California\_Housing

Aditya Pansari

2024-02-02

### California Housing

### Let’s see what are the pre-installed libraries in the system

# California Housing Prices 1990   
  
library(gdata)

##   
## Attaching package: 'gdata'

## The following object is masked from 'package:stats':  
##   
## nobs

## The following object is masked from 'package:utils':  
##   
## object.size

## The following object is masked from 'package:base':  
##   
## startsWith

library(plyr)  
library(readxl)  
library(dplyr)

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:plyr':  
##   
## arrange, count, desc, failwith, id, mutate, rename, summarise,  
## summarize

## The following objects are masked from 'package:gdata':  
##   
## combine, first, last, starts\_with

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

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

library(tidyr)

##   
## Attaching package: 'tidyr'

## The following object is masked from 'package:gdata':  
##   
## starts\_with

library(gridExtra)

##   
## Attaching package: 'gridExtra'

## The following object is masked from 'package:dplyr':  
##   
## combine

## The following object is masked from 'package:gdata':  
##   
## combine

library(caTools) # for splitting the data  
library(digest) # For hash functions  
library(ggplot2)   
library(recipes)

##   
## Attaching package: 'recipes'

## The following object is masked from 'package:stats':  
##   
## step

library(caret)

## Loading required package: lattice

library(data.table)

##   
## Attaching package: 'data.table'

## The following objects are masked from 'package:dplyr':  
##   
## between, first, last

## The following objects are masked from 'package:gdata':  
##   
## first, last

library(caret)  
library(leaflet) # for map. visualization  
library(sf)

## Linking to GEOS 3.11.2, GDAL 3.7.2, PROJ 9.3.0; sf\_use\_s2() is TRUE

library(viridis)

## Loading required package: viridisLite

### Data Loading

houses <- read.csv("housing.csv")

### Data Preprocessing / Taking a Quick Look at the Data Structure

summary(houses)

## longitude latitude housing\_median\_age total\_rooms   
## Min. :-124.3 Min. :32.54 Min. : 1.00 Min. : 2   
## 1st Qu.:-121.8 1st Qu.:33.93 1st Qu.:18.00 1st Qu.: 1448   
## Median :-118.5 Median :34.26 Median :29.00 Median : 2127   
## Mean :-119.6 Mean :35.63 Mean :28.64 Mean : 2636   
## 3rd Qu.:-118.0 3rd Qu.:37.71 3rd Qu.:37.00 3rd Qu.: 3148   
## Max. :-114.3 Max. :41.95 Max. :52.00 Max. :39320   
##   
## total\_bedrooms population households median\_income   
## Min. : 1.0 Min. : 3 Min. : 1.0 Min. : 0.4999   
## 1st Qu.: 296.0 1st Qu.: 787 1st Qu.: 280.0 1st Qu.: 2.5634   
## Median : 435.0 Median : 1166 Median : 409.0 Median : 3.5348   
## Mean : 537.9 Mean : 1425 Mean : 499.5 Mean : 3.8707   
## 3rd Qu.: 647.0 3rd Qu.: 1725 3rd Qu.: 605.0 3rd Qu.: 4.7432   
## Max. :6445.0 Max. :35682 Max. :6082.0 Max. :15.0001   
## NA's :207   
## median\_house\_value ocean\_proximity   
## Min. : 14999 Length:20640   
## 1st Qu.:119600 Class :character   
## Median :179700 Mode :character   
## Mean :206856   
## 3rd Qu.:264725   
## Max. :500001   
##

head(houses)

## longitude latitude housing\_median\_age total\_rooms total\_bedrooms population  
## 1 -122.23 37.88 41 880 129 322  
## 2 -122.22 37.86 21 7099 1106 2401  
## 3 -122.24 37.85 52 1467 190 496  
## 4 -122.25 37.85 52 1274 235 558  
## 5 -122.25 37.85 52 1627 280 565  
## 6 -122.25 37.85 52 919 213 413  
## households median\_income median\_house\_value ocean\_proximity  
## 1 126 8.3252 452600 NEAR BAY  
## 2 1138 8.3014 358500 NEAR BAY  
## 3 177 7.2574 352100 NEAR BAY  
## 4 219 5.6431 341300 NEAR BAY  
## 5 259 3.8462 342200 NEAR BAY  
## 6 193 4.0368 269700 NEAR BAY

str(houses)

## 'data.frame': 20640 obs. of 10 variables:  
## $ longitude : num -122 -122 -122 -122 -122 ...  
## $ latitude : num 37.9 37.9 37.9 37.9 37.9 ...  
## $ housing\_median\_age: num 41 21 52 52 52 52 52 52 42 52 ...  
## $ total\_rooms : num 880 7099 1467 1274 1627 ...  
## $ total\_bedrooms : num 129 1106 190 235 280 ...  
## $ population : num 322 2401 496 558 565 ...  
## $ households : num 126 1138 177 219 259 ...  
## $ median\_income : num 8.33 8.3 7.26 5.64 3.85 ...  
## $ median\_house\_value: num 452600 358500 352100 341300 342200 ...  
## $ ocean\_proximity : chr "NEAR BAY" "NEAR BAY" "NEAR BAY" "NEAR BAY" ...

colnames(houses)

## [1] "longitude" "latitude" "housing\_median\_age"  
## [4] "total\_rooms" "total\_bedrooms" "population"   
## [7] "households" "median\_income" "median\_house\_value"  
## [10] "ocean\_proximity"

### Let’s look if we have missing values

missing\_counts <- colSums(is.na(houses)) # counting missing values for each column  
missing\_counts

## longitude latitude housing\_median\_age total\_rooms   
## 0 0 0 0   
## total\_bedrooms population households median\_income   
## 207 0 0 0   
## median\_house\_value ocean\_proximity   
## 0 0

