homework_1

2023-09-22

Problem1
Problem2
Problem3
Problem4

Importing libraries

```
library(ggplot2)
library(GGally)
library(dplyr)
library(tidyr)
```

Problem 1

For this question, we will use the US census data set from 1994, which is in adult.csv.

a. First, we look at the summary statistics for all the variables. Based on those metrics, including the quartiles , compare two variables. What can you tell about their shape from these summaries?

Reading US census data set csv file. (adult.csv)

```
data <- read.csv2("C:/Users/bunty/Desktop/funda/adult.csv", header = T,sep = ",")
```

```
df <- as.data.frame(data)
dim(df)
```

```
## [1] 32561    15
```

```
summary(df)
```

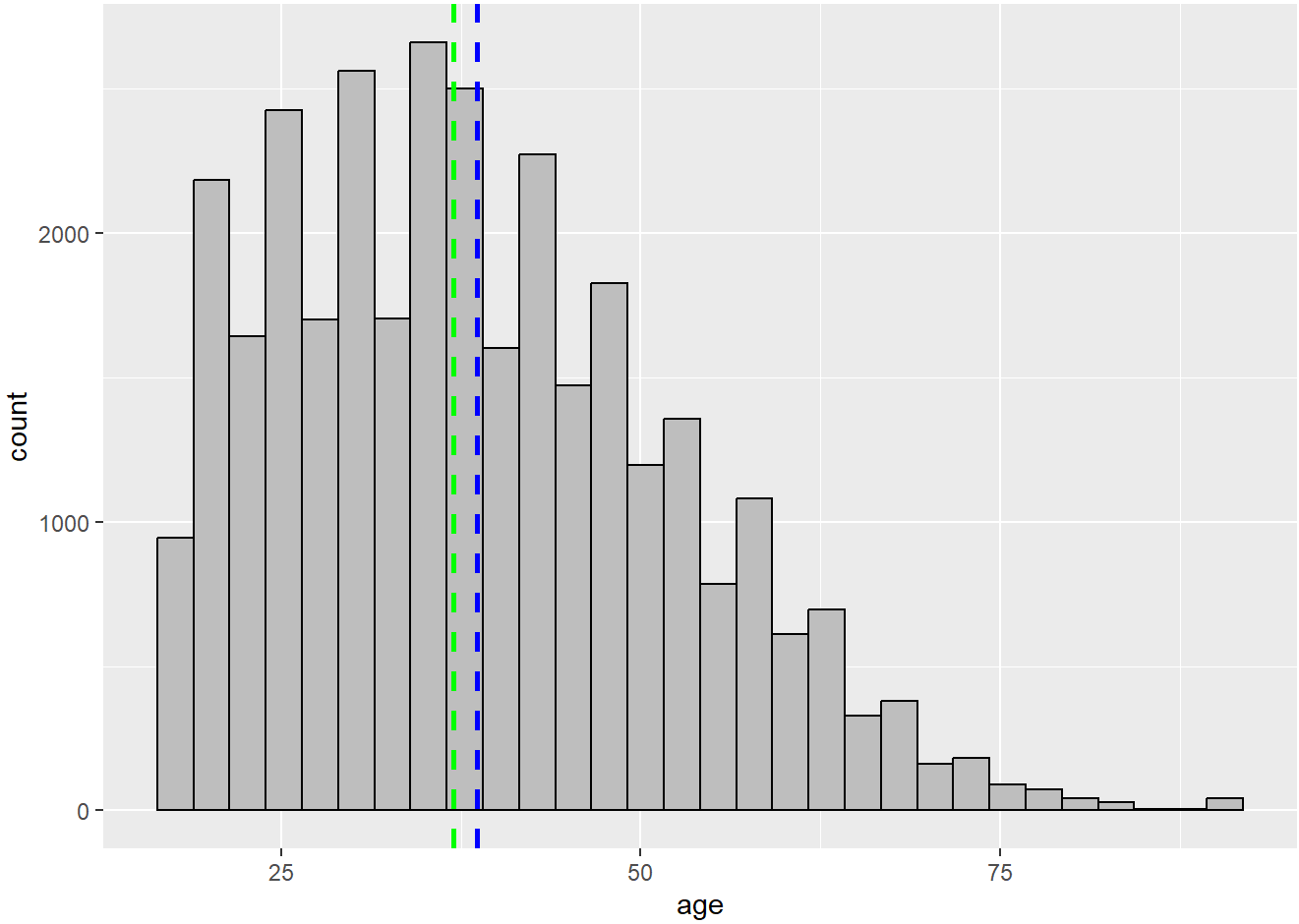
```
##       age         workclass             fnlwgt         education
##  Min.   :17.00   Length:32561       Min.   :  12285   Length:32561
##  1st Qu.:28.00   Class :character   1st Qu.: 117827   Class :character
##  Median :37.00   Mode  :character   Median : 178356   Mode  :character
##  Mean   :38.58                      Mean   : 189778
##  3rd Qu.:48.00                      3rd Qu.: 237051
##  Max.   :90.00                      Max.   :1484705
##  education.num   marital.status      occupation        relationship
##  Min.   : 1.00   Length:32561       Length:32561       Length:32561
##  1st Qu.: 9.00   Class :character   Class :character   Class :character
##  Median :10.00   Mode  :character   Mode  :character   Mode  :character
##  Mean   :10.08
##  3rd Qu.:12.00
##  Max.   :16.00
##      race               sex             capital.gain    capital.loss
##  Length:32561       Length:32561       Min.   :    0   Min.   :   0.0
##  Class :character   Class :character   1st Qu.:    0   1st Qu.:   0.0
##  Mode  :character   Mode  :character   Median :    0   Median :   0.0
##                                        Mean   : 1078   Mean   :  87.3
##                                        3rd Qu.:    0   3rd Qu.:   0.0
##                                        Max.   :99999   Max.   :4356.0
##  hours.per.week  native.country     income.bracket
##  Min.   : 1.00   Length:32561       Length:32561
##  1st Qu.:40.00   Class :character   Class :character
##  Median :40.00   Mode  :character   Mode  :character
##  Mean   :40.44
##  3rd Qu.:45.00
##  Max.   :99.00
```

From the above statistics of data set, we can observe the dimensions of the data set 32561 x 15. Taking two variables or attribute [age, hours.per.week] and based on the summary we can tell the shape, as mean > median for column age which indicates it is positive skewed and for the hours.per.week, the mean ~= median, which indicated it is near to normally distributed (nearly symmetrical).The entire data is mostly positive skewed.

b. Use a visualization to get a fine-grain comparison (you don't have to use QQ plots, though) of the distributions of those two variables. Why did you choose the type of visualization that you chose? How do your part (a) assumptions compare to what you can see visually?

