16 ***
applicant_race_1      -15.197943   0.602783  -25.21   <2e-16 ***
co_applicantTRUE       30.937813   1.201040   25.76   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 209.5 on 127722 degrees of freedom
Multiple R-squared:  0.2324,	Adjusted R-squared:  0.2324 
F-statistic: 1.289e+04 on 3 and 127722 DF,  p-value: < 2.2e-16
```

```
> summary(lm_model_02)

Call:
lm(formula = loan_amount_000s ~ applicant_income_000s + applicant_race_1 +
    co_applicant_race_1, data = mortgage_data_subset0)

Residuals:
     Min       1Q   Median       3Q      Max
-15084.3    -95.8    -27.9     59.6  20076.4

Coefficients:
                        Estimate Std. Error t value Pr(>|t|)
(Intercept)           274.143276   3.477607   78.83   <2e-16 ***
applicant_income_000s   0.591496   0.003116  189.84   <2e-16 ***
applicant_race_1      -12.608832   0.602750  -20.92   <2e-16 ***
co_applicant_race_1    -9.440509   0.337071  -28.01   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 209.4 on 127722 degrees of freedom
Multiple R-squared:  0.2332,    Adjusted R-squared:  0.2331
F-statistic: 1.294e+04 on 3 and 127722 DF,  p-value: < 2.2e-16
```

```
> summary(lm_model_03)

Call:
lm(formula = applicant_income_000s ~ loan_amount_000s + applicant_race_1 +
    co_applicant, data = mortgage_data_subset0)

Residuals:
    Min      1Q  Median      3Q     Max
-2561.7   -37.5   -12.0    18.4 26573.4

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
(Intercept)     -6.697964   2.292424  -2.922  0.00348 **
loan_amount_000s 0.371542   0.001958 189.722  < 2e-16 ***
applicant_race_1 4.509857   0.478658   9.422  < 2e-16 ***
co_applicantTRUE 20.631821  0.952408  21.663  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 166 on 127722 degrees of freedom
Multiple R-squared:  0.2288,    Adjusted R-squared:  0.2288
F-statistic: 1.263e+04 on 3 and 127722 DF,  p-value: < 2.2e-16
```

KNN Classification:

- I wanted to use the KNN classification algorithm to try and see if I could make a model which could predict whether a loan would be approved or denied, given the loan amount requested, the applicant’s income, race and presence of co-applicant

- I took several steps to further clean my data to prepare for this process, including (but not limited to) normalizing my data so that they are all on the same scale (0 to 1).
- I used roughly 90% of my data to train the model and 10% to test it
- I made two models: one with a NN coefficient (k) of 357, which is roughly the square root of my total observations and one where k is 5 to see if one of them was better than the other
- I used a confusion matrix to determine the accuracy of my models.
  - I found that my first model, where K=357, was more accurate than my other model
  - Model_00 was accurate roughly 74.3% of the time, where as model_01 was accurate only 68.6% of the time
  - I chose to stick with model_00, but either way, they were both decently accurate.

```r
"KNN models"

#First, I will further clean the data to get a more workable data set
knn_data_set <- mortgage_data_subset0 %>%
  mutate(applicant_white = if_else(applicant_race_1==5,
                                   true = 0,
                                   false = 1)) %>%
  mutate(co_applicant_white = if_else(co_applicant_race_1==5,
                                      true = 0,
                                      false = 1)) %>%
  select(-(contains("name") | contains("race"))) %>%
  #Changing co-applicant from boolean to binary so KNN algorithm can
  #Calculate the Euclidean distance
  mutate(co_applicant = if_else(co_applicant==TRUE,
                                true = 0,
                                false = 1)) %>%
  #similar thing for pre-aproval
  mutate(preapproval = if_else(preapproval == 3,
                               true = 1,
                               false = 0)) %>%
  select(-(c(1, 5, denial_reason_1:number_of_1_to_4_family_units))) %>%
  relocate(loan_approved, .after= last_col())

#Creating a function to normalize all my variables
normalize <- function(x) {
  return( (x - min(x))/(max(x) - min(x)))
}