### Let’s calculate the proportion of missing values in each column

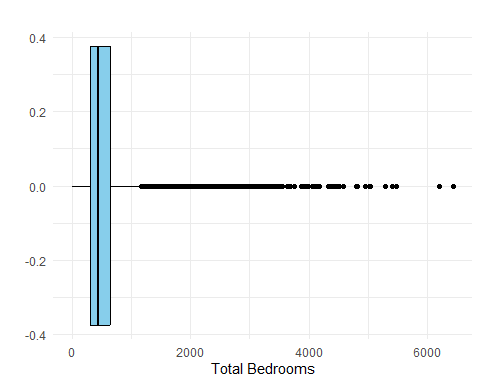
missing\_proportion <- houses %>%  
 summarise\_all(~ mean(is.na(.))) %>%  
 pivot\_longer(everything(), names\_to = "Variable", values\_to = "ProportionMissing")  
missing\_proportion

## # A tibble: 10 × 2  
## Variable ProportionMissing  
## <chr> <dbl>  
## 1 longitude 0   
## 2 latitude 0   
## 3 housing\_median\_age 0   
## 4 total\_rooms 0   
## 5 total\_bedrooms 0.0100  
## 6 population 0   
## 7 households 0   
## 8 median\_income 0   
## 9 median\_house\_value 0   
## 10 ocean\_proximity 0

median\_total\_bedrooms <- median(houses$total\_bedrooms, na.rm = TRUE)  
houses$total\_bedrooms[is.na(houses$total\_bedrooms)] <- median\_total\_bedrooms

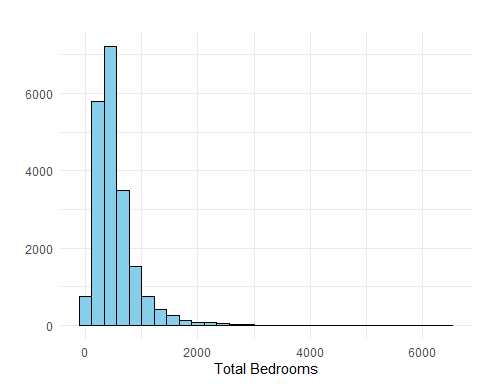
### A vizualization of the distribution of bedrooms

total\_bedrooms <- ggplot(houses, aes(x = total\_bedrooms)) +  
 geom\_boxplot(fill = "skyblue", color = "black") +  
 labs(title = "",  
 x = "Total Bedrooms",  
 y = "") +  
 theme\_minimal()  
total\_bedrooms

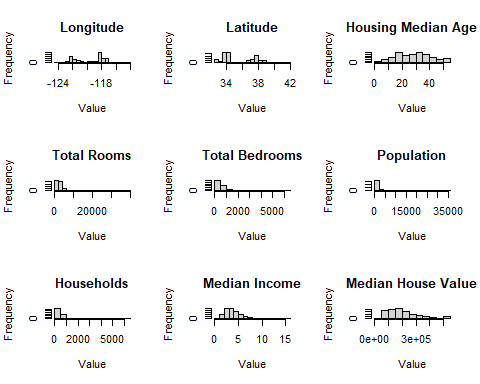


total\_bedrooms2 <- ggplot(houses, aes(x = total\_bedrooms)) +  
 geom\_histogram(fill = "skyblue", color = "black") +  
 labs(title = "",  
 x = "Total Bedrooms",  
 y = "") +  
 theme\_minimal()  
total\_bedrooms2

## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.

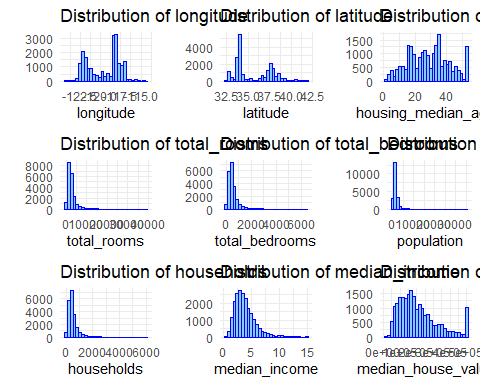
 ### let’s vizualize each numeric column with histograms to see the distribution of its data

par(mfrow=c(3, 3)) # allows us to insert several graphs in one  
  
hist(houses$longitude, main="Longitude", xlab="Value")  
hist(houses$latitude, main="Latitude", xlab="Value")  
hist(houses$housing\_median\_age, main="Housing Median Age", xlab="Value")  
hist(houses$total\_rooms, main="Total Rooms", xlab="Value")  
hist(houses$total\_bedrooms, main="Total Bedrooms", xlab="Value")  
hist(houses$population, main="Population", xlab="Value")  
hist(houses$households, main="Households", xlab="Value")  
hist(houses$median\_income, main="Median Income", xlab="Value")  
hist(houses$median\_house\_value, main="Median House Value", xlab="Value")



numeric\_columns <- c(  
 "longitude",  
 "latitude",  
 "housing\_median\_age",  
 "total\_rooms",  
 "total\_bedrooms",  
 "population",  
 "households",  
 "median\_income",  
 "median\_house\_value"  
)  
hists <- list()  
  
for (column in numeric\_columns) {  
 p <- ggplot(houses, aes(x = !!sym(column))) +  
 geom\_histogram(fill = "skyblue", col="blue") +  
 labs(title = paste("Distribution of", column),  
 x = column,  
 y = "") +  
 theme\_minimal()   
 hists[[column]] <- p  
}  
  
histograms <- grid.arrange(grobs = hists, ncol = 3)

## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.  
## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.  
## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.  
## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.  
## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.  
## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.  
## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.  
## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.  
## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.

