County.Rmd

Alessandra Bielli

2024-12-09

Load the data

```
library(readr)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
# Load the dataset
County_Data <- read_csv("County Data Project.csv")
```

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

```
## Rows: 49244 Columns: 99
```

```
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (26): lei, state_code, conforming_loan_limit, derived_loan_product_type,...
## dbl (63): activity_year, derived_msa-md, county_code, census_tract, action_t...
## lgl (10): applicant_ethnicity-4, applicant_ethnicity-5, co-applicant_ethnici...
##
## 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.
```

```
problems(County_Data)
```

```
## # A tibble: 173 x 5
##      row   col expected           actual  file
##    <int> <int> <chr>              <chr>   <chr>
##  1    11    44 a double           5-24    ""
##  2  1873    44 a double           100-149 ""
##  3  1874    44 a double           5-24    ""
##  4  3447    65 1/0/T/F/TRUE/FALSE 21      ""
##  5  3447    66 1/0/T/F/TRUE/FALSE 23      ""
```

```
##  6  4720    65 1/0/T/F/TRUE/FALSE 27      ""
##  7  6030    65 1/0/T/F/TRUE/FALSE 4       ""
##  8  6872    44 a double           100-149 ""
##  9  7070    44 a double           50-99   ""
## 10  7071    44 a double           >149    ""
## # i 163 more rows
```

Data Cleaning and Tranformation

Clean the Column 44

```
County_Data <- County_Data %>%
  mutate(total_units = as.character(total_units))
```

Clean the Logical Columns

```
County_Data <- County_Data %>%
  mutate(
    applicant_race_observed = case_when(
      applicant_race_observed %in% c(1, "TRUE") ~ TRUE,
      applicant_race_observed %in% c(0, "FALSE") ~ FALSE,
      TRUE ~ NA
    ),
    `co-applicant_ethnicity-3` = case_when(
      `co-applicant_ethnicity-3` %in% c(1, "TRUE") ~ TRUE,
      `co-applicant_ethnicity-3` %in% c(0, "FALSE") ~ FALSE,
      TRUE ~ NA
    )
  )

# Verify the Data
summary(County_Data)
```

```
##  activity_year      lei            derived_msa-md   state_code
##  Min.   :2023   Length:49244       Min.   :16740   Length:49244
##  1st Qu.:2023   Class :character   1st Qu.:16740   Class :character
##  Median :2023   Mode  :character   Median :16740   Mode  :character
##  Mean   :2023                      Mean   :16740
##  3rd Qu.:2023                      3rd Qu.:16740
##  Max.   :2023                      Max.   :16740
##
##   county_code     census_tract         conforming_loan_limit
##  Min.   :37119   Min.   :37119000101   Length:49244
##  1st Qu.:37119   1st Qu.:37119003302   Class :character
##  Median :37119   Median :37119005619   Mode  :character
##  Mean   :37119   Mean   :37119004896
##  3rd Qu.:37119   3rd Qu.:37119005921
##  Max.   :37119   Max.   :37119980300
##                  NA's   :39
##  derived_loan_product_type derived_dwelling_category derived_ethnicity
##  Length:49244              Length:49244              Length:49244
##  Class :character          Class :character          Class :character
##  Mode  :character          Mode  :character          Mode  :character
```

```
##
##
##
##
##  derived_race       derived_sex         action_taken   purchaser_type
##  Length:49244       Length:49244       Min.   :1.000   Min.   : 0.000
##  Class :character   Class :character   1st Qu.:1.000   1st Qu.: 0.000
##  Mode  :character   Mode  :character   Median :1.000   Median : 0.000
##                                        Mean   :2.422   Mean   : 5.566
##                                        3rd Qu.:4.000   3rd Qu.: 2.000
##                                        Max.   :8.000   Max.   :72.000
##
##   preapproval      loan_type      loan_purpose    lien_status
##  Min.   :1.000   Min.   :1.000   Min.   : 1.00   Min.   :1.000
##  1st Qu.:2.000   1st Qu.:1.000   1st Qu.: 1.00   1st Qu.:1.000
##  Median :2.000   Median :1.000   Median : 1.00   Median :1.000
##  Mean   :1.981   Mean   :1.208   Mean   : 7.94   Mean   :1.282
##  3rd Qu.:2.000   3rd Qu.:1.000   3rd Qu.: 4.00   3rd Qu.:2.000
##  Max.   :2.000   Max.   :4.000   Max.   :32.00   Max.   :2.000
##
##  reverse_mortgage   open-end_line_of_credit business_or_commercial_purpose
##  Min.   :   1.000   Min.   :   1.00         Min.   :   1.000
##  1st Qu.:   2.000   1st Qu.:   1.00         1st Qu.:   2.000
##  Median :   2.000   Median :   2.00         Median :   2.000
##  Mean   :   3.978   Mean   :   3.72         Mean   :   3.903
##  3rd Qu.:   2.000   3rd Qu.:   2.00         3rd Qu.:   2.000
##  Max.   :1111.000   Max.   :1111.00         Max.   :1111.000
##
##   loan_amount        loan_to_value_ratio interest_rate      rate_spread
##  Min.   :     5000   Length:49244        Length:49244       Length:49244
##  1st Qu.:   125000   Class :character    Class :character   Class :character
##  Median :   275000   Mode  :character    Mode  :character   Mode  :character
##  Mean   :   347824
##  3rd Qu.:   395000
##  Max.   :179545000
##
##   hoepa_status   total_loan_costs   total_points_and_fees origination_charges
##  Min.   :1.000   Length:49244       Length:49244          Length:49244
##  1st Qu.:2.000   Class :character   Class :character      Class :character
##  Median :2.000   Mode  :character   Mode  :character      Mode  :character
##  Mean   :2.433
##  3rd Qu.:3.000
##  Max.   :3.000
##
##  discount_points    lender_credits      loan_term
##  Length:49244       Length:49244       Length:49244
##  Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##  prepayment_penalty_term intro_rate_period  negative_amortization
##  Length:49244            Length:49244       Min.   :   1.000
```

```
##  Class :character        Class :character   1st Qu.:   2.000
##  Mode  :character        Mode  :character   Median :   2.000
##                                             Mean   :   3.982
##                                             3rd Qu.:   2.000
##                                             Max.   :1111.000
##
##  interest_only_payment balloon_payment    other_nonamortizing_features
##  Min.   :   1.000      Min.   :   1.000   Min.   :   1.000
##  1st Qu.:   2.000      1st Qu.:   2.000   1st Qu.:   2.000
##  Median :   2.000      Median :   2.000   Median :   2.000
##  Mean   :   3.878      Mean   :   3.936   Mean   :   3.974
##  3rd Qu.:   2.000      3rd Qu.:   2.000   3rd Qu.:   2.000
##  Max.   :1111.000      Max.   :1111.000   Max.   :1111.000
##
##  property_value     construction_method occupancy_type
##  Length:49244       Min.   :1.000       Min.   :1.000
##  Class :character   1st Qu.:1.000       1st Qu.:1.000
##  Mode  :character   Median :1.000       Median :1.000
##                     Mean   :1.009       Mean   :1.171
##                     3rd Qu.:1.000       3rd Qu.:1.000
##                     Max.   :2.000       Max.   :3.000
##
##  manufactured_home_secured_property_type
##  Min.   :   1.000
##  1st Qu.:   3.000
##  Median :   3.000
##  Mean   :   4.968
##  3rd Qu.:   3.000
##  Max.   :1111.000
##
##  manufactured_home_land_property_interest total_units
##  Min.   :   1.00                          Length:49244
##  1st Qu.:   5.00                          Class :character
##  Median :   5.00                          Mode  :character
##  Mean   :   6.95
##  3rd Qu.:   5.00
##  Max.   :1111.00
##
##  multifamily_affordable_units     income        debt_to_income_ratio
##  Length:49244                 Min.   :-3501.0   Length:49244
##  Class :character             1st Qu.:   74.0   Class :character
##  Mode  :character             Median :  116.0   Mode  :character
##                               Mean   :  168.3
##                               3rd Qu.:  186.0
##                               Max.   :40641.0
##                               NA's   :5721
##  applicant_credit_score_type co-applicant_credit_score_type
##  Min.   :   1.000            Min.   :   1.00
##  1st Qu.:   2.000            1st Qu.:   9.00
##  Median :   3.000            Median :   9.00
##  Mean   :   7.275            Mean   :  10.75
##  3rd Qu.:   9.000            3rd Qu.:  10.00
##  Max.   :1111.000            Max.   :1111.00
##
```

```
##  applicant_ethnicity-1 applicant_ethnicity-2 applicant_ethnicity-3
##  Min.   : 1.00         Min.   : 1.00         Min.   : 1.00
##  1st Qu.: 2.00         1st Qu.:11.00         1st Qu.:13.00
##  Median : 2.00         Median :14.00         Median :14.00
##  Mean   : 2.39         Mean   :12.61         Mean   :12.88
##  3rd Qu.: 3.00         3rd Qu.:14.00         3rd Qu.:14.00
##  Max.   :14.00         Max.   :14.00         Max.   :14.00
##  NA's   :22            NA's   :47082         NA's   :49193
##  applicant_ethnicity-4 applicant_ethnicity-5 co-applicant_ethnicity-1
##  Mode:logical          Mode:logical          Min.   : 1.000
##  NA's:49244            NA's:49244            1st Qu.: 3.000
##                                              Median : 5.000
##                                              Mean   : 4.057
##                                              3rd Qu.: 5.000
##                                              Max.   :14.000
##                                              NA's   :9
##  co-applicant_ethnicity-2 co-applicant_ethnicity-3 co-applicant_ethnicity-4
##  Min.   : 1.00            Mode:logical             Mode:logical
##  1st Qu.:11.00            NA's:49244               NA's:49244
##  Median :13.00
##  Mean   :12.48
##  3rd Qu.:14.00
##  Max.   :14.00
##  NA's   :48519
##  co-applicant_ethnicity-5 applicant_ethnicity_observed
##  Mode:logical             Min.   :1.000
##  NA's:49244               1st Qu.:2.000
##                           Median :2.000
##                           Mean   :2.109
##                           3rd Qu.:2.000
##                           Max.   :3.000
##
##  co-applicant_ethnicity_observed applicant_race-1 applicant_race-2
##  Min.   :1.0                     Min.   : 1.000   Min.   : 1.00
##  1st Qu.:2.0                     1st Qu.: 3.000   1st Qu.:21.00
##  Median :4.0                     Median : 5.000   Median :21.00
##  Mean   :3.3                     Mean   : 5.098   Mean   :19.45
##  3rd Qu.:4.0                     3rd Qu.: 6.000   3rd Qu.:23.00
##  Max.   :4.0                     Max.   :44.000   Max.   :44.00
##                                  NA's   :11       NA's   :46358
##  applicant_race-3 applicant_race-4 applicant_race-5 co-applicant_race-1
##  Min.   : 3       Mode:logical     Mode:logical     Min.   : 1.000
##  1st Qu.: 5       NA's:49244       NA's:49244       1st Qu.: 6.000
##  Median :23                                         Median : 8.000
##  Mean   :20                                         Mean   : 7.005
##  3rd Qu.:26                                         3rd Qu.: 8.000
##  Max.   :44                                         Max.   :44.000
##  NA's   :49069                                      NA's   :4
##  co-applicant_race-2 co-applicant_race-3 co-applicant_race-4
##  Min.   : 2.00       Min.   : 3.00       Mode:logical
##  1st Qu.:21.00       1st Qu.:22.00       NA's:49244
##  Median :21.00       Median :24.00
##  Mean   :19.75       Mean   :22.71
##  3rd Qu.:24.00       3rd Qu.:25.00
```

```
##  Max.   :44.00       Max.   :44.00
##  NA's   :48423       NA's   :49189
##  co-applicant_race-5 applicant_race_observed co-applicant_race_observed
##  Mode:logical        Mode:logical            Min.   :1.0
##  NA's:49244          TRUE:1652               1st Qu.:2.0
##                      NA's:47592              Median :4.0
##                                              Mean   :3.3
##                                              3rd Qu.:4.0
##                                              Max.   :4.0
##
##  applicant_sex   co-applicant_sex applicant_sex_observed
##  Min.   :1.000   Min.   :1.000    Min.   :1.00
##  1st Qu.:1.000   1st Qu.:2.000    1st Qu.:2.00
##  Median :2.000   Median :5.000    Median :2.00
##  Mean   :1.819   Mean   :3.934    Mean   :2.11
##  3rd Qu.:2.000   3rd Qu.:5.000    3rd Qu.:2.00
##  Max.   :6.000   Max.   :6.000    Max.   :3.00
##
##  co-applicant_sex_observed applicant_age      co-applicant_age
##  Min.   :1.0               Length:49244       Length:49244
##  1st Qu.:2.0               Class :character   Class :character
##  Median :4.0               Mode  :character   Mode  :character
##  Mean   :3.3
##  3rd Qu.:4.0
##  Max.   :4.0
##
##  applicant_age_above_62 co-applicant_age_above_62 submission_of_application
##  Length:49244           Length:49244              Min.   :   1.000
##  Class :character       Class :character          1st Qu.:   1.000
##  Mode  :character       Mode  :character          Median :   1.000
##                                                   Mean   :   3.262
##                                                   3rd Qu.:   1.000
##                                                   Max.   :1111.000
##
##  initially_payable_to_institution     aus-1              aus-2
##  Min.   :   1.000                 Min.   :   1.000   Min.   :1.00
##  1st Qu.:   1.000                 1st Qu.:   1.000   1st Qu.:1.00
##  Median :   1.000                 Median :   6.000   Median :2.00
##  Mean   :   3.242                 Mean   :   6.072   Mean   :2.15
##  3rd Qu.:   1.000                 3rd Qu.:   6.000   3rd Qu.:2.00
##  Max.   :1111.000                 Max.   :1111.000   Max.   :7.00
##                                                      NA's   :47015
##      aus-3           aus-4           aus-5       denial_reason-1
##  Min.   :1.00    Min.   :1.00    Min.   :1.00    Min.   :   1.00
##  1st Qu.:2.00    1st Qu.:1.00    1st Qu.:1.00    1st Qu.:  10.00
##  Median :2.00    Median :1.00    Median :1.00    Median :  10.00
##  Mean   :2.51    Mean   :1.38    Mean   :1.42    Mean   :  10.46
##  3rd Qu.:2.00    3rd Qu.:2.00    3rd Qu.:2.00    3rd Qu.:  10.00
##  Max.   :5.00    Max.   :3.00    Max.   :3.00    Max.   :1111.00
##  NA's   :48693   NA's   :49162   NA's   :49171
##  denial_reason-2 denial_reason-3 denial_reason-4 tract_population
##  Min.   :1.00    Min.   :1       Mode:logical    Min.   :   0
##  1st Qu.:3.00    1st Qu.:4       TRUE:1          1st Qu.:3272
##  Median :4.00    Median :6       NA's:49243      Median :3976
```

```
##  Mean   :4.44    Mean   :6                      Mean   :4161
##  3rd Qu.:6.00    3rd Qu.:9                      3rd Qu.:4941
##  Max.   :9.00    Max.   :9                      Max.   :8512
##  NA's   :47579   NA's   :49019
##  tract_minority_population_percent ffiec_msa_md_median_family_income
##  Min.   :  0.00                    Min.   :98700
##  1st Qu.: 29.12                    1st Qu.:98700
##  Median : 54.71                    Median :98700
##  Mean   : 53.85                    Mean   :98700
##  3rd Qu.: 78.88                    3rd Qu.:98700
##  Max.   :100.00                    Max.   :98700
##
##  tract_to_msa_income_percentage tract_owner_occupied_units
##  Min.   :  0.00                 Min.   :   0.0
##  1st Qu.: 78.01                 1st Qu.: 678.0
##  Median :111.66                 Median : 957.0
##  Mean   :121.62                 Mean   : 977.9
##  3rd Qu.:152.97                 3rd Qu.:1250.0
##  Max.   :310.61                 Max.   :2097.0
##
##  tract_one_to_four_family_homes tract_median_age_of_housing_units
##  Min.   :   0                   Min.   : 0.00
##  1st Qu.: 984                   1st Qu.:18.00
##  Median :1268                   Median :25.00
##  Mean   :1296                   Mean   :28.38
##  3rd Qu.:1618                   3rd Qu.:36.00
##  Max.   :2455                   Max.   :72.00
##
```

```r
problems(County_Data)
```