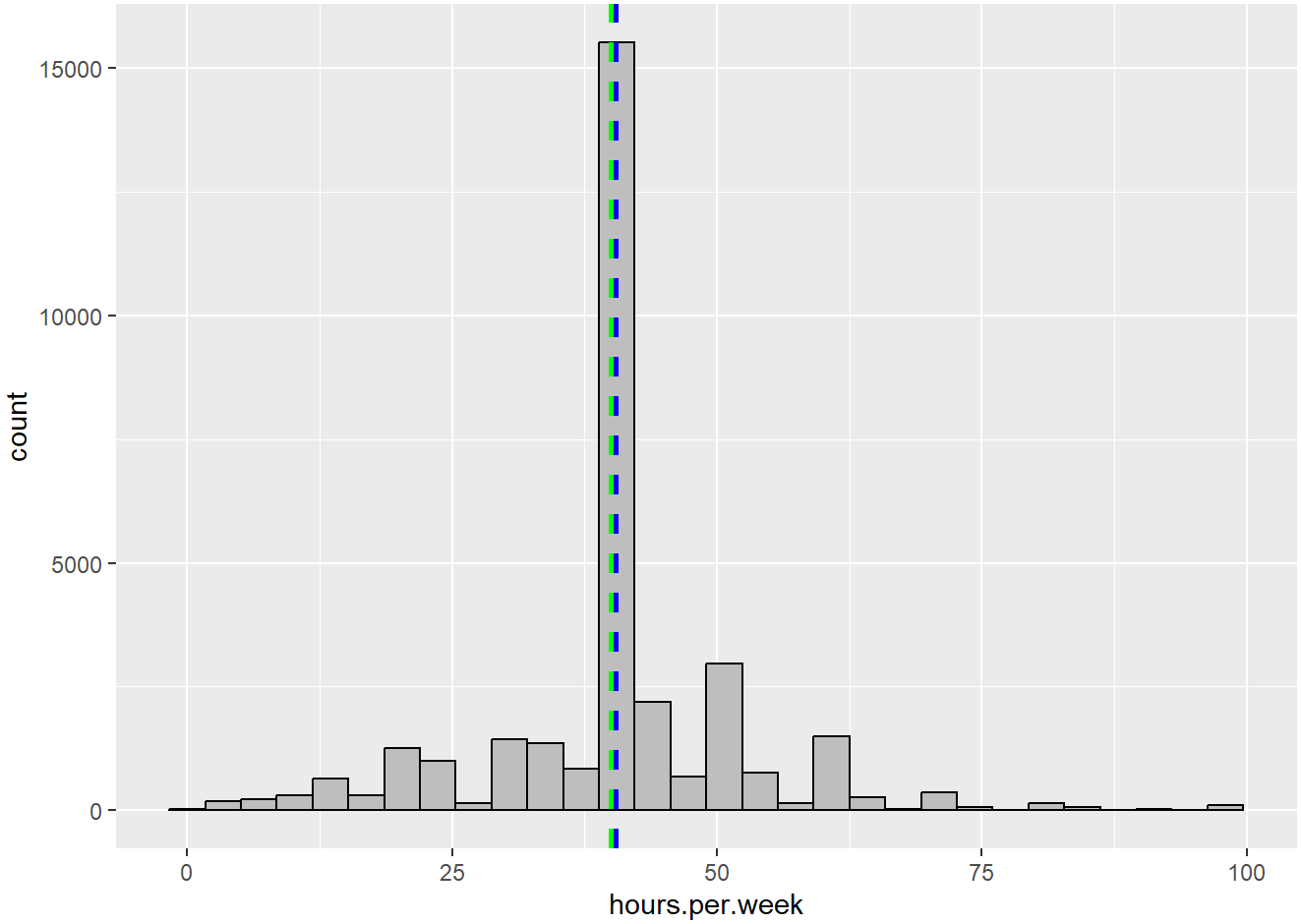
In the below visualization,I have chosen histogram with the line of mean and median to show the distribution of the age and hours.per.week.

```
ggplot(df, aes(x=age)) +
 geom_histogram(colour="black",fill="grey")+
  geom_vline(aes(xintercept=mean(age)),color="blue", linetype="dashed",size=1)+
  geom_vline(aes(xintercept=median(age)),color="green", linetype="dashed", size=1)
```



As discussed from the summary, we can see the distribution is postive skewed and the blue and green line shows mean and median respectively.

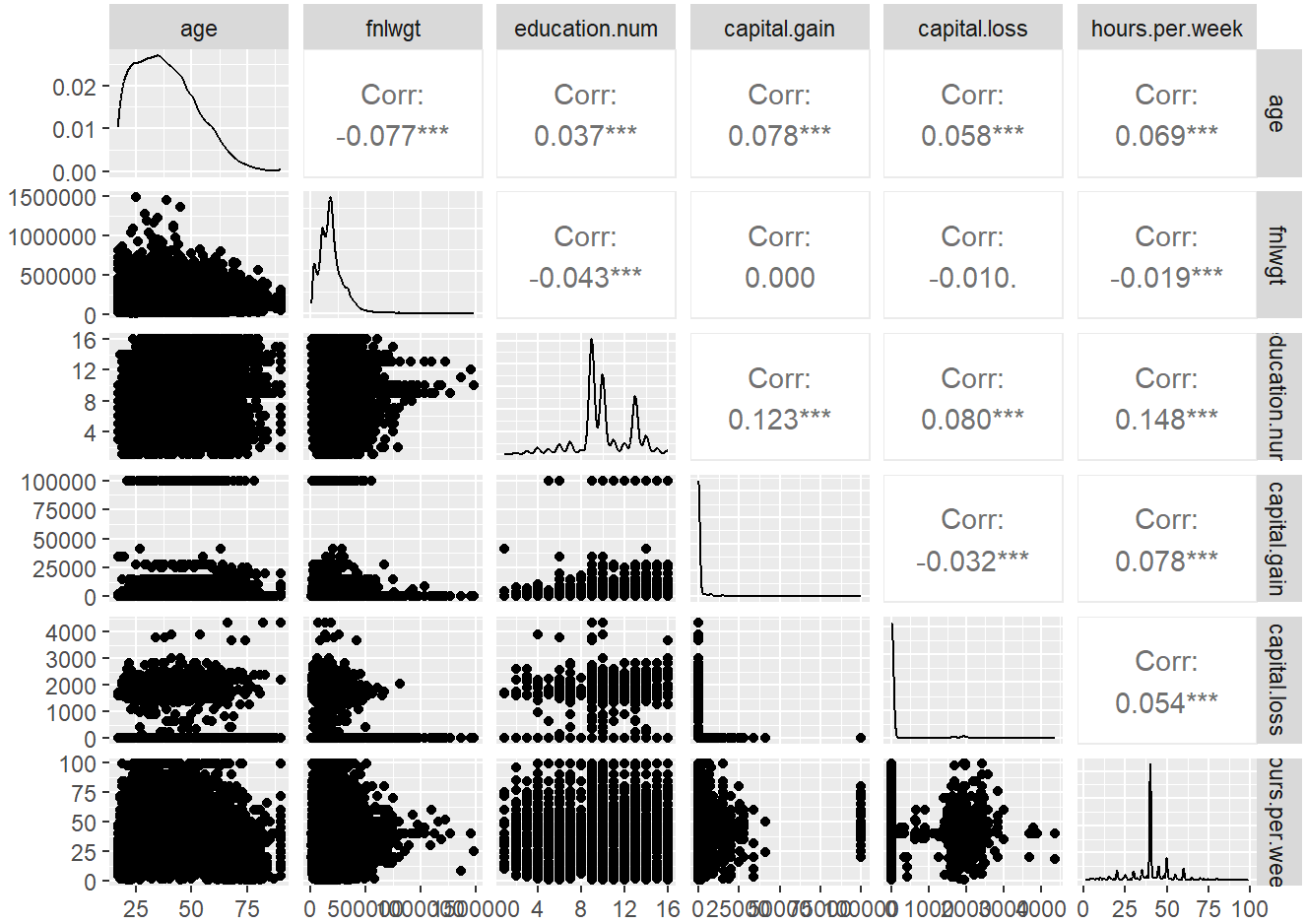
```
ggplot(df, aes(x=hours.per.week)) +
 geom_histogram(colour="black",fill="grey")+geom_vline(aes(xintercept=mean(hours.per.week)),color
="blue", linetype="dashed",size=1)+geom_vline(aes(xintercept=median(hours.per.week)),color="gree
n", linetype="dashed", size=1)
```