 ### Creating a test set

# Let's split the dataset into training and testing sets (80% train, 20% test)  
set.seed(123) #to make this output identical at every run  
  
#Option 1   
training\_data <- sample.split(houses$median\_house\_value, SplitRatio = 0.8)  
  
# Creating the training and test sets  
train\_set <- houses[training\_data, ]  
test\_set <- houses[!training\_data, ]  
  
nrow(train\_set) #number of rows for each set

## [1] 16690

nrow(test\_set)

## [1] 3950

summary(train\_set)

## longitude latitude housing\_median\_age total\_rooms   
## Min. :-124.3 Min. :32.55 Min. : 1.0 Min. : 6   
## 1st Qu.:-121.8 1st Qu.:33.93 1st Qu.:18.0 1st Qu.: 1452   
## Median :-118.5 Median :34.26 Median :29.0 Median : 2135   
## Mean :-119.6 Mean :35.63 Mean :28.6 Mean : 2650   
## 3rd Qu.:-118.0 3rd Qu.:37.71 3rd Qu.:37.0 3rd Qu.: 3160   
## Max. :-114.3 Max. :41.95 Max. :52.0 Max. :39320   
## total\_bedrooms population households median\_income   
## Min. : 2.0 Min. : 3 Min. : 2.0 Min. : 0.4999   
## 1st Qu.: 298.0 1st Qu.: 784 1st Qu.: 280.0 1st Qu.: 2.5737   
## Median : 435.0 Median : 1166 Median : 409.5 Median : 3.5568   
## Mean : 537.8 Mean : 1425 Mean : 500.1 Mean : 3.8958   
## 3rd Qu.: 644.0 3rd Qu.: 1722 3rd Qu.: 605.8 3rd Qu.: 4.7891   
## Max. :6445.0 Max. :35682 Max. :6082.0 Max. :15.0001   
## median\_house\_value ocean\_proximity   
## Min. : 14999 Length:16690   
## 1st Qu.:120100 Class :character   
## Median :181000 Mode :character   
## Mean :208586   
## 3rd Qu.:267800   
## Max. :500001

summary(test\_set)

## longitude latitude housing\_median\_age total\_rooms   
## Min. :-124.3 Min. :32.54 Min. : 2.00 Min. : 2   
## 1st Qu.:-121.8 1st Qu.:33.93 1st Qu.:19.00 1st Qu.: 1437   
## Median :-118.5 Median :34.24 Median :29.00 Median : 2090   
## Mean :-119.6 Mean :35.63 Mean :28.82 Mean : 2577   
## 3rd Qu.:-118.0 3rd Qu.:37.71 3rd Qu.:37.00 3rd Qu.: 3096   
## Max. :-114.5 Max. :41.81 Max. :52.00 Max. :37937   
## total\_bedrooms population households median\_income   
## Min. : 1.0 Min. : 5 Min. : 1.0 Min. : 0.536   
## 1st Qu.: 294.0 1st Qu.: 795 1st Qu.: 278.0 1st Qu.: 2.517   
## Median : 435.0 Median : 1166 Median : 408.0 Median : 3.429   
## Mean : 532.6 Mean : 1426 Mean : 497.1 Mean : 3.764   
## 3rd Qu.: 640.0 3rd Qu.: 1746 3rd Qu.: 601.0 3rd Qu.: 4.591   
## Max. :5471.0 Max. :16122 Max. :5189.0 Max. :15.000   
## median\_house\_value ocean\_proximity   
## Min. : 14999 Length:3950   
## 1st Qu.:118225 Class :character   
## Median :175000 Mode :character   
## Mean :199544   
## 3rd Qu.:251725   
## Max. :500001

# Creating test set. Option 2  
# We need to define a function to check whether an instance should be in the test set, but before we need to find an identifier.   
# Since we do not have index or ID column, I will create ID column by combining latitude and longtitude columns together to make them identifiers for test\_set\_check function.   
  
  
houses$ID <- paste(houses$latitude, houses$longitude, sep = "\_")

set.seed(42)  
  
# Let's use createDataPartition function to split the data based on the identifier column,   
# it tries to ensure that each partition (train or test) has a representative sample of the unique values in the identifier column (ID)  
  
index <- createDataPartition(houses$ID, p = 0.7, list = FALSE) # Creating an index for splitting the data (70% training, 30% test)