Tranform the Debt-to-Income-Ratio Column

```r
library(dplyr)
library(tidyr)


County_Data <- County_Data %>%
  mutate(debt_to_income_ratio_numeric = case_when(
    debt_to_income_ratio == "<20%" ~ 19,
    debt_to_income_ratio == "20%-<30%" ~ 25,
    debt_to_income_ratio == "30%-<36%" ~ 33,
    debt_to_income_ratio == "50%-60%" ~ 55,
    debt_to_income_ratio == ">60%" ~ 61,
    TRUE ~ NA_real_
  )) %>%
  # Remove rows with NA in the debt_to_income_ratio_numeric column
  drop_na(debt_to_income_ratio_numeric)

# Check the results
summary(County_Data$debt_to_income_ratio_numeric)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   19.00   25.00   33.00   35.81   55.00   61.00
```

For this analysis, debt_to_income_ratios <20% will be converted to 19, >60% will be converted to 61, and for the ranges like 20%-<30%, 30%-<36%, and 50%-60%, the midpoint will be used. NA values will be removed.

Clean the Derived_Race Column

```
# Remove specific values from derived_race
County_Data <- County_Data %>%
  filter(!derived_race %in% c("Free Form Text Only", "Race Not Available", "Joint"))
```

Now, 1 is "2 or more minority races" , 2 is "American Indian or Alaska Native", 3 is "Asian", 4 is "Black or African American", 5 is "Native Hawaiian or Other Pacific Islander", and 6 is White".

Ensure the Correct Column Types

```
county_data_clean <- County_Data %>%
  mutate(
    action_taken = as.factor(action_taken),
    derived_race = as.factor(derived_race),
    loan_to_value_ratio = as.numeric(loan_to_value_ratio),
    income = as.numeric(income),
    applicant_credit_score_type = as.factor(applicant_credit_score_type)
  )
```

Filter columns

```
# Filter the dataset to include only the relevant columns
county_data_clean <- county_data_clean %>%
  select(loan_to_value_ratio, income, debt_to_income_ratio_numeric,
         applicant_credit_score_type, derived_race, action_taken)

# Drop rows with missing values
county_data_clean <- county_data_clean %>%
  drop_na(income, derived_race, applicant_credit_score_type, loan_to_value_ratio, debt_to_income_ratio_n

# Summary of cleaned data
summary(county_data_clean)
```

```
##  loan_to_value_ratio     income        debt_to_income_ratio_numeric
##  Min.   :   0.333    Min.   : -208.0   Min.   :19.00
##  1st Qu.:  57.597    1st Qu.:   81.0   1st Qu.:25.00
##  Median :  75.516    Median :  133.0   Median :33.00
##  Mean   :  70.937    Mean   :  201.3   Mean   :35.52
##  3rd Qu.:  89.192    3rd Qu.:  218.0   3rd Qu.:55.00
##  Max.   : 999.990    Max.   :40641.0   Max.   :61.00
##
##  applicant_credit_score_type                                    derived_race
##  1      :3754                2 or more minority races                  :  56
##  3      :3402                American Indian or Alaska Native          :  78
##  8      :2527                Asian                                     :1384
##  2      :2501                Black or African American                 :3598
##  9      : 931                Native Hawaiian or Other Pacific Islander:  28
##  11     : 619                White                                     :9116
```

```
##  (Other): 526
##  action_taken
##  1:9841
##  2: 432
##  3:3967
##  7:   9
##  8:  11
##
##
```

Fit Linear Regression, Ridge Regression, and PLS Models

Check for MultiCollinearity

```
# Load required libraries
library(dplyr)
library(corrplot)
```

```
## corrplot 0.95 loaded
```

```
library(car)
```

```
## Loading required package: carData
```

