Spring 2023 **Firstname M. Lastname**

University of Central Florida
College of Business

QMB 6911
Capstone Project in Business Analytics

Solutions: Problem Set #2

0.1 Introduction

This note summarizes the findings in the script `Tractor_Price_Density.R`, which analyzes the prices of tractors, the dependent variable in the `TRACTOR7.csv` dataset. The output includes plots of the histogram and kernel-smoothed densities, using a selection of tuning parameters.

0.2 Data Description

This analysis follows the script `Tractor_Reg_Model.R` to produce a more accurate model for used tractor prices with the data from `TRACTOR7.csv` in the `Data` folder. The dataset includes the following variables.

$saleprice_i$	=	the price paid for tractor i in dollars
$horsepower_i$	=	the horsepower of tractor i
age_i	=	the number of years since tractor i was manufactured
$enghours_i$	=	the number of hours of use recorded for tractor i
$diesel_i$	=	an indicator of whether tractor i runs on diesel fuel
fwd_i	=	an indicator of whether tractor i has four-wheel drive
$manual_i$	=	an indicator of whether tractor i has a manual transmission
$johndeere_i$	=	an indicator of whether tractor i is manufactured by John Deere
cab_i	=	an indicator of whether tractor i has an enclosed cab
$spring_i$	=	an indicator of whether tractor i was sold in April or May
$summer_i$	=	an indicator of whether tractor i was sold between June and September
$winter_i$	=	an indicator of whether tractor i was sold between December and March

0.3 Relative Histogram of Tractor Prices

First plot a histogram with the default options.

```
hist(tractor_sales[, 'saleprice'],
     main = 'Relative Histogram of Tractor Prices',
     xlab = 'Price',
     probability = TRUE)
```

Figure 1 is a histogram of the tractor prices generated by the code block above. Notice that there are some very large values. Consider taking logs

Figure 1: Relative Histogram of Tractor Prices

to bring outliers closer to the others.

```
# Generate a new variable log_saleprice.
tractor_sales[, 'log_saleprice'] <- log(tractor_sales[, 'saleprice'])
```

Now plot the histogram for log of saleprice, depicted in Figure 2.

```
hist(tractor_sales[, 'log_saleprice'],
     main = 'Histogram of the Logarithm of Tractor Prices',
     xlab = 'Logarithm of Price',
     probability = TRUE)
```

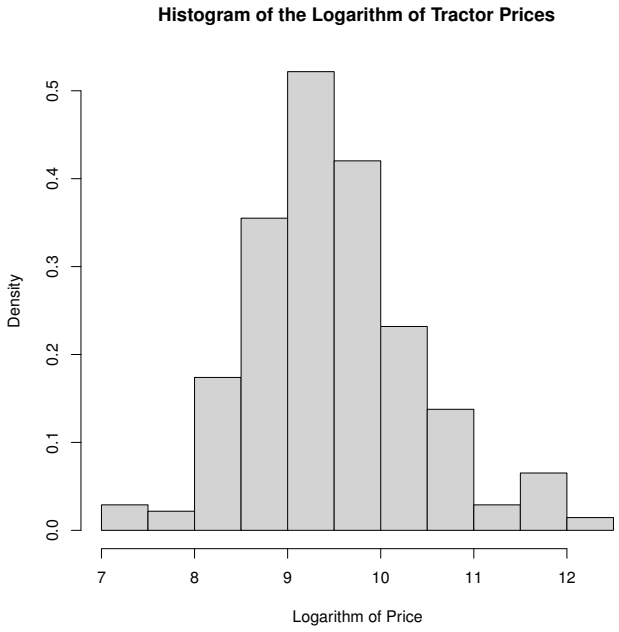

Figure 2: Relative Histogram of the Logarithm of Tractor Prices

With this transformation, the variable appears much better behaved. It looks almost normally distributed.

Now we will consider adjusting the tuning parameter for the number of bars in the chart for the histogram. A low number of breaks may give a smoother plot but it may not be very informative, as in Figure 3.

```
hist(tractor_sales[, 'log_saleprice'], breaks = 5,
     main = c('Histogram of the Logarithm of Tractor Prices',
              'Number of Breaks: 5'),
     xlab = 'Logarithm of Price',
     probability = TRUE)
```

On the other extreme, a high number of breaks gives a sparsely populated and jagged plot. Figure 4 shows such an example.

