ISYE 6501 Homework 10

2023-10-24

Question 14.1

The breast cancer data set breast-cancer-wisconsin.data.txt has missing values.

Part 1

Use the mean/mode imputation method to impute values for the missing data.

First, I loaded and cleaned the data set. As you can see from the summary, there are 16 missing data points in the Bare_nuclei column.

```
#load data
cancer_data <- read.csv("~/Z. OMSA/Intro to Analytics Modeling/Week 10 Homework/breast-cancer-wisconsin
                        header=FALSE)

#add column names
colnames(cancer_data) <- c("Sample_code_number",
                           "Clump_thickness",
                           "Uniformity_of_cell_size",
                           "Uniformity_of_cell_shape",
                           "Marginal_adhesion",
                           "Single_epithelial_size",
                           "Bare_nuclei",
                           "Bland_chromatin",
                           "Normal_nucleoli",
                           "Mitoses",
                           "Class")

#update binary variables
cancer_data$Class[cancer_data$Class == 2] <- 0 #benign
cancer_data$Class[cancer_data$Class == 4] <- 1 #malignant

#change ? to NA
cancer_data[cancer_data == "?"] <- NA

#make Bare_nuclei column numeric
cancer_data$Bare_nuclei <- as.numeric(cancer_data$Bare_nuclei)

#summarize data
summary(cancer_data)
```

```
##  Sample_code_number Clump_thickness  Uniformity_of_cell_size
##  Min.   :   61634   Min.   : 1.000   Min.   : 1.000
```

```
##  1st Qu.:  870688   1st Qu.: 2.000   1st Qu.: 1.000
##  Median : 1171710   Median : 4.000   Median : 1.000
##  Mean   : 1071704   Mean   : 4.418   Mean   : 3.134
##  3rd Qu.: 1238298   3rd Qu.: 6.000   3rd Qu.: 5.000
##  Max.   :13454352   Max.   :10.000   Max.   :10.000
##
##  Uniformity_of_cell_shape Marginal_adhesion Single_epithelial_size
##  Min.   : 1.000           Min.   : 1.000    Min.   : 1.000
##  1st Qu.: 1.000           1st Qu.: 1.000    1st Qu.: 2.000
##  Median : 1.000           Median : 1.000    Median : 2.000
##  Mean   : 3.207           Mean   : 2.807    Mean   : 3.216
##  3rd Qu.: 5.000           3rd Qu.: 4.000    3rd Qu.: 4.000
##  Max.   :10.000           Max.   :10.000    Max.   :10.000
##
##   Bare_nuclei     Bland_chromatin  Normal_nucleoli     Mitoses
##  Min.   : 1.000   Min.   : 1.000   Min.   : 1.000   Min.   : 1.000
##  1st Qu.: 1.000   1st Qu.: 2.000   1st Qu.: 1.000   1st Qu.: 1.000
##  Median : 1.000   Median : 3.000   Median : 1.000   Median : 1.000
##  Mean   : 3.545   Mean   : 3.438   Mean   : 2.867   Mean   : 1.589
##  3rd Qu.: 6.000   3rd Qu.: 5.000   3rd Qu.: 4.000   3rd Qu.: 1.000
##  Max.   :10.000   Max.   :10.000   Max.   :10.000   Max.   :10.000
##  NA's   :16
##      Class
##  Min.   :0.0000
##  1st Qu.:0.0000
##  Median :0.0000
##  Mean   :0.3448
##  3rd Qu.:1.0000
##  Max.   :1.0000
##
```

As you can see in the code block below, the 16 missing data points represent less than 5% of the data set. Therefore, it is appropriate to use imputation methods to address missing data.

```
#total missing data points
missing_data <- sum(is.na(cancer_data))
missing_data
```

```
## [1] 16
```

```
#percentage of missing data points
pct_missing <- missing_data/nrow(cancer_data)
pct_missing
```

```
## [1] 0.02288984
```

