Practice0 - Stat 240

Umaimah Ahmed

2025-02-13

The overall assignment is due by midnight, Thursday, Feb. 13th to Gradescope. The intermediate part of the assignment - see separate checklist and instructions - is due by midnight, Friday, Feb. 6th to Gradescope.

Practicing Academic Integrity

If you worked with others or used resources outside of provided course material (anything besides our textbook(s), course materials in Moodle, R help menu) to complete this assignment, please acknowledge them below using a bulleted list.

I acknowledge the following individuals with whom I worked on this assignment:

Name(s) and what they helped with

-

I used the following sources to help complete this assignment:

Source(s) and where you used them

- Boehmke, B., & Greenwell, B.M. (2019). Hands-On Machine Learning with R (1st ed.). Chapman and Hall/CRC. https://doi.org/10.1201/9780367816377

Prompt

You've been retained as a statistical consultant for the company WidgetsRUs, which sells widgets. They have three stores in a particular city, and they are trying to develop a model to predict the daily number of a particular widget sold based on some other variables: price of the widget that day (dollars), whether there was a sale on the widget that day (yes/no), the inside (store) and outside temperatures (degrees Fahrenheit), whether the weather was good or bad, whether there was a sign displayed that the store had the item in stock, and the location of the store. They have provided data for you from the past 200 days of sales, which you can assume is a representative sample of their sales. (Normally, you'd have concern here about correlation over time, we are ignoring that for this exercise.)

Here is the variable list:

Location - one of three locations - North, South, Central
OutsideTemp - average outside temperature near store in degrees F
InsideTemp - average store temperature in degrees F
Weather - was weather good or bad?
Sign - was there a sign advertising the store currently had the item in stock?
NumSold - number of the item sold
Price - price of the item
Sale - whether there was a sale on the item or not

Your task is to explore the data and propose (at least) 2 regression models for NumSold that will be compared using a CV-based approach, and argue that a final model you select is "best".

Introduction

The retailer WidgetsRUs seeks to develop a predictive model to better understand the factors influencing daily sales of a specific widget. To achieve this goal, we will analyze widget sales data from their three store locations, collected over 200 days, which includes a range of variables that may impact widget sales. Broadly, this includes price, promotional efforts (like sales and signage), various environmental conditions (inside and outside temperatures, weather quality), and store location (North, South, and Central). By evaluating the relationships between these variables, we aim to construct a model that can predict the daily number of widgets sold under varying conditions. Our approach will involve exploratory data analysis to identify trends and relationships. Then we will discuss our methods before sharing the results and performance of the predictive model we developed.

We begin with an overview of the dataset.

Data

```
widgets <-
  read.csv("https://awagaman.people.amherst.edu/stat240/WidgetData.csv")
```

The widgets dataset has 600 observations with 8 variables recording information about widget sales on a given day. There are four quantitative variables:

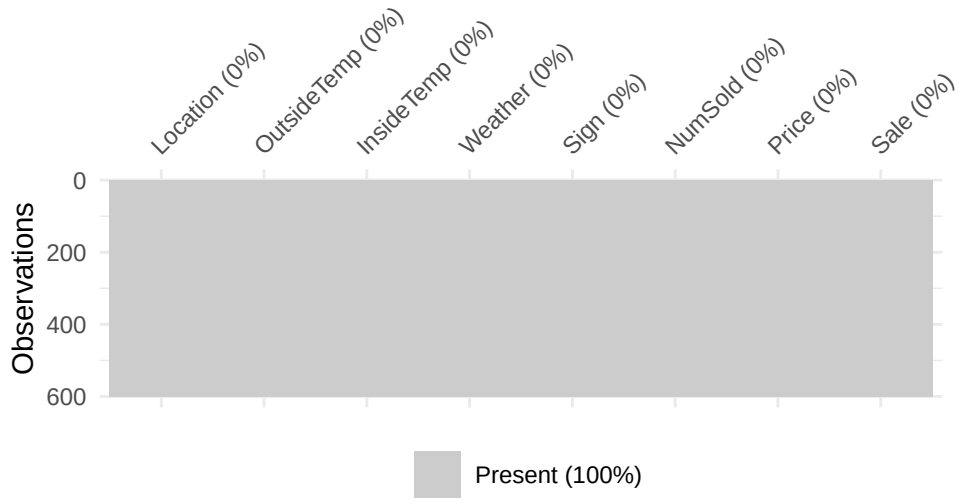
- `OutsideTemp`, which records the temperature outside on that day;
- `InsideTemp`, which records the temperature inside the store;
- `NumSold`, which records the quantity of widgets sold on that day; and
- `Price`, which states the price of the widget on that day.