Similarly for this attribute, it is nearly symmetrical and we can observe the mean and median are much closer to each other.

c. Now create a scatter plot matrix of the numerical variables. What does this view show you that would be difficult to see looking at distributions?

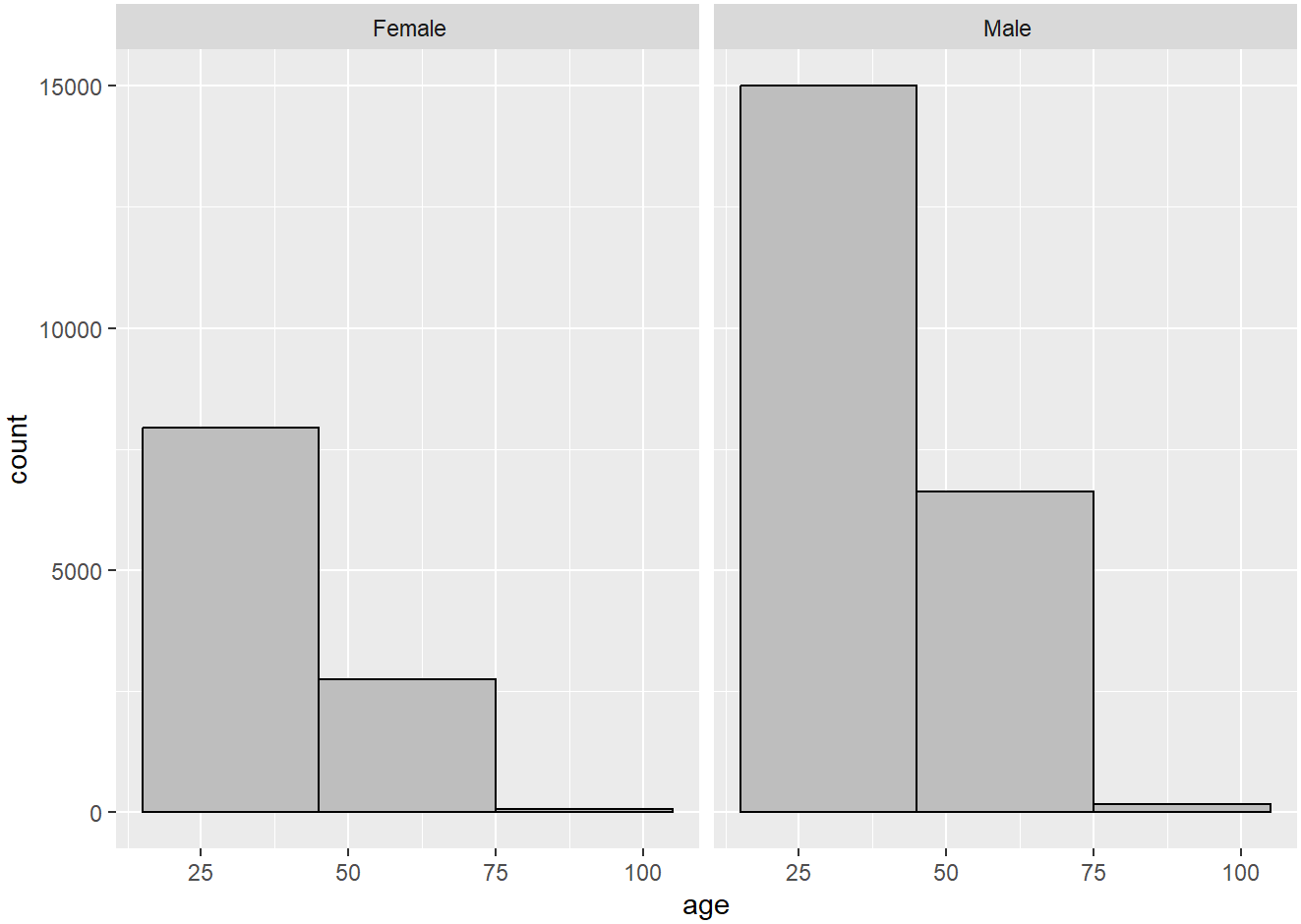
```
d_num = select_if(df,is.numeric)
ggpairs(d_num)
```



The scatter matrix view gives the relationship between each variables in the data set. This view is useful because it allows us to identify the correlation which would be difficult to see looking at the distribution. Here, we can observe there is a positive relation between age and capital.gain, negative relation between fnlwgt and hours.per.week and no relation between fnlwgt and capital.gain.

d. These data are a selection of US adults. It might not be a very balanced sample, though. Take a look at some categorical variables and see if any have a lot more of one category than others. There are many ways to do this, including histograms and following tidyererse group_by with count. I recommend you try a few for practice.

```
ggplot(df, aes(x=age,)) +
  geom_histogram(binwidth=30,colour="black",fill="grey")+facet_wrap(~sex)
```