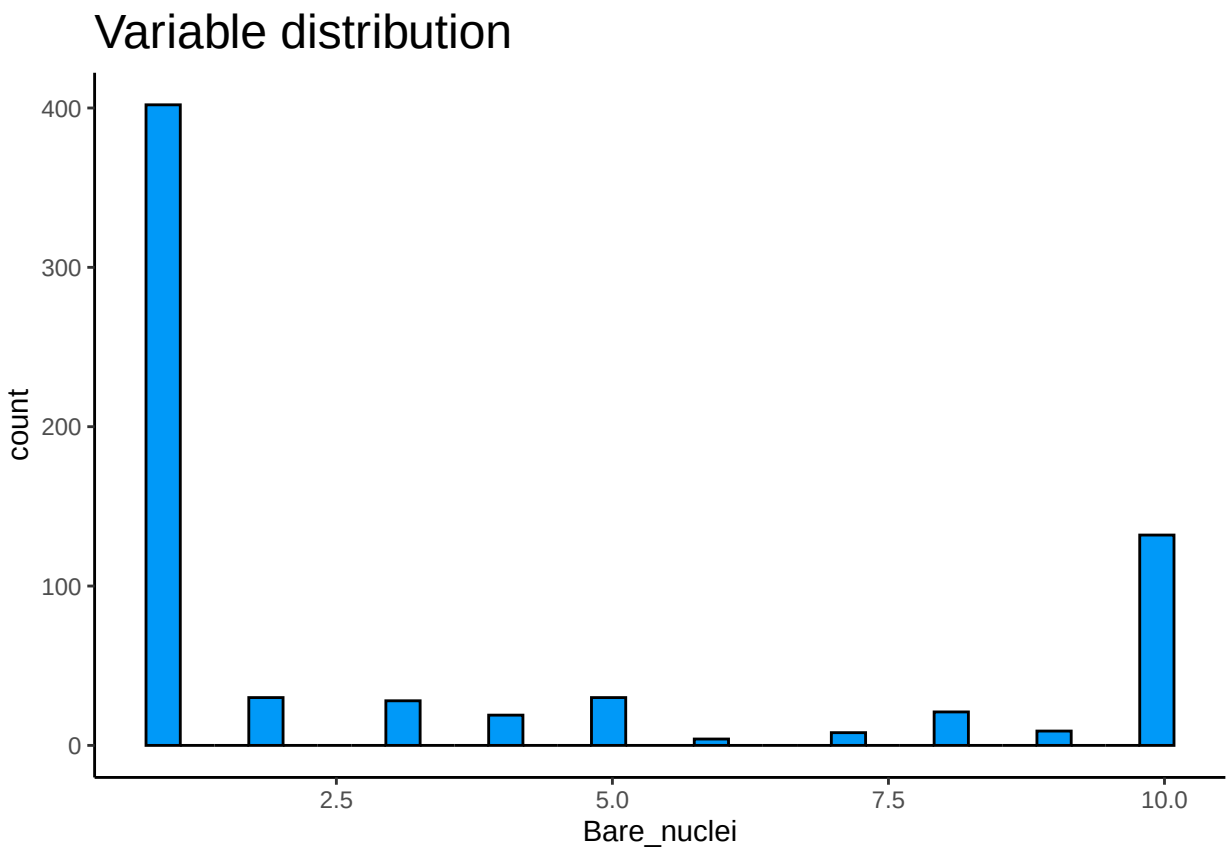
Next, I created a histogram to visualize the spread of Bare_nuceli data. I used this as a baseline to compare with my imputed data sets. The imputed data sets should have a very similar distribution to the baseline.

```
#visualize spread of data
ggplot(cancer_data, aes(Bare_nuclei)) +
  geom_histogram(color = "#000000", fill = "#0099F8") +
```

```
  ggtitle("Variable distribution") +
  theme_classic() +
  theme(plot.title = element_text(size = 18))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
## Warning: Removed 16 rows containing non-finite values (`stat_bin()`).
```



Then, I calculated the mean of the Bare_nuclei column and used this value to replace the missing data.

```
#copy data set
cancer_data_mean <- cancer_data

#impute with mean
mean <- round(mean(cancer_data_mean$Bare_nuclei, na.rm=TRUE), 2)
print(paste0("Mean: ", mean))
```