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

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

```
# View the first few rows of the cleaned data
head(county_data_clean)
```

```
## # A tibble: 6 x 6
##   loan_to_value_ratio income debt_to_income_ratio_numeric applicant_credit_sco~1
##                 <dbl>  <dbl>                         <dbl> <fct>
## 1                68.6     59                            61 11
## 2                92.9     89                            61 11
## 3                79.1     62                            61 11
## 4                72.7    220                            55 11
## 5                58.3    268                            25 3
## 6                58.8     35                            55 3
## # i abbreviated name: 1: applicant_credit_score_type
## # i 2 more variables: derived_race <fct>, action_taken <fct>
```

```
# Ensure that 'county_data_clean' has only numeric columns for correlation
# Select only numeric columns
numeric_data <- county_data_clean %>% select(where(is.numeric))

# Compute the correlation matrix, handle missing values
cor_matrix <- cor(numeric_data, use = "complete.obs")

# Fit the linear model using numeric predictors
lm_model <- lm(loan_to_value_ratio ~ income + debt_to_income_ratio_numeric +
                 applicant_credit_score_type + derived_race + action_taken,
```

```r
               data = county_data_clean)

# Summary of the linear model
summary(lm_model)
```

```
##
## Call:
## lm(formula = loan_to_value_ratio ~ income + debt_to_income_ratio_numeric +
##     applicant_credit_score_type + derived_race + action_taken,
##     data = county_data_clean)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -85.73 -12.92   4.25  16.33 935.16
##
## Coefficients:
##                                                              Estimate  Std. Error
## (Intercept)                                                 84.8162625   3.5626679
## income                                                       0.0012104   0.0004376
## debt_to_income_ratio_numeric                                 0.2678860   0.0180890
## applicant_credit_score_type2                                 1.6981081   0.6721114
## applicant_credit_score_type3                                -1.2524416   0.6164263
## applicant_credit_score_type4                                11.2935080  26.0132261
## applicant_credit_score_type6                                -6.0781358   5.1194188
## applicant_credit_score_type7                                 3.6312974   1.2441567
## applicant_credit_score_type8                               -10.2395517   0.6949914
## applicant_credit_score_type9                                -3.1199300   0.9580533
## applicant_credit_score_type11                              -13.5408445   1.1421143
## derived_raceAmerican Indian or Alaska Native               -23.2778373   4.5560192
## derived_raceAsian                                          -15.8295301   3.5502396
## derived_raceBlack or African American                      -19.7435973   3.5028000
## derived_raceNative Hawaiian or Other Pacific Islander      -27.4099437   6.0235141
## derived_raceWhite                                          -19.7199919   3.4931631
## action_taken2                                               -2.0231414   1.2816758
## action_taken3                                               -6.4730100   0.6004564
## action_taken7                                                9.1163092   8.6815492
## action_taken8                                               16.6564064   7.8461380
##                                                            t value
## (Intercept)                                                 23.807
## income                                                       2.766
## debt_to_income_ratio_numeric                                14.809
## applicant_credit_score_type2                                 2.527
## applicant_credit_score_type3                                -2.032
## applicant_credit_score_type4                                 0.434
## applicant_credit_score_type6                                -1.187
## applicant_credit_score_type7                                 2.919
## applicant_credit_score_type8                               -14.733
## applicant_credit_score_type9                                -3.257
## applicant_credit_score_type11                              -11.856
## derived_raceAmerican Indian or Alaska Native                -5.109
## derived_raceAsian                                           -4.459
## derived_raceBlack or African American                       -5.637
## derived_raceNative Hawaiian or Other Pacific Islander       -4.550
## derived_raceWhite                                           -5.645
```

```
## action_taken2                                          -1.579
## action_taken3                                         -10.780
## action_taken7                                           1.050
## action_taken8                                           2.123
##                                                                Pr(>|t|)
## (Intercept)                                         < 0.0000000000000002 ***
## income                                                           0.00569 **
## debt_to_income_ratio_numeric                        < 0.0000000000000002 ***
## applicant_credit_score_type2                                     0.01153 *
## applicant_credit_score_type3                                     0.04219 *
## applicant_credit_score_type4                                     0.66419
## applicant_credit_score_type6                                     0.23514
## applicant_credit_score_type7                                     0.00352 **
## applicant_credit_score_type8                        < 0.0000000000000002 ***
## applicant_credit_score_type9                                     0.00113 **
## applicant_credit_score_type11                       < 0.0000000000000002 ***
## derived_raceAmerican Indian or Alaska Native                  0.0000003276 ***
## derived_raceAsian                                             0.0000083080 ***
## derived_raceBlack or African American                         0.0000000177 ***
## derived_raceNative Hawaiian or Other Pacific Islander         0.0000053964 ***
## derived_raceWhite                                             0.0000000168 ***
## action_taken2                                                    0.11447
## action_taken3                                       < 0.0000000000000002 ***
## action_taken7                                                    0.29370
## action_taken8                                                    0.03378 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 26 on 14240 degrees of freedom
## Multiple R-squared:  0.05801,    Adjusted R-squared:  0.05675
## F-statistic: 46.16 on 19 and 14240 DF,  p-value: < 0.00000000000000022
```

```
# Calculate Variance Inflation Factor (VIF) to check for multicollinearity
vif(lm_model)
```

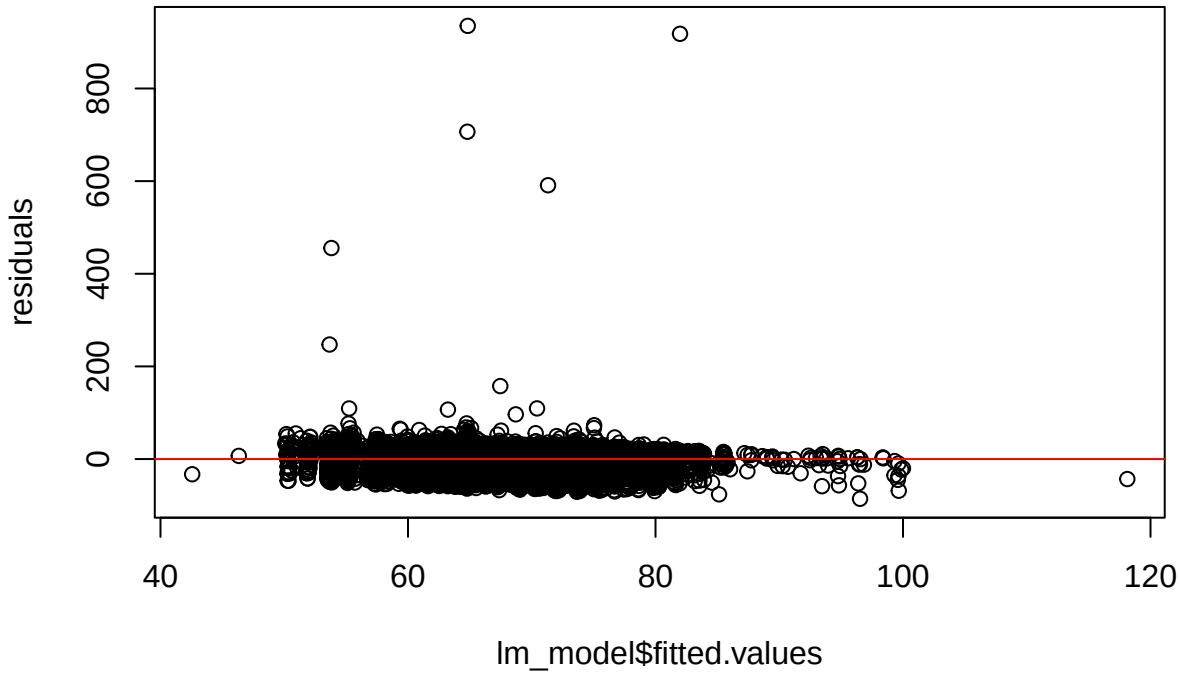
```
##                                GVIF Df GVIF^(1/(2*Df))
## income                     1.035959  1        1.017821
## debt_to_income_ratio_numeric 1.438257  1        1.199273
## applicant_credit_score_type  1.137682  8        1.008095
## derived_race                 1.134362  5        1.012687
## action_taken                 1.519355  4        1.053677
```

The model indicates that income has a small positive effect on loan-to-value ratio (Estimate = 0.00121, p = 0.00569), while debt-to-income ratio has a larger positive effect (Estimate = 0.2679, p < 2e-16). Several credit score types are statistically significant, such as type 8, which reduces the loan-to-value ratio by 10.24 (p < 2e-16), and type 7, which increases it by 3.63 (p = 0.00352). Race categories show substantial negative effects, with Native Hawaiian or Other Pacific Islander reducing the ratio by 27.41 (p = 5.40e-06) and Black or African American by 19.74 (p = 1.77e-08). The low R-squared value of 5.8% indicates the model explains only a small fraction of the variance in the loan-to-value ratio, while acceptable VIF values (all < 1.2) confirm no multicollinearity concerns.

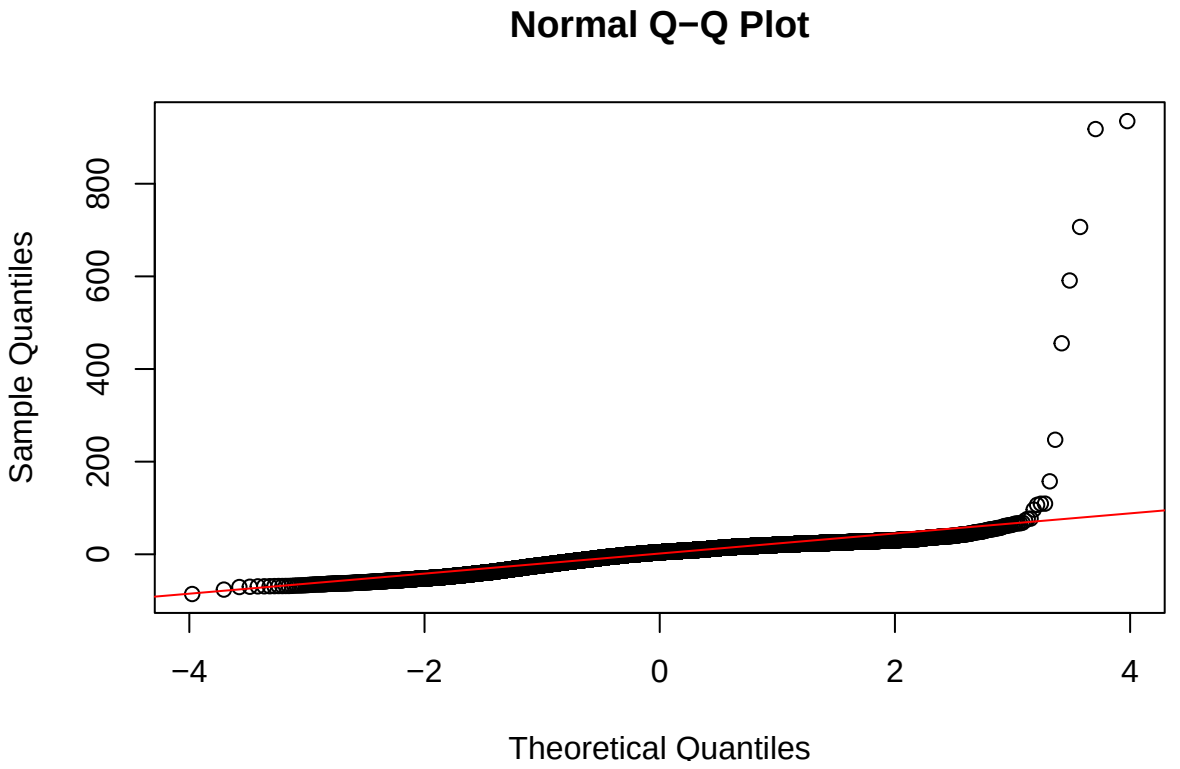
Check for Normality

```
residuals <- lm_model$residuals
# Residuals vs Fitted plot
```

```r
plot(lm_model$fitted.values, residuals)
abline(h = 0, col = "red")
```