There are also four categorical variables: * `Sale` and `Sign` state if there was a sale on widgets that day, and if there was a sign present announcing the sale, respectively. * `Weather`, which tells us the quality of the weather on that day, either 'good' or 'poor.' * Finally, `Location` tells us which of the three stores this data is from.

Our intended response variable is `NumSold`, the number of widgets sold in a day.

First, we check if there is any missingness, any zero variance variables, or near-zero variance variables that may have impeded our analysis.

```
# visualization of missing data
visdat::vis_miss(widgets, cluster = TRUE)
```




```
caret::nearZeroVar(widgets, saveMetrics = TRUE) %>%
  tibble::rownames_to_column() %>%
  filter(nzv)
```

```
##   rowname freqRatio percentUnique zeroVar  nzv
## 1 Weather        99      0.333333   FALSE TRUE
```

No missing values and no zero or near zero variance variables were detected. We are able to move onto univariate data analysis.

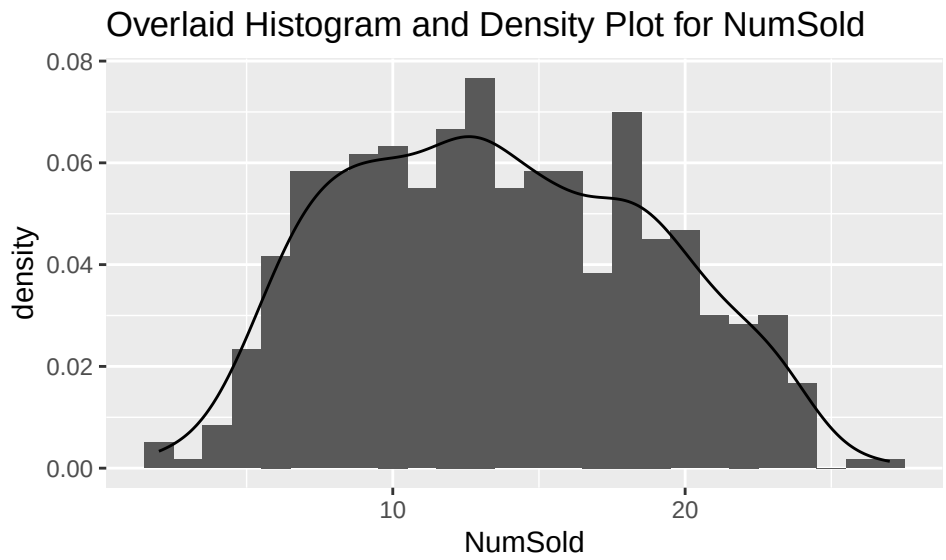
Univariate Data Analysis

We start out by looking at the distribution of our variable of interest, `NumSold`.

```r
# looking at response variable
favstats(~ NumSold, data = widgets)
```

```
##  min Q1 median Q3 max  mean     sd   n missing
##    2  9     13 18  27 13.59 5.1965 600       0
```

```r
# visual representation of the distribution of response
ggplot(widgets, aes(x = NumSold)) +
  geom_histogram(aes(y = after_stat(density)), binwidth = 1) +
  geom_density() +
  labs(title = "Overlaid Histogram and Density Plot for NumSold")
```



`NumSold` appears to be somewhat symmetrical with a mean of around 13 widgets sold. `NumSold` does not appear to have outliers or any other unusual features that require further investigation.

```r
ggplot(widgets, aes(x = Price)) +
  geom_histogram(aes(y = after_stat(density))) +
  geom_density()

ggplot(widgets, aes(x = OutsideTemp)) +
  geom_histogram(aes(y = after_stat(density))) +
  geom_density()

ggplot(widgets, aes(x = `OutsideTemp`)) +
  geom_histogram(aes(y = after_stat(density))) +
  geom_density()
```

We also took a look at the distribution of the other quantitative variables, `OutsideTemp`, `InsideTemp`, and Price, but there was nothing of note to report.

Next, we check the counts of the categorical variables.

```r
tally(~ Location, data = widgets)
```

```
## Location
## Central   North   South
##     200     200     200
```

```
tally(~ Weather, data = widgets)
```

```
## Weather
## Good Poor
##  594    6
```

```
tally(~ Sign, data = widgets)
```

```
## Sign
##  No Yes
## 416 184
```

```
tally(~ Sale, data = widgets)
```

```
## Sale
##  No Yes
## 402 198
```

We note that there are very few observations where `Weather` was reported as Poor, and anticipate that it may cause trouble when we split the data. For now we proceed with caution.