## Warning in createDataPartition(houses$ID, p = 0.7, list = FALSE): Some classes  
## have a single record ( 32.54\_-117.04, 32.55\_-117.04, 32.55\_-117.06,  
## 32.55\_-117.09, 32.56\_-116.97, 32.56\_-117.07, 32.56\_-117.09, 32.56\_-117.1,  
## 32.56\_-117.12, 32.57\_-117.11, 32.58\_-117.07, 32.58\_-117.09, 32.58\_-117.16,  
## 32.59\_-117.02, 32.59\_-117.06, 32.59\_-117.07, 32.59\_-117.1, 32.59\_-117.12,  
## 32.6\_-117.03, 32.6\_-117.08, 32.61\_-116.79, 32.61\_-117.08, 32.62\_-116.98,  
## 32.62\_-117.05, 32.62\_-117.06, 32.62\_-117.07, 32.62\_-117.11, 32.63\_-117.01,  
## 32.63\_-117.05, 32.63\_-117.13, 32.63\_-117.17, 32.64\_-116.2, 32.64\_-116.99,  
## 32.64\_-117, 32.64\_-117.02, 32.64\_-117.04, 32.64\_-117.06, 32.64\_-117.1,  
## 32.64\_-117.11, 32.65\_-116.45, 32.65\_-116.97, 32.65\_-117.03, 32.65\_-117.04,  
## 32.65\_-117.07, 32.66\_-117.01, 32.66\_-117.02, 32.66\_-117.04, 32.66\_-117.08,  
## 32.66\_-117.1, 32.66\_-117.11, 32.66\_-117.12, 32.67\_-115.5, 32.67\_-115.52,  
## 32.67\_-116.89, 32.67\_-117, 32.67\_-117.01, 32.67\_-117.02, 32.67\_-117.03,  
## 32.67\_-117.04, 32.67\_-117.06, 32.67\_-117.08, 32.68\_-115.48, 32.68\_-115.5,  
## 32.68\_-115.51, 32.68\_-116.98, 32.68\_-117.02, 32.68\_-117.07, 32.68\_-117.11,  
## 32.68\_-117.17, 32.68\_-117.18, 32.69\_-115.49, 32.69\_-115.59, 32.69\_-115.9,  
## 32.69\_-116.58, 32.69\_-117.02, 32.69\_-117.13, 32.69\_-117.19, 32.7\_-115.4,  
## 32.7\_-116.99, 32.7\_-117, 32.7\_-117.03, 32.7\_-117.04, 32.7\_-117.06,  
## 32.7\_-117.07, 32.7\_-117.17, 32.71\_-116.96, 32.71\_-116.98, 32.71\_-116.99,  
## 32.71\_-117, 32.71\_-117.01, 32.71\_-117.04, 32.71\_-117.05, 32.71\_-117.08,  
## 32.71\_-117.24, 32.72\_-116.87, 32.72\_-117, 32.72\_-117.01, 32.72\_-117.02,  
## 32.72\_-117.04, 32.72\_-117.06, 32.72\_-117.07, 32.72\_-117.08, 32.72\_-117.1,  
## 32.72\_-117.11, 32.72\_-117.12, 32.72\_-117.17, 32.73\_-115.52, 32.73\_-115.53,  
## 32.73\_-116.91, 32.73\_-116.95, 32.73\_-116.98, 32.73\_-116.99, 32.73\_-117,  
## 32.73\_-117.01, 32.73\_-117.02, 32.73\_-117.05, 32.73\_-117.06, 32.73\_-117.07,  
## 32.73\_-117.1, 32.73\_-117.14, 32.73\_-117.22, 32.73\_-117.24, 32.73\_-117.28,  
## 32.74\_-114.66, 32.74\_-116, 32.74\_-116.35, 32.74\_-116.76, 32.74\_-116.95,  
## 32.74\_-116.97, 32.74\_-117, 32.74\_-117.01, 32.74\_-117.05, 32.74\_-117.28,  
## 32.75\_-115.5, 32.75\_-115.64, 32.75\_-115.72, 32.75\_-116.87, 32.75\_-116.91,  
## 32.75\_-116.94, 32.75\_-116.97, 32.75\_-117.02, 32.75\_-117.03, 32.75\_-117.04,  
## 32.75\_-117.28, 32.76\_-114.63, 32.76\_-115.39, 32.76\_-115.56, 32.76\_-116.95,  
## 32.76\_-116.99, 32.76\_-117.17, 32.76\_-117.19, 32.76\_-117.2, 32.76\_-117.25,  
## 32.77\_-115.52, 32.77\_-116.9, 32.77\_-116.92, 32.77\_-116.95, 32.77\_-116.98,  
## 32.77\_-116.99, 32.77\_-117.1, 32.77\_-117.13, 32.77\_-117.15, 32.77\_-117.28,  
## 32.78\_-115.55, 32.78\_-115.58, 32.78\_-116.91, 32.78\_-116.92, 32.78\_-116.94,  
## 32.78\_-116.96, 32.78\_-116.99, 32.78\_-117.01, 32.78\_-117.03, 32.78\_-117.05,  
## 32.78\_-117.06, 32.78\_-117.08, 32.78\_-117.09, 32.78\_-117.12, 32.78\_-117.15,  
## 32.78\_-117.18, 32.78\_-117.19, 32.78\_-117.2, 32.78\_-117.22, 32.78\_-117.23,  
## 32.78\_-117.24, 32.79\_-114.65, 32.79\_-115.58, 32.79\_-115.59, 32.79\_-115.69,  
## 32.79\_-116.66, 32.79\_-116.9, 32.79\_-116.92, 32.79\_-116.95, 32.79\_-116.98,  
## 32.79\_-117, 32.79\_-117.02, 32.79\_-117.04, 32.79\_-117.16, 32.79\_-117.18,  
## 32.8\_-114.55, 32.8\_-115.48, 32.8\_-115.55, 32.8\_-115.64, 32.8\_-115.73,  
## 32.8\_-116.8, 32.8\_-116.86, 32.8\_-116.91, 32.8\_-116.92, 32.8\_-116.93,  
## 32.8\_-116.97, 32.8\_-116.98, 32.8\_-116.99, 32.8\_-117.03, 32.8\_-117.06,  
## 32.8\_-117.11, 32.8\_-117.17, 32.8\_-117.19, 32.8\_-117.22, 32.8\_-117.26,  
## 32.8\_-117.28, 32.81\_-115.38, 32.81\_-116.83, 32.81\_-116.88, 32.81\_-116.91,  
## 32.81\_-116.93, 32.81\_-116.96, 32.81\_-116.97, 32.81\_-117.04, 32.81\_-117.05,  
## 32.81\_-117.06, 32.81\_-117.07, 32.81\_-117.09, 32.81\_-117.11, 32.81\_-117.14,  
## 32.81\_-117.15, 32.81\_-117.19, 32.81\_-117.29, 32.82\_-115.32, 32.82\_-115.38,  
## 32.82\_-115.55, 32.82\_-116.75, 32.82\_-116.77, 32.82\_-116.89, 32.82\_-116.91,  
## 32.82\_-116.93, 32.82\_-116.94, 32.82\_-117.03, 32.82\_-117.05, 32.82\_-117.16,  
## 32.82\_-117.18, 32.82\_-117.21, 32.82\_-117.24, 32.82\_-117.26, 32.82\_-117.31,  
## 32.83\_-115.57, 32.83\_-116.83, 32.83\_-116.85, 32.83\_-116.91, 32.83\_-116.94,  
## 32.83\_-116.95, 32.83\_-116.99, 32.83\_-117.01, 32.83\_-117.04, 32.83\_-117.08,  
## 32.83\_-117.09, 32.83\_-117.14, 32.83\_-117.17, 32.83\_-117.24, 32.83\_-117.25,  
## 32.83\_-117.26, 32.83\_-117.28, 32.83\_-117.31, 32.84\_-115.57, 32.84\_-116.28,  
## 