#Normalizing my dataset
knn_data_set_norm <- as.data.frame(lapply(knn_data_set[, 1:8], normalize))

n <- 114953 #roughly 90% of my data
k <- 357 #roughly the sqrt of my total observations

knn_train <- knn_data_set_norm[1:n,]
knn_test <- knn_data_set_norm[(n+1):nrow(knn_data_set_norm),]

knn_train_target <- knn_data_set[1:n, 9]
knn_test_target <- knn_data_set[(n+1):nrow(knn_data_set), 9]

#Time for the actual model
knn_model_00 <- knn(train = knn_train, test = knn_test,
                    cl= knn_train_target, k= k)

knn_model_00

tbl <- table(knn_test_target, knn_model_00)
```

```r
knn_model_00

tbl_00 <- table(knn_test_target, knn_model_00)
tbl_00

plot(knn_model_00)
plot(tbl_00)
confusionMatrix(tbl_00)

#K = 5
knn_model_01 <- knn(train = knn_train, test = knn_test,
                    cl= knn_train_target, k= 5)
tbl_01 <- table(knn_test_target, knn_model_01)
tbl_01
confusionMatrix(tbl_01)
```

```
> library(caret)
> confusionMatrix(tbl_00)
Confusion Matrix and Statistics

               knn_model_00
knn_test_target FALSE TRUE
          FALSE   739 2640
          TRUE    636 8758

               Accuracy : 0.7435
                 95% CI : (0.7359, 0.7511)
    No Information Rate : 0.8924
    P-Value [Acc > NIR] : 1
```

```
> confusionMatrix(tbl_01)
Confusion Matrix and Statistics

               knn_model_01
knn_test_target FALSE TRUE
          FALSE  1173 2206
          TRUE   1802 7592

               Accuracy : 0.6862
                 95% CI : (0.6781, 0.6943)
    No Information Rate : 0.7671
    P-Value [Acc > NIR] : 1
```

KMeans:

- I used this algorithm in hopes of organizing my data into similar clusters so that I may be able to extrapolate some useful insights from these clusters.
- I made a few different models. The first one I built; I initially used the normalized dataset I used for my KNN classification.

    - While this model had was the best fit of the models I tried (had the highest between_SS / total_SS, 75.1 %), it did not allow for any useful interpretation of the data. At least none that I could find.
  - So then, I tried making a couple different models with non-normalized knn-data with 5 clusters (the number 5 was chosen arbitrarily).
    - These models were not as good of a fit as my first one (between_SS / total_SS = 66.4%), but in model_02, I found that loan amounts clustered around 347,000 had the highest rate of approval, which I found was interesting.

```r
#KMeans - feed it multiple columns, i.e. loan_amount, race, income, and see wha
"K-Means"
kmeans_dataset <- knn_data_set_norm
kmeans_model_00 <- kmeans(kmeans_dataset,5)
kmeans_model_00

clnnames <- c("loan_amount_000s", "preapproval", "co_applicant",
              "applicant_sex", "co_applicant_sex", "applicant_income_000s",
              "applicant_white", "co_applicant_white","loan_approved")

kmeans_dataset <- knn_data_set
kmeans_model_01 <- kmeans(kmeans_dataset,5)
kmeans_model_01

kmeans_dataset <-knn_data_set %>% select(clnnames[c(1,2,3,6,7,9)])
kmeans_model_02 <- kmeans(kmeans_dataset,5)
kmeans_model_02
```

```
> kmeans_dataset <- knn_data_set_norm
> kmeans_model_00 <- kmeans(kmeans_dataset,5)
> kmeans_model_00
K-means clustering with 5 clusters of sizes 30066, 44880, 6636, 10242, 35902

Cluster means:
  loan_amount_000s preapproval co_applicant applicant_sex co_applicant_sex applicant_income_000s applicant_white co_applicant_white
1      0.005663869   0.7979778            1     1.0000000        1.0000000           0.002518030      0.24502761         1.00000000
2      0.006879755   0.7844029            1     0.0000000        1.0000000           0.003521546      0.18638592         1.00000000
3      0.009714916   0.7501507            0     0.2703436        0.1838080           0.004527733      0.96745027         0.96986136
4      0.006972549   0.7862722            0     1.0000000        0.0270455           0.004213168      0.02890061         0.03134153
5      0.008202660   0.8178096            0     0.0000000        0.2406621           0.004776645      0.01877333         0.02206005

