in collaboration with

Report on Recommended System

Ronit Bhujel

College ID: 220050

Coventry ID: 12980521

BSc. (Hons.) Computing, Softwarica College of IT and E-commerce, Coventry University

ST5014CEM Data Science for Developers

Siddhartha Neupane

January 25,2024

Table of Contents

List of Abbreviations 4

Introduction 4

Problem and Solution Statement 5

Aim 5

Objective 6

Data Collection Source and Justification 6

Cleaning Data 7

1. House Price Cleaning 8
2. Towns and Post Codes Cleaning 9
3. Population Cleaning 10
4. Broadband Speed Cleaning 11
5. Crime Cleaning 11
6. School Cleaning 12

Exploratory Data Analysis 13

1. House Price Analysis 14
2. Broadband Speed Analysis 17
3. Crime Analysis 19
4. School Analysis 23

Linear Modeling 26

Recommended System 33

Reflection 38

Ethical and Legal Issues 39

Future Scope 40

Conclusion 41

GitHub Link: 41

References 42

Table of Figures

Figure 1: House Price Data Cleaning........ 8

Figure 2: House Price Data Cleaning........ 9

Figure 3: Towns and Post Codes Data Cleaning 9

Figure 4: Population Data Cleaning........ 10

Figure 5: Broadband Speed Cleaning 11

Figure 6: Crime Data Cleaning........ 11

Figure 7: School Data Cleaning........ 12

Figure 8: House Price Analysis Box Plot Code 14

Figure 9: House Price Analysis Box Plot 14

Figure 10: House Price Analysis Bar Chart Code........ 15

Figure 11: House Price Analysis Bar Chart........ 15

Figure 12: House Price Analysis Line Graph 16

Figure 13: House Price Analysis Line Graph 16

List of Abbreviations

1. CSV- Comma Separate Value

2. LSOA – Lower Layer Super Output Area

3. EDA – Exploratory Data Analysis

4. DPA – Data Protection Act

Introduction

Welcome to the Town Recommender System, created to assist those of us from around the world who are thinking of doing a study exchange in England. As students currently enrolled in this dynamic nation, we recognize the significance of selecting the ideal location based on a variety of factors, including safety ratings, cost of living, broadband speed, and educational institutions. We will explore the charming counties of Kent and Surrey in this tailored itinerary, providing information on several towns in these areas to assist you in making decisions regarding your study abroad experience. Whether you value excellent educational opportunities, reasonably priced housing, fast internet, or a secure environment, the recommender system can help it sort through the unique qualities of Kent and Surrey towns, guaranteeing a fulfilling and comprehensive stay in England.

This individual assignment uses R, Python, and a carefully chosen set of datasets that are exclusively accessible through the data.gov.uk website of the UK government to analyze the data mining lifecycle. The main goals are to achieve 3NF normalization through an extensive data cleaning process, and then to design and implement a reliable database system. The primary goal of providing international students with useful insights is to direct the study of statistical models and the creation of a recommendation system.

Problem and Solution Statement

Our Town Recommender System addresses the challenges encountered by foreign students arranging to study abroad in England, namely in the counties of Kent and Surrey. This solution was developed in response to the need for a centralized resource for thorough town evaluations. The approach seeks to simplify the decision-making process by taking into consider essential factors like safety ratings, cost of living, broadband speed, and educational institutions. With the help of these personalized recommendations, this tool assists students select towns that best suit their academic objectives and improve their study abroad experience in general.

Aim

In order to provide a well-rounded and educated decision-making process, our goal is to assist international students select the best study exchange locations in Kent and Surrey by providing tailored suggestions based on important variables like education, cost of living, broadband speed, and safety ratings.

Objective

1. Compile extensive data about Kent and Surrey communities' educational opportunities, cost of living, broadband speed, and safety ratings.
2. Create an intuitive user interface for easy navigation and interaction.
3. Design algorithms that evaluate and classify town suggestions according to individual preferences.
4. Assure timely updates to maintain accurate and up-to-date information.
5. Empower international students with a tool that facilitates well-informed decisions for an optimal study exchange experience.

Data Collection Source and Justification

The official UK government website, data.gov.uk, will be the source of all data for the Town Recommender System. This covers details about local crimes, broadband speed, cost of living (home prices), educational institutions, and any other pertinent statistics. Using the official government website guarantees the information's currency, accuracy, and dependability, giving international students a solid foundation on which to make judgments on the study exchange towns in Kent and Surrey. Utilizing data.gov.uk also complies with accountability and transparency criteria, giving users confidence about the veracity of the data they are receiving. Using this official source also makes updates easier and guarantees that the Town Recommender System always provides potential international students with the most accurate and up-to-date information.

Cleaning Data

A vital stage in the creation of our Town Recommender System is data cleaning, which assures the dependability and accuracy of the data obtained from data.gov.uk. This procedure entails locating and correcting any discrepancies, errors, or missing data in the datasets relevant to user preferences, cost of living, broadband speed, safety ratings, and educational institutions. Our goal is to offer reliable and accurate suggestions to international students for their study exchange program in Kent and Surrey through the use of stringent data cleaning procedures. This will help to facilitate a more seamless and informed decision-making process. (Salesforce, n.d.)

To improve database performance and reduce redundancy, we emphasize normalizing datasets using the Third Normal Form (3NF). This guarantees that the data structures used by our Town Recommender System are streamlined and well-organized, encouraging consistency and lowering the possibility of anomalies. We are committed to providing a strong and trustworthy recommendation tool for international students, providing a well-informed choosing process for their study exchange in Kent and Surrey, by combining both data cleaning and normalization.

1. House Price Cleaning

```r
library(tidyverse)
library(dplyr)
library(lubridate)

setwd("C:/Users/hacki/OneDrive/Desktop/Ronit_Bhujel_220050")

#-----------2019 Dataset Cleaning-----------#

#Cleaning data through the use of pipe operator
houseprices_2019 <-read_csv("Obtained Data/House Price Dataset/House Price Dataset 2019.csv", col_names = FALSE) %>%  #Importing CSV into R
  setNames(c("Transaction unique identifier", "Price", "Date of Transfer", "Postcode", "Property Type", "Old/New", "Duration", "PAON",
             "SAON", "Street", "Locality", "Town/City", "District", "County", "PPD Category type", "Record Status")) %>% #Changing Column name
  as_tibble() %>% #Converting into tibble
  na.omit() %>% #Removing rows with null value
  select(Price, `Date of Transfer`, Postcode, `Town/City`, District, County) %>% #selecting only columns that are required
  filter(County =="KENT" | County== "SURREY") %>% #Preserving rows with Kent and Surrey as county
  mutate(`Date of Transfer` = year(as.Date(`Date of Transfer`, format = "%y/%m/%d"))) %>%  #modifying the date of transfer column to only show year
  mutate(S_No = row_number()) %>% #Adding a new serial number column
  select(S_No, everything()) #moving the serial number column at first

#-----------2020 Dataset Cleaning-----------#

houseprices_2020 <-read_csv("Obtained Data/House Price Dataset/House Price Dataset 2020.csv", col_names = FALSE) %>%  #Importing CSV into R
  setNames(c("Transaction unique identifier", "Price", "Date of Transfer", "Postcode", "Property Type", "Old/New", "Duration", "PAON",
             "SAON", "Street", "Locality", "Town/City", "District", "County", "PPD Category type", "Record Status")) %>% #Changing Column name
  as_tibble() %>% #Converting into tibble
  na.omit() %>% #Removing rows with null value
  select(Price, `Date of Transfer`, Postcode, `Town/City`, District, County) %>% #selecting only columns that are required
  filter(County =="KENT" | County== "SURREY") %>% #Preserving rows with Kent and Surrey as county
  mutate(`Date of Transfer` = year(as.Date(`Date of Transfer`, format = "%y/%m/%d"))) %>%  #modifying the date of transfer column to only show year
  mutate(S_No = row_number()) %>% #Adding a new serial number column
  select(S_No, everything()) #moving the serial number column at first
```

```r
#-----------2021 Dataset Cleaning-----------#

houseprices_2021 <-read_csv("Obtained Data/House Price Dataset/House Price Dataset 2021.csv", col_names = FALSE) %>%  #Importing CSV into R
  setNames(c("Transaction unique identifier", "Price", "Date of Transfer", "Postcode", "Property Type", "Old/New", "Duration", "PAON",
             "SAON", "Street", "Locality", "Town/City", "District", "County", "PPD Category type", "Record Status")) %>% #Changing Column name
  as_tibble() %>% #Converting into tibble
  na.omit() %>% #Removing rows with null value
  select(Price, `Date of Transfer`, Postcode, `Town/City`, District, County) %>% #selecting only columns that are required
  filter(County =="KENT" | County== "SURREY") %>% #Preserving rows with Kent and Surrey as county
  mutate(`Date of Transfer` = year(as.Date(`Date of Transfer`, format = "%y/%m/%d"))) %>%  #modifying the date of transfer column to only show year
  mutate(S_No = row_number()) %>% #Adding a new serial number column
  select(S_No, everything()) #moving the serial number column at first

#-----------2022 Dataset Cleaning-----------#

houseprices_2022 <-read_csv("Obtained Data/House Price Dataset/House Price Dataset 2022.csv", col_names = FALSE) %>%  #Importing CSV into R
  setNames(c("Transaction unique identifier", "Price", "Date of Transfer", "Postcode", "Property Type", "Old/New", "Duration", "PAON",
             "SAON", "Street", "Locality", "Town/City", "District", "County", "PPD Category type", "Record Status")) %>% #Changing Column name
  as_tibble() %>% #Converting into tibble
  na.omit() %>% #Removing rows with null value
  select(Price, `Date of Transfer`, Postcode, `Town/City`, District, County) %>% #selecting only columns that are required
  filter(County =="KENT" | County== "SURREY") %>% #Preserving rows with Kent and Surrey as county
  mutate(`Date of Transfer` = year(as.Date(`Date of Transfer`, format = "%y/%m/%d"))) %>%  #modifying the date of transfer column to only show year
  mutate(S_No = row_number()) %>% #Adding a new serial number column
  select(S_No, everything()) #moving the serial number column at first
```

Figure 1: House Price Data Cleaning

The data cleaning procedure is shown in above figure, illustrating the changes from 2019 to 2020 and 2021 to 2022, respectively. The steps involved in each section are importing CSV files, giving columns meaningful names, converting datasets into Tibbles, and eliminating null values. Relevant columns are highlighted and the data is narrowed down to just include the counties of Kent and Surrey. The 'Date of Transfer' field has been modified to only display the year, and a new column for serial numbers has been added to improve classification. This procedure provides clean datasets that ensure consistency and clarity throughout integration and subsequent analyses.

```
#merging all the cleaned dataset into a single tibble
combined_houseprices<- bind_rows(houseprices_2019, houseprices_2020, houseprices_2021, houseprices_2022) %>%
  mutate(`Short Postcode`= substr(Postcode, 1,5)) #adding another column to the combine dataset

#defining path to save the cleaned dataset
file_path <- "Cleaned Data/Cleaned House Prices.csv"

#saving the cleaned dataset
write.csv(combined_houseprices,file_path, row.names = FALSE)
```

Figure 2: House Price Data Cleaning

The code creates a single Tibble named "combined house prices" by combining clean datasets of home prices from 2019 to 2022. The datasets are arranged sequentially to achieve this consolidation. Furthermore, 'Short Postcode' is included as a new column to enhance the postcode representation by eliminating the first five characters from the current 'Postcode' column. This process aims to produce a comprehensive dataset for further investigation. This improved and merged dataset must be saved as a CSV file for later usage in order to finish the procedure.

2. Towns and Post Codes Cleaning

```
library(tidyverse)
library(dplyr)
library(lubridate)

setwd("C:/Users/hacki/OneDrive/Desktop/Ronit_Bhujel_220050")

#importing cleaned house price dataset
cleaned_houseprices <- read_csv("Cleaned Data/Cleaned House Prices.csv")

#Cleaning and joining data through the use of pipe operator
postcode_to_lsoa <- read_csv("Obtained Data/Postcode to LSOA.csv") %>% #importing Postcode to LSOA csv file
  select(pcd7, lsoa11cd) %>% #selecting only required columns
  rename(Postcode= pcd7, `LSOA Code`= lsoa11cd) %>% #renaming columns
  right_join(cleaned_houseprices, by="Postcode") %>% #Joining with the cleaned house price dataset by matching Postcode
  select(`LSOA Code`, Postcode,`Short Postcode`,`Town/City`, District, County, ) %>% #selecting only required columns
  mutate(S_No = row_number()) %>% #Adding a new serial number column
  select(S_No, everything()) #moving the serial number column at first

#defining path to save the cleaned dataset
file_path <- "Cleaned Data/Cleaned Towns and Post Codes.csv"

#saving the cleaned dataset
write.csv(postcode_to_lsoa,file_path, row.names = FALSE)
```

Figure 3: Towns and Post Codes Data Cleaning

The image above shows how to integrate the cleaned house price dataset with Postcode to Lower Layer Super Output Area (LSOA) mappings in order to provide geographic information to the study. Import postcode and clean house pricing data into LSOA first. For the last case, pertinent columns are chosen and column names are made simpler by using the pipe operator. The Postcode, Short Postcode, Town/City, District, and County are all retained in the final dataset after a right join joins the Postcode data from the two datasets. Including a serial number column makes things more organized. The cleaned and enriched dataset with LSOA Code information is saved as a CSV file for further use as the last step.

3. Population Cleaning

```
library(tidyverse)
library(dplyr)
library(lubridate)
library(stringr)

setwd("C:/Users/hacki/OneDrive/Desktop/Ronit_Bhujel_220050")

#Importing cleaned postcode to LSOA csv into R
cleaned_postcode_to_LSOA<- read_csv("Cleaned Data/Cleaned Towns and Post Codes.csv")

#importing population dataset and managing the postcode column
population <- read_csv("Obtained Data/Population Dataset.csv")%>%
  rename(`Short Postcode`= Postcode) %>% #renaming postcode to short postcode
  mutate(`Short Postcode` = gsub(" ", "", `Short Postcode`),  # Remove all spaces
         `Short Postcode` = if_else(nchar(`Short Postcode`) == 5,
                             paste0(substr(`Short Postcode`, 1, 4), " ", substr(`Short Postcode`, 5, 6)),
                             paste0(substr(`Short Postcode`, 1, 3), " ", substr(`Short Postcode`, 4, 5)))) #fixing inconsistent spacing in postcode column %>%

#cleaning the population dataset further and joining with Postcode to LSOA table
population<- population %>%
  as_tibble() %>% #converting into tibble
  right_join(cleaned_postcode_to_LSOA, by="Short Postcode") %>%  #Joining with the cleaned Postcode to LSOA dataset by matching Postcode
  na.omit() %>%  #removing null values
  select(S_No, everything()) #moving the serial number column at first

#defining path to save the cleaned dataset
file_path <- "Cleaned Data/Cleaned Population.csv"

#saving the cleaned dataset
write.csv(population,file_path, row.names = FALSE)
```

Figure 4: Population Data Cleaning

This process involves adding population data to the study by combining additional population data with the revised Postcode to the LSOA dataset. To ensure consistency, the 'Short Postcode' column is monitored, and the population dataset is improved and combined with the cleaned Postcode to create the LSOA dataset. After filtering null values, the dataset is organized so that the serial number column remains at the top. After processing, a new CSV file with the improved population dataset is saved. By providing a more complete image of the towns for the town recommender system, the study is enhanced by the inclusion of demographic data.

4. Broadband Speed Cleaning

```
library(tidyverse)
library(dplyr)
library(lubridate)

setwd("C:/Users/hacki/OneDrive/Desktop/Ronit_Bhujel_220050")

#Importing cleaned postcode to LSOA csv into R
cleaned_postcode_to_LSOA<- read_csv("Cleaned Data/Cleaned Towns and Post Codes.csv")

#Cleaning and joining data through the use of pipe operator
broadband_speed<-read_csv("Obtained Data/Broadband Speed.csv") %>% #Importing broadband speed csv into R
  as_tibble() %>% #converting into tibble
  select('Average download speed (Mbit/s)', postcode_space) %>%  #only selecting columns that are required
  rename(Postcode= `postcode_space`) %>% #renaming the post_space column to Postcode
  right_join(cleaned_postcode_to_LSOA, by="Postcode") %>% #Joining with the cleaned house price dataset by matching Postcode
  select('Average download speed (Mbit/s)',Postcode, `Short Postcode`, `Town/City`, District, County,) %>% #selecting only required columns
  na.omit() %>%  #Removing rows with null value
  mutate(`Short Postcode`= substr(Postcode, 1,5)) %>% #Filling missing short code values
  mutate(S_No = row_number()) %>% #Adding a new serial number column
  select(S_No, everything()) #moving the serial number column at first

#defining path to save the cleaned dataset
file_path <- "Cleaned Data/Cleaned Broadband Speed Dataset.csv"

#saving the cleaned dataset
write.csv(broadband_speed,file_path, row.names = FALSE)
```

Figure 5: Broadband Speed Cleaning

By merging the collected broadband speed dataset with the cleaned Postcode to the LSOA dataset, this method applies broadband speed data to the study. First, the pipe operator is used to import and process the broadband speed data, selecting and naming only the relevant columns. Using the cleaned Postcode to LSOA dataset, a right merge is performed with postcode-based matching. The dataset is reorganized to preserve the serial number column at the beginning and cleaned up by removing null values.

