# Final_exam

2023-05-28

#Part1

```
#Load the required packages
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.4.1     v purrr   1.0.1
## v tibble  3.2.1     v dplyr   1.1.0
## v tidyr   1.3.0     v stringr 1.5.0
## v readr   2.1.3     v forcats 1.0.0
```

```
## Warning: package 'tibble' was built under R version 4.2.3
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(sf)
```

```
## Warning: package 'sf' was built under R version 4.2.3
```

```
## Linking to GEOS 3.9.3, GDAL 3.5.2, PROJ 8.2.1; sf_use_s2() is TRUE
```

```
#Read the dataset for part 1
hate_crimes <- read.csv("data/hate_crimes.csv")
us_states <- st_read("data/States_shapefile.shp", quiet = TRUE)
us_states %>%
  slice(1:6)
```

```
## Simple feature collection with 6 features and 6 fields
## Geometry type: MULTIPOLYGON
## Dimension:     XY
## Bounding box:  xmin: -178.2176 ymin: 30.2336 xmax: -84.89402 ymax: 71.40624
## Geodetic CRS:  WGS 84
##   FID         Program State_Code State_Name Flowing_St FID_1
## 1   1 PERMIT TRACKING         AL    ALABAMA          F   919
## 2   2            <NA>         AK     ALASKA          N   920
## 3   3         AZURITE         AZ    ARIZONA          F   921
## 4   4             PDS         AR   ARKANSAS          F   922
## 5   5            <NA>         CA CALIFORNIA          N   923
## 6   6          ECOMAP         CO   COLORADO          F   924
```

```
##                         geometry
## 1 MULTIPOLYGON (((-85.07007 3...
## 2 MULTIPOLYGON (((-161.3338 5...
## 3 MULTIPOLYGON (((-114.5206 3...
## 4 MULTIPOLYGON (((-94.46169 3...
## 5 MULTIPOLYGON (((-121.6652 3...
## 6 MULTIPOLYGON (((-102.0445 3...
```

Join both data frames that you have (hate_crimes and us_states) by state name. In order to complete this task, you need to do the following: 1- Rename the State_Name variable in us_states dataset to state using rename() function. 2- Change the states' names in both datasets to lower case using mutate() and tolower() functions. 3- Use left_join() function to merge both data sets into a new data tibble called state_crimes

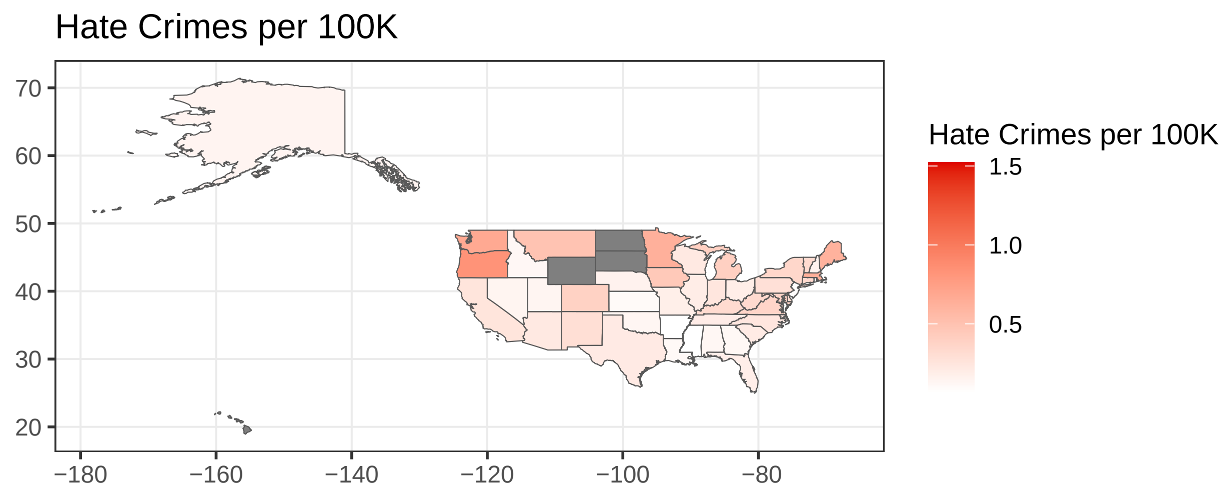
##task1

```
us_states <- us_states %>%
        rename("state" = "State_Name")


us_states <- us_states %>%
        mutate(state = tolower(state))

hate_crimes <- hate_crimes %>%
        mutate(state = tolower(state))

hate_crime_us_state <- left_join(hate_crimes,us_states,by="state")
```