Clustering vector:
   [1] 5 1 1 1 1 2 4 2 2 2 1 1 2 4 1 4 3 1 5 2 5 5 1 2 5 5 2 5 5 2 2 2 2 2 2 2 2 2 5 2 3 1 5 2 5 4 5 2 1 1 5 1 2 4 5 1 2 2 5 1 5 5 2 2
  [70] 1 2 1 5 2 1 1 5 2 5 5 5 1 1 2 5 1 2 2 1 1 1 2 1 5 2 2 1 2 2 1 1 1 1 2 5 3 5 5 2 2 4 2 2 2 5 1 2 1 2 5 2 4 2 2 1 1 5 4 5 1 2 4 1
 [139] 3 2 5 1 5 2 5 2 5 2 1 2 1 1 5 3 2 1 2 1 1 5 4 1 1 2 3 5 2 2 1 1 2 2 2 4 1 4 4 5 1 1 2 2 1 2 2 1 4 2 2 5 2 2 4 2 2 1 1 1 1 1 1 2
 [208] 3 1 4 2 3 5 2 5 5 1 4 2 2 1 2 1 2 2 2 2 5 4 2 4 2 2 2 1 1 1 4 3 5 2 1 1 2 4 2 1 2 2 1 2 5 1 2 1 3 1 1 1 5 1 2 1 2 1 1 3 2 1 2 2
 [277] 4 5 1 1 1 2 5 3 3 1 1 5 4 2 1 1 5 1 2 4 4 2 2 3 5 1 2 5 2 5 5 1 2 2 2 1 1 2 5 5 2 3 2 5 2 4 5 1 5 2 1 2 1 2 2 1 1 3 5 5 5 1 5 2
 [346] 5 1 5 2 3 1 2 2 2 5 2 1 5 5 4 2 2 2 1 4 2 1 5 4 5 4 5 2 1 1 2 2 5 2 1 5 1 2 5 5 1 1 2 2 3 5 1 1 5 1 5 2 4 1 2 1 5 1 1 1 2 1 2 1
 [415] 1 2 3 3 2 2 1 1 5 1 1 5 2 2 2 5 4 2 5 1 2 1 2 5 1 3 1 1 5 5 1 1 5 1 4 5 4 2 1 4 2 2 2 1 1 2 2 5 2 1 5 2 3 2 5 2 4 2 1 1 5 2 5 5
 [484] 5 2 5 3 2 2 2 3 1 2 2 2 5 2 5 5 5 1 2 5 1 2 1 5 2 1 2 4 2 1 2 2 1 5 5 5 5 1 3 1 2 5 3 5 5 2 1 2 1 1 2 2 5 5 1 2 2 1 2 5 5 1 2 1
 [553] 2 5 2 2 1 5 1 5 4 2 2 4 3 5 5 5 1 2 4 4 3 2 2 5 1 4 5 1 2 5 1 1 5 2 5 2 1 5 5 2 5 5 2 5 4 1 2 3 2 5 5 2 2 1 2 5 5 1 1 5 5 2 2 5
 [622] 5 1 2 4 5 2 5 3 5 1 1 5 3 2 1 5 1 1 2 1 5 1 4 5 2 5 5 1 2 4 5 5 1 2 5 2 4 2 1 5 2 5 5 1 1 2 1 1 3 5 1 5 2 2 1 1 1 2 1 5 5 1 2 1
 [691] 2 4 4 1 5 4 5 4 1 2 5 2 2 2 5 1 3 2 4 2 1 5 4 2 2 2 2 1 1 2 3 5 2 2 5 2 5 2 2 1 2 2 1 5 1 2 1 5 5 3 2 1 5 2 4 5 2 5 3 4 5 5 2 2
 [760] 2 5 5 1 2 5 2 2 5 1 5 1 3 5 1 2 2 5 3 1 1 3 1 1 1 1 5 2 1 5 5 5 2 5 5 5 2 5 5 1 4 3 1 1 5 2 1 4 2 1 3 1 3 4 5 2 4 5 2 2 2 2 2 3
 [829] 5 5 1 1 3 5 4 1 2 4 3 5 5 2 2 1 2 1 1 3 5 2 1 2 2 5 5 1 2 2 1 5 5 2 5 2 2 4 2 5 5 5 2 2 1 5 5 1 2 4 2 4 2 2 2 3 5 1 5 5 5 2 4 2
 [898] 2 1 5 2 5 1 5 2 4 1 5 5 1 2 2 5 5 3 2 3 5 1 2 5 5 1 2 2 1 2 2 2 5 4 1 5 3 2 4 1 5 2 1 2 5 2 5 4 2 1 5 2 1 5 5 2 5 2 2 2 1 1 2 1
 [967] 2 2 1 5 1 1 1 1 4 5 5 4 1 2 2 5 4 2 5 1 2 5 1 5 2 2 2 1 1 2 5 2 1 2
 [ reached getOption("max.print") -- omitted 126726 entries ]