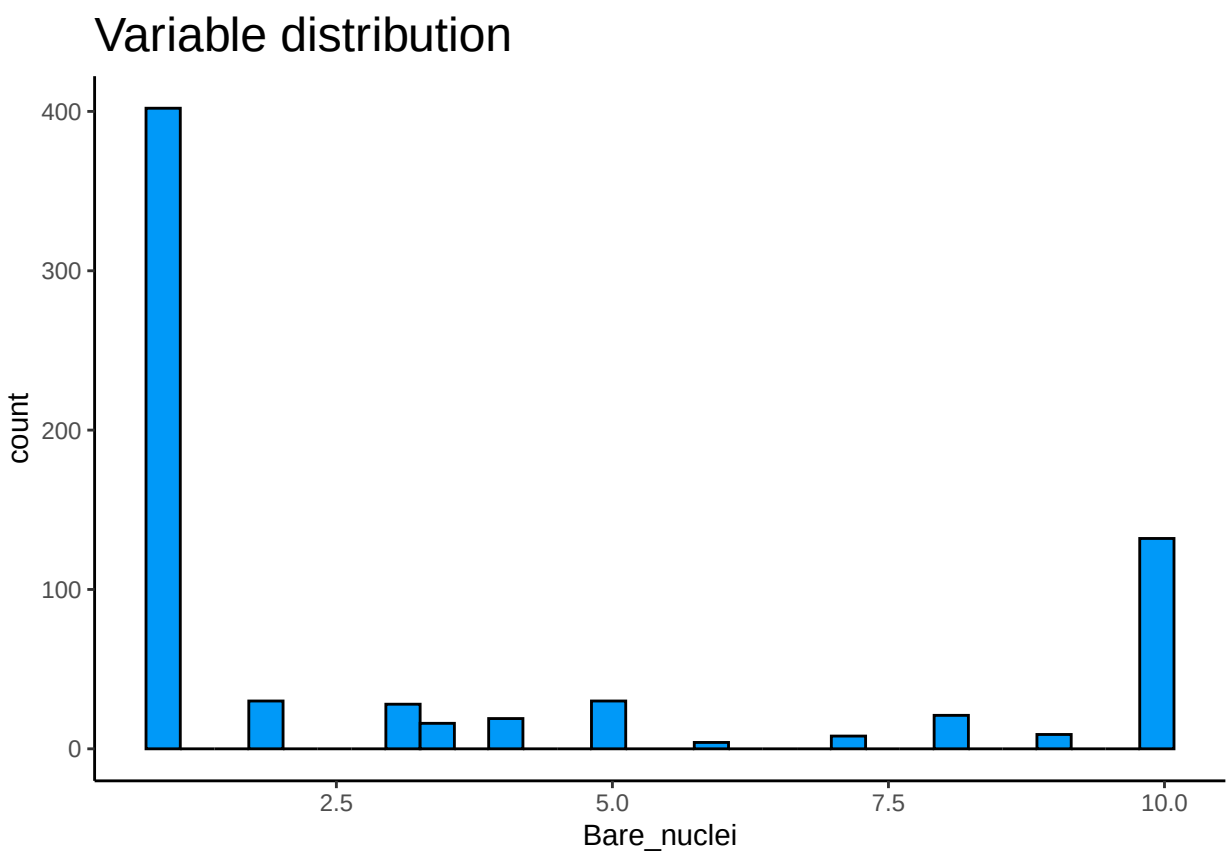
```
## [1] "Mean: 3.54"
```

```
cancer_data_mean[is.na(cancer_data_mean)] <- mean
```

The code block below shows the new spread of data.

```
#visualize spread of data after imputation
ggplot(cancer_data_mean, aes(Bare_nuclei)) +
  geom_histogram(color = "#000000", fill = "#0099F8") +
  ggtitle("Variable distribution") +
  theme_classic() +
  theme(plot.title = element_text(size = 18))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



I also used the mode to impute missing values.