32.84\_-116.76, 32.84\_-116.79, 32.84\_-116.92, 32.84\_-116.94, 32.84\_-116.95,  
## 32.84\_-117.01, 32.84\_-117.02, 32.84\_-117.11, 32.84\_-117.21, 32.84\_-117.22,  
## 32.84\_-117.25, 32.85\_-115.59, 32.85\_-116.75, 32.85\_-116.91, 32.85\_-116.92,  
## 32.85\_-116.94, 32.85\_-116.96, 32.85\_-116.98, 32.85\_-116.99, 32.85\_-117,  
## 32.85\_-117.01, 32.85\_-117.19, 32.85\_-117.22, 32.85\_-117.23, 32.85\_-117.3,  
## 32.86\_-115.4, 32.86\_-116.62, 32.86\_-116.84, 32.86\_-116.88, 32.86\_-116.92,  
## 32.86\_-116.96, 32.86\_-117.2, 32.86\_-117.22, 32.87\_-115.49, 32.87\_-115.6,  
## 32.87\_-116.86, 32.87\_-116.91, 32.87\_-116.93, 32.87\_-116.94, 32.87\_-116.96,  
## 32.87\_-117, 32.87\_-117.21, 32.88\_-116.98, 32.88\_-117.23, 32.89\_-116.94,  
## 32.89\_-117.16, 32.89\_-117.21, 32.9\_-116.52, 32.9\_-116.9, 32.9\_-116.96,  
## 32.9\_-117.04, 32.9\_-117.09, 32.9\_-117.1, 32.9\_-117.11, 32.9\_-117.13,  
## 32.9\_-117.14, 32.9\_-117.15, 32.91\_-117.07, 32.91\_-117.11, 32.91\_-117.12,  
## 32.91\_-117.14, 32.91\_-117.16, 32.92\_-116.84, 32.92\_-117.18, 32.92\_-117.29,  
## 32.93\_-115.88, 32.93\_-117.08, 32.93\_-117.12, 32.93\_-117.13, 32.93\_-117.15,  
## 32.94\_-117.13, 32.94\_-117.24, 32.95\_-117.02, 32.95\_-117.03, 32.95\_-117.11,  
## 32.95\_-117.18, 32.95\_-117.22, 32.95\_-117.24, 32.96\_-115.56, 32.96\_-115.59,  
## 32.96\_-116.95, 32.96\_-116.99, 32.96\_-117.03, 32.96\_-117.05, 32.96\_-117.1,  
## 32.96\_-117.13, 32.96\_-117.14, 32.96\_-117.21, 32.96\_-117.25, 32.96\_-117.26,  
## 32.96\_-117.3, 32.97\_-115.54, 32.97\_-116.67, 32.97\_-117.03, 32.97\_-117.04,  
## 32.97\_-117.05, 32.97\_-117.06, 32.97\_-117.08, 32.97\_-117.1, 32.97\_-117.11,  
## 32.97\_-117.13, 32.97\_-117.26, 32.98\_-115.54, 32.98\_-117.04, 32.98\_-117.09,  
## 32.98\_-117.13, 32.98\_-117.24, 32.98\_-117.26, 32.98\_-117.27, 32.98\_-117.31,  
## 32.99\_-115.41, 32.99\_-115.51, 32.99\_-115.53, 32.99\_-116.89, 32.99\_-117.01,  
## 32.99\_-117.04, 32.99\_-117.06, 32.99\_-117.23, 32.99\_-117.25, 32.99\_-117.26,  
## 32.99\_-117.27, 33.01\_-116.84, 33.01\_-116.99, 33.01\_-117.04, 33.01\_-117.08,  
## 33.01\_-117.23, 33.01\_-117.25, 33.01\_-117.32, 33.02\_-116.84, 33.02\_-116.86,  
## 33.02\_-116.88, 33.02\_-117.05, 33.02\_-117.07, 33.02\_-117.15, 33.02\_-117.18,  
## 33.02\_-117.21, 33.02\_-117.26, 33.02\_-117.28, 33.03\_-116.9, 33.03\_-117.04,  
## 33.03\_-117.06, 33.03\_-117.09, 33.03\_-117.21, 33.03\_-117.25, 33.03\_-117.33,  
## 33.04\_-115.6, 33.04\_-116.61, 33.04\_-117.01, 33.04\_-117.05, 33.04\_-117.06,  
## 33.04\_-117.07, 33.04\_-117.08, 33.04\_-117.24, 33.04\_-117.27, 33.04\_-117.28,  
## 33.04\_-117.29, 33.05\_-116.56, 33.05\_-116.86, 33.05\_-116.88, 33.05\_-117.05,  
## 33.05\_-117.25, 33.05\_-117.26, 33.05\_-117.27, 33.05\_-117.29, 33.05\_-117.3,  
## 33.06\_-116.6, 33.06\_-116.93, 33.06\_-117.16, 33.06\_-117.25, 33.06\_-117.26,  
## 33.06\_-117.27, 33.06\_-117.28, 33.06\_-117.29, 33.06\_-117.34, 33.07\_-114.98,  
## 33.07\_-116.26, 33.07\_-117.07, 33.07\_-117.1, 33.07\_-117.12, 33.07\_-117.14,  
## 33.07\_-117.2, 33.07\_-117.31, 33.08\_-116.77, 33.08\_-116.84, 33.08\_-117.08,  
## 33.08\_-117.26, 33.08\_-117.27, 33.08\_-117.29, 33.08\_-117.3, 33.09\_-115.73,  
## 33.09\_-116.58, 33.09\_-116.66, 33.09\_-117.04, 33.09\_-117.06, 33.09\_-117.08,  
## 33.09\_-117.1, 33.09\_-117.23, 33.1\_-117.05, 33.1\_-117.09, 33.1\_-117.23,  
## 33.1\_-117.25, 33.1\_-117.28, 33.1\_-117.29, 33.1\_-117.31, 33.11\_-117.05,  
## 33.11\_-117.09, 33.11\_-117.11, 33.11\_-117.18, 33.11\_-117.24, 33.11\_-117.31,  
## 33.12\_-115.51, 33.12\_-115.52, 33.12\_-117.03, 33.12\_-117.09, 33.12\_-117.1,  
## 33.12\_-117.11, 33.12\_-117.14, 33.12\_-117.2, 33.12\_-117.21, 33.12\_-117.25,  
## 33.12\_-117.29, 33.12\_-117.32, 33.13\_-115.52, 33.13\_-116.97, 33.13\_-117.03,  
## 33.13\_-117.06, 33.13\_-117.08, 33.13\_-117.09, 33.13\_-117.13, 33.13\_-117.29,  
## 33.14\_-117.05, 33.14\_-117.06, 33.14\_-117.07, 33.14\_-117.1, 33.14\_-117.11,  
## 33.14\_-117.15, 33.14\_-117.19, 33.14\_-117.2, 33.14\_-117.21, 33.14\_-117.22,  
## 33.14\_-117.38, 33.15\_-117.04, 33.15\_-117.06, 33.15\_-117.09, 33.15\_-117.1,  
## 33.15\_-117.11, 33.15\_-117.12, 33.15\_-117.18, 33.15\_-117.2, 33.15\_-117.27,  
## 33.15\_-117.29, 33.15\_-117.32, 33.16\_-116.68, 33.16\_-117.08, 33.16\_-117.14,  
## 33.16\_-117.15, 33.16\_-117.18, 33.16\_-117.2, 33.16\_-117.21, 33.16\_-117.33,  
## 33.17\_-117.06, 33.17\_-117.1, 33.17\_-117.21, 33.17\_-117.22, 33.17\_-117.24,  
## 33.17\_-117.3, 33.17\_-117.31, 33.17\_-117.32, 33.17\_-117.33, 33.17\_-117.36,  
## 33.18\_-117.03, 33.18\_-117.14, 33.18\_-117.17, 33.18\_-117.19, 33.18\_-117.22,  
## 33.18\_-117.24, 33.18\_-117.26, 33.18\_-117.27, 33.18\_-117.31,