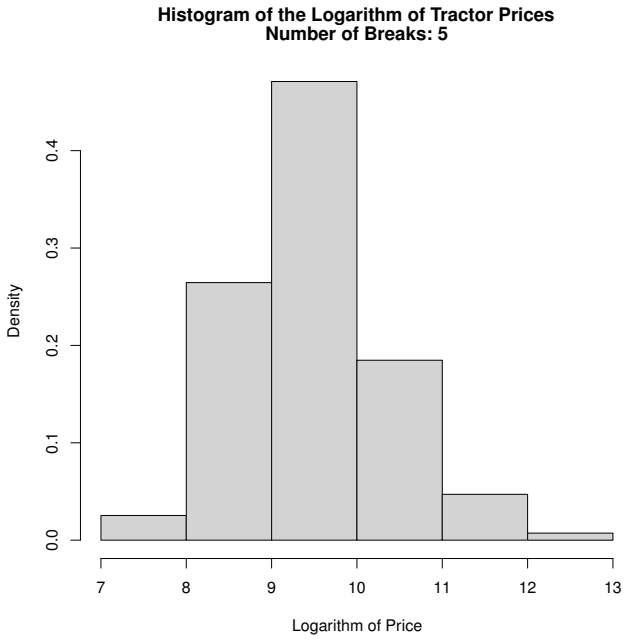

Figure 3: Relative Histogram of the Logarithm of Tractor Prices (Breaks: 5)

```
hist(tractor_sales[, 'log_saleprice'], breaks = 50,
     main = c('Histogram of the Logarithm of Tractor Prices',
              'Number of Breaks: 5'),
     xlab = 'Logarithm of Price',
     probability = TRUE)
```

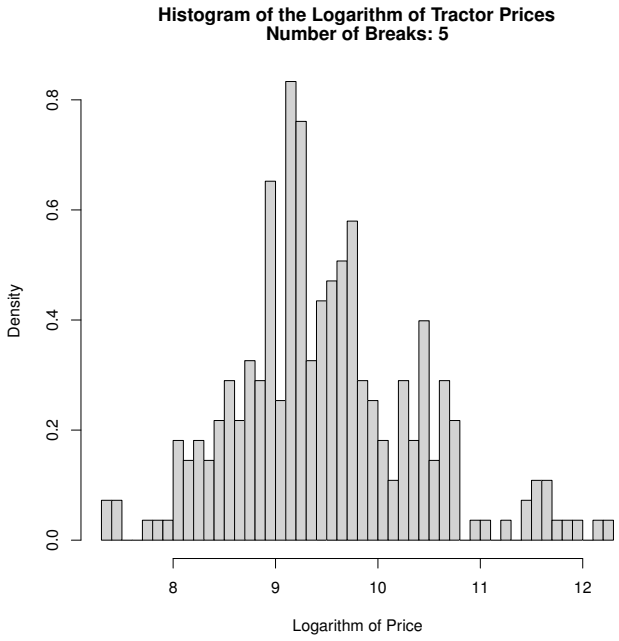

Figure 4: Relative Histogram of the Logarithm of Tractor Prices (Breaks: 50)

In the limit, the bars can be very small to the point that each bar is a pixel wide: approximating a continuous function. To smooth it out, the nonparametric technique of kernel smoothing with produce a continuous function.

0.4 Probability Density Function of Tractor Prices

Kernel-density smoothing is an example of a nonparametric method, that is, a model without parameters, such as the slope coefficients β in a regression model. You may have used nonparametric methods to plot a density. In fact, we have just used a rudimentary form of nonparanetric method when we plotted the histograms above.

Figure 5 depicts the kernel-smoothed probability density function of the natural logarithm of price.

```
price_density <- density(tractor_sales[, 'saleprice'])
plot(price_density,
     main = 'Kernel-Smoothed Density of Tractor Prices',
     xlab = 'Price')
```