```
#copy data set
cancer_data_mode <- cancer_data

#impute with mode
mode <- Mode(cancer_data_mode$Bare_nuclei, na.rm =TRUE)
print(paste0("Mode: ", mode))
```

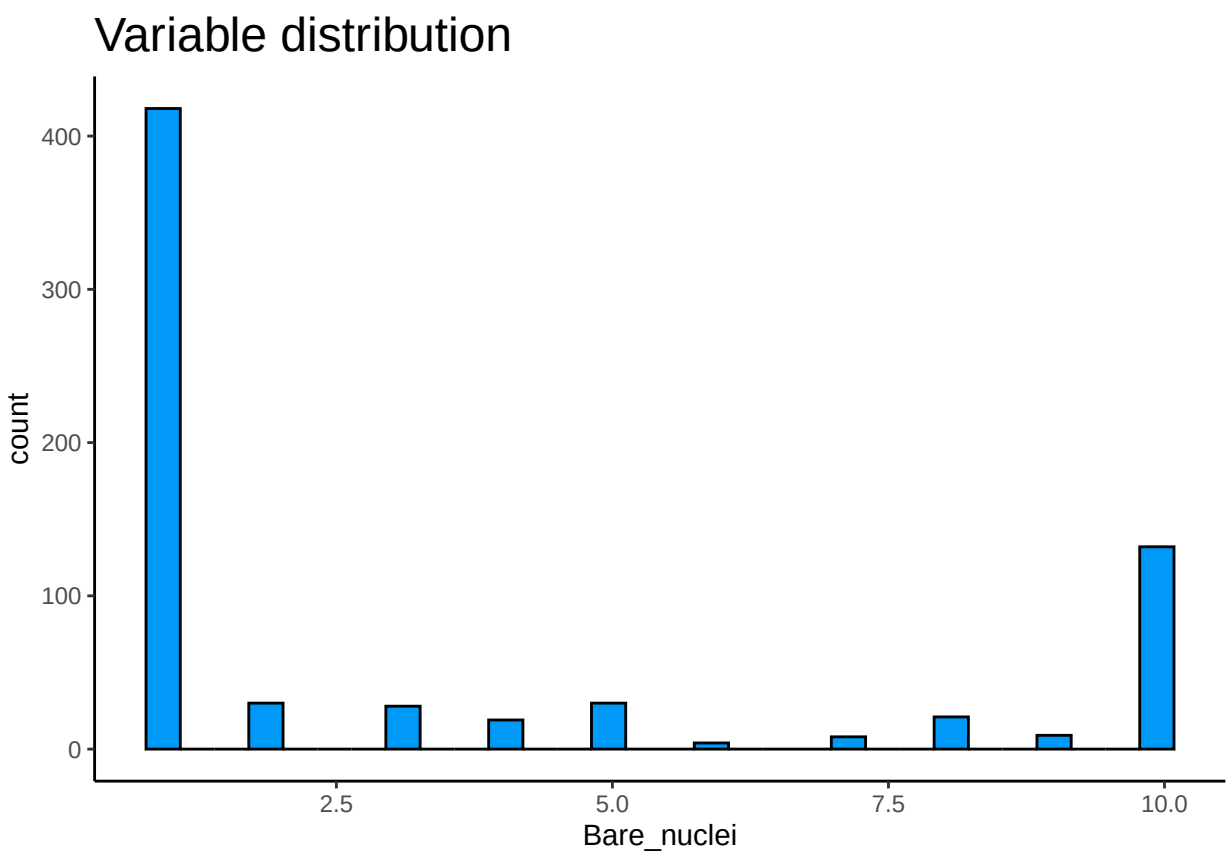
```
## [1] "Mode: 1"
```

```
cancer_data_mode[is.na(cancer_data_mode)] <- mode
```

The code block below shows the new spread of data.

```
#visualize spread of data after imputation
ggplot(cancer_data_mode, aes(Bare_nuclei)) +
  geom_histogram(color = "#000000", fill = "#0099F8") +
  ggtitle("Variable distribution") +
  theme_classic() +
  theme(plot.title = element_text(size = 18))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Part 2

Use regression to impute values for the missing data.