# Training and splitting the data by ID column  
train\_set2 <- houses[index, ]  
test\_set2 <- houses[-index, ]  
  
summary(train\_set2)

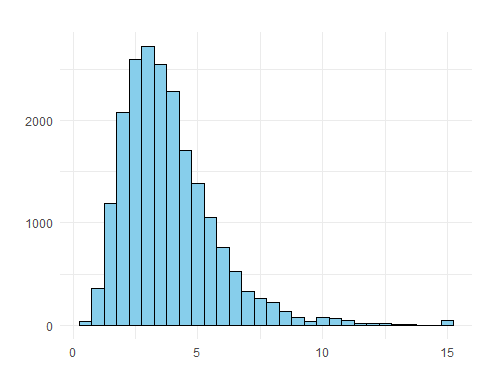
## longitude latitude housing\_median\_age total\_rooms   
## Min. :-124.3 Min. :32.54 Min. : 1.00 Min. : 2   
## 1st Qu.:-121.8 1st Qu.:33.93 1st Qu.:18.00 1st Qu.: 1462   
## Median :-118.5 Median :34.28 Median :28.00 Median : 2157   
## Mean :-119.6 Mean :35.66 Mean :28.17 Mean : 2674   
## 3rd Qu.:-118.0 3rd Qu.:37.72 3rd Qu.:37.00 3rd Qu.: 3193   
## Max. :-114.3 Max. :41.95 Max. :52.00 Max. :39320   
## total\_bedrooms population households median\_income   
## Min. : 1.0 Min. : 3 Min. : 1.0 Min. : 0.4999   
## 1st Qu.: 298.0 1st Qu.: 788 1st Qu.: 280.0 1st Qu.: 2.5768   
## Median : 435.0 Median : 1172 Median : 411.0 Median : 3.5551   
## Mean : 540.6 Mean : 1436 Mean : 502.8 Mean : 3.8960   
## 3rd Qu.: 648.0 3rd Qu.: 1739 3rd Qu.: 608.0 3rd Qu.: 4.7813   
## Max. :6445.0 Max. :35682 Max. :6082.0 Max. :15.0001   
## median\_house\_value ocean\_proximity ID   
## Min. : 14999 Length:19624 Length:19624   
## 1st Qu.:118800 Class :character Class :character   
## Median :179100 Mode :character Mode :character   
## Mean :206039   
## 3rd Qu.:264000   
## Max. :500001

head(train\_set2)

## longitude latitude housing\_median\_age total\_rooms total\_bedrooms population  
## 1 -122.23 37.88 41 880 129 322  
## 2 -122.22 37.86 21 7099 1106 2401  
## 3 -122.24 37.85 52 1467 190 496  
## 4 -122.25 37.85 52 1274 235 558  
## 5 -122.25 37.85 52 1627 280 565  
## 6 -122.25 37.85 52 919 213 413  
## households median\_income median\_house\_value ocean\_proximity ID  
## 1 126 8.3252 452600 NEAR BAY 37.88\_-122.23  
## 2 1138 8.3014 358500 NEAR BAY 37.86\_-122.22  
## 3 177 7.2574 352100 NEAR BAY 37.85\_-122.24  
## 4 219 5.6431 341300 NEAR BAY 37.85\_-122.25  
## 5 259 3.8462 342200 NEAR BAY 37.85\_-122.25  
## 6 193 4.0368 269700 NEAR BAY 37.85\_-122.25