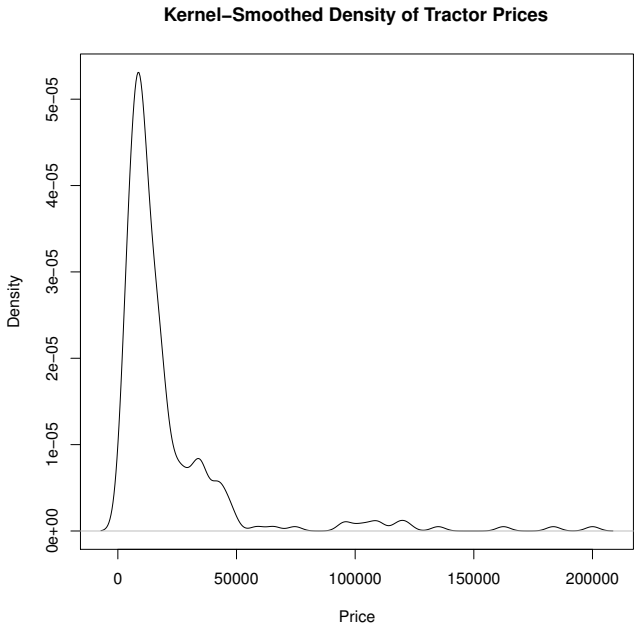

Figure 5: Probability Density Function of Tractor Prices

In the default, the bandwidth is chosen using an algorithm. See the help for density.

```
> attributes(price_density)
$names
[1] "x"         "y"         "bw"        "n"
[5] "call"      "data.name" "has.na"

$class
[1] "density"

> price_density$bw
[1] 2875.455
>
```

The default algorithm for this variable sets the bandwidth to $\$2,875$. You can choose the bandwidth as a tuning parameter. Try a larger value as in the following code block.

```
price_density <- density(tractor_sales[, 'saleprice'],
                         bw = 10000)
plot(price_density,
     main = c('Kernel-Smoothed Density of Tractor Prices',
              'Bandwidth: 10,000'),
     xlab = 'Price')
```

Figure 6 depicts the kernel-smoothed probability density function with a bandwidth of $10,000$.

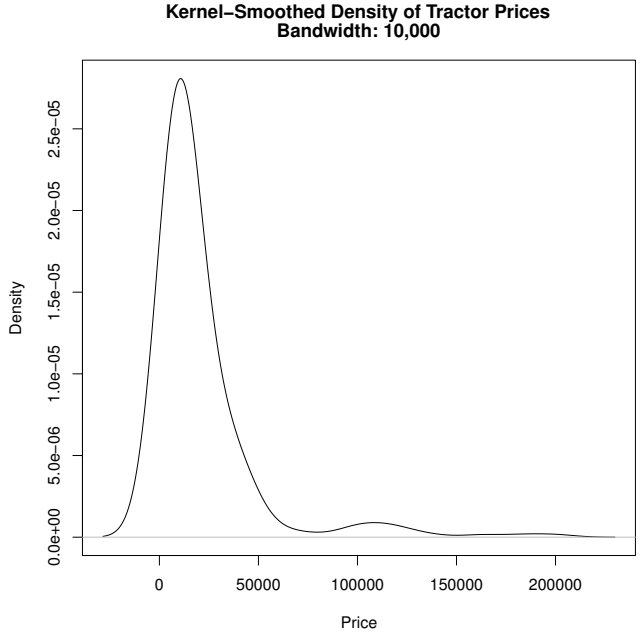

Figure 6: Probability Density Function of Tractor Prices (Bandwidth: $10,000$)

A bigger bandwidth gives you a smooth density, but might smooth over the details. A smaller bandwidth might make the density too noisy.

```
price_density <- density(tractor_sales[, 'saleprice'],
                         bw = 1000)
plot(price_density,
     main = c('Kernel-Smoothed Density of Tractor Prices',
              'Bandwidth: 1,000'),
     xlab = 'Price')
```

In Figure 7, we see many jagged changes that are more closely related with the particular values observed than with the population density.

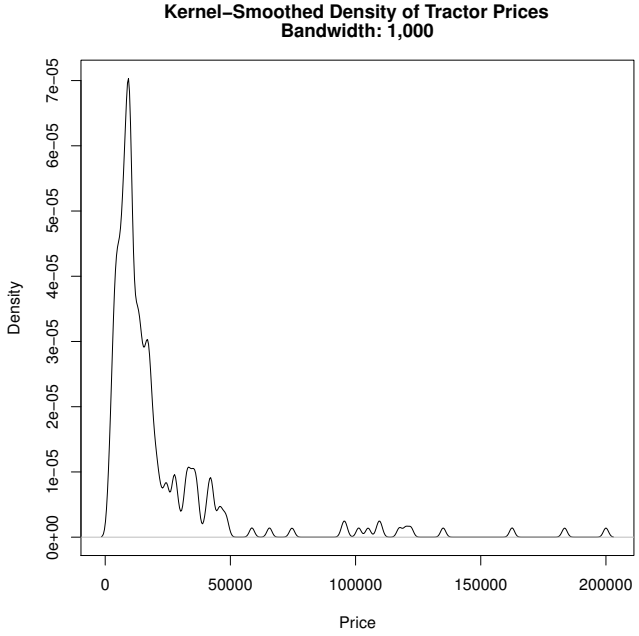

Figure 7: Probability Density Function of Tractor Prices (Bandwidth: $10,00$)