Within cluster sum of squares by cluster:
[1] 10409.879 14400.830  3037.004  2381.838  6872.722
 (between_SS / total_SS =  75.1 %)
```

```
> kmeans_dataset <- knn_data_set
> kmeans_model_01 <- kmeans(kmeans_dataset,5)
> kmeans_model_01
K-means clustering with 5 clusters of sizes 3736, 38686, 85099, 196, 9

Cluster means:
  loan_amount_000s preapproval co_applicant applicant_sex co_applicant_sex applicant_income_000s applicant_white co_applicant_white
1         865.0356   0.8078158    0.4480728      1.199946         3.237687             376.04684       0.2194861          0.5685225
2         347.4261   0.7612573    0.4777697      1.267384         3.321951             135.12666       0.2159696          0.5714987
3         118.5041   0.8101270    0.6426750      1.364117         3.839834              69.71181       0.1634802          0.6833570
4        3047.2500   0.8724490    0.4795918      1.127551         3.341837            1764.32143       0.1632653          0.5357143
5        6162.6667   0.8888889    0.4444444      1.000000         3.111111           14951.33333       0.3333333          0.4444444
  loan_approved
1     0.6686296
2     0.7517965
3     0.6406538
4     0.5969388
5     0.3333333

Clustering vector:
   [1] 3 3 3 3 3 3 2 3 3 2 3 3 3 3 3 3 3 3 3 2 1 3 3 3 3 2 2 2 3 2 3 3 2 2 3 3 3 3 3 3 2 3 2 3 3 3 3 2 3 3 2 3 2 3 3 3 2 2 3 2 3 2 3 3 2
  [70] 2 3 2 2 3 3 3 2 3 3 3 3 2 3 2 3 3 3 3 3 3 3 3 3 3 3 3 3 2 3 3 3 3 2 3 3 2 3 3 3 2 2 2 2 2 2 3 3 3 3 3 3 3 3 1 3 1 3 2 3 3 2 2 2 2
 [139] 3 3 3 3 3 2 2 3 2 3 3 3 2 2 3 3 3 2 2 3 3 2 2 3 3 2 3 3 3 2 3 2 3 3 3 3 3 3 3 2 3 3 2 3 2 3 3 3 3 3 3 2 3 3 2 3 3 3 3 3 3 2 2 3 2 3
 [208] 2 3 3 3 3 2 3 2 3 3 2 3 3 2 2 3 2 2 3 3 3 3 3 3 2 2 3 3 2 3 3 2 3 2 3 3 3 2 3 3 3 3 3 3 2 3 3 3 3 3 3 2 3 3 3 3 2 2 2 2 3 2 3 3 3 3
 [277] 3 3 3 3 1 3 3 3 3 3 3 3 2 3 3 3 3 3 3 2 3 3 3 2 3 3 3 3 3 3 3 2 3 2 3 3 2 3 3 3 2 3 2 3 3 3 3 3 1 3 2 2 3 2 2 3 2 2 3 2 3 2 3 3 3
 [346] 2 3 2 2 2 3 3 2 3 3 2 2 2 3 1 3 3 3 3 2 2 3 3 1 2 3 2 3 3 2 2 2 3 3 3 3 3 3 3 3 2 3 3 3 1 3 3 3 3 3 2 2 2 3 3 3 3 3 3 3 3 3 3 2 3 2
 [415] 3 3 3 2 3 3 3 3 3 2 2 3 3 3 3 1 3 2 3 3 3 2 2 2 3 3 3 3 2 2 3 3 2 3 3 2 2 3 3 2 2 1 3 3 3 3 2 3 3 2 3 3 3 3 3 2 3 3 3 3 3 2 3 3 3 2