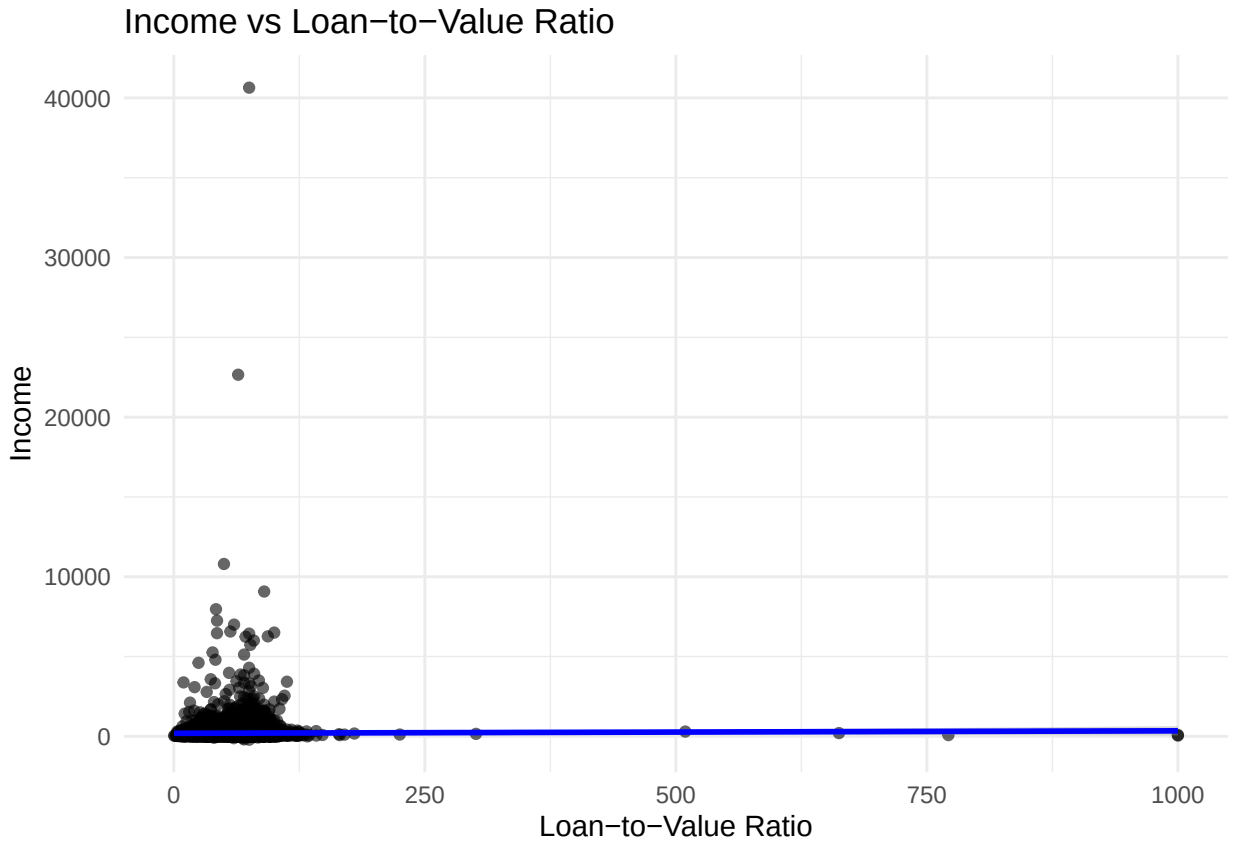

```r
# Normality of residuals
qqnorm(residuals)
qqline(residuals, col = "red")
```



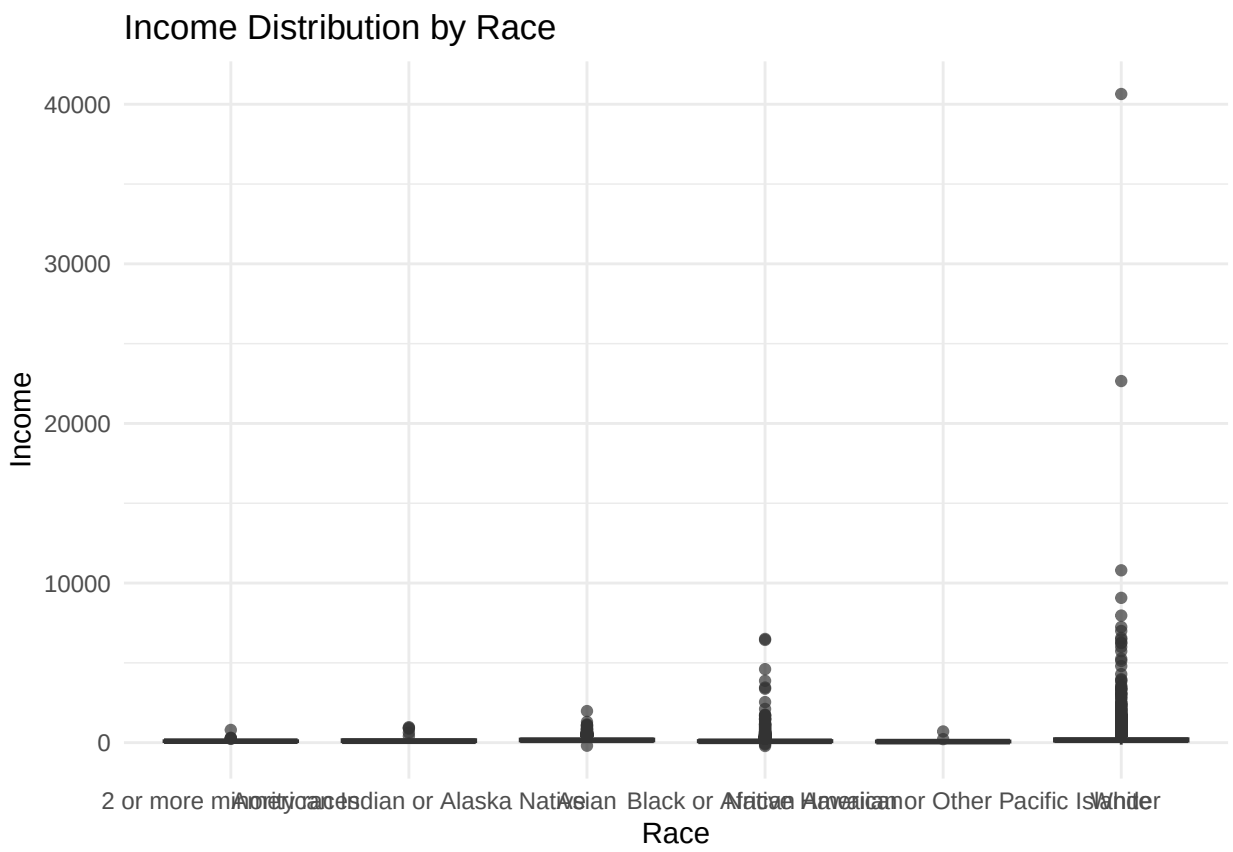
Linear Regression Visualizations

```
library(ggplot2)

# Scatter plot with regression line
ggplot(county_data_clean, aes(x = loan_to_value_ratio, y = income)) +
  geom_point(alpha = 0.6) +  # Scatter points
  geom_smooth(method = "lm", col = "blue") +  # Regression line
  labs(title = "Income vs Loan-to-Value Ratio",
       x = "Loan-to-Value Ratio",
       y = "Income") +
  theme_minimal()
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

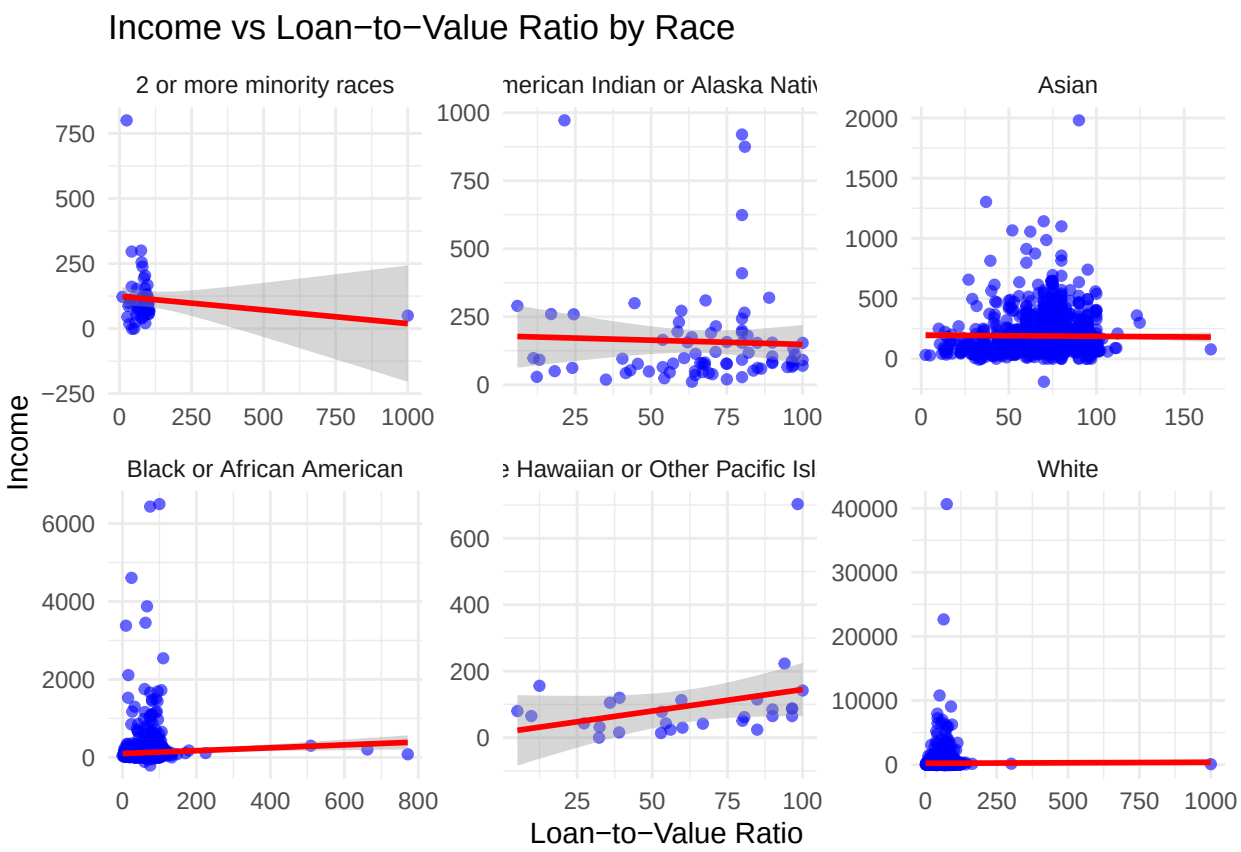