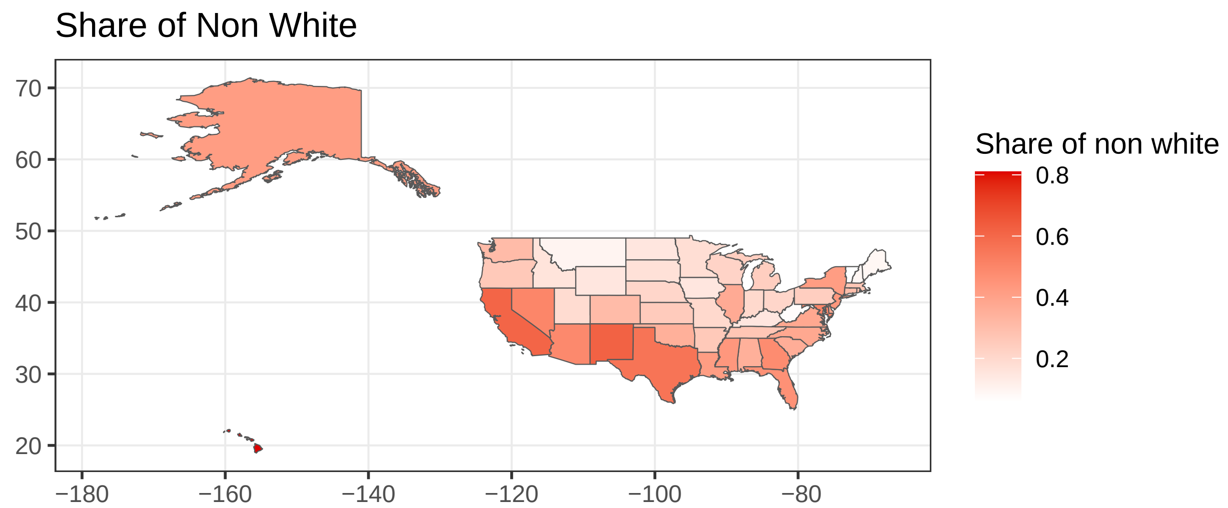
##task2

Usin ggplot() and geom_sf(), draw the US states maps that shows the following: 1- A map that shows US stats colored by hate_crimes_per_100k_splc variable. 2- A map that shows US stats colored by share_non_white variable. Do you see any relation between both maps? Stae that have a low rate of hate crimes per 100K have a high share of non white citezen.

```
ggplot(hate_crime_us_state) +
geom_sf(aes(fill = hate_crimes_per_100k_splc, geometry = geometry)) +
scale_fill_gradient(low = "white", high = "#DE0100")+
labs(title = "Hate Crimes per 100K",
     fill = "Hate Crimes per 100K") +
theme_bw()
```




```
ggplot(hate_crime_us_state) +
geom_sf(aes(fill = share_non_white, geometry = geometry))+
scale_fill_gradient(low = "white", high = "#DE0100")+
labs(title = "Share of Non White",
    fill = "Share of non white") +
theme_bw()
```