I used the mice() function to automate the process of imputation with regression. I input my data set, a method for imputation ("norm.predict" for linear regression), and a value for "m" (the number of multiple imputations). The code block below shows the imputed values for the missing data over 1 cycle.

```
#copy data set
cancer_data_regression <- cancer_data

#impute with regression using mice()
mice_data <- mice(cancer_data_regression, method = "norm.predict", m=1)
```

```
##
```

```
##  iter imp variable
##   1   1  Bare_nuclei
##   2   1  Bare_nuclei
##   3   1  Bare_nuclei
##   4   1  Bare_nuclei
##   5   1  Bare_nuclei
```

```
#see imputed values over the five trials
mice_data[["imp"]]$Bare_nuclei
```

```
##            1
## 24  7.191237
## 41  3.419208
## 140 1.188951
## 146 1.579936
## 159 1.260453
## 165 1.428797
## 236 1.943842
## 250 1.562574
## 276 1.740990
## 293 6.432884
## 295 1.303253
## 298 1.186205
## 316 2.083334
## 322 1.469119
## 412 1.179806
## 618 1.059370
```

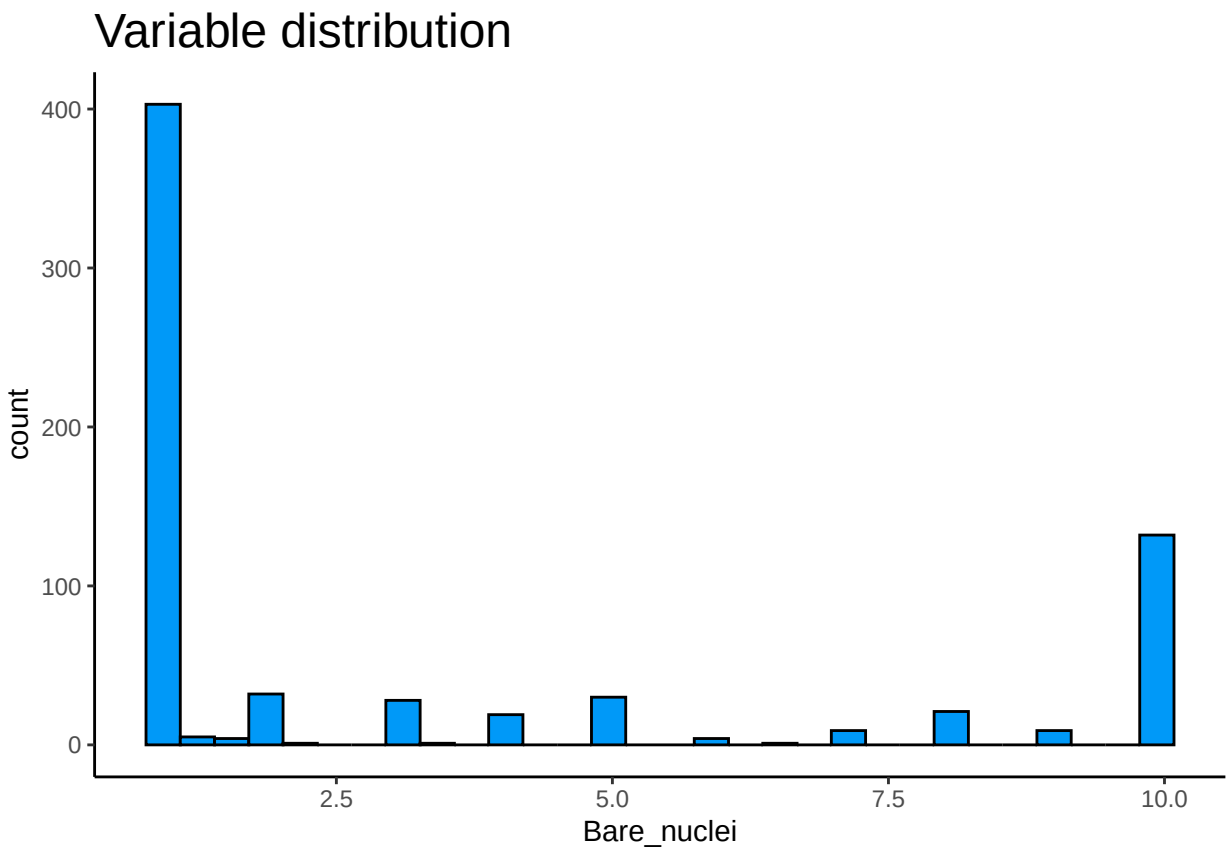
I used the complete() function to replace missing values in my original data set with the values imputed by the mice() function.

```
#update original data set
cancer_data_regression <- complete(mice_data, action = 1)
```