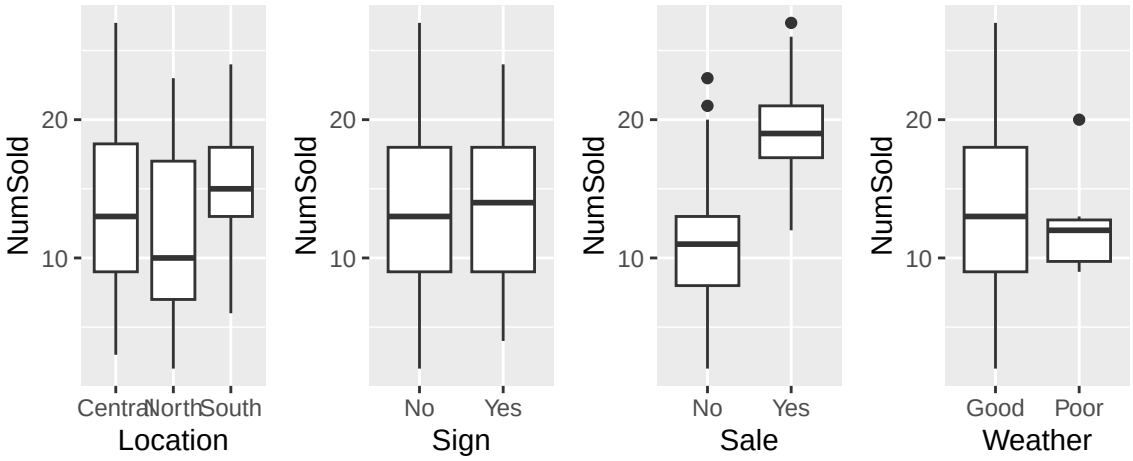
Next, we perform bivariate data analysis to understand the relationship between the quantity of widgets sold with the rest of the variables in our dataset.

Bivariate Data Analysis

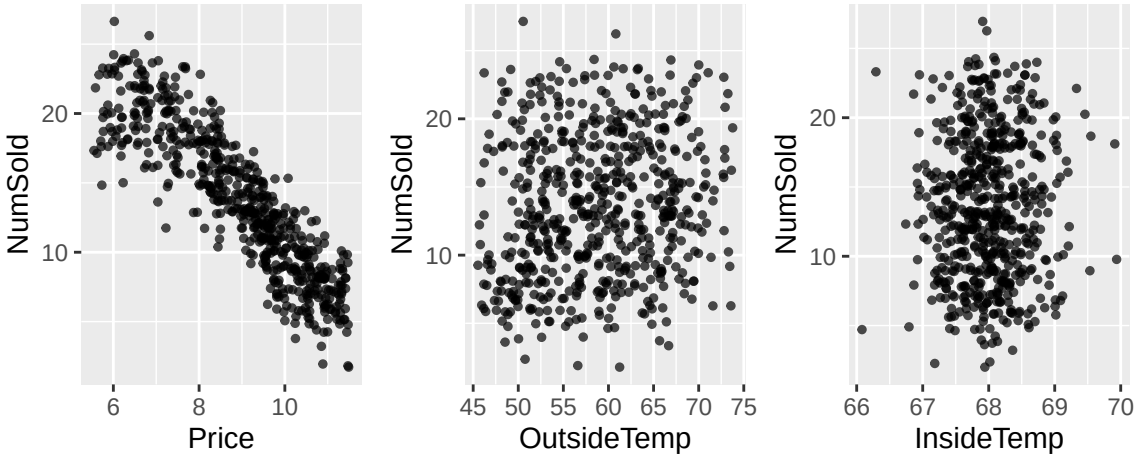
```
# boxplots
bp1 <- gf_boxplot(NumSold ~ Location, data = widgets)
bp2 <- gf_boxplot(NumSold ~ Sign, data = widgets)
bp3 <- gf_boxplot(NumSold ~ Sale, data = widgets)
bp4 <- gf_boxplot(NumSold ~ Weather, data = widgets)

# scatterplots
sp1 <- gf_point(NumSold ~ OutsideTemp, data = widgets, size = 1,
                position = "jitter", alpha = 0.7)
sp2 <- gf_point(NumSold ~ InsideTemp, data = widgets, size = 1,
                position = "jitter", alpha = 0.7)
sp3 <- gf_point(NumSold ~ Price, data = widgets, size = 1,
                position = "jitter", alpha = 0.7)

grid.arrange(bp1, bp2, bp3, bp4, ncol = 4)
```



```
grid.arrange(sp3, sp1, sp2, ncol = 3)
```



We note that `Sale` and `Location` may have some kind of relationship with `NumSold`. Both are categorical variables, and based on the tallys above, we decide that it would be best to stratify by `Sale` when splitting the data into training and test sets. Whereas each location (North, South, Central) had exactly equal numbers of observations, `Sale` is not equal across levels. To achieve similar distributions for `Sale` in the training and test sets, we stratify by `Sale`. There also appears to be a moderately strong, negative, linear relationship between `Price` and `NumSold`. As the price of the widget goes up, the quantity sold goes down. The scatterplots of `OutsideTemp` vs `NumSold` and `InsideTemp` vs `NumSold` show a lot of randomness suggesting there isn't much of an association between temperature and the number of widgets sold.