5. Crime Cleaning

```
library(tidyverse)
library(dplyr)
library(lubridate)

setwd("C:/Users/hacki/OneDrive/Desktop/Ronit_Bhujel_220050")

# Define the path to the main directory containing all the year-month folders
main_dir <- "Obtained Data/Crime Dataset"

# Create a list of all CSV file paths
file_paths <- list.files(main_dir, pattern = "\\.csv$", full.names = TRUE, recursive = TRUE)

# Read and combine all CSV files into one dataframe
combined_crime_dataset <- file_paths %>%
  set_names() %>% # Ensure each element in file_paths is named
  map_df(~read_csv(.x)) %>%   # Apply read_csv to each file path
  as_tibble() #converting into tibble

#Importing cleaned postcode to LSOA csv into R
cleaned_postcode_to_LSOA<- read_csv("Cleaned Data/Cleaned Towns and Post Codes.csv")

#Cleaning the combined crime data set through the use of pipe operator
combined_crime_dataset<- combined_crime_dataset %>%
  select(Month, `Falls within`, `Crime type`, `LSOA code`) %>% #selecting only columns that are required
  rename(`Date of crime`= `Month`, `LSOA Code`= `LSOA code`) %>% #renaming the month column
  right_join(cleaned_postcode_to_LSOA, join_by(`LSOA Code`)) %>% #joining with another table to show towns
  select(`Date of crime`, `Falls within`, `Crime type`, `LSOA Code`, `Postcode`, `Short Postcode`, `Town/City`) %>% #selecting only columns that are required
  na.omit() %>% #removing null values
  mutate(S_No = row_number()) %>% #Adding a new serial number column
  select(S_No, everything()) #moving the serial number column at first

#defining path to save the cleaned dataset
file_path <- "Cleaned Data/Cleaned Crime.csv"

#saving the cleaned dataset
write.csv(combined_crime_dataset,file_path, row.names = FALSE)
```

Figure 6: Crime Data Cleaning

The above figure illustrates the cleaning of crime data. All the csv files are combined into one data frame and cleaned postcode to LSOA is imported. Then, the combined crime data is cleaned using pipe operator. Lastly, the cleaned crime data is saved defining a path.

6. School Cleaning

```r
library(tidyverse)
library(dplyr)
library(lubridate)

setwd("C:/Users/hacki/OneDrive/Desktop/Ronit_Bhujel_220050")

#-----------2018 Kent School Dataset Cleaning-----------#

kent_2018_2019_school <- read_csv("Obtained Data/School Dataset/Kent 2018-2019 School Dataset.csv") %>%
  select(SCHNAME,ATT8SCR, TOWN, PCODE ) %>%  #Selecting only the required columns
  rename(`School Name`=SCHNAME,`Attainment Score`=ATT8SCR, Town= TOWN, `Postcode`= PCODE) %>%
  as_tibble() %>% #Converting into tibble
  mutate('Short Post Code'= substr(Postcode, 1, 5)) %>%
  na.omit() %>%  #Removing rows with null value
  filter (`Attainment Score` != "NE" & `Attainment Score` != "SUPP") %>%  #removing NE and SUPP from Attainment Score row
  mutate(County = "Kent") %>% #adding a new column for county
  mutate(Year= "2018") %>% #adding a new column for year
  mutate(S_No = row_number()) %>% #Adding a new serial number column
  select(S_No, everything())  #moving the serial number column at first

#-----------2021 Kent School Dataset Cleaning-----------#

kent_2021_2022_school <- read_csv("Obtained Data/School Dataset/Kent 2021-2022 School Dataset.csv") %>%
  select(SCHNAME,ATT8SCR, TOWN, PCODE ) %>%  #Selecting only the required columns
  rename(`School Name`=SCHNAME,`Attainment Score`=ATT8SCR, Town= TOWN, `Postcode`= PCODE) %>%
  as_tibble() %>% #Converting into tibble
  mutate('Short Post Code'= substr(Postcode, 1, 5)) %>%
  na.omit() %>%  #Removing rows with null value
  filter (`Attainment Score` != "NE" & `Attainment Score` != "SUPP") %>%  #removing NE and SUPP from Attainment Score row
  mutate(County = "Kent") %>% #adding a new column for county
  mutate(Year= "2021") %>% #adding a new column for year
  mutate(S_No = row_number()) %>% #Adding a new serial number column
  select(S_No, everything())  #moving the serial number column at first
```

```r
#-----------2018 Surrey School Dataset Cleaning-----------#

Surrey_2018_2019_school <- read_csv("Obtained Data/School Dataset/Surrey 2018-2019 School Dataset.csv") %>%
  select(SCHNAME,ATT8SCR, TOWN, PCODE ) %>%  #Selecting only the required columns
  rename(`School Name`=SCHNAME,`Attainment Score`=ATT8SCR, Town= TOWN, `Postcode`= PCODE) %>%
  as_tibble() %>% #Converting into tibble
  mutate('Short Post Code'= substr(Postcode, 1, 5)) %>%
  na.omit() %>%  #Removing rows with null value
  filter (`Attainment Score` != "NE" & `Attainment Score` != "SUPP") %>%  #removing NE and SUPP from Attainment Score row
  mutate(County = "Surrey") %>% #adding a new column for county
  mutate(Year= "2018") %>% #adding a new column for year
  mutate(S_No = row_number()) %>% #Adding a new serial number column
  select(S_No, everything())  #moving the serial number column at first

#-----------2021 Surrey School Dataset Cleaning-----------#

Surrey_2021_2022_school <- read_csv("Obtained Data/School Dataset/Surrey 2021-2022 School Dataset.csv") %>%
  select(SCHNAME,ATT8SCR, TOWN, PCODE ) %>%  #Selecting only the required columns
  rename(`School Name`=SCHNAME,`Attainment Score`=ATT8SCR, Town= TOWN, `Postcode`= PCODE) %>%
  as_tibble() %>% #Converting into tibble
  mutate('Short Post Code'= substr(Postcode, 1, 5)) %>%
  na.omit() %>%  #Removing rows with null value
  filter (`Attainment Score` != "NE" & `Attainment Score` != "SUPP") %>%  #removing NE and SUPP from Attainment Score row
  mutate(County = "Surrey") %>% #adding a new column for county
  mutate(Year= "2021") %>% #adding a new column for year
  mutate(S_No = row_number()) %>% #Adding a new serial number column
  select(S_No, everything())  #moving the serial number column at first

#merging all the cleaned dataset into a single tibble
combined_school_dataset= bind_rows(kent_2018_2019_school, kent_2021_2022_school, Surrey_2018_2019_school, Surrey_2021_2022_school)

#defining path to save the cleaned dataset
file_path <- "Cleaned Data/Cleaned School.csv"

#saving the cleaned dataset
write.csv(combined_school_dataset,file_path, row.names = FALSE)
```

Figure 7: School Data Cleaning

The school dataset of the year 2018 and 2021 from Kent and Surrey is cleaned and merged into a single tibble. Then, saved as a clean school dataset defining a path to it.

Exploratory Data Analysis

Exploratory Data Analysis (EDA) methods are integrated into our Town Recommender System in order to extract insightful information from the gathered datasets. We analyze patterns, trends, and connections in the data related to educational institutions, cost of living, broadband speed, safety ratings, and user preferences using graphical representations, statistical summaries, and data visualization tools. EDA offers a greater understanding of the underlying data structure in addition to helping in the identification of abnormalities and outliers. Our recommendation model is developed using the insights from this comprehensive study, guaranteeing that overseas students are provided with thoughtful and sophisticated town options that consider the nuances that were discovered during the exploratory stage.

Moreover, by using exploratory data analysis, we can find potential connections between various elements, leading to a more comprehensive comprehension of the town's features. We may improve our algorithms for individualized recommendations by visualizing patterns in the data, which will help us precisely match the system to the varied interests of international students. EDA is essential to improve our Town Recommender System's precision and efficacy, which helps students organizing their study exchange in Kent and Surrey make more informed and personalized decisions.

In compliance with the project requirements, the data is visually analyzed, and some of the main findings are as follows:

1. House Price Analysis

```r
library(tidyverse)
library(dplyr)
library(lubridate)
library(ggplot2)
library("scales")

setwd("C:/Users/hacki/OneDrive/Desktop/Ronit_Bhujel_220050")

#importing the cleaned house prices
cleaned_houseprices= read_csv('Cleaned Data/Cleaned House Prices.csv')

#-----------2022 House Price Box plot-----------#

#grouping the cleaned house prices by county , towns and DOT and showing the average price for each group
Grouped_houseprice = cleaned_houseprices%>%
  group_by(`Town/City`,District,County,`Date of Transfer`) %>%
  summarise(`Average Price`= mean(Price)) %>%
  ungroup(`Town/City`,District,County,`Date of Transfer`)

#creating box plot to visualize average house prices in Kent and Surrey in 2022
Grouped_houseprice %>%
  filter(`Date of Transfer`==2022) %>% #filtering to show only house price data of 2022
  group_by(County) %>% #grouping by county since we are comparing counties only
  ggplot(aes(x = County, y = `Average Price`, fill=County)) + #setting x-axis and y-axis values
  scale_y_continuous(limits=c(0,2000000), breaks = seq(0,2000000,300000))+ #setting limits and breaks
  geom_boxplot() + #specifying the type of plot we need
  labs(title="2022 Average House Prices By County Box Plot") + #setting label for the chart
  scale_fill_manual(values = c("red","blue"))
```

Figure 8: House Price Analysis Box Plot Code

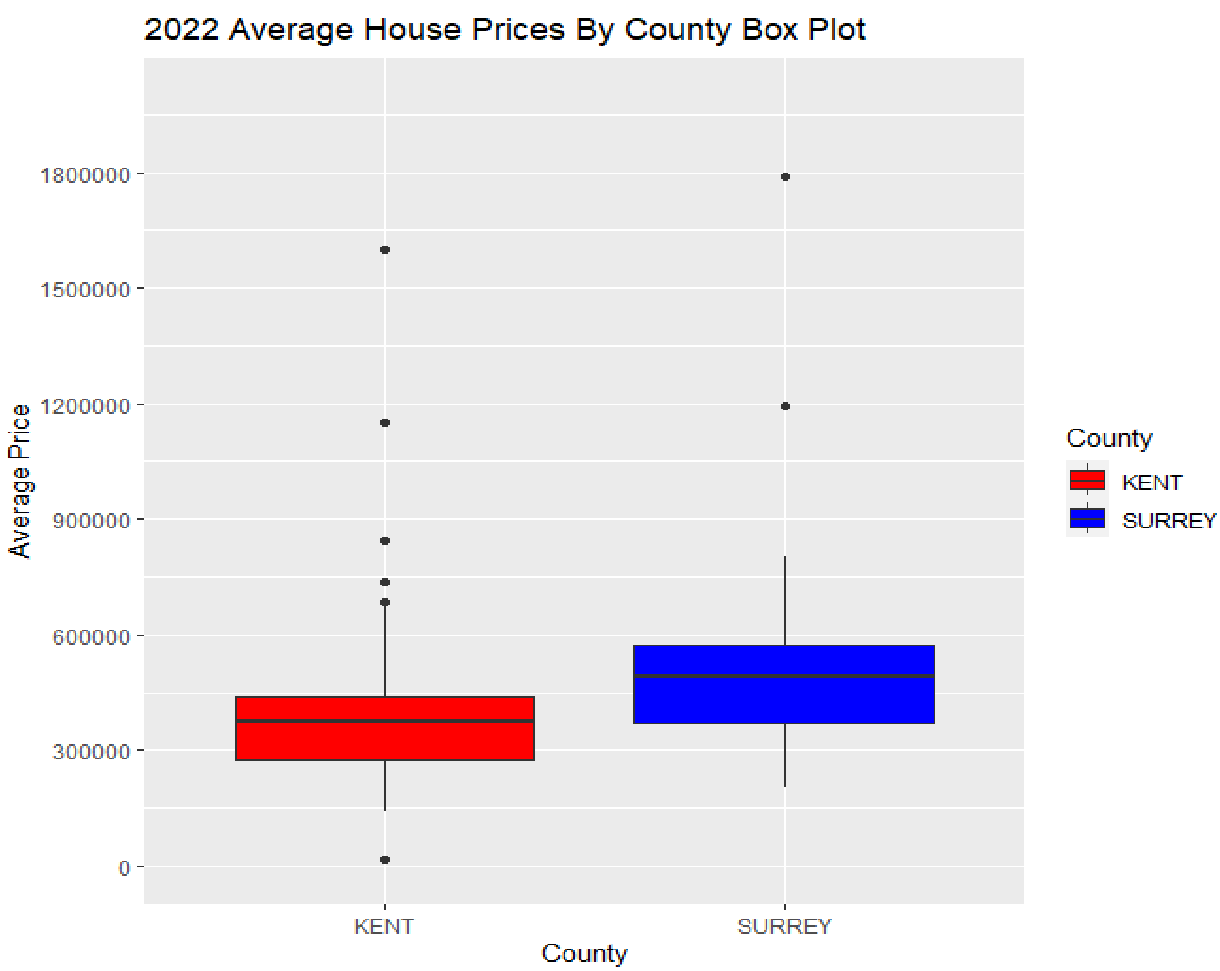


Figure 9: House Price Analysis Box Plot

```r
#-----------2022 Average House Price Bar Chart-----------#

#creating bar chart to visualize average house prices in Kent and Surrey
Grouped_houseprice %>%
  filter(`Date of Transfer`==2022) %>% #filtering to show only house price data of 2022
  group_by(County) %>% #grouping by county since we are comparing counties only
  ggplot(aes(x = County, y = `Average Price`, fill= County)) + #defining x-axis and y-axis values
  geom_bar(stat = "identity") + #using average prices as height of the bar
  scale_y_continuous(limits=c(0,2000000), breaks = seq(0,2000000,300000))+ #setting limits and breaks
  labs(title = "2022 Average House Prices Barchart") +
  scale_fill_manual(values = c("red","blue"))
```

Figure 10: House Price Analysis Bar Chart Code

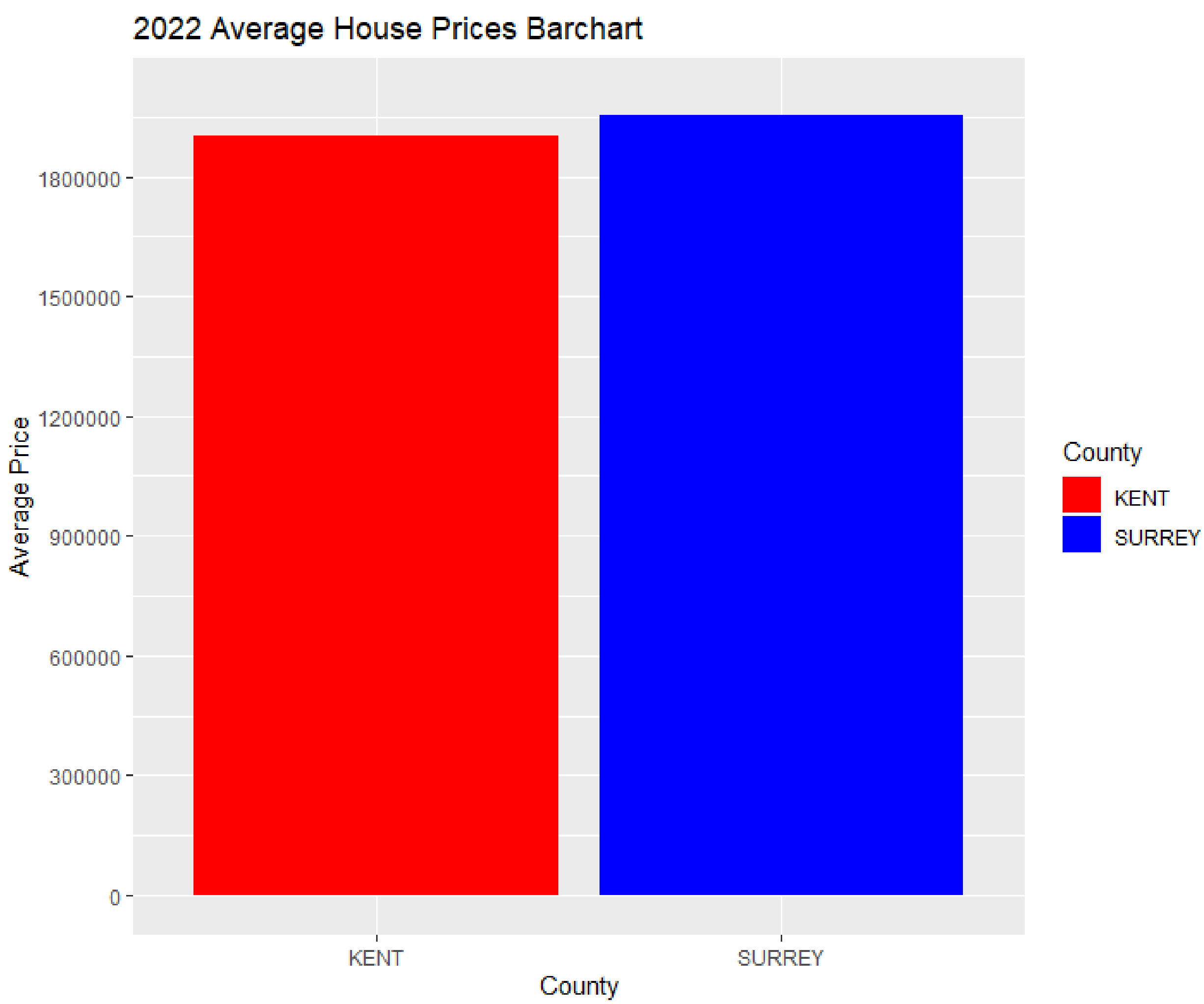


Figure 11: House Price Analysis Bar Chart

```r
#-----------2019-2022 Average House Line Graph-----------#

#grouping the cleaned house prices by county and year and showing the average price for each group
Grouped_houseprice2 = cleaned_houseprices%>%
  group_by(County,`Date of Transfer`) %>%
  summarise(`Average Price`= mean(Price))

#creating line graph of average house prices from 2021-2022
Grouped_houseprice2 %>%
  filter(`Date of Transfer`==2019 | `Date of Transfer`==2020 | `Date of Transfer`==2021 |`Date of Transfer`==2022) %>% #filtering to show only house price data of 2019,2020,2021,2022
  group_by(County, `Date of Transfer`) %>%  #grouping by county and date of transfer since we are comparing prices of counties year after year
  ggplot( aes(x = `Date of Transfer`, y = `Average Price`, group = County, color = County)) + #defining x-axis and y-axis values and colors of line
  geom_line(linewidth = 1) + #defining line width
  geom_point(size = 2, color = "brown") + #defining point size and color
  scale_y_continuous(limits=c(0,700000), breaks = seq(0,700000,100000), labels = label_number()) + #defining limits, breaks and setting label as number instead of scientific notation
  labs(title = "2019-2022 Average House Prices Line Graph", #defining labels
       x = "Year",
       y = "Average Price")
```