# Let’s create a histogram for visualizig median income because some experts assume that this attribute is important in predicting median housing prices

med\_income\_hist <- ggplot(houses, aes(x = median\_income)) +  
 geom\_histogram(fill = "skyblue", color = "black", bins = 30) +  
 labs(title = "",  
 x = "",  
 y = "") +  
 theme\_minimal()  
med\_income\_hist

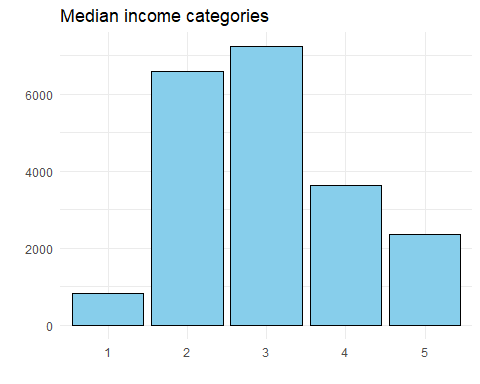


# Most median income values are clustered around 1 to 6 (Keep in mind that this values are capped. So 1-6 are basically $10,000–$60,000),   
# but some median incomes go far beyond 6. It is important to have a sufficient number of instances in our dataset for each stratum,   
# This means that we should not have too many strata, and each stratum should be large enough.   
# We will create an income category attribute with 5 categories (labeled from 1 to 5): category 1 ranges from 0 to 1.5   
# (i.e., less than $15,000), category 2 from 1.5 to 3, and so on:  
  
houses$income\_cat <- cut(houses$median\_income,  
 breaks = c(0, 1.5, 3.0, 4.5, 6.0, Inf),  
 labels = c(1, 2, 3, 4, 5))  
  
  
# Get value counts for the 'income\_cat' column  
income\_cat\_counts <- table(houses$income\_cat)  
income\_cat\_counts

##   
## 1 2 3 4 5   
## 822 6581 7236 3639 2362

### let’s make another hist with income categories

med\_income\_cat\_bar <- ggplot(houses, aes(x = factor(income\_cat))) +  
 geom\_bar(fill = "skyblue", color = "black") +  
 labs(title = "Median income categories",  
 x = "",  
 y = "") +  
 theme\_minimal()  
med\_income\_cat\_bar



# So now we can see that the most median income is in the category 3 which ranges from 3 to 4.5 (or in other words median income between $30k to $45k. Do not forget that it is in 1990!)  
# Now we are ready to do stratified sampling based on the income category.  
  
set.seed(42) # Set a random seed for reproducibility  
split <- createDataPartition(houses$income\_cat, p = 0.2, list = FALSE)  
  
# We will create another stratified training and test sets  
strat\_train\_set <- houses[split, ]  
strat\_test\_set <- houses[-split, ]  
  
# let's check the distribution of income categories in the test set  
income\_cat\_prop <- prop.table(table(strat\_test\_set$income\_cat))  
  
# I want to arrange the proportions in descending order  
income\_cat\_props\_desc <- income\_cat\_prop[order(-income\_cat\_prop)]  
income\_cat\_props\_desc

##   
## 3 2 4 5 1   
## 0.35059664 0.31885638 0.17632806 0.11442244 0.03979647

### Let’s find proportion of income categories on the whole dataset to compare how stratified sampling is different from random sampling.

income\_cat\_prop\_full <- prop.table(table(houses$income\_cat)) # the proportions of income categories in the whole dataset  
income\_cat\_prop\_full

##   
## 1 2 3 4 5   
## 0.03982558 0.31884690 0.35058140 0.17630814 0.11443798

# To compare between stratified and random sampling, we need to ensure that income\_cat column exists in test\_set2 too  
test\_set$income\_cat <- cut(test\_set$median\_income,  
 breaks = c(0, 1.5, 3.0, 4.5, 6.0, Inf),  
 labels = c(1, 2, 3, 4, 5))  
  
  
random\_income\_cat\_prop <- prop.table(table(test\_set$income\_cat)) # the proportions of income categories in the random test set  
random\_income\_cat\_prop

##   
## 1 2 3 4 5   
## 0.04632911 0.33240506 0.35544304 0.16784810 0.09797468

stratified\_income\_cat\_prop <- income\_cat\_props\_desc # the proportions of income categories in the stratified test set  
stratified\_income\_cat\_prop

##   
## 3 2 4 5 1   
## 0.35059664 0.31885638 0.17632806 0.11442244 0.03979647

### Let’s create a data frame to compare the proportions

compare\_prop <- data.frame(  
 "Overall" = income\_cat\_prop\_full,  
 "Random" = random\_income\_cat\_prop,  
 "Stratified" = stratified\_income\_cat\_prop  
)  
compare\_prop

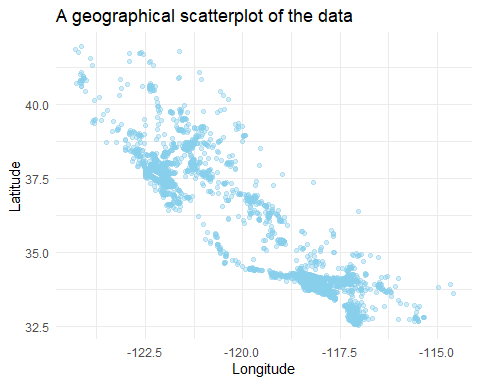
## Overall.Var1 Overall.Freq Random.Var1 Random.Freq Stratified.Var1  
## 1 1 0.03982558 1 0.04632911 3  
## 2 2 0.31884690 2 0.33240506 2  
## 3 3 0.35058140 3 0.35544304 4  
## 4 4 0.17630814 4 0.16784810 5  
## 5 5 0.11443798 5 0.09797468 1  
## Stratified.Freq  
## 1 0.35059664  
## 2 0.31885638  
## 3 0.17632806  
## 4 0.11442244  
## 5 0.03979647