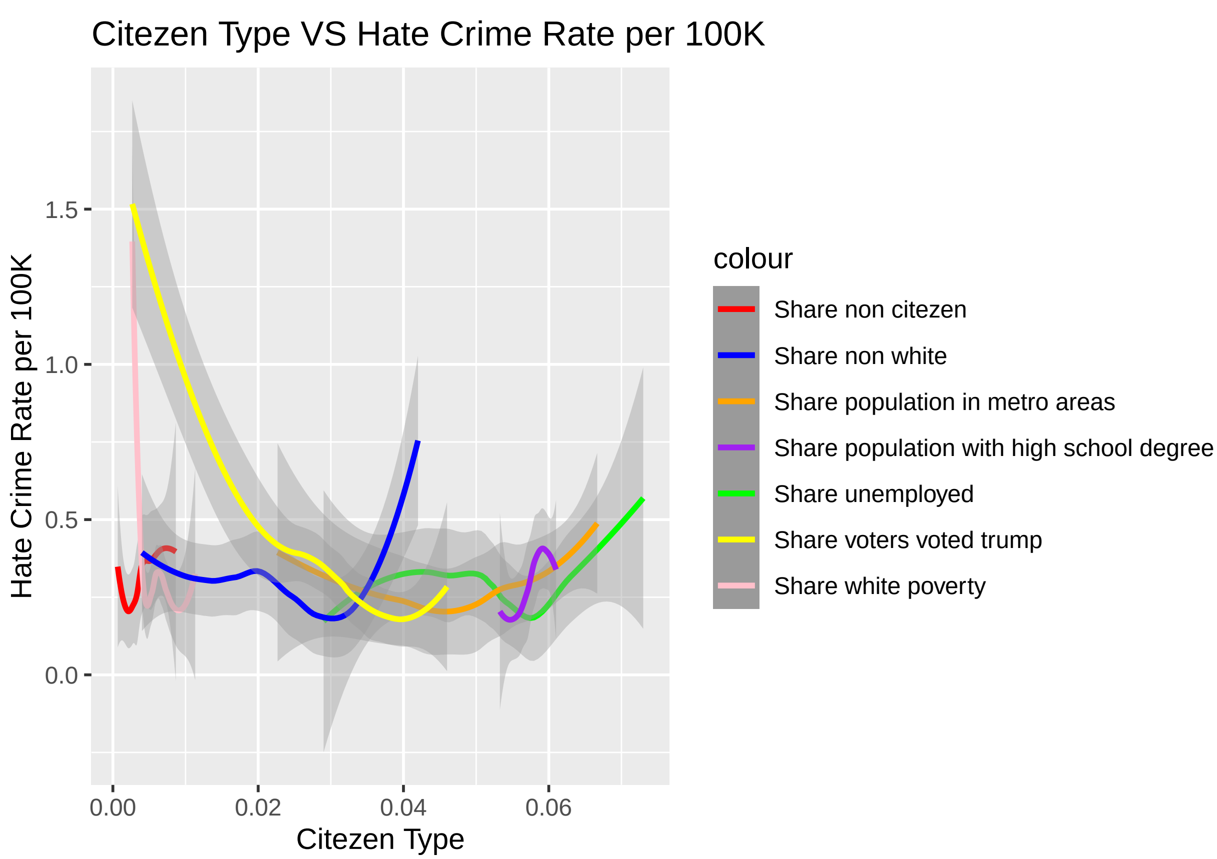

Come up with a research question based on these data and write it down. Then, create an effective data visualization that answers the question and write a brief paragraph explaining how your visualization answers the question.

What is the relation between hate crimes and citizen status? I created a plot that visulize the distrbution of hate crimes by citezen type. The citezen who have the highest rate crimes are the non-white, trump voters, and white citezen who suffer from poverty ##task3

```
scale = 15

ggplot(hate_crime_us_state, aes(x = share_unemployed_seasonal , y = hate_crimes_per_100k_splc)) +
  geom_smooth(aes(color = "Share unemployed")) +
  geom_smooth(aes(x = share_population_in_metro_areas/scale, color = "Share population in metro areas"))
    geom_smooth(aes(x = share_population_with_high_school_degree/scale, color = "Share population with h
      geom_smooth(aes(x = share_non_citizen/scale, color = "Share non citezen")) +
      geom_smooth(aes(x = share_white_poverty/scale, color = "Share white poverty")) +
  geom_smooth(aes(x = share_non_white/scale, color = "Share non white")) +
   geom_smooth(aes(x = share_voters_voted_trump/scale, color = "Share voters voted trump")) +
  labs(title = "Citezen Type VS Hate Crime Rate per 100K",
       x = "Citezen Type", y = "Hate Crime Rate per 100K") +
  scale_color_manual(values = c("red", "blue", "orange", "purple", "green", "yellow", "pink"))
```

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

## Warning: Removed 4 rows containing non-finite values (`stat_smooth()`).

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

```
## Warning: Removed 4 rows containing non-finite values (`stat_smooth()`).

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

## Warning: Removed 4 rows containing non-finite values (`stat_smooth()`).

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

## Warning: Removed 6 rows containing non-finite values (`stat_smooth()`).

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

## Warning: Removed 4 rows containing non-finite values (`stat_smooth()`).

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

## Warning: Removed 4 rows containing non-finite values (`stat_smooth()`).

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

## Warning: Removed 4 rows containing non-finite values (`stat_smooth()`).
```