Figure 12: House Price Analysis Line Graph

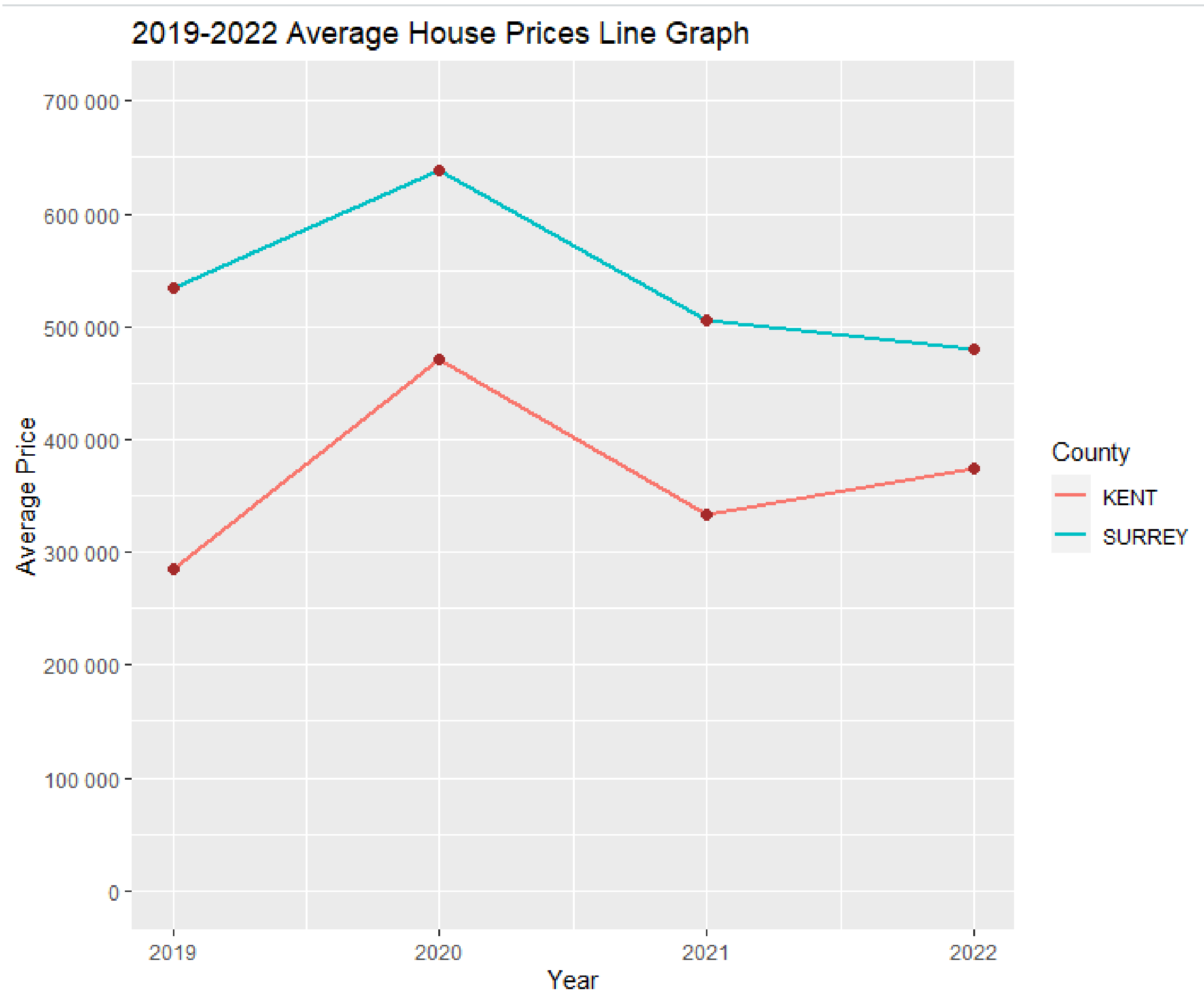
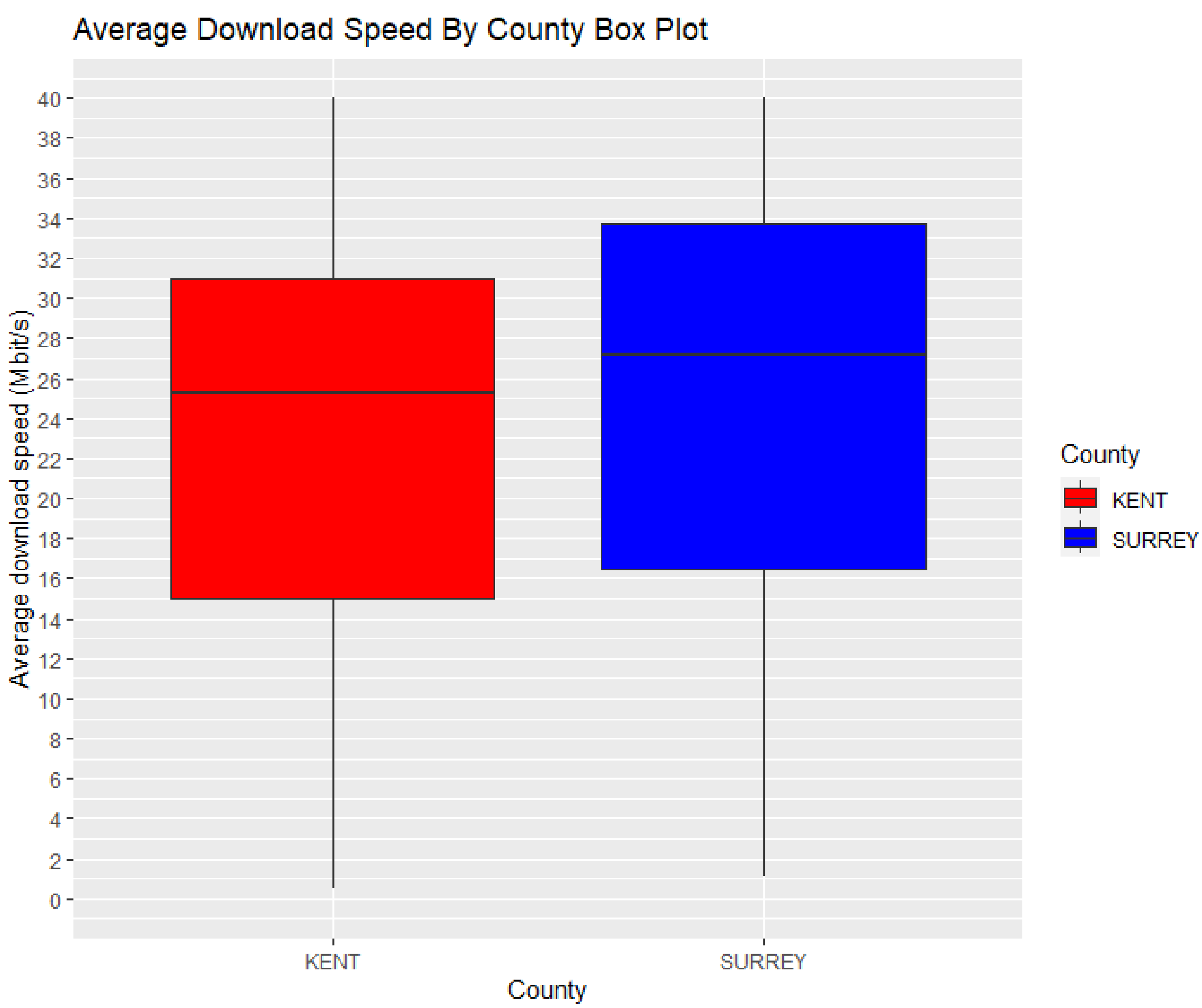


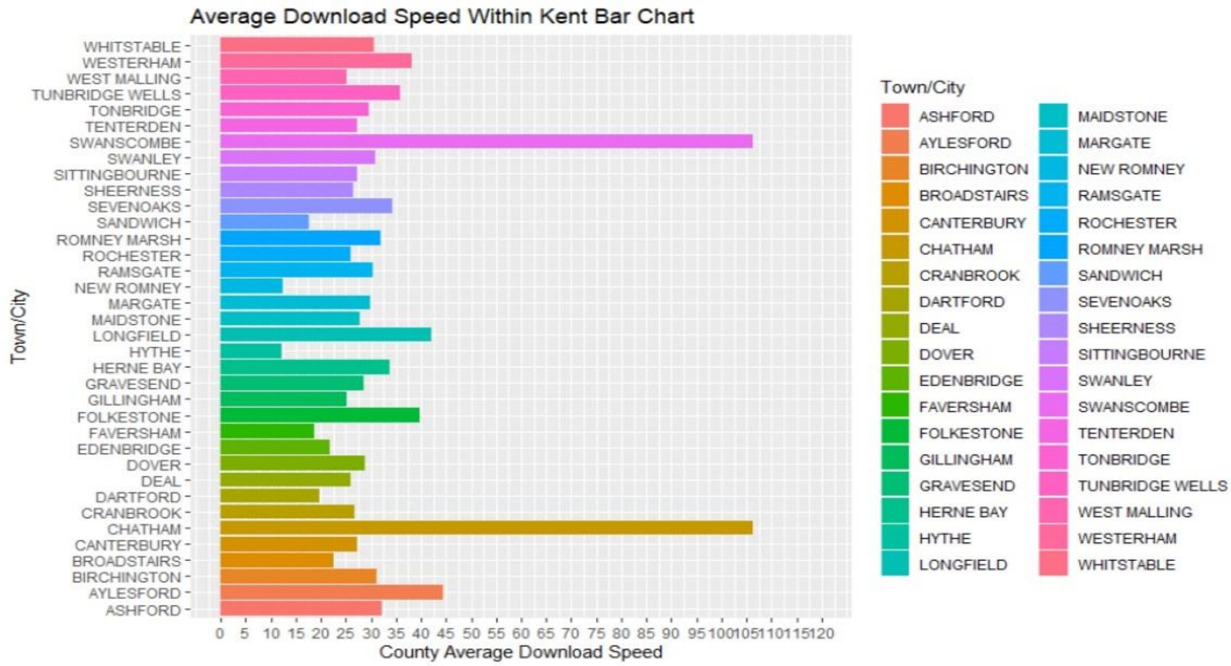
Figure 13: House Price Analysis Line Graph

2. Broadband Speed Analysis

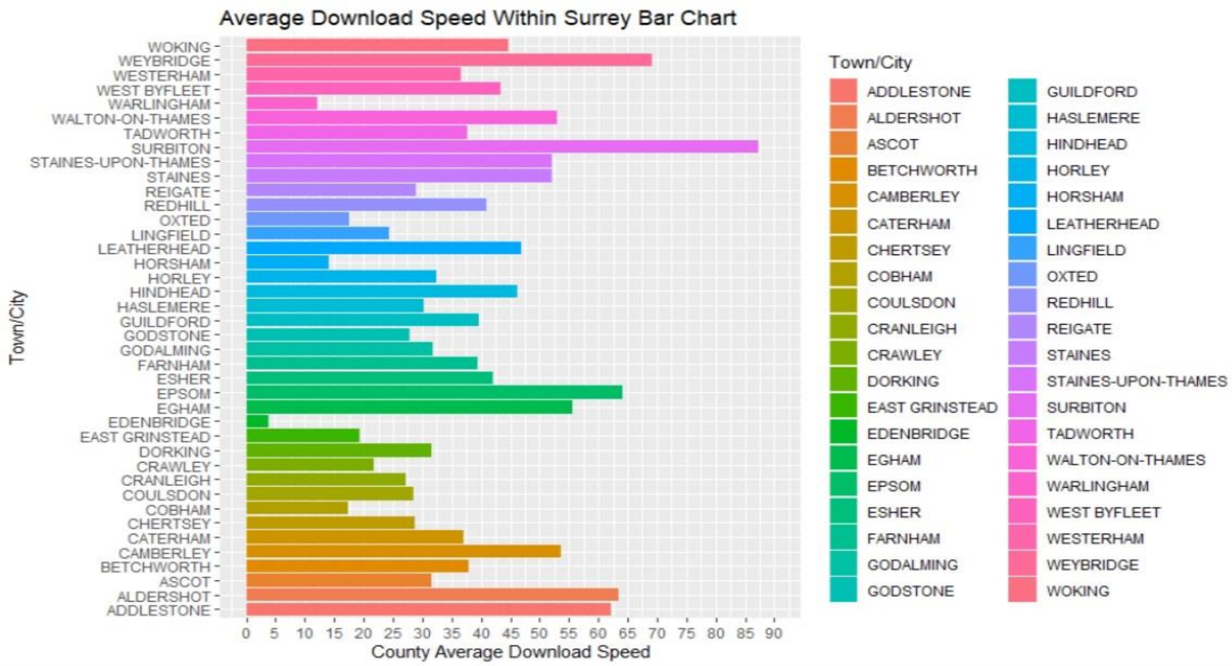
```
library(tidyverse)
library(dplyr)
library(lubridate)
library(ggplot2)
library("scales")

setwd("C:/Users/hacki/OneDrive/Desktop/Ronit_Bhujel_220050")

#importing the cleaned house prices
cleaned_broadband_speed= read_csv('Cleaned Data/Cleaned Broadband Speed.csv')

#----------- Broadband Speed Box plot-----------#

#creating box plot to visualize average download speed in Kent and Surrey
cleaned_broadband_speed %>%
  group_by(County) %>% #grouping by county since we are comparing counties only
  ggplot(aes(x = County, y = `Average download speed (Mbit/s)`, fill=County)) + #setting x-axis and y-axis values
  scale_y_continuous(limits=c(0,40), breaks = seq(0,40,2))+ #setting limits and breaks
  geom_boxplot() + #specifying the type of plot we need
  labs(title="Average Download Speed By County Box Plot") + #setting label for the chart
  scale_fill_manual(values = c("red","blue"))
```



```
#----------- Broadband Speed Bar Charts-----------#

#creating bar chart to visualize average download speed in Kent
cleaned_broadband_speed %>%
  filter(County== "KENT") %>%
  group_by(`Town/City`) %>% #grouping by county since we are comparing counties only
  summarise(`County Average Download Speed`= mean(`Average download speed (Mbit/s)`)) %>%
  ggplot(aes(x = `Town/City`, y = `County Average Download Speed`, fill=`Town/City`)) + #setting x-axis and y-axis values
  scale_y_continuous(limits=c(0,120), breaks = seq(0,120,5))+ #setting limits and breaks
  geom_bar(stat = "identity") + #specifying the type of plot we need
  labs(title="Average Download Speed Within Kent Bar Chart") + #setting label for the chart
  coord_flip()
```