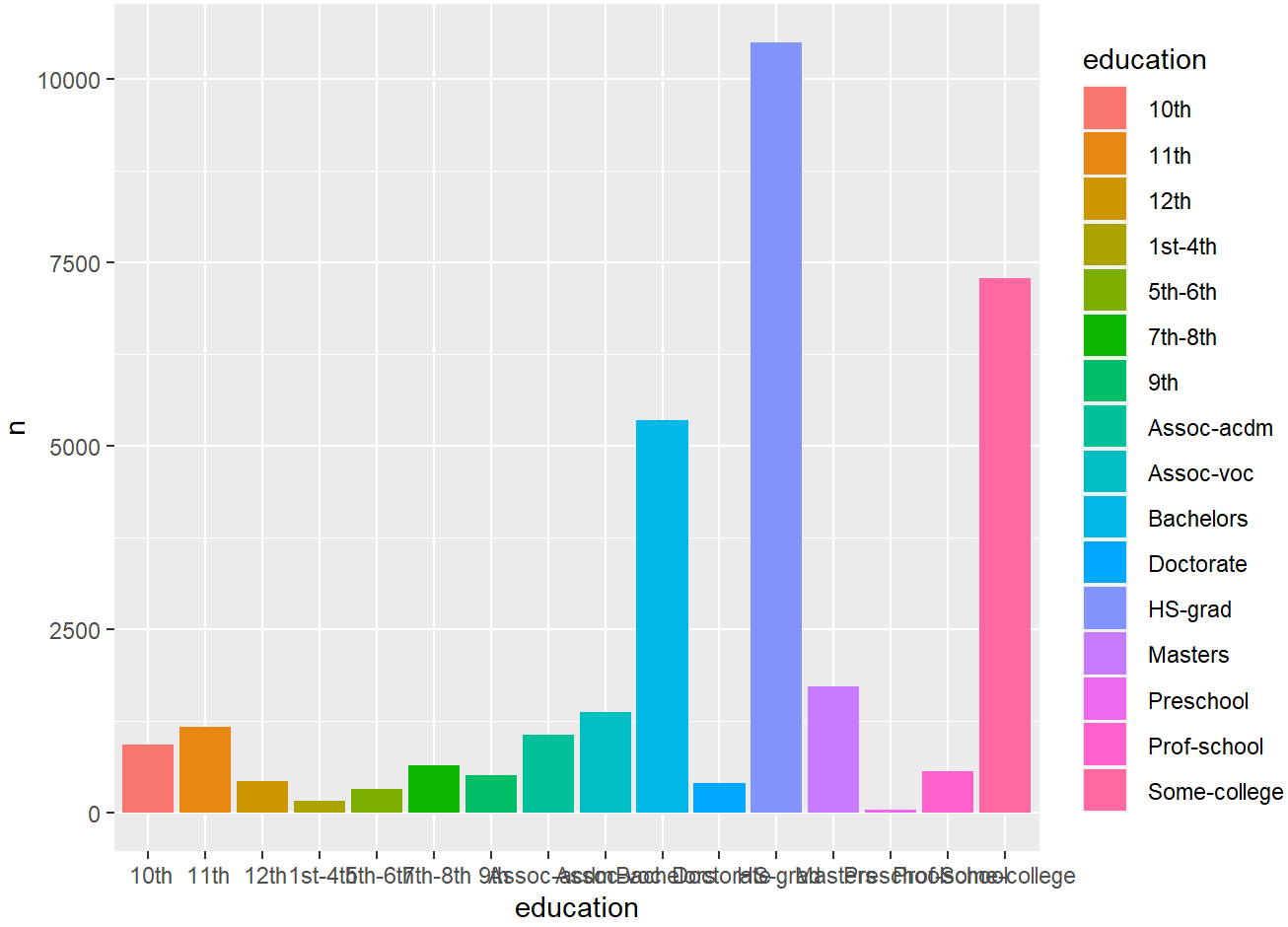
Here, we can see the sex variable has two categories male and female with higher number of males than females.

```
unique(df$education)
```

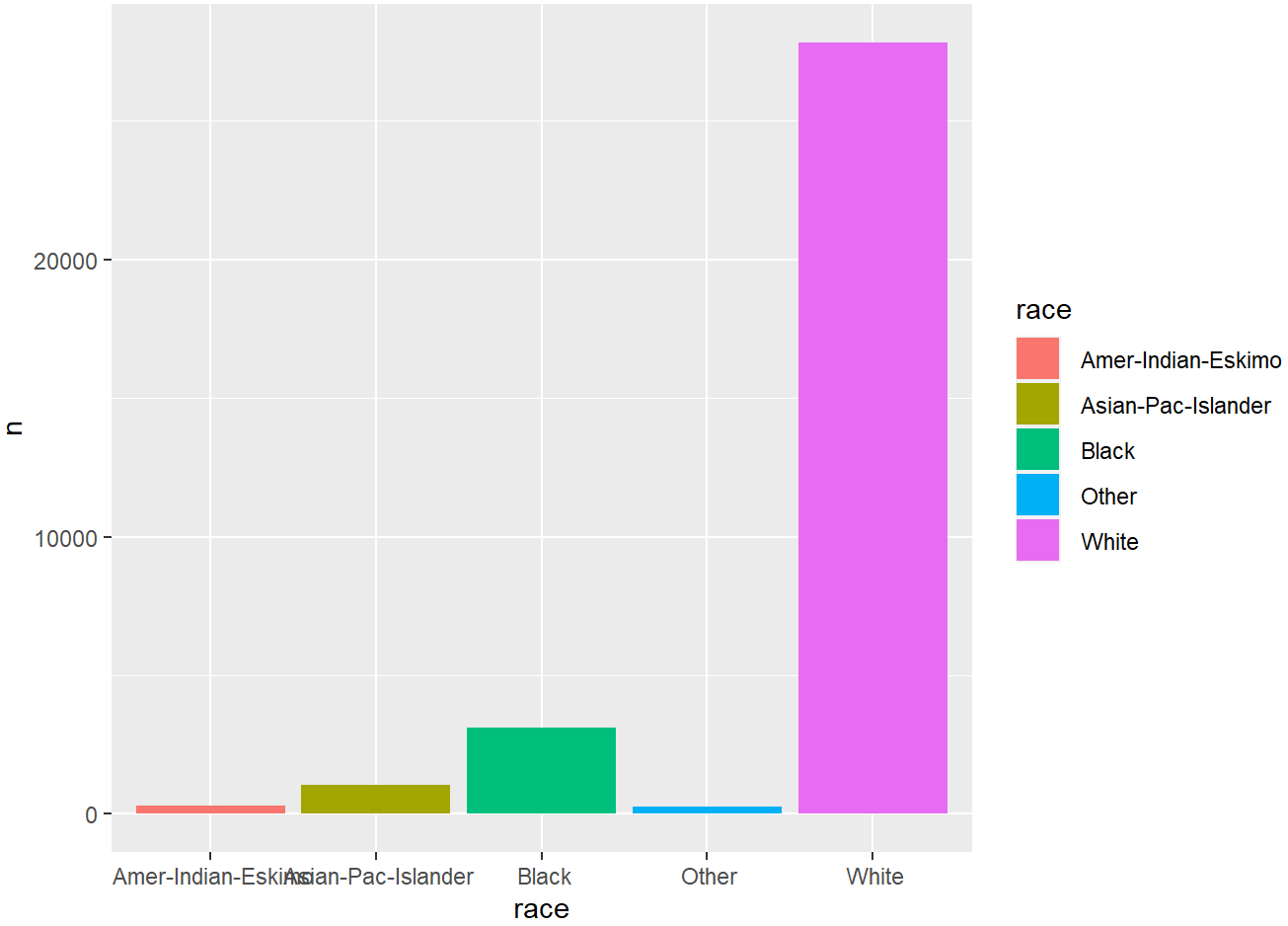
```
##  [1] " Bachelors"    " HS-grad"      " 11th"         " Masters"
##  [5] " 9th"          " Some-college" " Assoc-acdm"   " Assoc-voc"
##  [9] " 7th-8th"      " Doctorate"    " Prof-school"  " 5th-6th"
## [13] " 10th"         " 1st-4th"      " Preschool"    " 12th"
```