#Part2

```r
#Load the required packages for part 2
library(caret)
```

```
## Warning: package 'caret' was built under R version 4.2.3
```

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

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

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

```r
library(skimr)
```

```
## Warning: package 'skimr' was built under R version 4.2.3
```

```r
#Read the dataset for part 2
bikeshare_day <- read.csv("data/bikeshare-day.csv")
```

Recode the season variable to be a factor with meaningful level names as outlined in the codebook, with spring as the baseline level. 1:winter, 2:spring, 3:summer, 4:fall ##task1

```r
bikeshare_day <- bikeshare_day %>%
  mutate(season = case_when(season == 1 ~ "winter",
                            season == 2 ~ "spring",
                            season == 3 ~ "summer",
                            season == 4 ~ "fall"))
```

##task2

Calculate raw temperature, feeling temperature, humidity, and windspeed as their values given in the dataset multiplied by the maximum raw values stated in the codebook for each variable. Instead of writing over the existing variables, create new ones with concise but informative names.

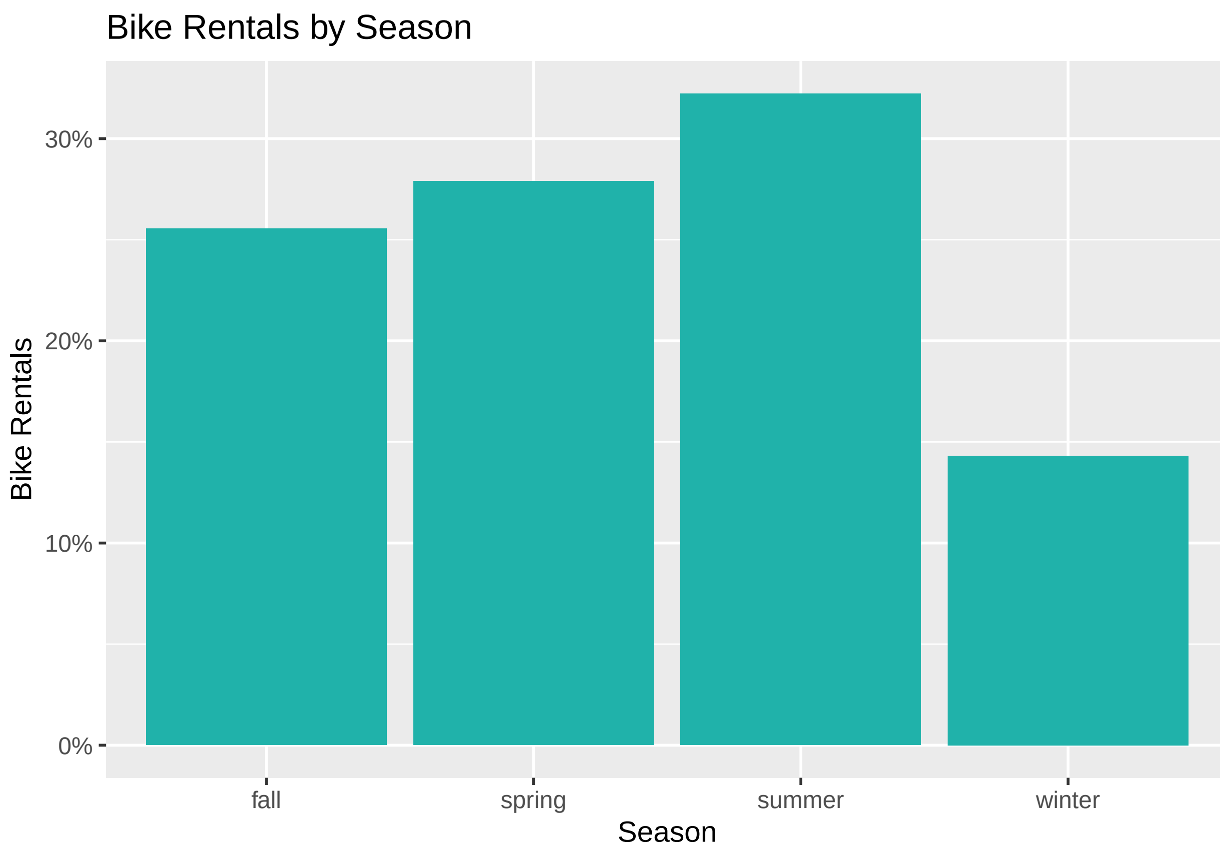
```r
bikeshare_day <- bikeshare_day %>%
  mutate(raw_temp_actual = temp * 41,
         feel_temp_actual = atemp * 50,
         humidity_actual = hum * 100,
         windspeed_actual = windspeed * 67)
```