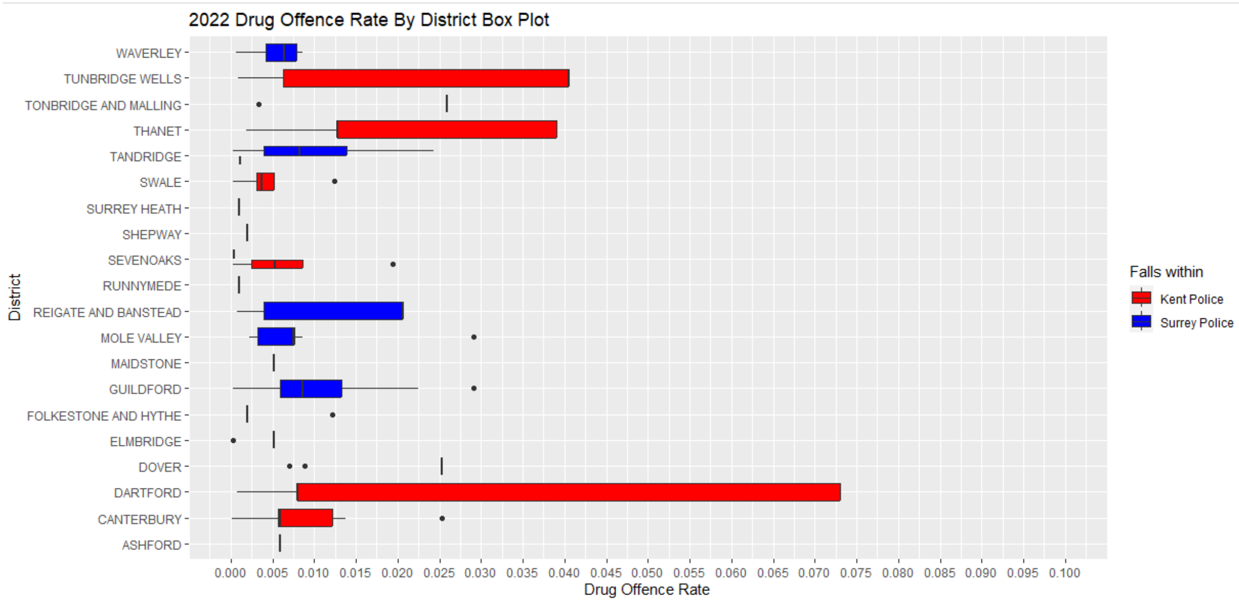



```
#creating bar chart to visualize average download speed in Surrey
cleaned_broadband_speed %>%
  filter(County== "SURREY") %>%
  group_by(`Town/City`) %>% #grouping by county since we are comparing counties only
  summarise(`County Average Download Speed`= mean(`Average download speed (Mbit/s)`)) %>%
  ggplot(aes(x = `Town/City`, y = `County Average Download Speed`, fill=`Town/City`)) + #setting x-axis and y-axis values
  scale_y_continuous(limits=c(0,90), breaks = seq(0,90,5))+ #setting limits and breaks
  geom_bar(stat = "identity") + #specifying the type of plot we need
  labs(title="Average Download Speed Within Surrey Bar Chart") + #setting label for the chart
  coord_flip()
```

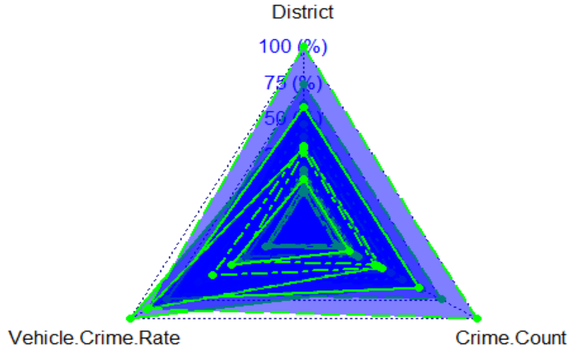


3. Crime Analysis

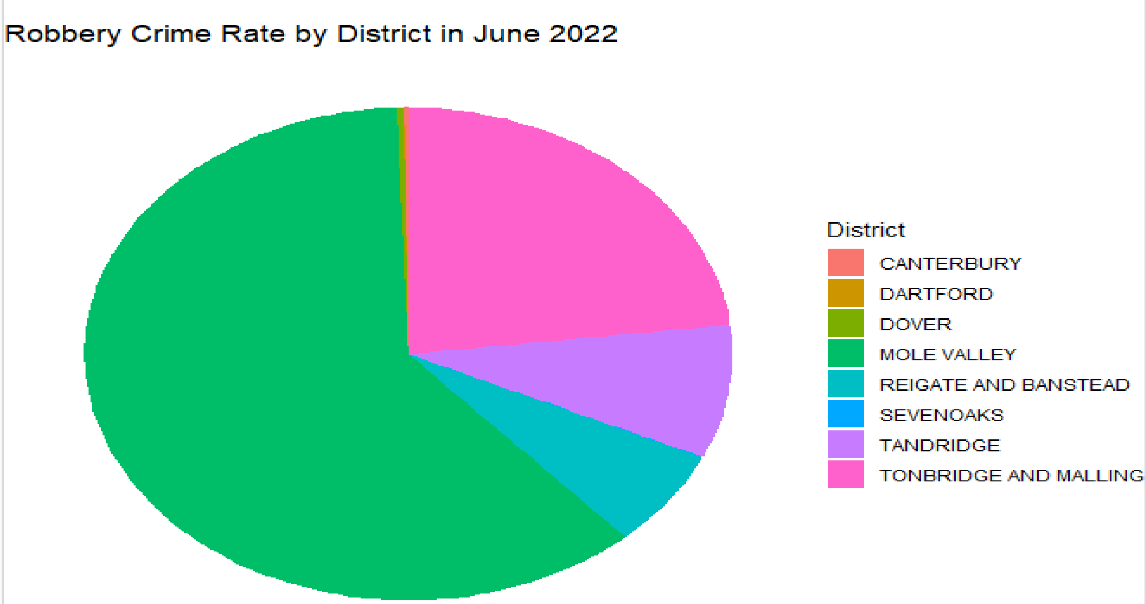
```r
install.packages("fmsb") #installing this package for radar chart

library(tidyverse)
library(dplyr)
library(lubridate)
library(ggplot2)
library("scales")
library(fmsb)

setwd("C:/Users/hacki/OneDrive/Desktop/Ronit_Bhujel_220050")

#importing the cleaned crime dataset
cleaned_crime_dataset= read_csv('Cleaned Data/Cleaned Crime.csv')

#importing population dataset
population_dataset<- read_csv('Cleaned Data/Cleaned Population.csv')

#-----------2022 Drug Offence Rate Box plot-----------#

#modifying our crime dataset to show drug offence rate and crime count for 2022
crime_dataset_drugs <-cleaned_crime_dataset %>%
  mutate(`Date of crime`= substr(`Date of crime`, 1, 4)) %>% #Mutating this column to only show year
  group_by(`Short Postcode`,`Crime type`,`Date of crime`, `Falls within`) %>% #Grouping to show crime count in each postcode by year
  select(`Short Postcode`,`Crime type`,`Date of crime`, `Falls within`) %>%
  na.omit() %>%
  tally() %>% #creating crime count column
  rename(`Crime Count`=n) %>%  #renaming crime count column %>%
  right_join(population_dataset, by = "Short Postcode") %>% #joining with population dataset to show district and population
  select(`Short Postcode`,`Crime type`,`Crime Count`, `Population`, `Date of crime`, `Falls within`, District) %>% #select the required columns
  na.omit() %>%
  filter(`Crime type`== "Drugs" & `Date of crime`==2022) %>% #filtering to show only drug crimes of 2022
  mutate(`Drug Offence Rate` = (`Crime Count` / Population)) #calculating drug offence rate

#creating box plot to visualize drug offence rate in Kent and Surrey's district in 2022
ggplot(data = crime_dataset_drugs, aes(x = District, y = `Drug Offence Rate`, fill = `Falls within`)) + #setting x-axis and y-axis values
  scale_y_continuous(limits=c(0,0.1), breaks = seq(0,0.1,0.005), labels = label_number()) + #defining limits, breaks
  geom_boxplot() + #defining the type of plot we want
  labs(title = "2022 Drug Offence Rate By District Box Plot") +
  scale_fill_manual(values = c("red","blue")) +
  coord_flip()
```



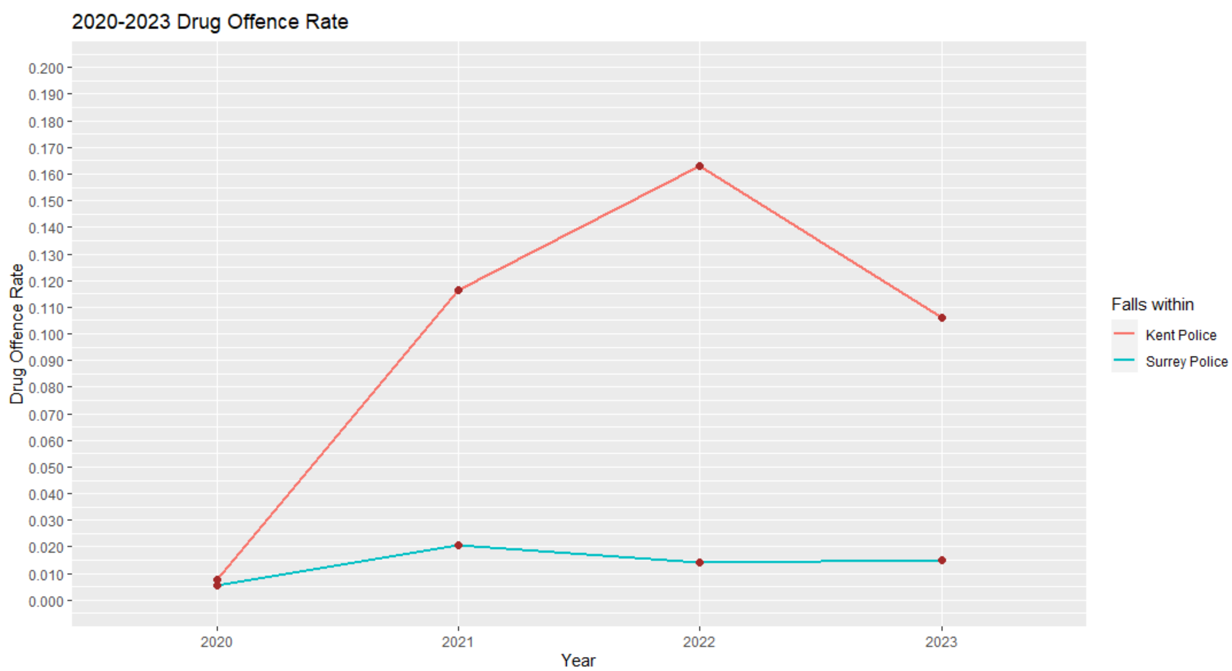

```
#-----------2022 June Vehicle Crime Rate Per 10000 people Radar Chart-----------#

#modifying our crime dataset to show vehicle crime rate and crime count
crime_dataset_vehicle <-cleaned_crime_dataset %>%
  group_by(`Short Postcode`,`Crime type`,`Date of crime`, `Falls within`) %>% #Grouping to show crime count in each postcode by year
  select(`Short Postcode`,`Crime type`,`Date of crime`, `Falls within`) %>%
  na.omit() %>%
  tally() %>% #creating crime count column
  rename(`Crime Count`=n) %>%  #renaming crime count column %>%
  ungroup(`Short Postcode`,`Crime type`,`Date of crime`, `Falls within`) %>%
  right_join(population_dataset, by = "Short Postcode") %>% #joining with population dataset to show district and population
  select(`Short Postcode`,`Crime type`,`Crime Count`, `Population`, `Date of crime`, `Falls within`, District) %>% #select the required columns
  na.omit() %>%
  filter(`Crime type`== "Vehicle crime" & `Date of crime`=="2022-06") %>%   #filtering to show only vehicle crimes of 2022 June
  mutate(`Vehicle Crime Rate` = (`Crime Count` / Population)*10000) #calculating vehicle crime rate per 10000 people

radar_data<- crime_dataset_vehicle %>%
  select(District, `Vehicle Crime Rate`, `Crime Count`) %>%
  unique()  # Assuming you want unique districts


# Find the max value for scaling the radar chart
max_value <- max(radar_data$`Vehicle Crime Rate`, na.rm = TRUE)
max_crime_count <- max(radar_data$`Crime Count`, na.rm = TRUE)

# Create a dataframe with max values
max_row <- data.frame(District = "Max", `Vehicle Crime Rate` = max_value, `Crime Count`=max_crime_count) %>%
  rename(`Vehicle Crime Rate`= `Vehicle.Crime.Rate`) %>%
  rename(`Crime Count`= `Crime.Count`)



# Add the max_row dataframe to the start of radar_data
radar_data <- rbind(max_row, radar_data)

# Normalize the data for radar chart
radar_data_normalized <- as.data.frame(lapply(radar_data[, -1], function(x) (x - min(x)) / (max(x) - min(x))))
radar_data_normalized <- cbind(District = radar_data$`Crime Count`, radar_data_normalized)

# Create the radar chart
radar_chart <- radarchart(radar_data_normalized, axistype = 1,
                          pcol = c("black", rep("green", nrow(radar_normalized) - 1)),
                          pfcol = c(NA, rep(rgb(0, 0, 1, 0.5), nrow(radar_data_normalized) - 1)),
                          plwd = 2)
```




```r
#-----------2022 June Robbery Rate Per 10000 people Pie Chart-----------#

#modifying our crime dataset to show robbery crime rate and crime count
crime_dataset_robbery <-cleaned_crime_dataset %>%
  group_by(`Short Postcode`,`Crime type`,`Date of crime`, `Falls within`) %>% #Grouping to show crime count in each postcode by year
  select(`Short Postcode`,`Crime type`,`Date of crime`, `Falls within`) %>%
  na.omit() %>%
  tally() %>% #creating crime count column
  rename(`Crime Count`=n) %>%  #renaming crime count column %>%
  ungroup(`Short Postcode`,`Crime type`,`Date of crime`, `Falls within`) %>%
  right_join(population_dataset, by = "Short Postcode") %>% #joining with population dataset to show district and population
  select(`Short Postcode`,`Crime type`,`Crime Count`, `Population`, `Date of crime`, `Falls within`, District) %>% #select the required columns
  na.omit() %>%
  filter(`Crime type`== "Robbery" & `Date of crime`=="2022-06") %>%   #filtering to show only vehicle crimes of 2022 June
  mutate(`Robbery Crime Rate` = (`Crime Count` / Population)*10000) %>% #calculating vehicle crime rate per 10000 people
  group_by(District) %>% #grouping by district
  summarise(TotalRobberyCrimeRate = sum(`Robbery Crime Rate`)) #aggregating crime rates by District

ggplot(crime_dataset_robbery, aes(x = "", y = TotalRobberyCrimeRate, fill = District)) + #defining x axis and y axis values
  geom_bar(width = 1, stat = "identity") +
  coord_polar("y", start = 0) +
  theme_void() +
  labs(fill = "District", title = "Robbery Crime Rate by District in June 2022") #defining labels
```