The education variable has total 16 categories with highest count in the category of HS-grad.

```
df %>%
  group_by(education) %>%
  summarise(n = n()) %>%
  ggplot(aes(x = education, y = n, fill = education)) +
  geom_bar(stat = "identity")
```

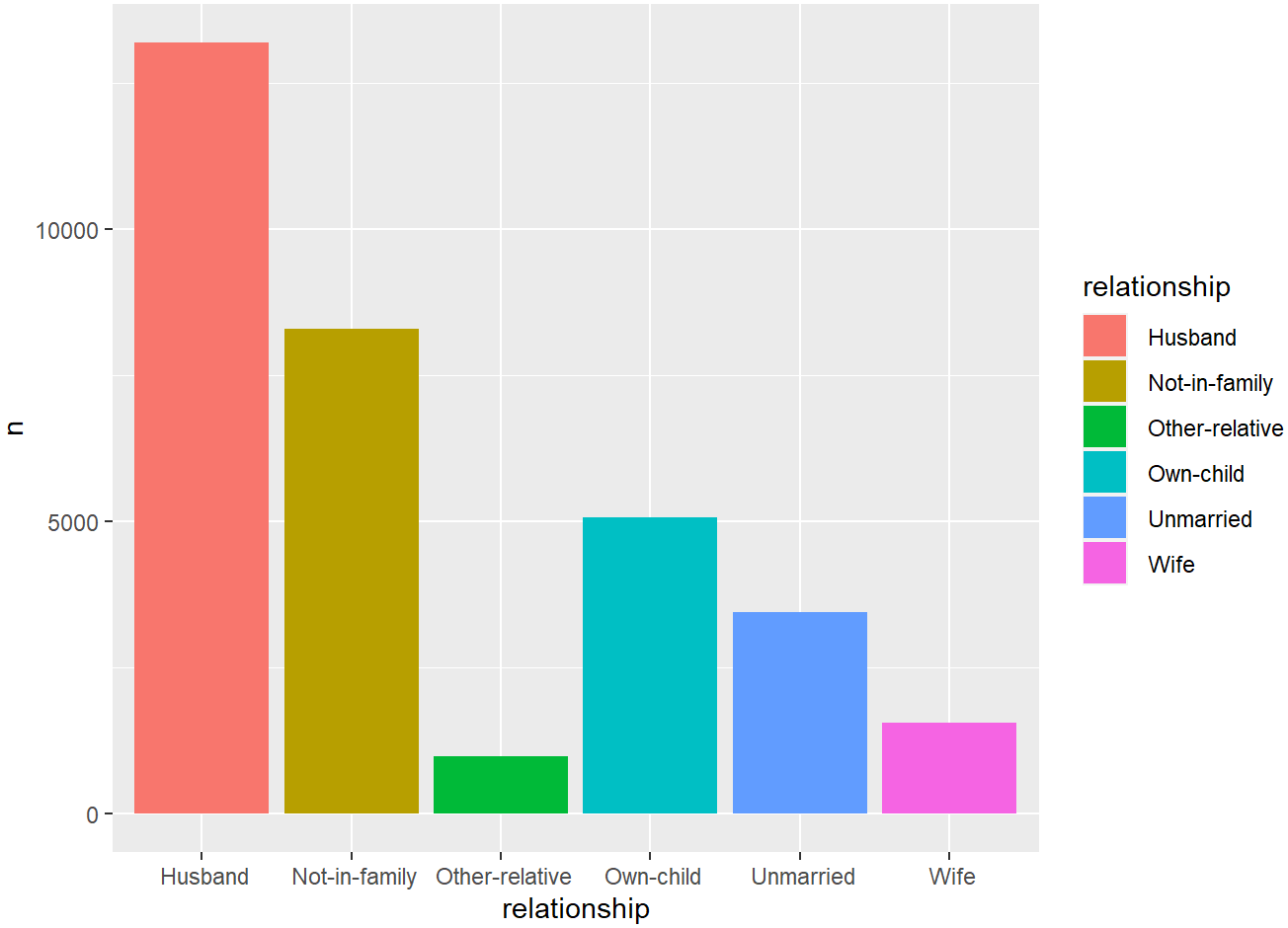



```
df %>%
group_by(race) %>%
  summarise(n = n()) %>%
  ggplot(aes(x = race, y = n, fill = race)) +
  geom_bar(stat = "identity")
```



variable race has total 5 categories as shown in the above chart with white race having highest count.

```
df %>%
group_by(relationship) %>%
  summarise(n = n()) %>%
  ggplot(aes(x = relationship, y = n, fill = relationship)) +
  geom_bar(stat = "identity")
```



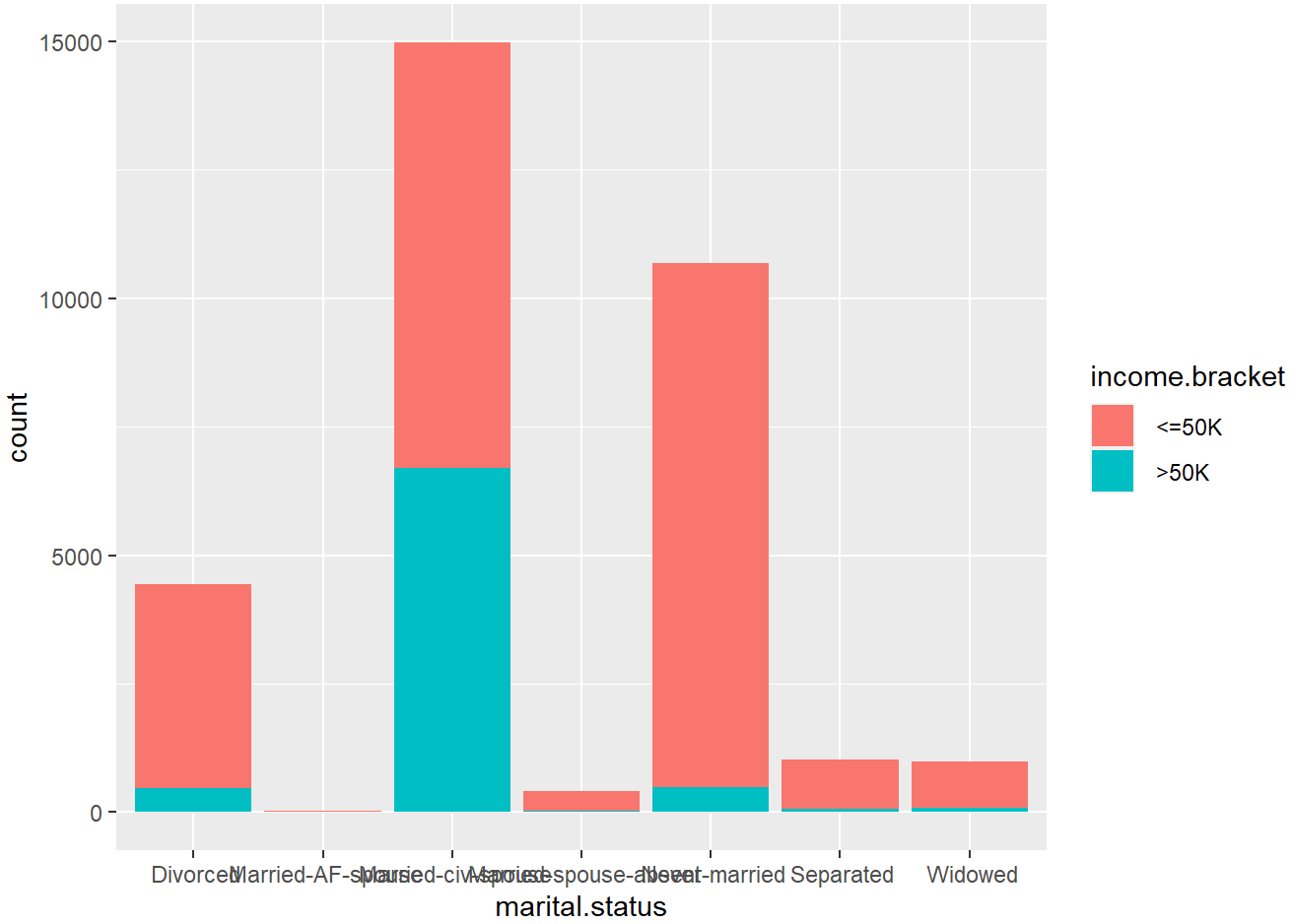
Similarly, variable relationship has 6 categories with husband as highest.