Create a visualization displaying the relationship between bike rentals and season. Interpret the plot in context of the data.

Mjrity of bike rents happen during summer with 32% rate while winter has the lowest rate of bike rents at arround 14%. Spring and fall have an average renting rate arround 25%.

##task3

```r
ggplot(bikeshare_day, aes(x = season, y = cnt/sum(cnt))) +
  geom_bar(stat = "identity", fill = "#20B2AA") +
  scale_y_continuous(labels = scales::percent) +
  labs(x = "Season", y = "Bike Rentals", title = "Bike Rentals by Season")
```




##task4

```r
# Create the knn imputation model on the training data
preProcess_missingdata_model <- preProcess(bikeshare_day, method='knnImpute')
preProcess_missingdata_model
```

```
## Created from 731 samples and 20 variables
##
## Pre-processing:
##   - centered (18)
##   - ignored (2)
##   - 5 nearest neighbor imputation (18)
##   - scaled (18)
```

```r
# Use the imputation model to predict the values of missing data points
library(RANN)  # required for knnImpute
```

```
## Warning: package 'RANN' was built under R version 4.2.3
```

```r
bikeshare_day_impute <- predict(preProcess_missingdata_model, newdata = bikeshare_day)
anyNA(bikeshare_day_impute)
```

```
## [1] FALSE
```

```r
# Store X and Y for later use.
X = bikeshare_day_impute[, 3:13]
y = bikeshare_day_impute$cnt
```

```r
# Split the data into training and testing sets
set.seed(42)
train_indices <- createDataPartition(y, p = 0.8, list = FALSE)
X_train <- X[train_indices,]
y_train <- y[train_indices]
X_test <- X[-train_indices,]
y_test <- y[-train_indices]
```

Linear Regression

```r
# Create and fit the linear regression model
model_LR <- train(X_train, y_train, method = "lm")

# Make predictions on the test set
y_pred_LR <- predict(model_LR, X_test)

# Evaluate the model using mean squared error
mse_LR <- mean((y_test - y_pred_LR)^2)
print(paste("Mean Squared Error:", mse_LR))
```

```
## [1] "Mean Squared Error: 0.159170263725586"
```

```r
# Calculate R-squared
r2_LR <- cor(y_pred_LR, y_test)^2
print(paste("R-squared:", r2_LR))
```

```
## [1] "R-squared: 0.844791340432038"
```

```r
# Print the weights
print(model_LR$finalModel$coefficients)
```

```
##  (Intercept) seasonspring seasonsummer seasonwinter           yr         mnth
##   0.34827806  -0.21856077  -0.36016795  -0.81578274   0.52831047  -0.01622689
##      holiday      weekday   workingday   weathersit         temp        atemp
##  -0.04923481   0.06140956   0.03297618  -0.16117309   0.38697234   0.09590595
##          hum    windspeed
##  -0.09264225  -0.11871848
```

Random Forest

```r
# Create and fit the Random Forest regression model
model_RF <- train(
  x = X_train, y = y_train,
  method = "rf",
  trControl = trainControl(method = "cv", number = 5),
  tuneLength = 10
)

# Make predictions on the test set
y_pred_RF <- predict(model_RF, X_test)

# Evaluate the model using mean squared error
mse_RF <- mean((y_pred_RF - y_test)^2)
print(paste("Mean Squared Error:", mse_RF))
```

```
## [1] "Mean Squared Error: 0.10251703800958"
```

```r
# Calculate R-squared value
r2_RF <- 1 - sum((y_test - y_pred_RF)^2) / sum((y_test - mean(y_test))^2)
print(paste("R-squared:", r2_RF))
```

```
## [1] "R-squared: 0.899968733289444"
```