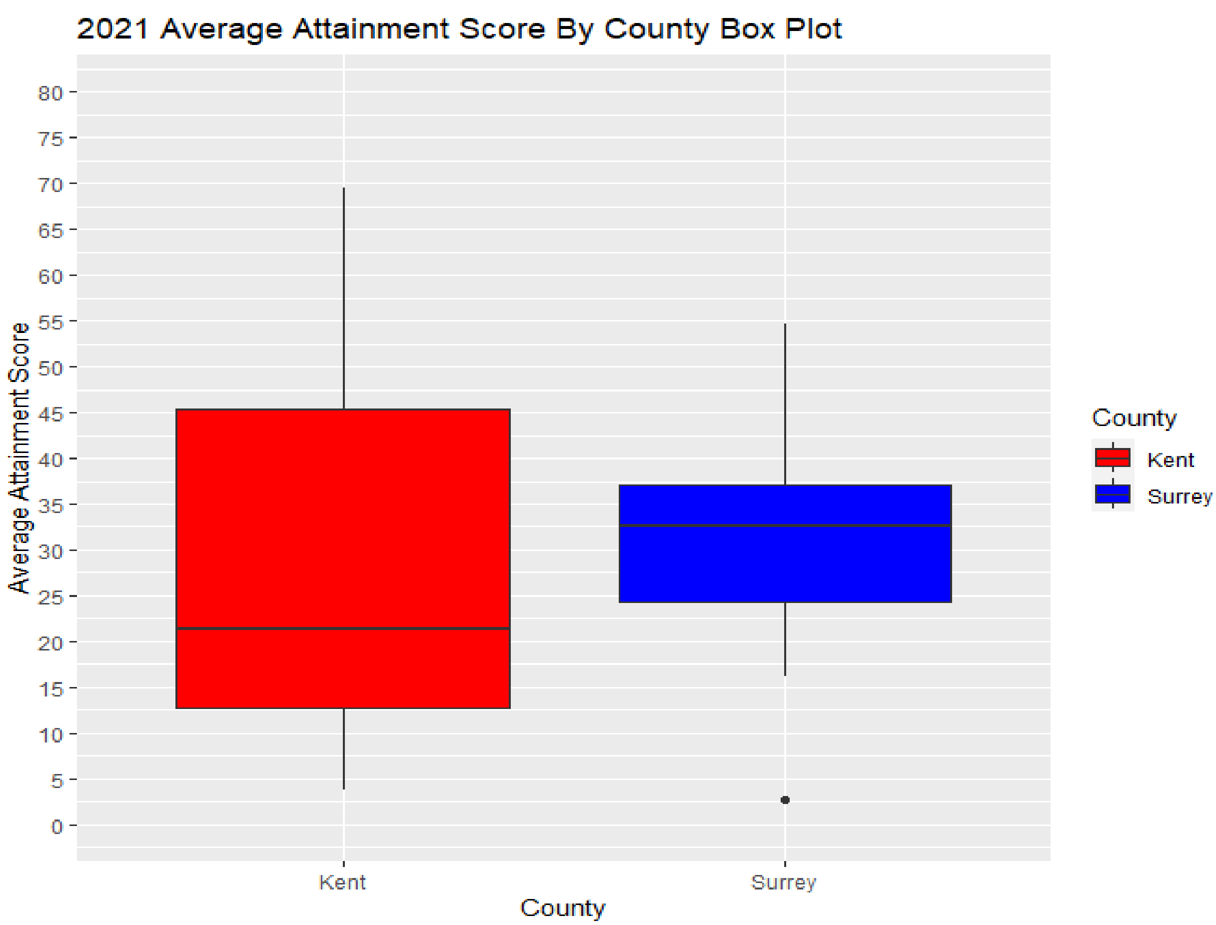


```r
#-----------2019-2022 Drug Offence Rate In Kent and Surrey Line Chart-----------#

#modifying our crime dataset to show drug offence rate and crime count
crime_dataset_drugs2 <-cleaned_crime_dataset %>%
  mutate(`Date of crime`= substr(`Date of crime`, 1, 4)) %>% #Mutating this column to only show year
  group_by(`Short Postcode`,`Crime type`,`Date of crime`, `Falls within`) %>% #Grouping to show crime count in each postcode by year
  select(`Short Postcode`,`Crime type`,`Date of crime`, `Falls within`) %>%
  na.omit() %>%
  tally() %>% #creating crime count column
  rename(`Crime Count`=n) %>%  #renaming crime count column %>%
  right_join(population_dataset, by = "Short Postcode") %>% #joining with population dataset to show district and population
  select(`Short Postcode`,`Crime type`,`Crime Count`, `Population`, `Date of crime`, `Falls within`, District) %>% #select the required columns
  na.omit() %>%
  filter(`Crime type`== "Drugs") %>% #filtering to show only drug crimes of 2022
  mutate(`Drug Offence Rate` = (`Crime Count` / Population)) #calculating drug offence rate

#grouping the drug crime dataset by county and year and showing the rate for each group
Grouped_drug_crime <- crime_dataset_drugs2%>%
  group_by(`Falls within`,`Date of crime`) %>%
  summarise(`Drug Offence Rate`= mean(`Drug Offence Rate`))

#creating line graph of average house prices from 2021-2022
Grouped_drug_crime %>%
  group_by(`Falls within`, `Date of crime`) %>%  #grouping by county and date of crime since we are comparing offence rate in counties year after year
  ggplot( aes(x = `Date of crime`, y = `Drug Offence Rate`, group = `Falls within`, color = `Falls within`)) + #defining x-axis and y-axis values and colors of line
  geom_line(linewidth = 1) + #defining line width
  geom_point(size = 2, color = "brown") + #defining point size and color
  scale_y_continuous(limits=c(0,0.2), breaks = seq(0,0.2,0.01), labels = label_number()) + #defining limit and breaks
  labs(title = "2020-2023 Drug Offence Rate", #defining labels
       x = "Year",
       y = "Drug Offence Rate") #setting labels
```

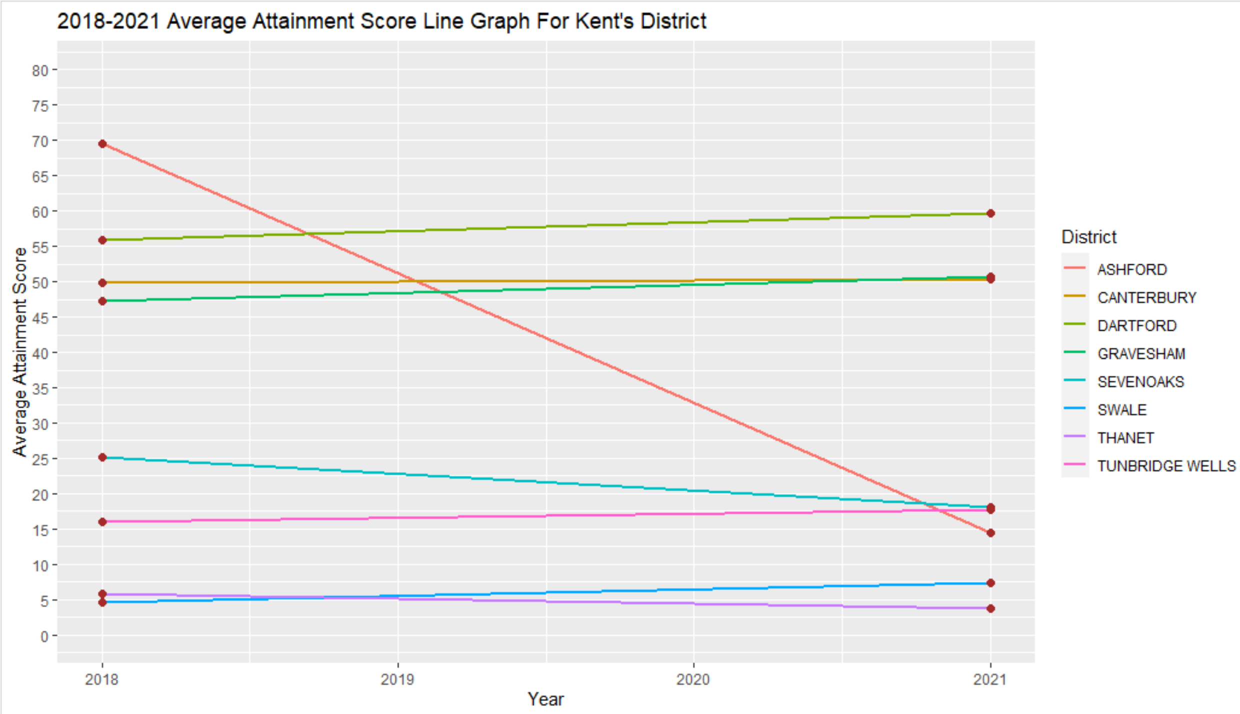


4. School Analysis

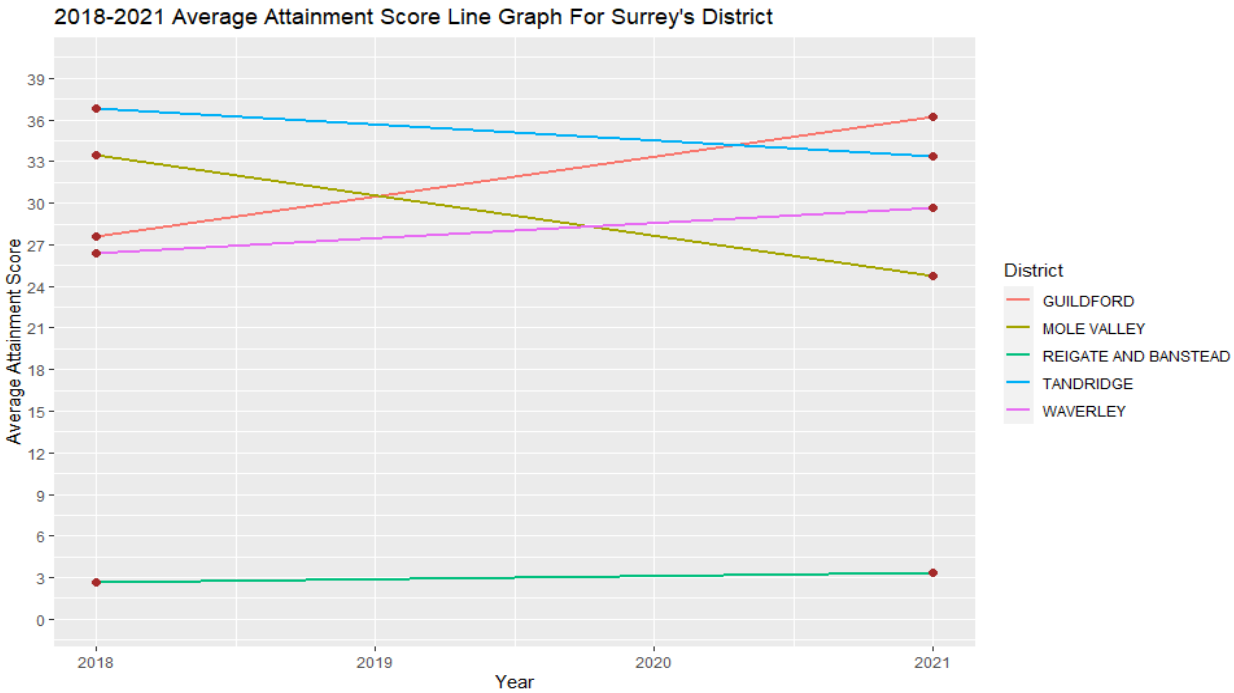
```
library(tidyverse)
library(dplyr)
library(lubridate)
library(ggplot2)
library("scales")

setwd("C:/Users/hacki/OneDrive/Desktop/Ronit_Bhujel_220050")

#importing the cleaned school dataset
cleaned_school_dataset= read_csv('Cleaned Data/Cleaned School.csv')

#Creating a new dataset consisting district and short postcode
district= read_csv('Cleaned Data/Cleaned Population.csv') %>%
  select(`Short Postcode`, District) %>%
  rename(`Short Post Code`= `Short Postcode`) #renaming to match the column name in school dataset

#Joining the district dataset into Schoo Dataset by Short Post Code
cleaned_school_dataset <- cleaned_school_dataset %>%
  left_join(district, by = "Short Post Code") %>%
  na.omit()

#-----------2021 Average Attainment Score Box plot-----------#

#grouping school dataset by town, distrct, county and year and showing avg. price for each group
Grouped_school_dataset = cleaned_school_dataset %>%
  group_by(`Town`,District,County,Year) %>%
  summarise(`Average Attainment Score`= mean(`Attainment Score`)) %>%
  ungroup(`Town`,District,County,`Year`)

#creating box plot to visualize average attainment score in kent and surrey in 2021
Grouped_school_dataset %>%
  filter(Year==2021) %>% #filtering to show only data of 2021
  group_by(County) %>% #grouping by county since we are comparing counties only
  ggplot(aes(x = County, y = `Average Attainment Score`, fill=County)) + #setting x-axis and y-axis values
  scale_y_continuous(limits=c(0,80), breaks = seq(0,80,5))+ #setting limits and breaks
  geom_boxplot() + #specifying the type of plot we need
  labs(title="2021 Average Attainment Score By County Box Plot") + #setting label for the chart
  scale_fill_manual(values = c("red","blue"))
```



```
#-----------2018-2021 Average Attainment Score Line Graph For Kent's District-----------#

#grouping the cleaned school dataset by county and year and showing the average price for each group
Grouped_school_dataset2 <- cleaned_school_dataset %>%
  filter(County=="Kent") %>% #filtering to show only rows with county as Kent
  group_by(District,Year) %>%
  summarise(`Average Attainment Score`= mean(`Attainment Score`))

#creating line graph of average Attainment score from 2018-2021
Grouped_school_dataset2 %>%
  group_by(District, Year) %>%  #grouping by District and year since we are comparing average score of districts, year after year
  ggplot( aes(x = `Year`, y = `Average Attainment Score`, group = District, color = District)) + #defining x-axis and y-axis values and colors of line
  geom_line(linewidth = 1) + #defining line width
  geom_point(size = 2, color = "brown") + #defining point size and color
  scale_y_continuous(limits=c(0,80), breaks = seq(0,80,5)) + #defining limits, breaks
  labs(title = "2018-2021 Average Attainment Score Line Graph For Kent's District", #defining labels
       x = "Year",
       y = "Average Attainment Score")
```



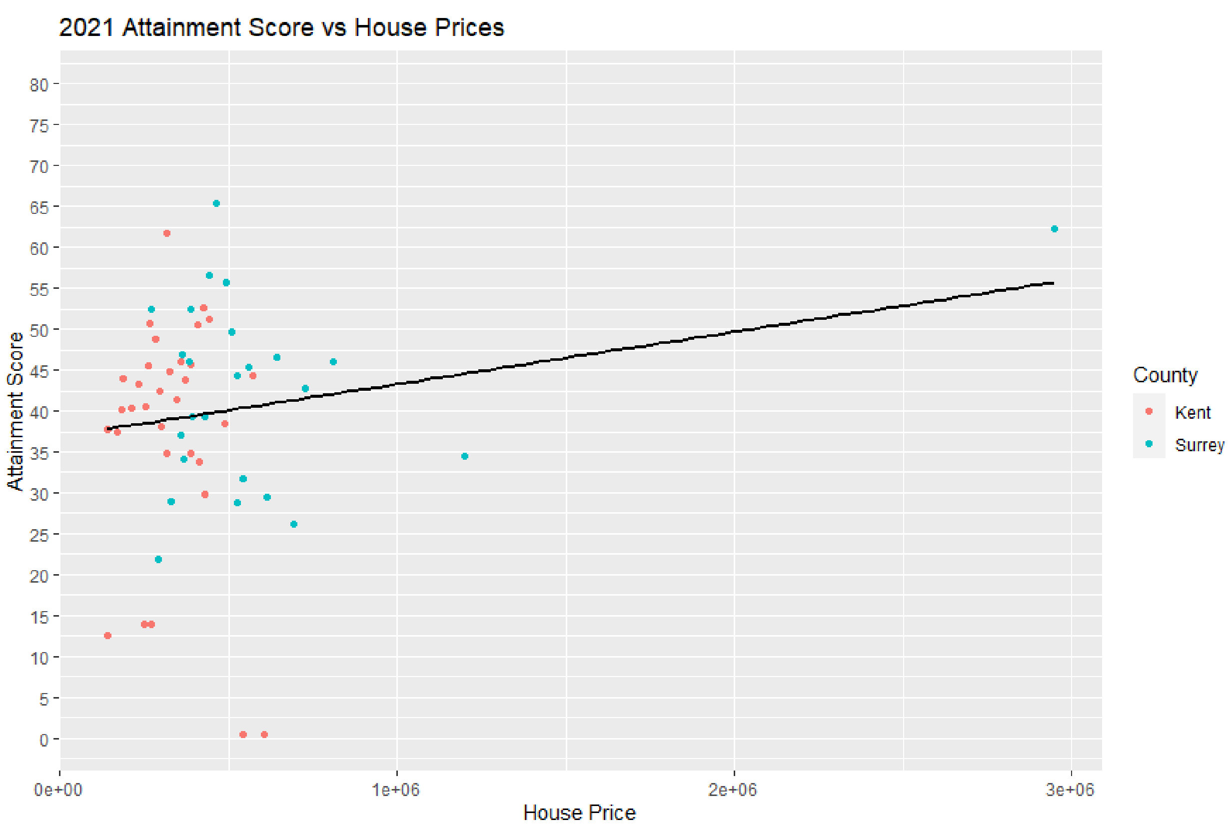

```
#-----------2018-2021 Average Attainment Score Line Graph For Surrey's District-----------#

#grouping the cleaned school dataset by county and year and showing the average price for each group
Grouped_school_dataset3 <- cleaned_school_dataset %>%
  filter(County=="Surrey") %>% #filtering to show only rows with county as Surrey
  group_by(District,Year) %>%
  summarise(`Average Attainment Score`= mean(`Attainment Score`))

#creating line graph of average Attainment score from 2018-2021
Grouped_school_dataset3 %>%
  group_by(District, Year) %>%  #grouping by District and year since we are comparing average score of districts, year after year
  ggplot( aes(x = `Year`, y = `Average Attainment Score`, group = District, color = District)) + #defining x-axis and y-axis values and colors of line
  geom_line(linewidth = 1) + #defining line width
  geom_point(size = 2, color = "brown") + #defining point size and color
  scale_y_continuous(limits=c(0,40), breaks = seq(0,40,3)) + #defining limits, breaks
  labs(title = "2018-2021 Average Attainment Score Line Graph For Surrey's District",
       x = "Year",
       y = "Average Attainment Score") #defining labels
```


2018-2021 Average Attainment Score Line Graph For Surrey's District
Average Attainment Score
0
3
6
9
12
15
18
21
24
27
30
33
36
39
Year
2018
2019
2020
2021
District
GUILDFORD
MOLE VALLEY
REIGATE AND BANSTEAD
TANDRIDGE
WAVERLEY