The code block below shows the new spread of data.

```
#visualize spread of data after imputation
ggplot(cancer_data_regression, aes(Bare_nuclei)) +
  geom_histogram(color = "#000000", fill = "#0099F8") +
  ggtitle("Variable distribution") +
  theme_classic() +
  theme(plot.title = element_text(size = 18))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Part 3

Use regression with perturbation to impute values for the missing data.

I also used the mice() function to impute values using regression with perturbation. I used the "norm.nob" instead of the "norm.predict" method to impute with added variability.

```
#copy data set
cancer_data_regression_perturb <- cancer_data

#imputed values
mice_data <- mice(cancer_data_regression_perturb, method = "norm.nob", m=1)
```

```
##
##  iter imp variable
##   1   1  Bare_nuclei
##   2   1  Bare_nuclei
##   3   1  Bare_nuclei
##   4   1  Bare_nuclei
##   5   1  Bare_nuclei
```

```
mice_data[["imp"]]$Bare_nuclei
```

```
##              1
```

```
## 24   7.4977412
## 41   2.3875268
## 140  0.3901479
## 146 -1.2845248
## 159 -2.7607492
## 165  2.1056021
## 236  3.2466977
## 250  6.4299315
## 276  4.1241822
## 293  8.2781692
## 295  2.5457169
## 298  1.8990268
## 316  2.2256577
## 322  2.9330411
## 412  2.4196539
## 618  0.5741458
```

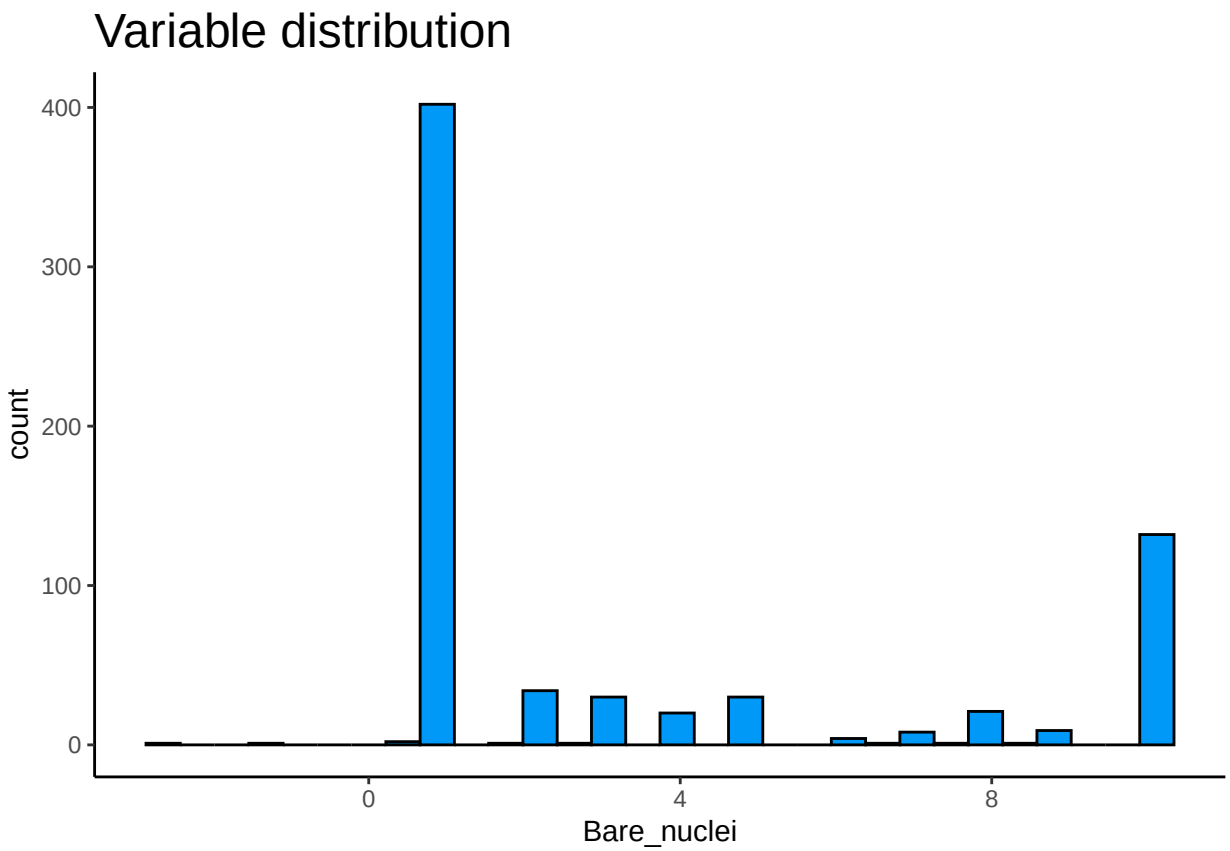
Then, I used the complete() function to merge the imputed values with my original data set.