```
# Box plot to visualize the distribution of income by derived_race
ggplot(county_data_clean, aes(x = derived_race, y = income, fill = derived_race)) +
  geom_boxplot(alpha = 0.7) +  # Boxplot with transparency
  labs(title = "Income Distribution by Race",
       x = "Race",
       y = "Income") +
  theme_minimal() +
  theme(legend.position = "none") +
  scale_fill_brewer(palette = "Set3")
```




```
# Scatter plot faceted by race
ggplot(county_data_clean, aes(x = loan_to_value_ratio, y = income)) +
  geom_point(alpha = 0.6, color = "blue") +  # Scatter points
  geom_smooth(method = "lm", color = "red") +  # Regression line
  facet_wrap(~ derived_race, scales = "free") +  # Facet by race
  labs(
    title = "Income vs Loan-to-Value Ratio by Race",
    x = "Loan-to-Value Ratio",
    y = "Income"
  ) +
  theme_minimal()
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



Split the Data into Training and Testing Sets

```
set.seed(123)

# Split into training (70%) and testing (30%)
sample_index <- sample(seq_len(nrow(county_data_clean)), size = 0.7 * nrow(county_data_clean))
train_data <- county_data_clean[sample_index, ]
test_data <- county_data_clean[-sample_index, ]

# Check dimensions
dim(train_data)
```

```
## [1] 9982    6
```

```
dim(test_data)
```

```
## [1] 4278    6
```

Logistic Regression Test

```
lm_model <- lm(loan_to_value_ratio ~ income + debt_to_income_ratio_numeric +
                 applicant_credit_score_type + derived_race + action_taken,
               data = train_data)

# Predict on test data and evaluate
predictions <- predict(lm_model, newdata = test_data)
```

```r
# Calculate performance metrics
mse <- mean((predictions - test_data$loan_to_value_ratio)^2)
r_squared <- 1 - (sum((predictions - test_data$loan_to_value_ratio)^2) /
                  sum((test_data$loan_to_value_ratio - mean(test_data$loan_to_value_ratio))^2))

cat("MSE on test data:", mse, "\n")
```

```
## MSE on test data: 569.3976
```

```r
cat("R-squared on test data:", r_squared, "\n")
```

```
## R-squared on test data: 0.06410993
```

Ridge Regression Test

```r
library(glmnet)
```

```
## Loading required package: Matrix
```

```
##
## Attaching package: 'Matrix'
```

```
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
```

```
## Loaded glmnet 4.1-8
```

```r
# Create matrix of predictors and response variable
X_train <- model.matrix(loan_to_value_ratio ~ income + debt_to_income_ratio_numeric +
                          applicant_credit_score_type + derived_race + action_taken,
                        data = train_data)[, -1]
y_train <- train_data$loan_to_value_ratio

# Fit ridge regression with cross-validation
ridge_model <- cv.glmnet(X_train, y_train, alpha = 0)

# Optimal lambda
optimal_lambda <- ridge_model$lambda.min

# Coefficients at optimal lambda
ridge_coefficients <- coef(ridge_model, s = optimal_lambda)

# Print optimal lambda and coefficients
print(paste("Optimal Lambda: ", optimal_lambda))
```

```
## [1] "Optimal Lambda:  0.548947034784625"
```

```r
print(ridge_coefficients)
```

```
## 20 x 1 sparse Matrix of class "dgCMatrix"
##                                                          s1
## (Intercept)                                  72.8639569494
## income                                        0.0009955725
## debt_to_income_ratio_numeric                  0.2701788535
## applicant_credit_score_type2                  1.8774433124
## applicant_credit_score_type3                 -1.3901981116
```

```
## applicant_credit_score_type4                              11.1579969048
## applicant_credit_score_type6                              -6.5033987347
## applicant_credit_score_type7                               3.1534610354
## applicant_credit_score_type8                              -9.6606806931
## applicant_credit_score_type9                              -2.7787207085
## applicant_credit_score_type11                            -13.0291829345
## derived_raceAmerican Indian or Alaska Native              -9.7050465474
## derived_raceAsian                                         -3.9827770883
## derived_raceBlack or African American                     -7.9895694711
## derived_raceNative Hawaiian or Other Pacific Islander -14.5994494994
## derived_raceWhite                                         -7.6778115155
## action_taken2                                             -1.5756735098
## action_taken3                                             -6.3215168269
## action_taken7                                             10.8874013832
## action_taken8                                             15.4258709476
```

```
# Evaluate on test data
X_test <- model.matrix(loan_to_value_ratio ~ income + debt_to_income_ratio_numeric +
                         applicant_credit_score_type + derived_race + action_taken,
                       data = test_data)[, -1]
ridge_predictions <- predict(ridge_model, newx = X_test, s = optimal_lambda)
ridge_test_mse <- mean((test_data$loan_to_value_ratio - ridge_predictions)^2)
ridge_test_mse
```

```
## [1] 567.2943
```

```
summary(test_data$loan_to_value_ratio)
```

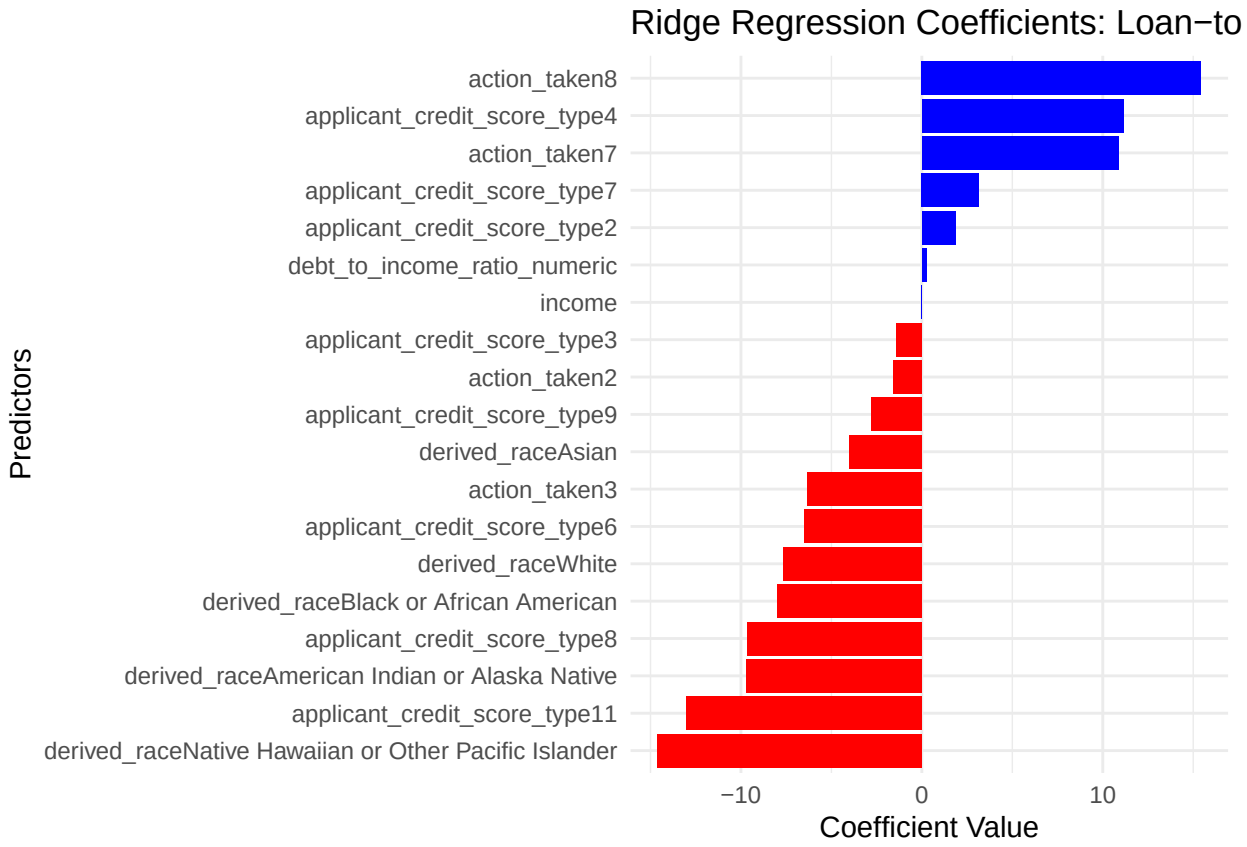
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.96   57.65   75.00   70.49   87.21  662.43
```

The optimal lambda for the ridge regression model is 0.549, which helps to shrink the coefficients and reduce overfitting. The model reveals that the intercept is significant with a value of 72.86, while variables like income (0.000995), debt-to-income ratio (0.270), and various credit score types exhibit notable coefficients, indicating their importance in predicting the loan-to-value ratio. For instance, applicant credit score types 4 and 11 show strong positive and negative relationships with the outcome, with values of 11.16 and -13.03, respectively, while some types, such as credit score type 8, show a negative relationship of -9.66. Race categories, particularly "American Indian or Alaska Native" (-9.71) and "Native Hawaiian or Other Pacific Islander" (-14.60), have substantial negative coefficients, suggesting a significant effect on the loan-to-value ratio. The action taken variables, particularly types 7 and 8, show positive associations with the outcome, with coefficients of 10.89 and 15.43, respectively, indicating the model's sensitivity to different actions taken in the loan process.

Ridge Regression Visualization

```
# Extract ridge regression coefficients at optimal lambda
ridge_coefficients <- coef(ridge_model, s = "lambda.min")
ridge_coefficients <- as.data.frame(as.matrix(ridge_coefficients))
ridge_coefficients$Variable <- rownames(ridge_coefficients)

# Rename columns for clarity
colnames(ridge_coefficients) <- c("Coefficient", "Variable")

# Remove intercept for clarity
ridge_coefficients <- ridge_coefficients[ridge_coefficients$Variable != "(Intercept)", ]
```

```r
# Remove rows with missing values
ridge_coefficients <- ridge_coefficients[!is.na(ridge_coefficients$Coefficient), ]