We can do something similar to predict one variable with the others. Before that, we will transform the dependent variable, after analyzing it in greater detail in future problem sets. For now, we can plot the logarithm of the dependent variable, with a bandwidth of 0.20, which is appropriate for the price levels on a log scale..

```
log_price_density <- density(tractor_sales[, 'log_saleprice'],
                             bw = 0.20)
plot(log_price_density,
     main = c('Density of the Logarithm of Tractor Prices',
              'Bandwidth: 0.20'),
     xlab = 'Logarithm of Price')
```

The density plot shown in Figure 8 is much better behaved and is a good starting point to analyze the data in a regression model.

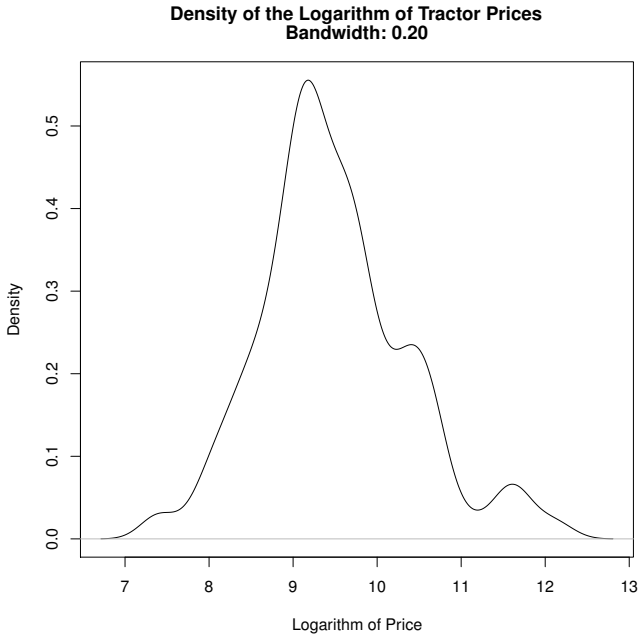

Figure 8: Probability Density Function of the Logarithm of Tractor Prices (Bandwidth: 0.20)

0.5 Separate Probability Density Function by Make

Now we investigate the value of John Deere tractors compared to other brands. We can analyze the log of sale prices because prices are highly skewed. After taking logs, the distribution is approximately symmetric, with some bunching in the upper tail. Now separate the sample by make of tractor, with John Deere tractors in one usbsample and the rest in the other subsample.

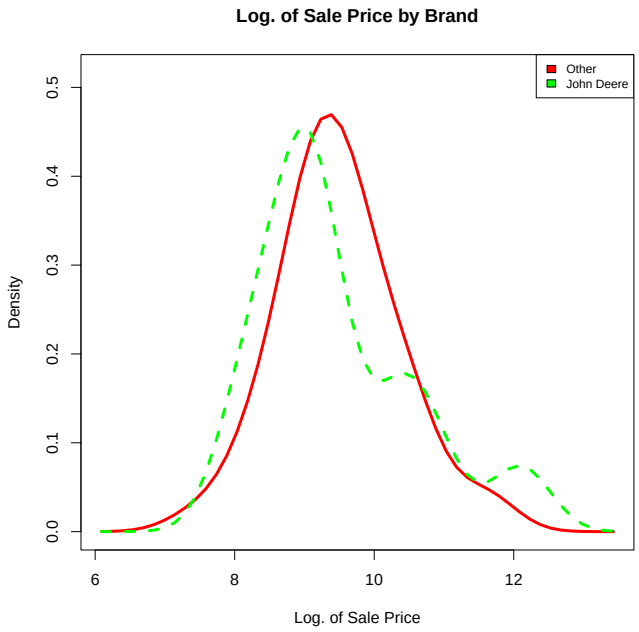

Figure 9: Densities of Log. Tractor Prices by Brand

Figure 9 shows the kernel density estimate of the sale prices of John Deere tractors in green and that of the other brands in red. The distribution of sale prices of John Deere tractors has several modes and is skewed to the right, with the highest mode lower than that for other brands.