```
#update original data set
cancer_data_regression_perturb <- complete(mice_data, action=1)
```

The code block below shows the new spread of data.

```
#visualize spread of data after imputation
ggplot(cancer_data_regression_perturb, aes(Bare_nuclei)) +
  geom_histogram(color = "#000000", fill = "#0099F8") +
  ggtitle("Variable distribution") +
  theme_classic() +
  theme(plot.title = element_text(size = 18))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Part 4 (Optional)

Compare the results and quality of classification models (e.g., SVM, KNN) build using the data sets from questions 1, 2, and 3.

As you can see in the code block below, knn model accuracy is the same for all 4 data sets. This is likely because very few values were changed in each data set. If more values were imputed, it is likely we would observe more variability.

```
#create global variables
sample <- sample(1:nrow(cancer_data), 0.6*nrow(cancer_data))
data_set_list <- list(cancer_data_mean, cancer_data_mode, cancer_data_regression, cancer_data_regressio
data_set_descriptions <- list("Imputation with Mean ",
                              "Imputation with Mode ",
                              "Imputation with Regression ",
                              "Imputation with Regression with Perturbation ")
ind <- 1

#for loop to fit knn model with each imputed data set
for(data_set in data_set_list){

  #split data into training and test sets
  train_data <- data_set[sample,]
  test_data <- data_set[-sample,]
```

```
  #fit knn model
  knn_model <- kknn(Class~., train_data, test_data, kernel="rectangular", scale=TRUE)

  #make predictions and determine accuracy
  predictions <- round(predict(knn_model))
  accuracy <- sum(test_data$Class == predictions)/nrow(test_data)

  #print result
  print(paste0(data_set_descriptions[[ind]], "model accuracy: ", round(accuracy, 4)*100, "%"))
  ind <- ind+1
}
```

```
## [1] "Imputation with Mean model accuracy: 96.79%"
## [1] "Imputation with Mode model accuracy: 96.79%"
## [1] "Imputation with Regression model accuracy: 96.79%"
## [1] "Imputation with Regression with Perturbation model accuracy: 96.79%"
```

Question 15.1

Describe a situation or problem from your job, everyday life, current events, etc., for which optimization would be appropriate. What data would you need?

Google Maps uses optimization methods to provide users with efficient and real-time route recommendations. The technology considers traffic estimates, distance to location, speed limits, road closures, and driver preferences (e.g., avoid highways), among other factors to suggest a route that minimizes travel time.