# Remove irrelevant levels in 'derived_race'
ridge_coefficients <- ridge_coefficients[!grepl("derived_raceFree Form Text Only|derived_raceRace Not A

# Plot ridge regression coefficients
library(ggplot2)
ggplot(ridge_coefficients, aes(x = reorder(Variable, Coefficient), y = Coefficient)) +
  geom_bar(stat = "identity", fill = ifelse(ridge_coefficients$Coefficient > 0, "blue", "red")) +
  coord_flip() +  # Flip the coordinates for better readability
  labs(
    title = "Ridge Regression Coefficients: Loan-to-Value Ratio",
    x = "Predictors",
    y = "Coefficient Value"
  ) +
  theme_minimal()
```



PLS Regression

```r
library(pls)
```

```
##
## Attaching package: 'pls'
```

```
## The following object is masked from 'package:corrplot':
##
```

```
##     corrplot
```

```
## The following object is masked from 'package:stats':
##
##     loadings
```

```
# Fit PLS model with cross-validation
pls_model <- plsr(loan_to_value_ratio ~ income + debt_to_income_ratio_numeric +
                    applicant_credit_score_type + derived_race + action_taken,
                  data = train_data, validation = "CV")

# Print the summary of the PLS model
summary(pls_model)
```

```
## Data:    X dimension: 9982 19
##  Y dimension: 9982 1
## Fit method: kernelpls
## Number of components considered: 19
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##        (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## CV           27.63    27.63    27.57    27.05    26.99    26.97    26.96
## adjCV        27.63    27.63    27.57    27.05    26.99    26.97    26.96
##        7 comps  8 comps  9 comps  10 comps  11 comps  12 comps  13 comps
## CV       26.96    26.95    26.96     26.98     26.99     26.99     26.98
## adjCV    26.96    26.95    26.96     26.98     26.98     26.98     26.98
##        14 comps  15 comps  16 comps  17 comps  18 comps  19 comps
## CV        26.98     26.98     26.98     26.98     26.98     26.98
## adjCV     26.97     26.97     26.97     26.97     26.97     26.97
##
## TRAINING: % variance explained
##                         1 comps   2 comps  3 comps  4 comps  5 comps  6 comps
## X                     99.916541   99.9995  100.000  100.000   100.00  100.000
## loan_to_value_ratio    0.003157    0.5128    4.369    4.837     5.08    5.105
##                       7 comps  8 comps  9 comps  10 comps  11 comps  12 comps
## X                     100.000  100.000  100.000   100.000   100.000    100.00
## loan_to_value_ratio     5.124    5.159    5.194     5.329     5.443      5.47
##                       13 comps  14 comps  15 comps  16 comps  17 comps  18 comps
## X                      100.000   100.000   100.000   100.000   100.000   100.000
## loan_to_value_ratio      5.489     5.503     5.514     5.515     5.515     5.515
##                       19 comps
## X                      100.000
## loan_to_value_ratio      5.515
```

```
# Optimal number of components
optimal_components <- which.min(pls_model$validation$PRESS)

# Print optimal number of components
print(paste("Optimal Number of Components: ", optimal_components))
```

```
## [1] "Optimal Number of Components:  8"
```

```
# Predict on test data
pls_predictions <- predict(pls_model, newdata = test_data, ncomp = optimal_components)

# Extract predicted values (pls_predictions is usually a matrix)
```

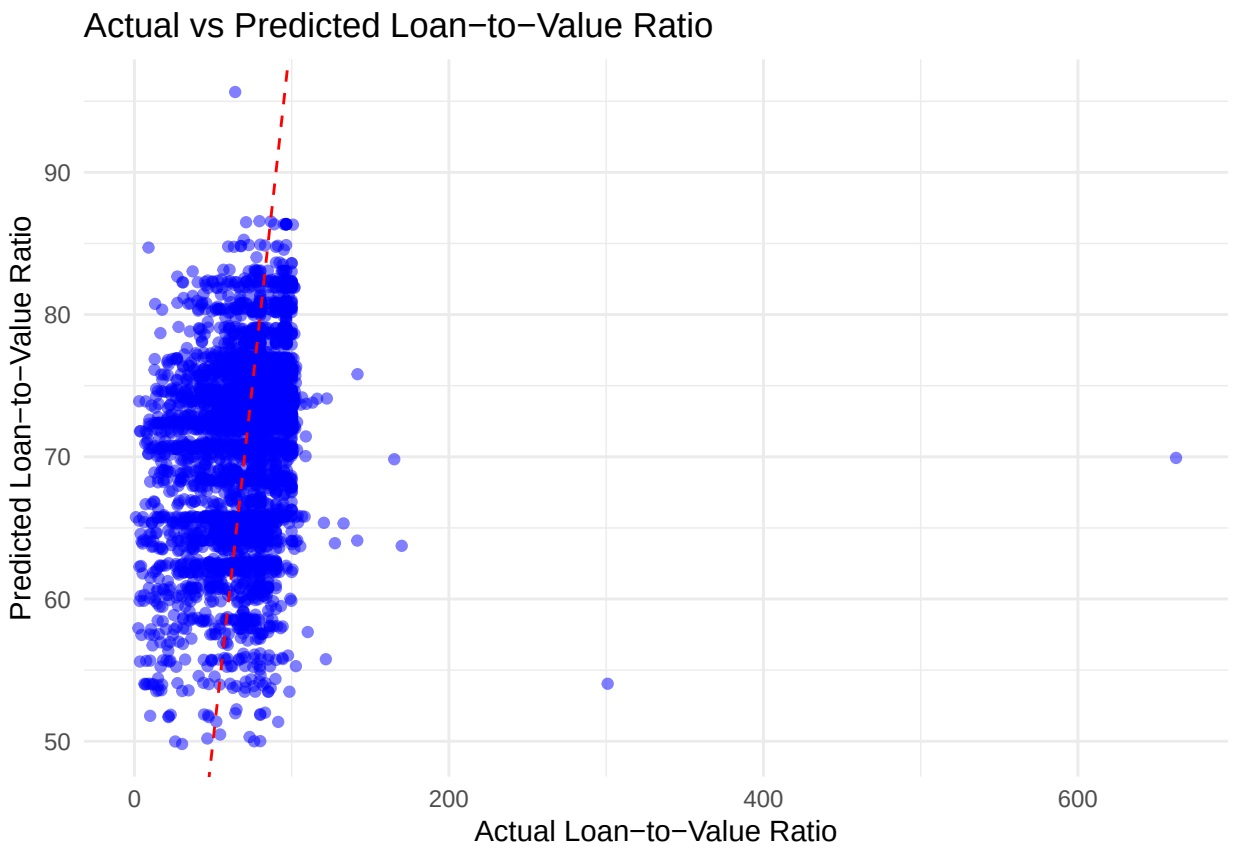
```r
pls_predictions_vector <- as.vector(pls_predictions)

# Compute the Mean Squared Error of the Test data
pls_test_mse <- mean((test_data$loan_to_value_ratio - pls_predictions_vector)^2)
pls_test_mse
```

```
## [1] 568.0606
```

The PLS regression results show that the optimal number of components is 8, as identified by minimizing the RMSEP value during cross-validation. The RMSEP values for the validation set decrease slightly as more components are added, with the lowest value of 26.96 occurring at 8 components, indicating that using additional components beyond 8 does not improve the model significantly. On the training set, the X matrix (predictor variables) explains nearly all the variance (close to 100%) across all components, while the variance explained by the dependent variable, loan_to_value_ratio, starts at 0.003157% for the first component and increases gradually with more components, reaching about 5.52% by the 19th component. The model's performance remains relatively stable after 8 components, with minimal improvements in RMSEP or variance explained, suggesting that 8 components are optimal for predicting the loan-to-value ratio. The test MSE (Mean Squared Error) for the model's predictions is 568.06, which provides an estimate of the prediction error for the chosen model.

PLS Visualization

```r
library(ggplot2)

# Create a dataframe for Actual vs Predicted values
results_df <- data.frame(
  Actual = test_data$loan_to_value_ratio,
  Predicted = pls_predictions_vector
)

# Plot the actual vs predicted values
library(ggplot2)
ggplot(results_df, aes(x = Actual, y = Predicted)) +
  geom_point(color = "blue", alpha = 0.5) +
  geom_abline(slope = 1, intercept = 0, color = "red", linetype = "dashed") +
  labs(
    title = "Actual vs Predicted Loan-to-Value Ratio",
    x = "Actual Loan-to-Value Ratio",
    y = "Predicted Loan-to-Value Ratio"
  ) +
  theme_minimal()
```



Compare the Models' Performance

Calculate MSE and R-squared for Each Model

```
# Linear Model MSE and R-squared
lm_predictions <- predict(lm_model, newdata = test_data)

lm_mse <- mean((lm_predictions - test_data$loan_to_value_ratio)^2)
lm_r2 <- 1 - sum((lm_predictions - test_data$loan_to_value_ratio)^2) /
         sum((test_data$loan_to_value_ratio - mean(test_data$loan_to_value_ratio))^2)

# Ridge Model MSE and R-squared
ridge_mse <- mean((ridge_predictions - test_data$loan_to_value_ratio)^2)
ridge_r2 <- 1 - sum((ridge_predictions - test_data$loan_to_value_ratio)^2) /
            sum((test_data$loan_to_value_ratio - mean(test_data$loan_to_value_ratio))^2)

# PLS Model MSE and R-squared
pls_mse <- mean((pls_predictions - test_data$loan_to_value_ratio)^2)
pls_r2 <- 1 - sum((pls_predictions - test_data$loan_to_value_ratio)^2) /
          sum((test_data$loan_to_value_ratio - mean(test_data$loan_to_value_ratio))^2)