e. Now we'll consider a relationship between two categorical variables. Create a cross tabulation and then a corresponding visualization and explain a relationship between some of the values of the categorical.

```
cross_tabulation <- table(df$marital.status,df$income.bracket)
cross_tabulation
```

```
##
##                          <=50K  >50K
##    Divorced               3980   463
##    Married-AF-spouse        13    10
##    Married-civ-spouse     8284  6692
##    Married-spouse-absent   384    34
##    Never-married         10192   491
##    Separated               959    66
##    Widowed                 908    85
```

```
ggplot(df,aes(marital.status, fill=income.bracket))+geom_bar()
```



From the cross tabulation and stacked bar graph we can observe there is a relation between marital.status and income.bracket. Within Never-married category there is a higher proportion of people who has income <=50k.

Problem 2

In this question, you will integrate data on different years into one table and use some reshaping to get a visualization. There are two data files: population_even.csv and population_odd.csv. These are population data for even and odd years respectively.

a. Join the two tables together so that you have one table with each state's population for years 2010- 2019. If you are unsure about what variable to use as the key for the join, consider what variable the two original tables have in common. (Show a head of the resulting table.)

```
data_even <- read.csv2("C:/Users/bunty/Desktop/funda/population_even.csv",header = T,sep=",")
even = as.data.frame(data_even)
data_odd<- read.csv2("C:/Users/bunty/Desktop/funda/population_odd.csv",header=T,sep=",")
odd = as.data.frame(data_odd)
```

```
merged_data = even %>% inner_join(odd,by="NAME")
head(merged_data)
```

	STATE.x <int>	NAME <chr>	POPESTIMATE2010 <int>	POPESTIMATE2012 <int>	POPESTIMATE2014 <int>	POPESTIMATE2016 <int>
1	1	Alabama	4785437	4815588	4841799	4863525
2	2	Alaska	713910	730443	736283	741456
3	4	Arizona	6407172	6554978	6730413	6941072

	STATE.x	NAME	POPESTIMATE2010	POPESTIMATE2012	POPESTIMATE2014	POPESTIMATE2016
	<int>	<chr>	<int>	<int>	<int>	<int>
4	5	Arkansas	2921964	2952164	2967392	2989918
5	6	California	37319502	37948800	38596972	39167117
6	8	Colorado	5047349	5192647	5350101	5539215

6 rows | 1-7 of 14 columns

b. Clean this data up a bit (show a head of the data after):

Remove the duplicate state ID column if your process created one.

```
merged_data <- select(merged_data,-STATE.y)
colnames(merged_data)[1] = "STATE"
head(merged_data)
```

	ST…	NAME	POPESTIMATE2010	POPESTIMATE2012	POPESTIMATE2014	POPESTIMATE2016
	<int>	<chr>	<int>	<int>	<int>	<int>
1	1	Alabama	4785437	4815588	4841799	4863525
2	2	Alaska	713910	730443	736283	741456
3	4	Arizona	6407172	6554978	6730413	6941072
4	5	Arkansas	2921964	2952164	2967392	2989918
5	6	California	37319502	37948800	38596972	39167117
6	8	Colorado	5047349	5192647	5350101	5539215

6 rows | 1-7 of 13 columns

Rename columns to be just the year number.

```
new = gsub("POPESTIMATE","",colnames(merged_data))
colnames(merged_data)=new
head(merged_data)
```

	STA…	NAME	2010	2012	2014	2016	2018	2011	2013
	<int>	<chr>	<int>	<int>	<int>	<int>	<int>	<int>	<int>
1	1	Alabama	4785437	4815588	4841799	4863525	4887681	4799069	4830081
2	2	Alaska	713910	730443	736283	741456	735139	722128	737068
3	4	Arizona	6407172	6554978	6730413	6941072	7158024	*NA*	6632764
4	5	Arkansas	2921964	2952164	2967392	2989918	3009733	2940667	2959400
5	6	California	37319502	37948800	38596972	39167117	39461588	37638369	38260787
6	8	Colorado	5047349	5192647	5350101	5539215	5691287	5121108	5269035

6 rows | 1-10 of 13 columns

Reorder the columns to be in year order.

```
merged_data = merged_data[,c("STATE","NAME",sort(colnames(merged_data)[3:ncol(merged_data)]))]
head(merged_data)
```

	STA…	NAME	2010	2011	2012	2013	2014	2015	2016
	<int>	<chr>	<int>	<int>	<int>	<int>	<int>	<int>	<int>
1	1	Alabama	4785437	4799069	4815588	4830081	4841799	4852347	4863525
2	2	Alaska	713910	722128	730443	737068	736283	737498	741456
3	4	Arizona	6407172	*NA*	6554978	6632764	6730413	6829676	6941072
4	5	Arkansas	2921964	2940667	2952164	2959400	2967392	2978048	2989918
5	6	California	37319502	37638369	37948800	38260787	38596972	38918045	39167117
6	8	Colorado	5047349	5121108	5192647	5269035	5350101	5450623	5539215

6 rows | 1-10 of 13 columns

c. Deal with missing values in the data by replacing them with the average of the surrounding years. For example, if you had a missing value for Georgia in 2016, you would replace it with the average of Georgia's 2015 and 2017 numbers. This may require some manual effort.