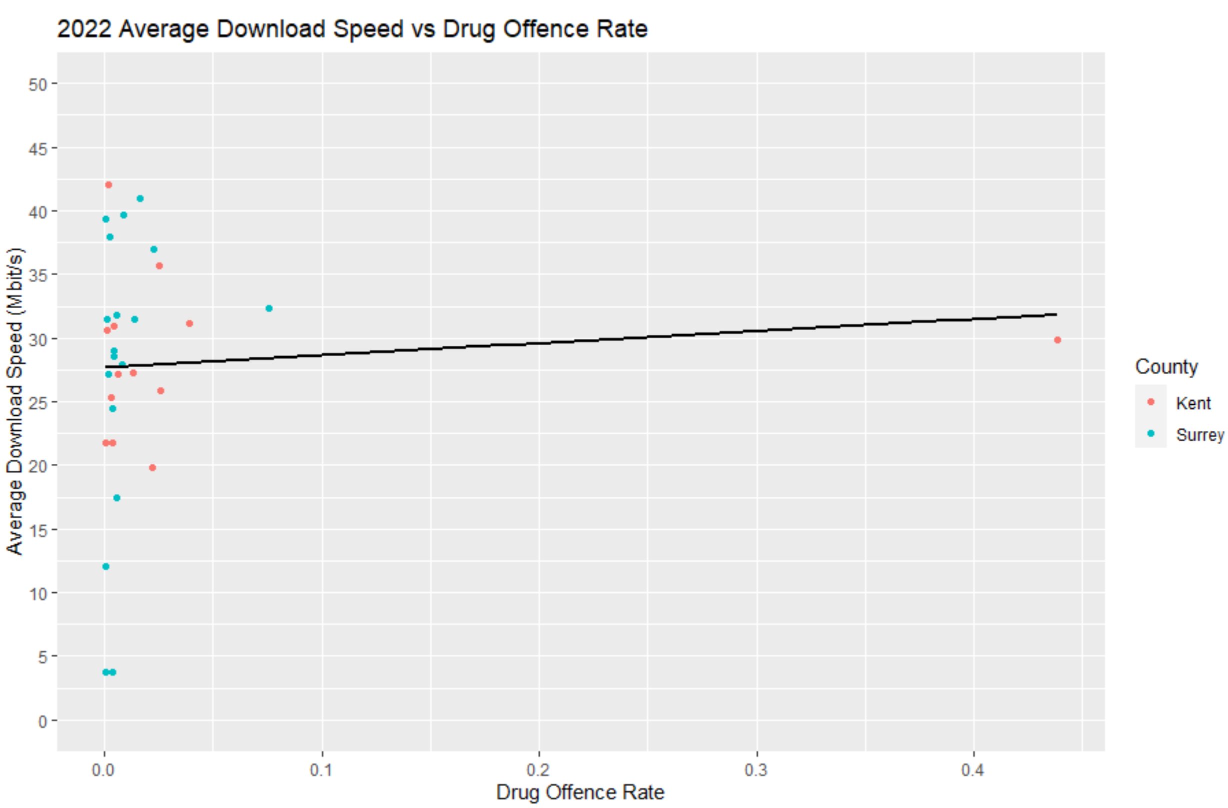
Linear Modeling

We apply Linear Modeling approaches in the development of our Town Recommender System to quantify the correlations between different town attributes and user preferences. Regression analysis allows us to determine how many parameters, such as internet speed, cost of living, safety ratings, and quality of education, affect overall desirability. This helps us build a prediction model that can be continuously improved upon as new data becomes available, in addition to offering tailored recommendations. Through the use of Linear Modeling, we hope to improve the accuracy and dependability of our system and provide overseas students with a useful resource for choosing their study exchange towns in Kent and Surrey. A robust solution for students looking for the best study exchange experiences in Kent and Surrey is offered by the integration of linear modeling, which not only makes it easier to identify important influencers but also guarantees a dynamic and adaptive recommendation system that changes with changing preferences and town dynamics.

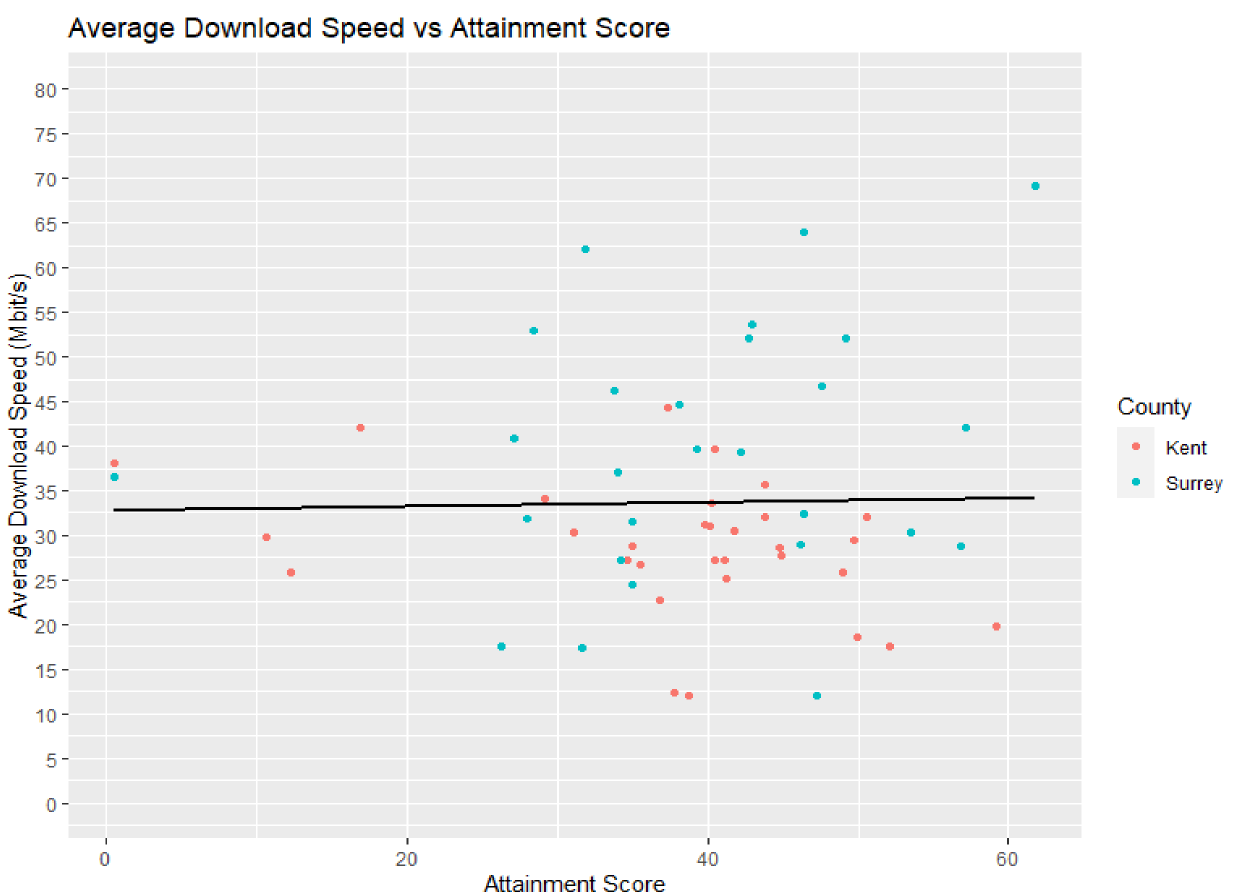
```
library(tidyverse)
library(dplyr)
library(lubridate)
library(ggplot2)
setwd("C:/Users/hacki/OneDrive/Desktop/Ronit_Bhujel_220050")

#importing the cleaned house prices
cleaned_houseprices= read_csv('Cleaned Data/Cleaned House Prices.csv')

#importing the cleaned school dataset
cleaned_school_dataset= read_csv('Cleaned Data/Cleaned School.csv')

#grouping house prices by town and county and finding average price for each group
grouped_house_prices = cleaned_houseprices %>%
  filter(`Date of Transfer`=="2021") %>%
  group_by(`Town/City`,County) %>%
  mutate(`Town/City` = tolower(`Town/City`)) %>% #converting the town from uppercase to all lowercase
  summarise(Price=mean(Price))

#grouping school data by town and county and finding average score for each group
grouped_school_dataset = cleaned_school_dataset %>%
  filter(`Year`=="2021") %>%
  group_by(`Town`,County) %>%
  mutate(Town= tolower(Town)) %>%  #converting the town from to all lowercase
  summarise(`Attainment Score`=mean(`Attainment Score`))

#joining school data and house price data in a single table
school_houseprice_data = grouped_school_dataset %>%
  left_join(grouped_house_prices,by=c("Town"="Town/City")) %>%
  na.omit #removing rows with null value

#creating a linear model
l_model = lm(data=school_houseprice_data, `Attainment Score`~Price) #this model predicts Average attainment score as a function of Average house prices

#showing summary of the Linear Model
summary(l_model)

#creating the linear model graph
ggplot(school_houseprice_data,aes(x=Price,y= `Attainment Score`)) +
  scale_y_continuous(limits=c(0,80), breaks = seq(0,80,5))+ #setting limits and breaks
  geom_point(data = filter(school_houseprice_data,County.x=="Kent"),aes(color=c("Red"="Kent")))+ #setting color as red for Kent's data point
  geom_point(data = filter(school_houseprice_data,County.x=="Surrey"), aes(color=c("Blue"="Surrey"))) + #setting color as blue for Surrey's data point
  geom_smooth(method=lm,se=FALSE,color="black")+ #adding linear regression line and omitting error bands
  labs(x="House Price",
       y="Attainment Score",
       title="2021 Attainment Score vs House Prices",color="County") #setting labels
```



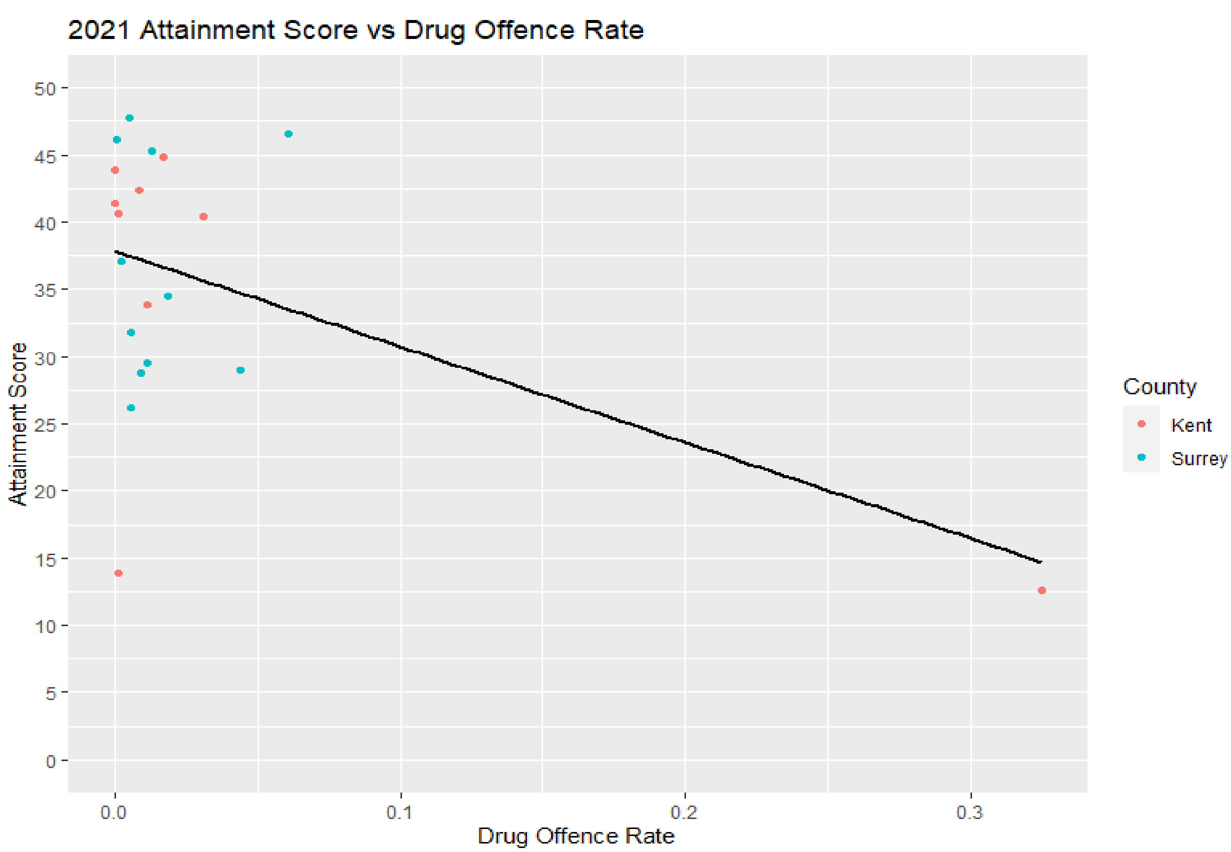
```r
#importing population dataset
population_dataset<- read_csv('Cleaned Data/Cleaned Population.csv')

#importing the cleaned crime dataset
cleaned_crime_dataset= read_csv('Cleaned Data/Cleaned Crime.csv')

#importing the cleaned broadband speed
cleaned_broadband_speed= read_csv('Cleaned Data/Cleaned Broadband Speed.csv')

#grouping broadband speed by town and county and finding average download speed for each group
grouped_broadband_speeds = cleaned_broadband_speed %>%
  group_by(`Town/City`,County) %>%
  summarise(`Average download speed (Mbit/s)`= mean(`Average download speed (Mbit/s)`))

#modifying our crime dataset to show drug offence rate and crime count
crime_dataset_drugs2 <-cleaned_crime_dataset %>%
  mutate(`Date of crime`= substr(`Date of crime`, 1, 4)) %>% #Mutating this column to only show year
  group_by(`Short Postcode`,`Crime type`,`Date of crime`, `Falls within`) %>% #Grouping to show crime count in each postcode by year
  select(`Short Postcode`,`Crime type`,`Date of crime`, `Falls within`) %>%
  na.omit() %>%
  tally() %>% #creating crime count column
  rename(`Crime Count`=n) %>%  #renaming crime count column %>%
  right_join(population_dataset, by = "Short Postcode") %>% #joining with population dataset to show district and population
  select(`Short Postcode`,`Crime type`,`Crime Count`, `Population`, `Date of crime`, `Falls within`, `Town/City`, District) %>% #select the required columns
  na.omit() %>%
  filter(`Crime type`== "Drugs") %>% #filtering to show only drug crimes of 2022
  mutate(`Drug Offence Rate` = (`Crime Count` / Population)) #calculating drug offence rate

#grouping the drug crime dataset by county and town and showing the rate for each group for the year 2022
grouped_drug_crime <- crime_dataset_drugs2 %>%
  filter(`Date of crime`=="2022") %>%
  group_by(`Falls within`,`Town/City`) %>%
  summarise(`Drug Offence Rate`= mean(`Drug Offence Rate`))

#joining broadband data and drug crime rate data in a single table
broadband_crime_data = grouped_broadband_speeds %>%
  left_join(grouped_drug_crime,by="Town/City") %>%
  na.omit #removing null values

#creating a linear model
l_model = lm(data=broadband_crime_data, `Average download speed (Mbit/s)`~`Drug Offence Rate`) #this model predicts Average download speed as a function of Drug offence rate

#showing summary of the Linear Model
summary(l_model)

#creating the linear model graph
ggplot(broadband_crime_data,aes(x=`Drug Offence Rate`,y=`Average download speed (Mbit/s)`)) +
  scale_y_continuous(limits=c(0,50), breaks = seq(0,50,5))+ #setting limits and breaks
  geom_point(data = filter(broadband_crime_data,County=="KENT"),aes(color=c("Red"="Kent")))+ #setting color as red for Kent's data point
  geom_point(data = filter(broadband_crime_data,County=="SURREY"), aes(color=c("Blue"="Surrey"))) + #setting color as blue for Surrey's data point
  geom_smooth(method=lm,se=FALSE,color="black")+ #adding linear regression line and omitting error bands
  labs(x="Drug Offence Rate",
       y="Average Download Speed (Mbit/s)",
       title="2022 Average Download Speed vs Drug Offence Rate",color="County") #setting labels
```