# Output the results
cat("Linear Model MSE: ", lm_mse, "\n")
```

```
## Linear Model MSE:  569.3976
```

```r
cat("Ridge Model MSE: ", ridge_mse, "\n")
```

```
## Ridge Model MSE:  567.2943
```

```r
cat("PLS Model MSE: ", pls_mse, "\n")
```

```
## PLS Model MSE:  568.0606
```

```r
cat("Linear Model R2: ", lm_r2, "\n")
```

```
## Linear Model R2:  0.06410993
```

```r
cat("Ridge Model R2: ", ridge_r2, "\n")
```

```
## Ridge Model R2:  0.06756697
```

```r
cat("PLS Model R2: ", pls_r2, "\n")
```

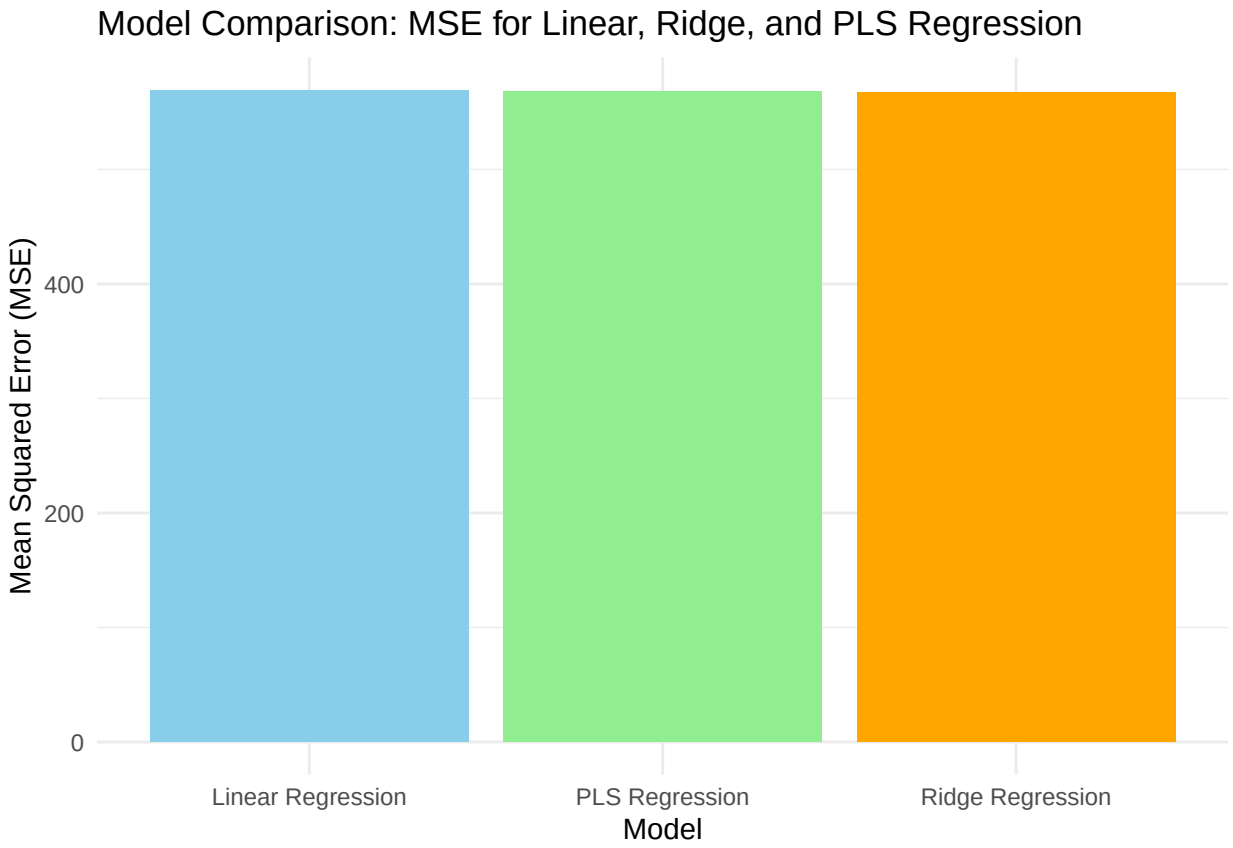
```
## PLS Model R2:  0.06630739
```

MSE Model Comparison

```r
# Store MSE for each model
model_comparison <- data.frame(
  Model = c("Linear Regression", "Ridge Regression", "PLS Regression"),
  MSE = c(lm_mse, ridge_mse, pls_mse)
)

# Print comparison
print(model_comparison)
```

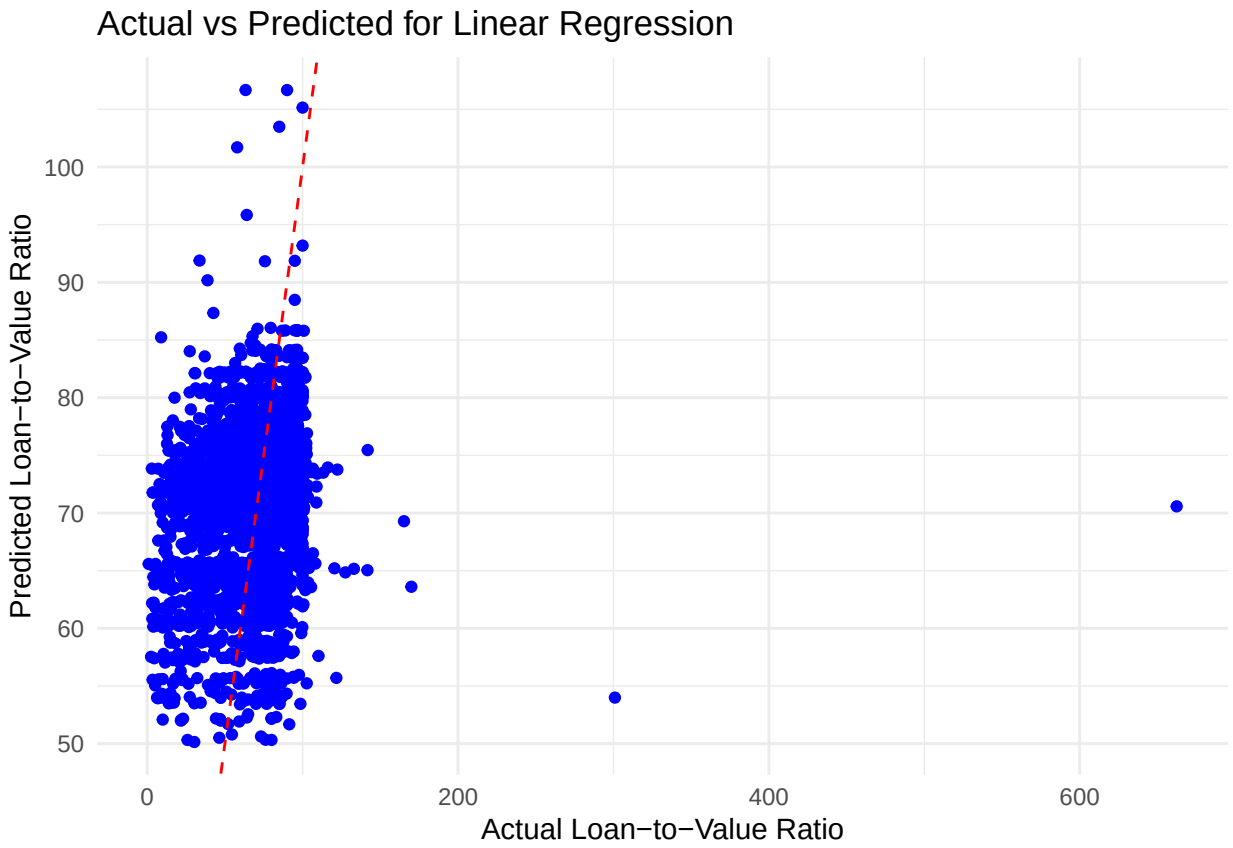
```
##               Model      MSE
## 1 Linear Regression 569.3976
## 2  Ridge Regression 567.2943
## 3    PLS Regression 568.0606
```