Summary of merge_data shows there are some NA's in the columns

```
summary(merged_data)
```

```
##      STATE           NAME                2010              2011
##  Min.   : 1.00   Length:52          Min.   :  564487   Min.   :  567299
##  1st Qu.:16.75   Class :character   1st Qu.: 1764843   1st Qu.: 1712291
##  Median :29.50   Mode  :character   Median : 4092836   Median : 3872036
##  Mean   :29.79                      Mean   : 6020061   Mean   : 6054176
##  3rd Qu.:42.50                      3rd Qu.: 6610438   3rd Qu.: 6720105
##  Max.   :72.00                      Max.   :37319502   Max.   :37638369
##                                                        NA's   :1
##       2012              2013              2014              2015
##  Min.   :  576305   Min.   :  582122   Min.   :  582531   Min.   :  585613
##  1st Qu.: 1788808   1st Qu.: 1732560   1st Qu.: 1794895   1st Qu.: 1866664
##  Median : 4142674   Median : 3922468   Median : 4188796   Median : 4425976
##  Mean   : 6105105   Mean   : 6039414   Mean   : 6189152   Mean   : 6322693
##  3rd Qu.: 6721518   3rd Qu.: 6673040   3rd Qu.: 6835611   3rd Qu.: 6996666
##  Max.   :37948800   Max.   :38260787   Max.   :38596972   Max.   :38918045
##                     NA's   :1                             NA's   :1
##       2016              2017              2018              2019
##  Min.   :  584215   Min.   :  578931   Min.   :  577601   Min.   :  578759
##  1st Qu.: 1793862   1st Qu.: 1866476   1st Qu.: 1790852   1st Qu.: 1789606
##  Median : 4264079   Median : 4452268   Median : 4321520   Median : 4217737
##  Mean   : 6275923   Mean   : 6416830   Mean   : 6343863   Mean   : 6384525
##  3rd Qu.: 7029497   3rd Qu.: 7233685   3rd Qu.: 7249485   3rd Qu.: 7446805
##  Max.   :39167117   Max.   :39358497   Max.   :39461588   Max.   :39512223
##                     NA's   :1                             NA's   :1
```

Replacing the NA's

```
for (i in 1:nrow(merged_data)){
  for(j in 3:ncol(merged_data)){
    if(is.na(merged_data[i,j])){
      p_=merged_data[i,j-1]
      n_=merged_data[i,j+1]
      avg = mean(c(p_,n_), na.rm=TRUE)
      merged_data[i,j]<-avg
    }
  }
}
summary(merged_data)
```

```
##      STATE           NAME                2010               2011
##  Min.   : 1.00   Length:52          Min.   :  564487   Min.   :  567299
##  1st Qu.:16.75   Class :character   1st Qu.: 1764843   1st Qu.: 1776482
##  Median :29.50   Mode  :character   Median : 4092836   Median : 4120928
##  Mean   :29.79                      Mean   : 6020061   Mean   : 6062385
##  3rd Qu.:42.50                      3rd Qu.: 6610438   3rd Qu.: 6666844
##  Max.   :72.00                      Max.   :37319502   Max.   :37638369
##       2012               2013               2014               2015
##  Min.   :  576305   Min.   :  582122   Min.   :  582531   Min.   :  585613
##  1st Qu.: 1788808   1st Qu.: 1793237   1st Qu.: 1794895   1st Qu.: 1795724
##  Median : 4142674   Median : 4163564   Median : 4188796   Median : 4220884
##  Mean   : 6105105   Mean   : 6145883   Mean   : 6189152   Mean   : 6232963
##  3rd Qu.: 6721518   3rd Qu.: 6775982   3rd Qu.: 6835611   3rd Qu.: 6913171
##  Max.   :37948800   Max.   :38260787   Max.   :38596972   Max.   :38918045
##       2016               2017               2018               2019
##  Min.   :  584215   Min.   :  578931   Min.   :  577601   Min.   :  578759
##  1st Qu.: 1793862   1st Qu.: 1792182   1st Qu.: 1790852   1st Qu.: 1790876
##  Median : 4264079   Median : 4297946   Median : 4321520   Median : 4342705
##  Mean   : 6275923   Mean   : 6313637   Mean   : 6343863   Mean   : 6373427
##  3rd Qu.: 7029497   3rd Qu.: 7138846   3rd Qu.: 7249485   3rd Qu.: 7362761
##  Max.   :39167117   Max.   :39358497   Max.   :39461588   Max.   :39512223
```

d. We can use some tidyverse aggregation to learn about the population.

a.Get the maximum population for a single year for each state. Note that because you are using an aggregation function (max) across a row, you will need the rowwise() command in your tidyverse pipe. If you do not, the max value will not be individual to the row. Of course there are alternative ways.

using pipeline and rowwise() as learned in tutorial 1

```
max_pop = merged_data %>%
  rowwise() %>%
  mutate(max_population = max(c_across(3:ncol(.)),na.rm = TRUE)) %>%
  select(STATE,max_population)
max_pop
```

STATE <int>	max_population <dbl>
1	4903185
2	741456
4	7278717

STATE <int>	max_population <dbl>
5	3017804
6	39512223
8	5758736
9	3594841
10	973764
11	705749
12	21477737

1-10 of 52 rows Previous **1** 2 3 4 5 6 Next

b.Now get the total population across all years for each state. This should be possible with a very minor change to the code from (d). Why is that?

Just we need to change the aggregate function.

```
total_sum_state <- merged_data %>%
  rowwise() %>%
  mutate(total_sum = sum(c_across(3:ncol(.)), na.rm = TRUE)) %>%
  select(STATE, total_sum)

total_sum_state
```

STATE <int>	total_sum <dbl>
1	48453198
2	7325170
4	68057899
5	29738435
6	386181900
8	54031986
9	35826676
10	9364455
11	6636276
12	201096314

1-10 of 52 rows Previous **1** 2 3 4 5 6 Next

e. Finally, get the total US population for one single year. Keep in mind that this can be done with a single line of code even without the tidyverse, so keep it simple.

```
total <- colSums(merged_data[3:ncol(merged_data)])
total
```

```
##      2010      2011      2012      2013      2014      2015      2016      2017
## 313043191 315244038 317465478 319585920 321835882 324114082 326347983 328309105
##      2018      2019
## 329880855 331418189
```

Problem 3

Continuing with the data from Problem 2, let's create a graph of population over time for a few states (choose at least three yourself). This will require another data transformation, a reshaping. In order to create a line graph, we will need a variable that represents the year, so that it can be mapped to the x axis. Use a transformation to turn all those year columns into one column that holds the year, reducing the 10 year columns down to 2 columns (year and population). Once the data are in the right shape, it will be no harder than any line graph: put the population on the y axis and color by the state.