# Now after exploring sampling bias comparison of stratified versus purely random sampling we should remove the income\_cat attribute so the data is back to its original state:  
  
houses <- subset(houses, select = -income\_cat) # removing income\_cat from houses  
  
test\_set2<- subset(test\_set, select = -income\_cat) # removing income\_cat from test\_set2  
  
test\_set <- subset(test\_set, select = -income\_cat) # removing income\_cat from test\_set

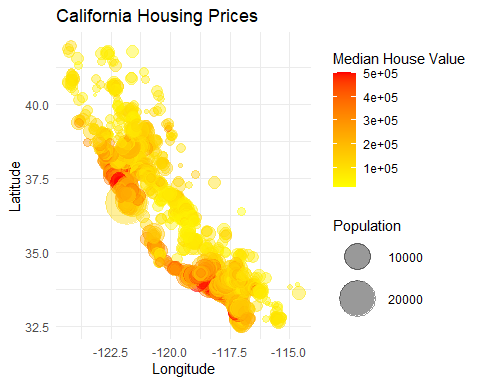
copy\_strat\_train\_set <- strat\_train\_set

### We will visualize geographical data

map <- ggplot(copy\_strat\_train\_set, aes(x = longitude, y = latitude)) +  
 geom\_point(color = "skyblue", alpha = 0.4) +  
 labs(title = "A geographical scatterplot of the data",  
 x = "Longitude",  
 y = "Latitude") +  
 theme\_minimal()  
map



library(ggplot2)  
map2 <- ggplot(copy\_strat\_train\_set, aes(x = longitude, y = latitude)) +  
 geom\_point(aes(size = population, color = median\_house\_value), alpha = 0.4) +  
 scale\_size\_continuous(range = c(1, 15)) +  
 scale\_color\_gradient(low = "yellow", high = "red") +  
 labs(  
 title = "California Housing Prices",  
 x = "Longitude",  
 y = "Latitude",  
 size = "Population",  
 color = "Median House Value"  
 ) +  
 theme\_minimal()+  
 guides(color = guide\_colorbar(title = "Median House Value"))  
  
  
map2

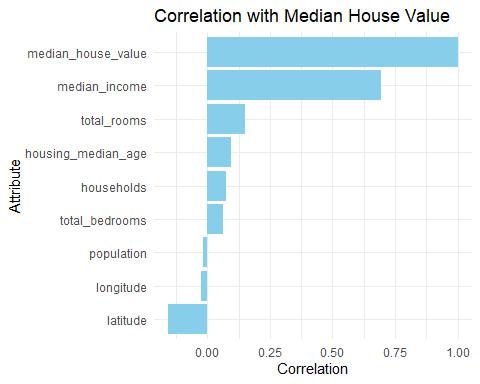


numeric\_attributes <- copy\_strat\_train\_set[, sapply(copy\_strat\_train\_set, is.numeric)]

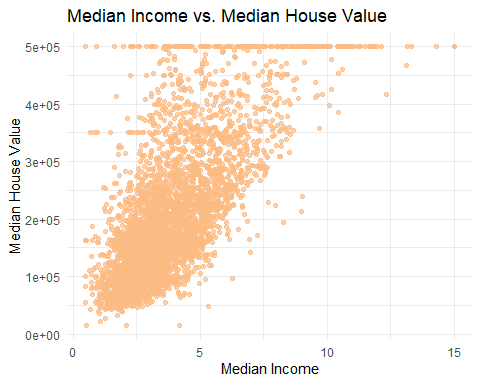
correlation\_matrix <- cor(numeric\_attributes)

median\_house\_value\_corr <- correlation\_matrix[,"median\_house\_value"]  
sorted\_correlation <- sort(median\_house\_value\_corr, decreasing = TRUE)

corr\_df <- data.frame(Attribute = names(sorted\_correlation), Correlation = sorted\_correlation)  
corr\_plot <- ggplot(corr\_df, aes(x = reorder(Attribute, Correlation), y = Correlation)) +  
 geom\_bar(stat = "identity", fill = "skyblue", alpha = 1.5) +  
 coord\_flip() +  
 labs(  
 title = "Correlation with Median House Value",  
 x = "Attribute",  
 y = "Correlation"  
 ) +  
 theme\_minimal()  
  
corr\_plot



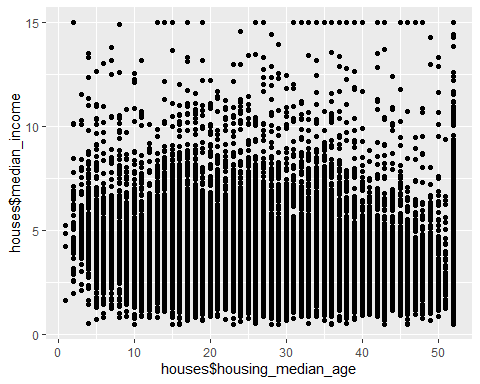
corr\_plot2 <- ggplot(copy\_strat\_train\_set, aes(x = median\_income, y = median\_house\_value)) +  
 geom\_point(color = "#fdbb84", alpha = 0.7) +  
 labs(  
 title = "Median Income vs. Median House Value",  
 x = "Median Income",  
 y = "Median House Value"  
 ) +  
 theme\_minimal()  
  
corr\_plot2



ggplot(houses,aes(x=houses$housing\_median\_age, y=houses$median\_income))+  
 geom\_point()

## Warning: Use of `houses$housing\_median\_age` is discouraged.  
## ℹ Use `housing\_median\_age` instead.

## Warning: Use of `houses$median\_income` is discouraged.  
## ℹ Use `median\_income` instead.



ggplot(houses,aes(x=houses$housing\_median\_age, y=houses$median\_income))+  
 geom\_violin()

## Warning: Use of `houses$housing\_median\_age` is discouraged.  
## ℹ Use `housing\_median\_age` instead.

## Warning: Use of `houses$median\_income` is discouraged.  
## ℹ Use `median\_income` instead.