```r
# Bar plot for MSE comparison
ggplot(model_comparison, aes(x = Model, y = MSE, fill = Model)) +
  geom_bar(stat = "identity", show.legend = FALSE) +
  labs(title = "Model Comparison: MSE for Linear, Ridge, and PLS Regression",
       x = "Model",
       y = "Mean Squared Error (MSE)") +
  theme_minimal() +
  scale_fill_manual(values = c("skyblue", "lightgreen", "orange"))
```

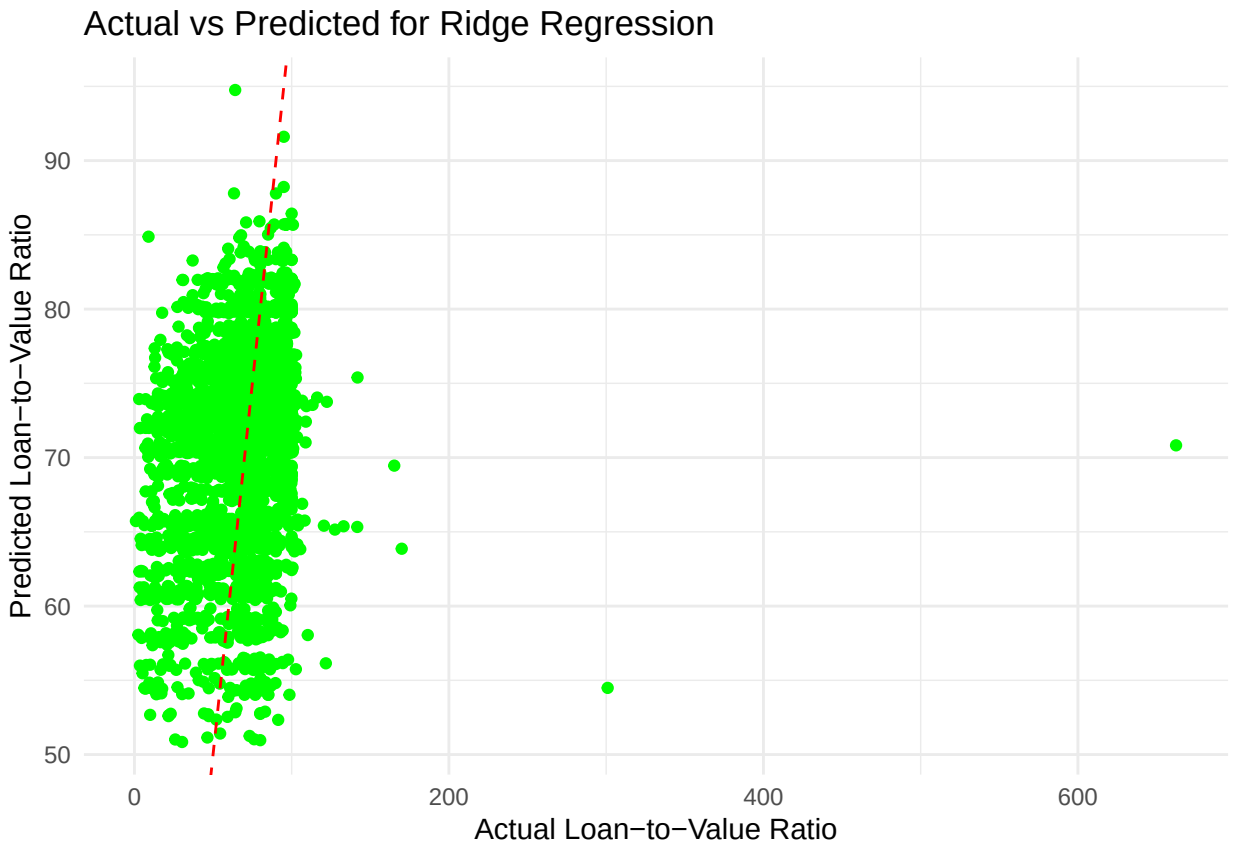


Actual vs Predicted for all Models

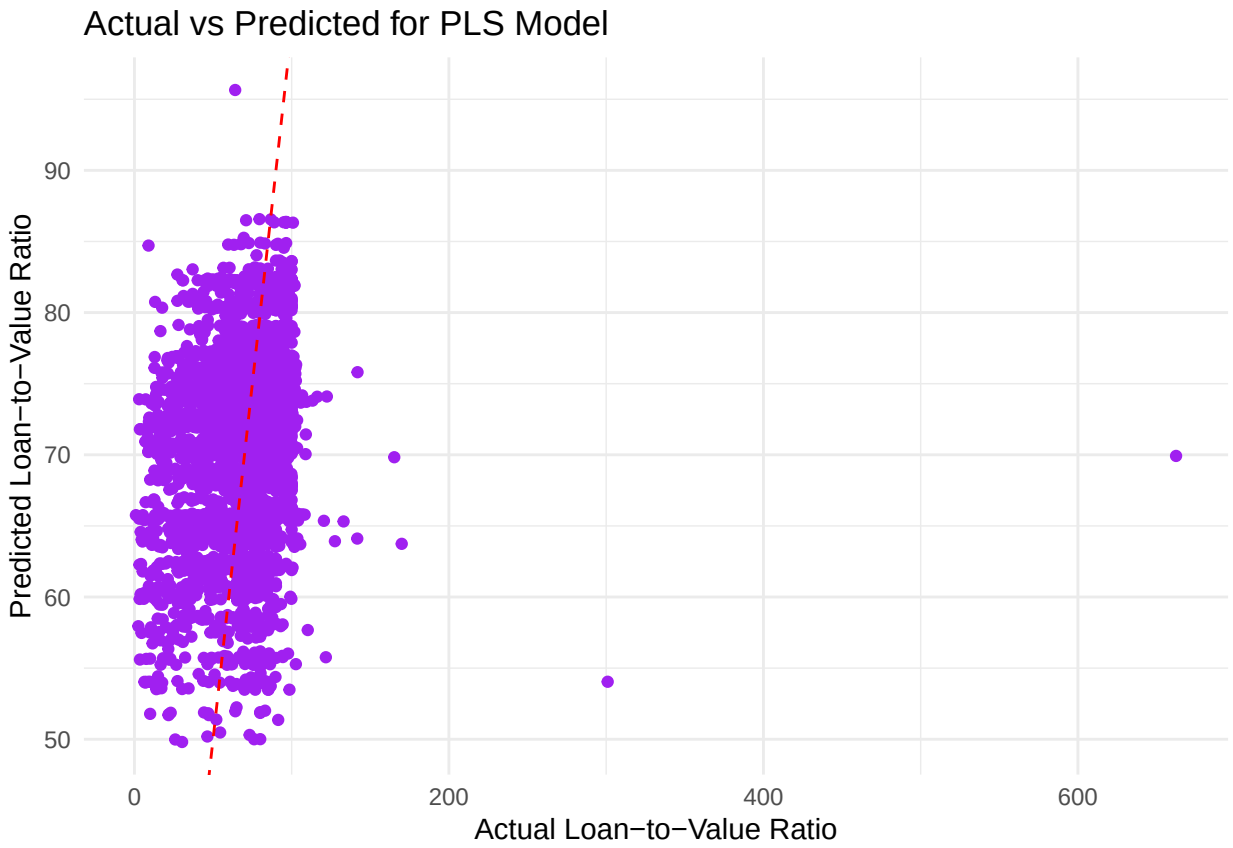
```
# Actual vs Predicted for Linear Model
ggplot() +
  geom_point(aes(x = test_data$loan_to_value_ratio, y = lm_predictions), color = "blue") +
  geom_abline(intercept = 0, slope = 1, color = "red", linetype = "dashed") +
  labs(title = "Actual vs Predicted for Linear Regression",
       x = "Actual Loan-to-Value Ratio",
       y = "Predicted Loan-to-Value Ratio") +
  theme_minimal()
```




```
# Actual vs Predicted for Ridge Model
ggplot() +
  geom_point(aes(x = test_data$loan_to_value_ratio, y = ridge_predictions), color = "green") +
  geom_abline(intercept = 0, slope = 1, color = "red", linetype = "dashed") +
  labs(title = "Actual vs Predicted for Ridge Regression",
       x = "Actual Loan-to-Value Ratio",
       y = "Predicted Loan-to-Value Ratio") +
  theme_minimal()
```




```
# Actual vs Predicted for PLS Model
ggplot() +
  geom_point(aes(x = test_data$loan_to_value_ratio, y = pls_predictions), color = "purple") +
  geom_abline(intercept = 0, slope = 1, color = "red", linetype = "dashed") +
  labs(title = "Actual vs Predicted for PLS Model",
       x = "Actual Loan-to-Value Ratio",
       y = "Predicted Loan-to-Value Ratio") +
  theme_minimal()
```



Perform Cross-Validation

```
library(caret)
```

```
## Loading required package: lattice
```

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

```
## The following object is masked from 'package:pls':
##
##     R2
```

```
library(glmnet)
library(pls)

# Set up K-Fold Cross Validation (10 folds)
train_control <- trainControl(method = "cv", number = 10)

# 1. Linear Regression Model with Cross-Validation
lm_model_cv <- train(loan_to_value_ratio ~ income + debt_to_income_ratio_numeric +
                       applicant_credit_score_type + derived_race + action_taken,
                       data = county_data_clean,
                       method = "lm",
                       trControl = train_control)
```

```
## Warning in predict.lm(modelFit, newdata): prediction from rank-deficient fit;
## attr(*, "non-estim") has doubtful cases
```

```
# 2. Ridge Regression Model with Cross-Validation
ridge_model_cv <- train(loan_to_value_ratio ~ income + debt_to_income_ratio_numeric +
```

```
                         applicant_credit_score_type + derived_race + action_taken,
                         data = county_data_clean,
                         method = "glmnet",
                         trControl = train_control,
                         tuneGrid = expand.grid(alpha = 0, lambda = seq(0.001, 0.1, length = 10)))  # R

# 3. PLS Model with Cross-Validation
pls_model_cv <- train(loan_to_value_ratio ~ income + debt_to_income_ratio_numeric +
                      applicant_credit_score_type + derived_race + action_taken,
                      data = county_data_clean,
                      method = "pls",
                      trControl = train_control,
                      tuneGrid = expand.grid(ncomp = 1:10))

# Output the results for each model
print("Linear Model Results:")
```

```
## [1] "Linear Model Results:"
```

```
print(lm_model_cv)
```

```
## Linear Regression
##
## 14260 samples
##     5 predictor
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 12835, 12832, 12835, 12836, 12833, 12834, ...
## Resampling results:
##
##   RMSE      Rsquared    MAE
##   25.50347  0.06022987  17.6243
##
## Tuning parameter 'intercept' was held constant at a value of TRUE
```

```
print("Ridge Model Results:")
```

```
## [1] "Ridge Model Results:"
```

```
print(ridge_model_cv)
```

```
## glmnet
##
## 14260 samples
##     5 predictor
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 12835, 12835, 12834, 12835, 12832, 12834, ...
## Resampling results across tuning parameters:
##
##   lambda  RMSE      Rsquared    MAE
##   0.001   25.26961  0.06679763  17.60534
##   0.012   25.26961  0.06679763  17.60534
##   0.023   25.26961  0.06679763  17.60534
```

```
##   0.034   25.26961  0.06679763  17.60534
##   0.045   25.26961  0.06679763  17.60534
##   0.056   25.26961  0.06679763  17.60534
##   0.067   25.26961  0.06679763  17.60534
##   0.078   25.26961  0.06679763  17.60534
##   0.089   25.26961  0.06679763  17.60534
##   0.100   25.26961  0.06679763  17.60534
##
## Tuning parameter 'alpha' was held constant at a value of 0
## RMSE was used to select the optimal model using the smallest value.
## The final values used for the model were alpha = 0 and lambda = 0.1.
```

```
print("PLS Model Results:")
```

```
## [1] "PLS Model Results:"
```

```
print(pls_model_cv)
```

```
## Partial Least Squares
##
## 14260 samples
##     5 predictor
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 12833, 12835, 12833, 12834, 12833, 12836, ...
## Resampling results across tuning parameters:
##
##   ncomp  RMSE      Rsquared      MAE
##    1     26.37132  0.0008087445  18.43978
##    2     26.31281  0.0073362301  18.44283
##    3     25.71077  0.0544948744  17.70074
##    4     25.63130  0.0605296606  17.64765
##    5     25.61902  0.0614921256  17.65003
##    6     25.60654  0.0624153886  17.62563
##    7     25.60209  0.0628135483  17.62432
##    8     25.59960  0.0630427219  17.61795
##    9     25.60475  0.0626112022  17.61668
##   10     25.62040  0.0613753096  17.62172
##
## RMSE was used to select the optimal model using the smallest value.
## The final value used for the model was ncomp = 8.
```

```
# Compare the performance metrics: RMSE, R-squared, etc.
model_comparison <- data.frame(
  Model = c("Linear Regression", "Ridge Regression", "PLS Regression"),
  RMSE = c(min(lm_model_cv$results$RMSE), min(ridge_model_cv$results$RMSE), min(pls_model_cv$results$RMS
  Rsquared = c(max(lm_model_cv$results$Rsquared), max(ridge_model_cv$results$Rsquared), max(pls_model_cv
)

# Print Model Comparison
print(model_comparison)
```

```
##               Model     RMSE   Rsquared
## 1 Linear Regression 25.50347 0.06022987
## 2  Ridge Regression 25.26961 0.06679763
```

```
## 3    PLS Regression 25.59960 0.06304272
```

```r
# Visualize Model Comparison (Bar plot for RMSE)
library(ggplot2)
ggplot(model_comparison, aes(x = Model, y = RMSE, fill = Model)) +
  geom_bar(stat = "identity", show.legend = FALSE) +
  labs(title = "Model Comparison: RMSE for Linear, Ridge, and PLS Regression",
       x = "Model",
       y = "Root Mean Squared Error (RMSE)") +
  theme_minimal() +
  scale_fill_manual(values = c("skyblue", "lightgreen", "orange"))
```