Reshaping data set.

```
reshaped_data <- merged_data %>%
  pivot_longer(cols = "2010":"2019", names_to = "Year", values_to = "Population")

reshaped_data
```

STATE <int>	NAME <chr>	Year <chr>	Population <dbl>
1	Alabama	2010	4785437
1	Alabama	2011	4799069
1	Alabama	2012	4815588
1	Alabama	2013	4830081
1	Alabama	2014	4841799
1	Alabama	2015	4852347
1	Alabama	2016	4863525
1	Alabama	2017	4874486
1	Alabama	2018	4887681
1	Alabama	2019	4903185

1-10 of 520 rows Previous **1** 2 3 4 5 6 … 52 Next

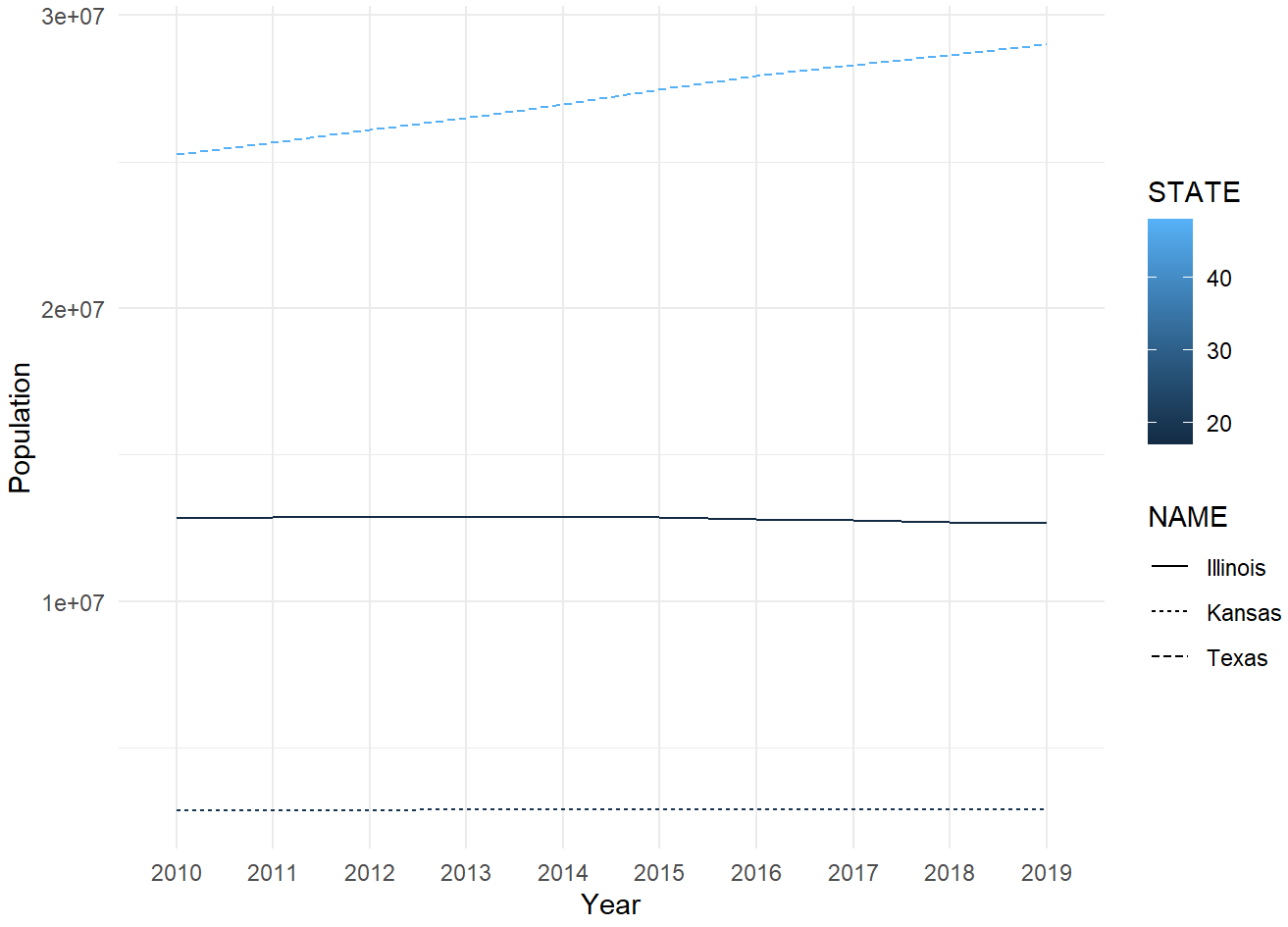
```
states <- reshaped_data %>% filter(NAME %in% c("Illinois","Kansas","Texas"))

states
```

STATE <int>	NAME <chr>	Year <chr>	Population <dbl>
17	Illinois	2010	12840503
17	Illinois	2011	12867454
17	Illinois	2012	12882510

STATE <int>	NAME <chr>	Year <chr>	Population <dbl>
17	Illinois	2013	12895129
17	Illinois	2014	12884493
17	Illinois	2015	12858913
17	Illinois	2016	12820527
17	Illinois	2017	12778828
17	Illinois	2018	12723071
17	Illinois	2019	12671821

1-10 of 30 rows Previous **1** 2 3 Next

```
ggplot(states,aes(x=Year,y=Population,group=NAME,color=STATE))+geom_line(aes(linetype=NAME))+
  theme_minimal()
```



Problem 4

This problem is short answer questions only. No code is needed.

a. Describe two ways in which data can be dirty, and for each one, provide a potential solution. There are many ways in which data can be dirty, some of it are: *Incomplete Data* : Incomplete data occurs when certain fields or values are missing from the data set. For example, a Student database might have missing some of the details and values. This can lead to inaccurate

analysis. Missing or incomplete data can be ignored, replaced or inferred. The Solution to it - we can drop all the rows which contain incomplete information only if it does not affect the analysis and predicting outputs, can be replaced with mean median mode if we do not have a class label and depending up on the data set.

Noisy Data : Noisy data means incomplete, errors that present in the data set. Solution to handle noisy data is to identify the outliers and clean the data. In univariate case, IQR and standard deviation method can be used to detect outliers and smoothing it to remove noise. In bivariate case, liner regression can be used for the same.

b. Explain which data mining functionality you would use to help with each of these data questions.

a. Suppose we have data where each row is a customer and we have columns that describe their purchases. What are five groups of customers who buy similar things?

clustering is the best approach by making 5 groups (clusters) who buy similar things.

b. For the same data: can I predict if a customer will buy milk based on what else they bought?

Yes we can use classification to predict if a customer will buy milk on what else they bought.

c. Suppose we have data listing items in individual purchases. What are different sets of products that are often purchased together?

Here, association rule mining can be used.

d. Explain if each of the following is a data mining task

a. Organizing the customers of a company according to education level.

This is not a data mining task.to do this task we use sorting and grouping.

b. Computing the total sales of a company.

No, this is not a data mining task.

c. Sorting a student database according to identification numbers.

No, this is not a data mining task.

d. Predicting the outcomes of tossing a (fair) pair of dice. No, this is not a data mining task as this is a part of probability.

e. Predicting the future stock price of a company using historical records.

Yes, this is a data mining task because it takes the historical data to anlayze future predictions of a company.