 [484] 2 3 3 3 2 2 3 3 3 3 3 2 2 3 2 3 3 3 3 2 3 3 2 3 3 3 3 3 2 3 2 2 3 2 3 3 2 3 2 3 3 2 3 3 3 2 2 3 3 3 3 3 3 2 3 3 2 2 3 3 3 3 3 3 3 2
 [553] 3 3 3 3 3 3 3 2 3 1 3 3 3 3 2 3 3 3 3 2 3 3 3 3 3 2 3 3 2 3 3 2 3 3 2 3 3 2 3 3 2 3 3 3 3 3 3 3 3 3 3 2 3 3 3 3 3 3 3 3 2 2 3 2 3
 [622] 3 2 2 3 2 3 3 3 2 3 3 2 3 3 3 3 3 3 3 3 3 3 2 2 3 2 3 3 3 2 3 2 3 3 2 3 3 3 3 2 3 3 3 3 3 2 3 2 2 3 3 2 3 3 2 3 2 3 2 3 3 3 3 2 3 3
 [691] 3 3 3 3 2 2 2 2 3 3 3 2 3 3 2 3 3 3 2 3 2 3 3 3 2 3 2 3 3 3 2 3 2 2 3 3 2 3 3 3 3 2 2 2 3 2 3 3 2 3 2 3 3 3 3 2 3 2 1 3 2 3 3 3 2
 [760] 3 3 3 3 3 3 2 3 3 2 3 3 3 3 3 3 2 3 3 3 3 3 2 3 3 3 3 2 3 1 3 3 3 1 2 3 3 3 3 3 3 2 3 3 3 3 2 2 3 3 3 3 2 3 2 2 3 2 3 3 1 3 3 2 2
 [829] 3 3 3 3 3 3 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3 2 3 3 3 3 3 2 2 3 3 3 2 2 2 3 3 3 3 2 3 2 3 3 3 3 1 2 2 3 3 3 3 2 2 2 3 3 3 2 2 3 3 3 3
 [898] 3 3 3 3 2 3 3 3 3 3 3 3 2 2 3 3 3 3 2 3 2 3 3 2 2 2 2 3 3 3 3 3 3 3 3 3 3 1 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2 3 2 3 2 3 2
 [967] 3 3 3 3 1 3 2 3 2 1 3 3 3 2 2 3 3 2 1 3 2 3 3 3 3 3 3 2 3 3 2 3 3 3
 [ reached getOption("max.print") -- omitted 126726 entries ]

Within cluster sum of squares by cluster:
[1]  714332550  665893685  544931106  957642977 1110048236
 (between_SS / total_SS =  66.4 %)
```

```
> kmeans_dataset <-knn_data_set %>% select(clnnames[c(1,2,3,6,7,9)])
> kmeans_model_02 <- kmeans(kmeans_dataset,5)
> kmeans_model_02
K-means clustering with 5 clusters of sizes 38686, 85099, 196, 3736, 9

Cluster means:
  loan_amount_000s preapproval co_applicant applicant_income_000s applicant_white loan_approved
1         347.4261   0.7612573    0.4777697             135.12666       0.2159696     0.7517965
2         118.5041   0.8101270    0.6426750              69.71181       0.1634802     0.6406538
3        3047.2500   0.8724490    0.4795918            1764.32143       0.1632653     0.5969388
4         865.0356   0.8078158    0.4480728             376.04684       0.2194861     0.6686296
5        6162.6667   0.8888889    0.4444444           14951.33333       0.3333333     0.3333333