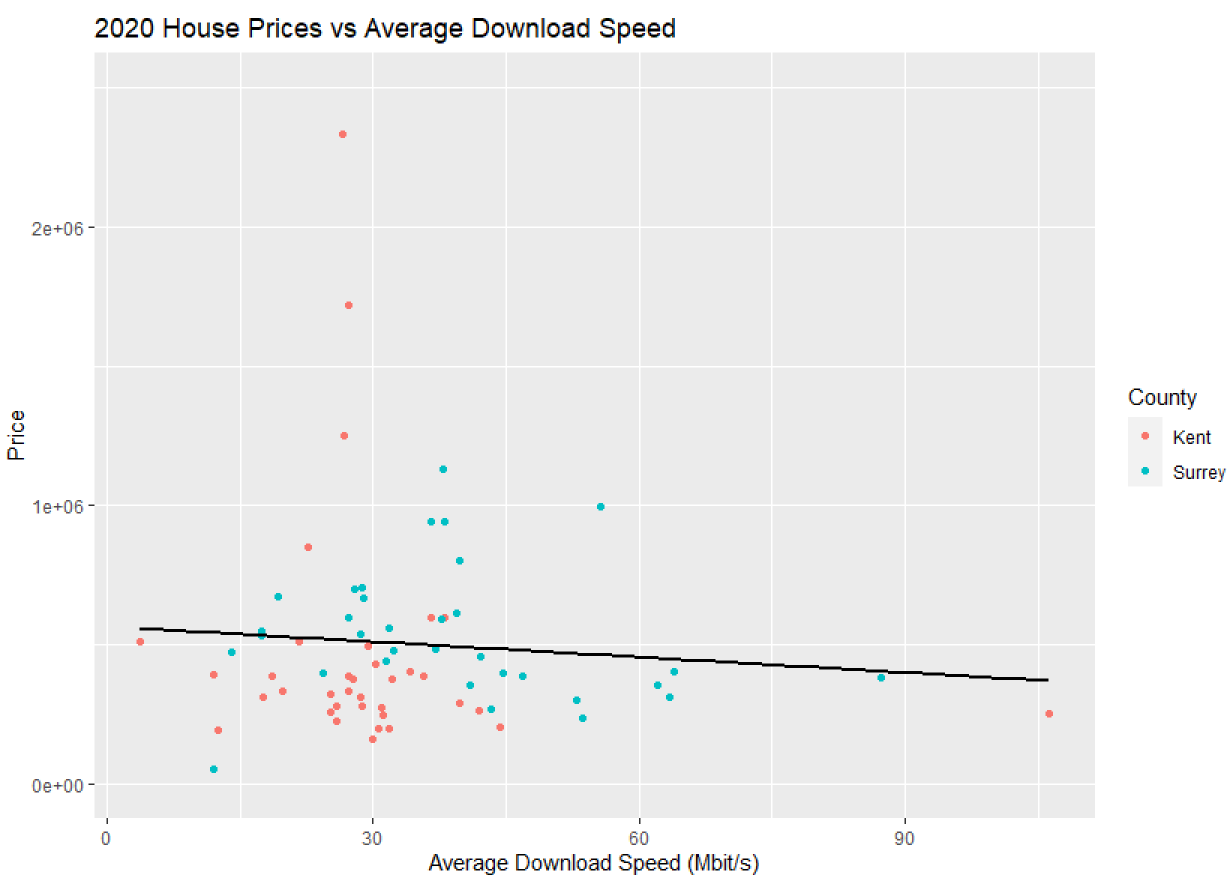
```r
library(tidyverse)
library(dplyr)
library(lubridate)
library(ggplot2)

setwd("C:/Users/hacki/OneDrive/Desktop/Ronit_Bhujel_220050")

#importing the cleaned broadband speed
cleaned_broadband_speed= read_csv('Cleaned Data/Cleaned Broadband Speed.csv')

#importing the cleaned school dataset
cleaned_school_dataset= read_csv('Cleaned Data/Cleaned School.csv')

#grouping broadband speed by town and county and finding average download speed for each group
grouped_broadband_speeds = cleaned_broadband_speed %>%
  group_by(`Town/City`,County) %>%
  mutate(`Town/City`= tolower(`Town/City`)) %>%  #converting the town from to all lowercase
  summarise(`Average download speed (Mbit/s)`= mean(`Average download speed (Mbit/s)`))

#grouping school data by town and county and finding average score for each group
grouped_school_dataset = cleaned_school_dataset %>%
  group_by(`Town`,County) %>%
  mutate(Town= tolower(Town)) %>%  #converting the town from to all lowercase
  summarise(`Attainment Score`=mean(`Attainment Score`))

#joining broadband data and school data in a single table
broadband_attainment_data = grouped_broadband_speeds %>%
  left_join(grouped_school_dataset,by=c("Town/City"="Town")) %>%
  na.omit #removing rows with null value

#creating a linear model
l_model = lm(data=broadband_attainment_data, `Average download speed (Mbit/s)`~`Attainment Score`) #this model predicts Average download speed as a function of Drug offence rate

#showing summary of the Linear Model
summary(l_model)

#creating the linear model graph
ggplot(broadband_attainment_data,aes(x=`Attainment Score`,y=`Average download speed (Mbit/s)`)) +
  scale_y_continuous(limits=c(0,80), breaks = seq(0,80,5))+ #setting limits and breaks
  geom_point(data = filter(broadband_attainment_data,County.x=="KENT"),aes(color=c("Red"="Kent")))+ #setting color as red for Kent's data point
  geom_point(data = filter(broadband_attainment_data,County.x=="SURREY"), aes(color=c("Blue"="Surrey"))) + #setting color as blue for Surrey's data point
  geom_smooth(method=lm,se=FALSE,color="black")+ #adding linear regression line and omitting error bands
  labs(x="Attainment Score",
       y="Average Download Speed (Mbit/s)",
       title="Average Download Speed vs Attainment Score",color="County") #setting labels
```



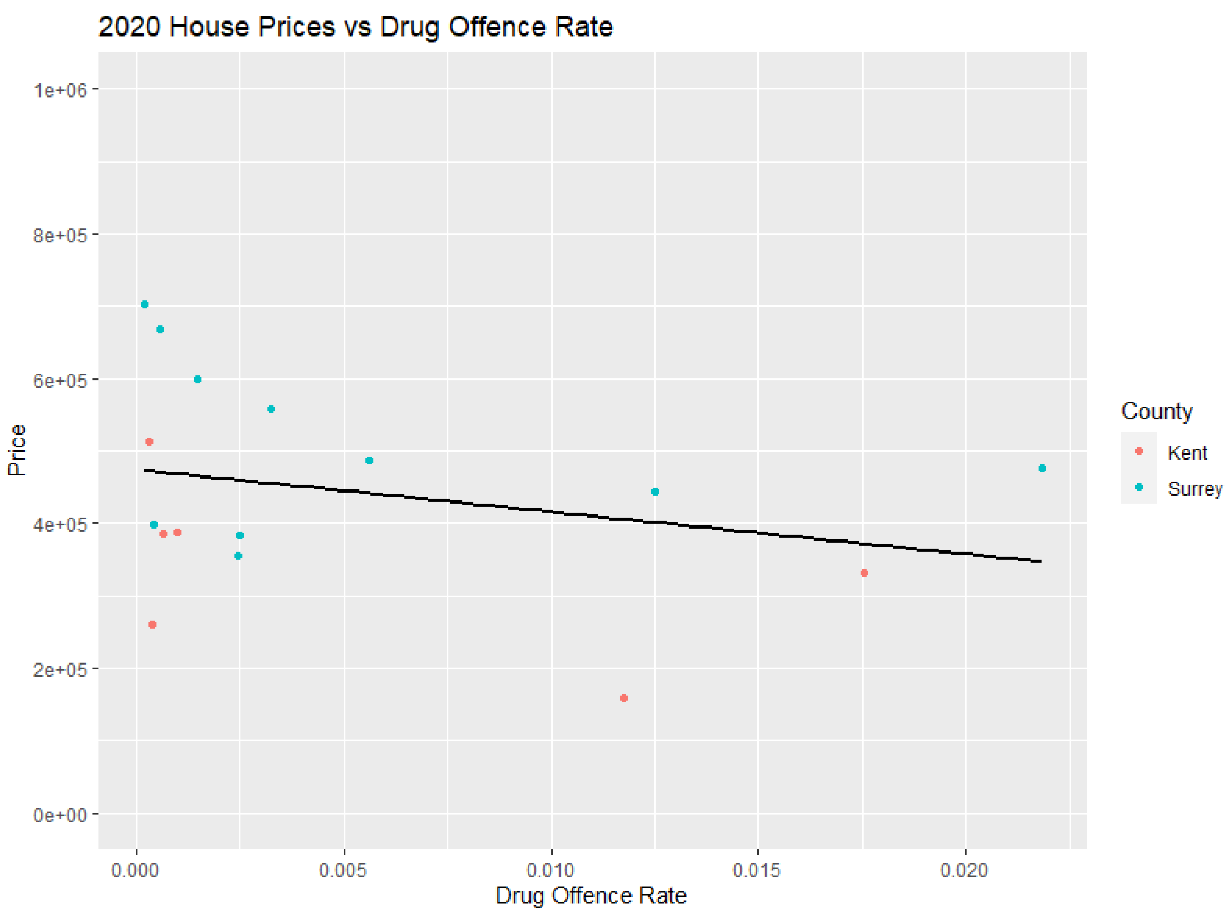
```r
#importing population dataset
population_dataset<- read_csv('Cleaned Data/Cleaned Population.csv')

#importing the cleaned school dataset
cleaned_school_dataset= read_csv('Cleaned Data/Cleaned School.csv')

#importing the cleaned crime dataset
cleaned_crime_dataset= read_csv('Cleaned Data/Cleaned Crime.csv')

#grouping school data by town and county and finding average score for each group
grouped_school_dataset = cleaned_school_dataset %>%
  filter(`Year`=="2021") %>%
  group_by(`Town`,County) %>%
  mutate(Town= tolower(Town)) %>%  #converting the town from to all lowercase
  summarise(`Attainment Score`=mean(`Attainment Score`))

#modifying our crime dataset to show drug offence rate and crime count
crime_dataset_drugs2 <-cleaned_crime_dataset %>%
  mutate(`Date of crime`= substr(`Date of crime`, 1, 4)) %>% #Mutating this column to only show year
  group_by(`Short Postcode`,`Crime type`,`Date of crime`, `Falls within`) %>% #Grouping to show crime count in each postcode by year
  select(`Short Postcode`,`Crime type`,`Date of crime`, `Falls within`) %>%
  na.omit() %>%
  tally() %>% #creating crime count column
  rename(`Crime Count`=n) %>%  #renaming crime count column %>%
  right_join(population_dataset, by = "Short Postcode") %>% #joining with population dataset to show district and population
  select(`Short Postcode`,`Crime type`,`Crime Count`, `Population`, `Date of crime`, `Falls within`, `Town/City`, District) %>% #select the required columns
  na.omit() %>%
  filter(`Crime type`== "Drugs") %>% #filtering to show only drug crimes of 2022
  mutate(`Drug Offence Rate` = (`Crime Count` / Population)) #calculating drug offence rate

#grouping the drug crime dataset by county and town and showing the rate for each group for the year 2021
grouped_drug_crime <- crime_dataset_drugs2 %>%
  filter(`Date of crime`=="2021") %>%
  group_by(`Falls within`,`Town/City`) %>%
  mutate(`Town/City`= tolower(`Town/City`)) %>%  #converting the town from to all lowercase
  summarise(`Drug Offence Rate`= mean(`Drug Offence Rate`))
#joining school data and house price data in a single table
school_drug_data = grouped_school_dataset %>%
  left_join(grouped_drug_crime ,by=c("Town"="Town/City")) %>%
  na.omit #removing rows with null value

#creating a linear model
l_model = lm(data=school_drug_data, `Attainment Score`~`Drug Offence Rate`) #this model predicts Average attainment score as a function of Drug offence rate

#showing summary of the Linear Model
summary(l_model)

#creating the linear model graph
ggplot(school_drug_data,aes(x=`Drug Offence Rate`,y= `Attainment Score`)) +
  scale_y_continuous(limits=c(0,50), breaks = seq(0,50,5))+ #setting limits and breaks
  geom_point(data = filter(school_drug_data,County=="Kent"),aes(color=c("Red"="Kent")))+ #setting color as red for Kent's data point
  geom_point(data = filter(school_drug_data,County=="Surrey"), aes(color=c("Blue"="Surrey"))) + #setting color as blue for Surrey's data point
  geom_smooth(method=lm,se=FALSE,color="black")+ #adding linear regression line and omitting error bands
  labs(x="Drug Offence Rate",
       y="Attainment Score",
       title="2021 Attainment Score vs Drug Offence Rate",color="County") #setting labels
```



```r
library(tidyverse)
library(dplyr)
library(lubridate)
library(ggplot2)

setwd("C:/Users/hacki/OneDrive/Desktop/Ronit_Bhujel_220050")

#importing the cleaned house prices
cleaned_houseprices= read_csv('Cleaned Data/Cleaned House Prices.csv')

#importing the cleaned broadband speed
cleaned_broadband_speed= read_csv('Cleaned Data/Cleaned Broadband Speed.csv')

#grouping house prices by town and county and finding average price for each group
grouped_house_prices = cleaned_houseprices %>%
  filter(`Date of Transfer`=="2020") %>%
  group_by(`Town/City`,County) %>%
  summarise(Price=mean(Price))

#grouping broadband speed by town and county and finding average download speed for each group
grouped_broadband_speeds = cleaned_broadband_speed %>%
  group_by(`Town/City`,County) %>%
  summarise(`Average download speed (Mbit/s)`= mean(`Average download speed (Mbit/s)`))

#joining house price data and broadband speed data in a single table
house_price_broadband_data = grouped_house_prices %>%
  left_join(grouped_broadband_speeds,by="Town/City")

#creating a linear model
l_model = lm(data=house_price_broadband_data, Price~`Average download speed (Mbit/s)`) #this model predicts Price as a function of Average download speed (Mbit/s)

#showing summary of the Linear Model
summary(l_model)

#creating the linear model graph
ggplot(house_price_broadband_data,aes(x=`Average download speed (Mbit/s)`,y=Price)) +
  scale_y_continuous(limits=c(0,2500000), breaks = seq(0,2500000,1000000))+ #setting limits and breaks
  geom_point(data = filter(house_price_broadband_data,County.x=="KENT"),aes(color=c("Red"="Kent")))+ #setting color as red for Kent's data point
  geom_point(data = filter(house_price_broadband_data,County.x=="SURREY"), aes(color=c("Blue"="Surrey"))) + #setting color as blue for Surrey's data point
  geom_smooth(method=lm,se=FALSE,color="black")+ #adding linear regression line and omitting error bands
  labs(x="Average Download Speed (Mbit/s)",
       y="Price",
       title="2020 House Prices vs Average Download Speed",color="County") #setting labels
```



```
setwd("C:/Users/hacki/OneDrive/Desktop/Ronit_Bhujel_220050")

#importing population dataset
population_dataset<- read_csv('Cleaned Data/Cleaned Population.csv')
#importing the cleaned house prices
cleaned_houseprices= read_csv('Cleaned Data/Cleaned House Prices.csv')
#importing the cleaned crime dataset
cleaned_crime_dataset= read_csv('Cleaned Data/Cleaned Crime.csv')
#grouping house prices by town and county and finding average price for each group
grouped_house_prices = cleaned_houseprices %>%
  filter(`Date of Transfer`=="2020") %>%
  group_by(`Town/City`,County) %>%
  summarise(Price=mean(Price))

#modifying our crime dataset to show drug offence rate and crime count
crime_dataset_drugs2 <-cleaned_crime_dataset %>%
  mutate(`Date of crime`= substr(`Date of crime`, 1, 4)) %>% #Mutating this column to only show year
  group_by(`Short Postcode`,`Crime type`,`Date of crime`, `Falls within`) %>% #Grouping to show crime count in each postcode by year
  select(`Short Postcode`,`Crime type`,`Date of crime`, `Falls within`) %>%
  na.omit() %>%
  tally() %>% #creating crime count column
  rename(`Crime Count`=n) %>%  #renaming crime count column %>%
  right_join(population_dataset, by = "Short Postcode") %>% #joining with population dataset to show district and population
  select(`Short Postcode`,`Crime type`,`Crime Count`, `Population`, `Date of crime`, `Falls within`, `Town/City`, District) %>% #select the required columns
  na.omit() %>%
  filter(`Crime type`== "Drugs") %>% #filtering to show only drug crimes of 2022
  mutate(`Drug Offence Rate` = (`Crime Count` / Population)) #calculating drug offence rate
#grouping the drug crime dataset by county and town and showing the rate for each group for the year 2020
grouped_drug_crime <- crime_dataset_drugs2 %>%
  filter(`Date of crime`=="2020") %>%
  group_by(`Falls within`,`Town/City`) %>%
  summarise(`Drug Offence Rate`= mean(`Drug Offence Rate`))

#joining house price data and drug crime rate data in a single table
house_price_drug_crime_data = grouped_house_prices %>%
  left_join(grouped_drug_crime,by="Town/City") %>%
  na.omit #removing null values
#creating a linear model
l_model = lm(data=house_price_drug_crime_data, Price~`Drug Offence Rate`) #this model predicts House Price as a function of Drug offence rate
#showing summary of the Linear Model
summary(l_model)

#creating the linear model graph
ggplot(house_price_drug_crime_data,aes(x=`Drug Offence Rate`,y=Price)) +
  scale_y_continuous(limits=c(0,1000000), breaks = seq(0,1000000,200000))+ #setting limits and breaks
  geom_point(data = filter(house_price_drug_crime_data,County=="KENT"),aes(color=c("Red"="Kent")))+ #setting color as red for Kent's data point
  geom_point(data = filter(house_price_drug_crime_data,County=="SURREY"), aes(color=c("Blue"="Surrey"))) + #setting color as blue for Surrey's data point
  geom_smooth(method=lm,se=FALSE,color="black")+ #adding linear regression line and omitting error bands
  labs(x="Drug Offence Rate",
       y="Price",
       title="2020 House Prices vs Drug Offence Rate",color="County") #setting labels
```