Clustering vector:
   [1] 2 2 2 2 2 2 1 2 2 1 2 2 2 2 2 2 2 2 2 1 4 2 2 2 2 1 1 1 2 1 2 2 1 1 2 2 2 2 2 2 1 2 1 2 2 2 2 1 2 2 1
  [70] 1 2 1 1 2 2 2 1 2 2 2 2 1 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 1 2 2 1 2 2 2 1 1 1 1 1 1 2 2 2 2 2
 [139] 2 2 2 2 2 1 1 2 1 2 2 2 1 1 2 2 2 1 1 2 2 1 1 2 2 1 2 2 2 1 2 1 2 2 2 2 2 2 2 1 2 2 1 2 1 2 2 2 2 2 2
 [208] 1 2 2 2 2 1 2 1 2 2 1 2 2 1 1 2 1 1 2 2 2 2 2 1 1 2 2 1 2 2 1 2 1 2 2 2 1 2 2 2 2 2 1 2 2 2 2 2 2 1 2
 [277] 2 2 2 2 4 2 2 2 2 2 2 2 1 2 2 2 2 2 2 1 2 2 2 1 2 2 2 2 2 2 2 1 2 1 2 2 1 2 2 2 1 2 1 2 2 2 2 2 4 2 1
 [346] 1 2 1 1 1 2 2 1 2 2 1 1 1 2 4 2 2 2 2 1 1 2 2 4 1 2 1 2 2 1 1 1 2 2 2 2 2 2 2 2 1 2 2 2 4 2 2 2 2 2 1
 [415] 2 2 2 1 2 2 2 2 2 1 1 2 2 2 2 4 2 1 2 2 2 1 1 1 2 2 2 2 1 1 2 2 1 2 2 1 1 2 2 1 1 4 2 2 2 2 1 2 2 1 2
 [484] 1 2 2 2 1 1 2 2 2 2 2 1 1 2 1 2 2 2 2 1 2 2 1 2 2 2 2 2 1 2 1 1 2 1 2 2 1 2 1 2 2 1 2 2 2 1 1 2 2 2 2
 [553] 2 2 2 2 2 2 2 1 2 4 2 2 2 2 1 2 2 2 2 1 2 2 2 2 2 1 2 2 1 2 2 1 2 2 1 2 2 1 2 2 1 2 2 2 2 2 2 2 2 2 2
 [622] 2 1 1 2 1 2 2 2 1 2 2 1 2 2 2 2 2 2 2 2 2 2 1 1 2 1 2 2 2 1 2 1 2 2 1 2 2 2 2 1 2 2 2 2 2 1 2 1 1 2 2
 [691] 2 2 2 2 1 1 1 1 2 2 2 1 2 2 1 2 2 2 1 2 1 2 2 2 1 2 1 2 2 2 1 2 1 1 2 2 1 2 2 2 2 1 1 1 2 1 2 2 1 2 1
 [760] 2 2 2 2 2 2 1 2 2 1 2 2 2 2 2 2 1 2 2 2 2 2 1 2 2 2 2 1 2 4 2 2 2 4 1 2 2 2 2 2 2 1 2 2 2 2 2 1 1 2 2 2
 [829] 2 2 2 2 2 2 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 1 1 2 2 2 1 1 1 2 2 2 2 1 2 1 2 2 2 2 4 1 1 2 2
 [898] 2 2 2 2 1 2 2 2 2 2 2 2 1 1 2 2 2 2 1 2 1 2 2 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 4 2 2 2 2 2 2 2 2 2 2 2 2
 [967] 2 2 2 2 4 2 1 2 1 4 2 2 2 1 1 2 2 1 4 2 1 2 2 2 2 2 2 1 2 2 1 2 2 2
 [ reached getOption("max.print") -- omitted 126726 entries ]

Within cluster sum of squares by cluster:
[1]  665773584  544681324  957642394  714321295 1110048207
 (between_SS / total_SS =  66.4 %)
```