Recommended System

These recommendations are based on the score.

```r
library(tidyverse)
library(dplyr)

setwd("C:/Users/hacki/OneDrive/Desktop/Ronit_Bhujel_220050")

#-----------House Price Ranking-----------#

#Importing cleaned population data
cleaned_population_data = read_csv("Cleaned Data/Cleaned Population.csv")

#Importing cleaned house price data
cleaned_houseprices = read_csv("Cleaned Data/Cleaned House Prices.csv")

#Creating a new house rank table
houseprice_rank= cleaned_houseprices %>%
  filter(`Date of Transfer`=="2020") %>%
  group_by(`Town/City`) %>%
  summarise(Price=mean(Price),County=first(County)) %>% #reducing the table by merging multiple same towns that belong to the same county
  arrange(Price) %>% #arranging price in ascending order
  mutate(`House Score`=10-(Price/100000)) %>%  #calculating score. We are subtracting from 10 because lower house prices need to have higher rank
  select(`Town/City`,County, `House Score`)

#defining path to save the house ranking csv
file_path <- "Recommended System/House Pricing Ranks.csv"

#saving the house ranking csv
write.csv(houseprice_rank, file_path, row.names = FALSE)
view(houseprice_rank)
```

	Town/City	County	House Score
1	WARLINGHAM	SURREY	9.45000000
2	MARGATE	KENT	8.41593280
3	NEW ROMNEY	KENT	8.05843750
4	ROMNEY MARSH	KENT	8.02142857
5	WHITSTABLE	KENT	8.00500000
6	AYLESFORD	KENT	7.94944444
7	DEAL	KENT	7.74549237
8	CAMBERLEY	SURREY	7.63759296
9	BIRCHINGTON	KENT	7.54500000
10	SWANSCOMBE	KENT	7.45460750
11	WEST MALLING	KENT	7.43141833
12	LONGFIELD	KENT	7.38800000
13	WEST BYFLEET	SURREY	7.29576923
14	SWANLEY	KENT	7.27285714
15	DOVER	KENT	7.21980000
16	ROCHESTER	KENT	7.19192632
17	FOLKESTONE	KENT	7.09256480
18	WALTON-ON-THAMES	SURREY	7.00515208
19	SANDWICH	KENT	6.88833333
20	GRAVESEND	KENT	6.87293000
21	ALDERSHOT	SURREY	6.86883333
22	GILLINGHAM	KENT	6.78200000
23	DARTFORD	KENT	6.68405769
24	TENTERDEN	KENT	6.65000000

```r
#-----------Download Speed Ranking-----------#

#importing the cleaned broadband speed
cleaned_broadband_speed= read_csv('Cleaned Data/Cleaned Broadband Speed.csv')

#Creating a new download speed rank table
download_speed_rank <- cleaned_broadband_speed %>%
  group_by(`Town/City`) %>%
  rename(Town=`Town/City`) %>% #renaming to maintain consistency
  summarise(`Average download speed (Mbit/s)`=`Average download speed (Mbit/s)`,County=first(County)) %>%
  arrange(desc(`Average download speed (Mbit/s)`)) %>% #arranging download speed in descending order
  mutate(`Download Score`= (`Average download speed (Mbit/s)`/100)) %>% #calculating score
  select(Town, County, `Download Score`) %>%
  distinct(Town, .keep_all = TRUE) #keeping .keep_all as true because we want to preserve other columns

#defining path to save the download speed speed ranking csv
file_path <- "Recommended System/Broadband speed rank.csv"
view(download_speed_rank)

#saving the download speed ranking csv
write.csv(download_speed_rank, file_path, row.names = FALSE)
```

	Town	County	Download Score
1	TONBRIDGE	KENT	6.037
2	SEVENOAKS	KENT	4.137
3	ADDLESTONE	SURREY	1.368
4	WOKING	SURREY	1.311
5	ALDERSHOT	SURREY	1.307
6	EGHAM	SURREY	1.146
7	WEST BYFLEET	SURREY	1.124
8	SURBITON	SURREY	1.098
9	FARNHAM	SURREY	1.097
10	CHATHAM	KENT	1.063
11	REDHILL	SURREY	1.063
12	SWANSCOMBE	KENT	1.063
13	ESHER	SURREY	1.049
14	CAMBERLEY	SURREY	1.036
15	EPSOM	SURREY	1.023
16	GRAVESEND	KENT	1.007
17	LEATHERHEAD	SURREY	1.005
18	COULSDON	SURREY	0.986
19	ASHFORD	KENT	0.983
20	SITTINGBOURNE	KENT	0.966
21	WALTON-ON-THAMES	SURREY	0.917
22	WEST MALLING	KENT	0.880
23	FOLKESTONE	KENT	0.853
24	GUILDFORD	SURREY	0.836

```r
#-----------Crime Ranking-----------#

#importing cleaned crime dataset
cleaned_crime_data =read_csv('Cleaned Data/Cleaned Crime.csv')

#importing cleaned population dataset
population_dataset<- read_csv('Cleaned Data/Cleaned Population.csv')

crime_dataset_count <-cleaned_crime_data %>%
  mutate(`Date of crime`= substr(`Date of crime`, 1, 4)) %>% #Mutating this column to only show year
  group_by(`Short Postcode`,`Crime type`,`Date of crime`, `Falls within`) %>% #Grouping to show crime count in each postcode by year
  select(`Short Postcode`,`Crime type`,`Date of crime`, `Falls within`) %>%
  na.omit() %>%
  tally() %>% #creating crime count column
  rename(`Crime Count`=n) %>%  #renaming crime count column %>%
  left_join(population_dataset, by = "Short Postcode") %>% #joining with population dataset to show district and population
  select(`Short Postcode`,`Crime Count`, `Town/City`, County) %>% #select the required columns
  na.omit()

crime_rank <-crime_dataset_count %>%
  rename(Town=`Town/City`) %>% #renaming to maintain consistency
  group_by(Town) %>%
  summarise(`Mean Crime`= mean(`Crime Count`),County=first(County)) %>%
  arrange((`Mean Crime`)) %>%  #arranging mean crime in ascending order
  mutate(`Crime Score`=10-(`Mean Crime`/1000)) %>% #calculating score. We are subtracting from 10 because lower mean crime need to have higher rank
  select(Town,County,`Crime Score`)

#defining path to save the crime rank csv
file_path <- "Recommended System/Crime rank.csv"

#saving the download crime rank csv
write.csv(crime_rank, file_path, row.names = FALSE)
View(crime_rank)
```

	Town	County	Crime Score
1	HERNE BAY	KENT	9.991629
2	ORPINGTON	KENT	9.989194
3	WADHURST	KENT	9.986774
4	ASCOT	SURREY	9.984820
5	FARNHAM	SURREY	9.968290
6	CHATHAM	KENT	9.966211
7	WHITSTABLE	KENT	9.965984
8	WARLINGHAM	SURREY	9.964088
9	REIGATE	SURREY	9.960200
10	BETCHWORTH	SURREY	9.954914
11	LONGFIELD	KENT	9.939113
12	CATERHAM	SURREY	9.911858
13	GILLINGHAM	KENT	9.896881
14	EDENBRIDGE	KENT	9.884104
15	CRANLEIGH	SURREY	9.878044
16	OXTED	SURREY	9.876051
17	SWANLEY	KENT	9.852904
18	SURBITON	SURREY	9.825114
19	LINGFIELD	SURREY	9.821340
20	GODSTONE	SURREY	9.805205
21	GUILDFORD	SURREY	9.790691
22	DARTFORD	KENT	9.772098
23	DORKING	SURREY	9.767942
24	CANTERBURY	KENT	9.761976

```r
#-----------School Ranking-----------#

#importing the cleaned school dataset
cleaned_school_dataset <-read_csv('Cleaned Data/Cleaned School.csv')

school_rank <-cleaned_school_dataset %>%
  mutate(Town= toupper(Town), County= toupper(County)) %>% #converting into all upper case for consistency
  group_by(Town) %>%
  mutate(`Mean Attainment`=mean(`Attainment Score`),County=first(County)) %>%
  arrange(desc(`Mean Attainment`)) %>% #arranging in descending order
  mutate(`School Score`= (`Mean Attainment`/10)) %>%
  select(Town, County, `School Score`) %>%
  distinct()

#defining path to save school rank csv
file_path <- "Recommended System/School rank.csv"

#saving the school rank csv
write.csv(school_rank, file_path, row.names = FALSE)
view(school_rank)
```

	Town	County	School Score
1	WEYBRIDGE	SURREY	6.180000
2	DARTFORD	KENT	5.921429
3	ESHER	SURREY	5.712000
4	CHERTSEY	SURREY	5.683333
5	HASLEMERE	SURREY	5.350000
6	SANDWICH	KENT	5.205000
7	SUNBURY-ON-THAMES	SURREY	5.111667
8	ASH	SURREY	5.060000
9	FAVERSHAM	KENT	4.992500
10	TONBRIDGE	KENT	4.972222
11	STAINES	SURREY	4.916667
12	WYE	KENT	4.915000
13	ROCHESTER	KENT	4.900000
14	LEATHERHEAD	SURREY	4.761000
15	WARLINGHAM	SURREY	4.730000
16	EPSOM	SURREY	4.643571
17	HORLEY	SURREY	4.640000
18	REIGATE	SURREY	4.617000
19	BANSTEAD	SURREY	4.570000
20	ASHFORD	KENT	4.535556
21	MAIDSTONE	KENT	4.491034
22	GRAVESEND	KENT	4.474444
23	WILMINGTON	KENT	4.455000
24	SHEPPERTON	SURREY	4.447500

```r
#-----------Joining all the ranking table-----------#

combined_ranking_table <-houseprice_rank %>% #starting with house price rank table
  left_join(download_speed_rank, by = c("Town/City" = "Town", "County" = "County")) %>% #joining with download speed rank table
  na.omit() %>%
  left_join(crime_rank, by = c("Town/City" = "Town", "County" = "County")) %>% #joining with crime rank table
  na.omit() %>%
  left_join(school_rank, by = c("Town/City" = "Town", "County" = "County")) %>% #joining with school rank table
  na.omit()

#-----------Calculation of total score-----------#

final_rank <- combined_ranking_table %>%
  mutate(`Total Score` = (`House Score` + `Download Score` + `Crime Score` + `School Score`) / 4) %>% #creating a new column to show the total score
  select(`Town/City`, County, `House Score`, `Download Score`, `Crime Score`,`School Score`, `Total Score`) %>% #arranging the order for columns
  arrange(desc(`Total Score`)) %>%   #showing the highest score first
  mutate(Rank= row_number()) %>%
  select(Rank, everything()) #moving the serial number column at first

#defining path to save final ranks csv
file_path <- "Recommended System/Final rank.csv"

#saving the final ranks csv
write.csv(final_rank, file_path, row.names = FALSE)
view(final_rank)
```

	Rank	Town/City	County	House Score	Download Score	Crime Score	School Score	Total Score
1	1	WARLINGHAM	SURREY	9.450000	0.121	9.964088	4.730000	6.066272
2	2	DARTFORD	KENT	6.684058	0.489	9.772098	5.921429	5.716646
3	3	WHITSTABLE	KENT	8.005000	0.668	9.965984	4.180000	5.704746
4	4	ROCHESTER	KENT	7.191926	0.720	9.320860	4.900000	5.533197
5	5	BIRCHINGTON	KENT	7.545000	0.558	9.753409	3.985000	5.460352
6	6	SWANLEY	KENT	7.272857	0.542	9.852904	4.015000	5.420690
7	7	TUNBRIDGE WELLS	KENT	6.143611	0.703	9.643441	4.377500	5.216888
8	8	CANTERBURY	KENT	6.125046	0.637	9.761976	3.462222	4.996561
9	9	HORLEY	SURREY	5.226786	0.643	9.371122	4.640000	4.970227
10	10	LINGFIELD	SURREY	6.014000	0.476	9.821340	3.495000	4.951585
11	11	REDHILL	SURREY	6.446202	1.063	9.490925	2.713750	4.928469
12	12	DORKING	SURREY	5.567341	0.716	9.767942	3.500000	4.887821
13	13	LONGFIELD	KENT	7.388000	0.502	9.939113	1.692000	4.880278
14	14	FARNHAM	SURREY	3.888833	1.097	9.968290	4.220833	4.793739
15	15	CATERHAM	SURREY	5.133333	0.618	9.911858	3.406250	4.767360
16	16	REIGATE	SURREY	3.321667	0.634	9.960200	4.617000	4.633217
17	17	CRANLEIGH	SURREY	4.000000	0.536	9.878044	3.425000	4.459761
18	18	GODALMING	SURREY	4.424111	0.619	9.748997	2.803571	4.398920
19	19	OXTED	SURREY	4.673571	0.401	9.876051	2.621667	4.393072
20	20	GUILDFORD	SURREY	2.000850	0.836	9.790691	3.928947	4.139122
21	21	MARGATE	KENT	8.415933	0.693	2.822104	1.068000	3.249759
22	22	SITTINGBOURNE	KENT	-7.202647	0.966	9.753521	4.050833	1.891927

Reflection

As we look back on the creation of our Town Recommender System, it is clear that our dedication to transparency, user privacy, and moral principles has played a critical role in creating a trustworthy and responsible tool. By integrating data cleansing, normalization, and exploratory data analysis, we have been able to guarantee the precision and potency of our recommendation model in addition to spotting trends in town features. Our system's precision has been further improved by utilizing Linear Modeling approaches, which enable the generation of dynamic and tailored recommendations. To ensure that our system develops responsibly and complies with changing norms for fairness and data protection, however, constant attention to growing legal and ethical issues is essential. Our dedication to development and ethical standards continues to be at the heart of our efforts as we work to provide foreign students more influence over their study abroad choices.

Our reflection approach also heavily relies on continuous user feedback and involvement, which helps us modify the Town Recommender System in response to changing student demands and real-world experiences. Our objective is to maintain the confidence that international students have in our system and make a positive impact on their study exchange experiences in Kent and Surrey by building a culture of accountability and continuous development.

Ethical and Legal Issues

Our Town Recommender System was developed with careful consideration of legal and ethical considerations. It is crucial to guarantee adherence to data protection regulations, including the General Data Protection Regulation (GDPR) in the European Union. By protecting and anonymizing sensitive data, getting required consents, and upholding openness regarding data usage, we put user privacy first. In addition, ethical concerns direct our decision-making process, avoiding bias in modeling or data selection to guarantee unbiased and equitable suggestions. In order to give international students a reliable and responsible tool to help them navigate their study exchange in Kent and Surrey, we must uphold the highest legal and ethical standards.

Additionally, we respect ethical standards by emphasizing openness in the way that data is gathered, handled, and used in our system. In order to maintain the integrity of the recommendation algorithms and reduce the possibility of unintentional biases, regular audits and monitoring systems are in place. Likewise, our technology respects user autonomy by giving users control over their personal information and choices. Our goal is to create a Town Recommender System that not only satisfies regulatory requirements but also gains the respect and faith of its users by thoroughly resolving legal and ethical issues.

Future Scope

The future plans for our Town Recommender System call for ongoing improvements and additions. We intend to apply the latest developments in machine learning to improve recommendation algorithms, making them more flexible and sensitive to changing user preferences. Furthermore, working together with local government agencies and communities can enhance the quality of our data sources and provide us a more complete picture of town dynamics. Integration with cutting-edge technology like augmented reality for immersive town exploration experiences may also be included in future generations. Our dedication to upholding the law, upholding moral principles, and making user-centered enhancements will not waver as technology advances, guaranteeing that our Town Recommender System will be a useful tool for overseas students looking for the best study exchange options in Kent and Surrey.

Conclusion

To sum up, the creation of our Town Recommender System is a big step in the right direction for helping foreign students choose Kent and Surrey for their study exchange. Through thorough data collecting, cleansing, and analysis, coupled with the application of ethical and legal norms, we've produced a dependable tool that prioritizes user privacy and transparency. Personalized and dynamic suggestions are provided by the predictive edge that comes from the use of linear modeling. In the future, we will prioritize ongoing enhancement, user involvement, and technological progress to guarantee that the Town Recommender System continues to be a useful and dynamic tool for students navigating their academic paths in England. Finally, our effort to adding new technologies and enhancing the system in response to customer feedback highlights our commitment to offering a flexible and ever-improving service. Our Town Recommender System focuses on user empowerment, ethical considerations, and technical innovation to provide international students in the dynamic counties of Kent and Surrey with a smooth and enriching study exchange experience.

GitHub Link:

https://github.com/Ronit-Bhujel/DataScience_220050

References

Salesforce. (n.d.). *Tableau*. Retrieved from tableau.com: https://www.tableau.com/learn/articles/what-is-data-cleaning