We decided to do a quick graphical check to see if there are any differences in the price of the widgets by the location of the store:

```
gf_point(NumSold ~ Price, data = widgets, color = ~ Location, alpha = 0.7,
         position = "jitter",
         xlab = "Price of Widget", ylab = "Number of Widgets Sold") %>%
  gf_lm()
```

```
## Warning: Using the `size` aesthetic with geom_line was deprecated in ggplot2 3.4.0.
## i Please use the `linewidth` aesthetic instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

We note an interesting relationship between `Location` and `Price`. Based on the plot above, it appears that the relationship between the number of widgets sold and the price of the widget is slightly different for each store location.

Splitting the Data

With 600 observations, we decided it would be appropriate to do a 80:20 split.

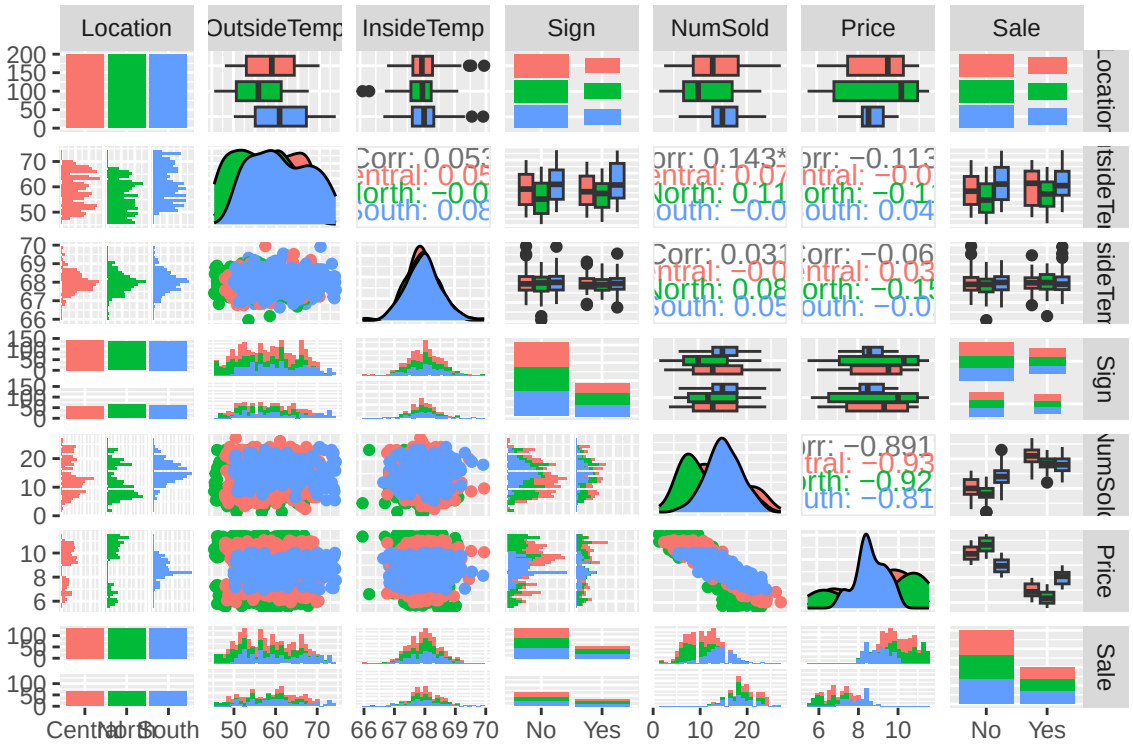
```
# Set up data set train/test split
# Stratified sampling with the rsample package
set.seed(240) # change to whatever seed you want
split <- initial_split(widgets, prop = 0.8, strata = "Sale")
widgets_train  <- training(split)
widgets_test   <- testing(split)
```

```
caret::nearZeroVar(widgets_train, saveMetrics = TRUE) %>%
  tibble::rownames_to_column() %>%
  filter(nzv)
```

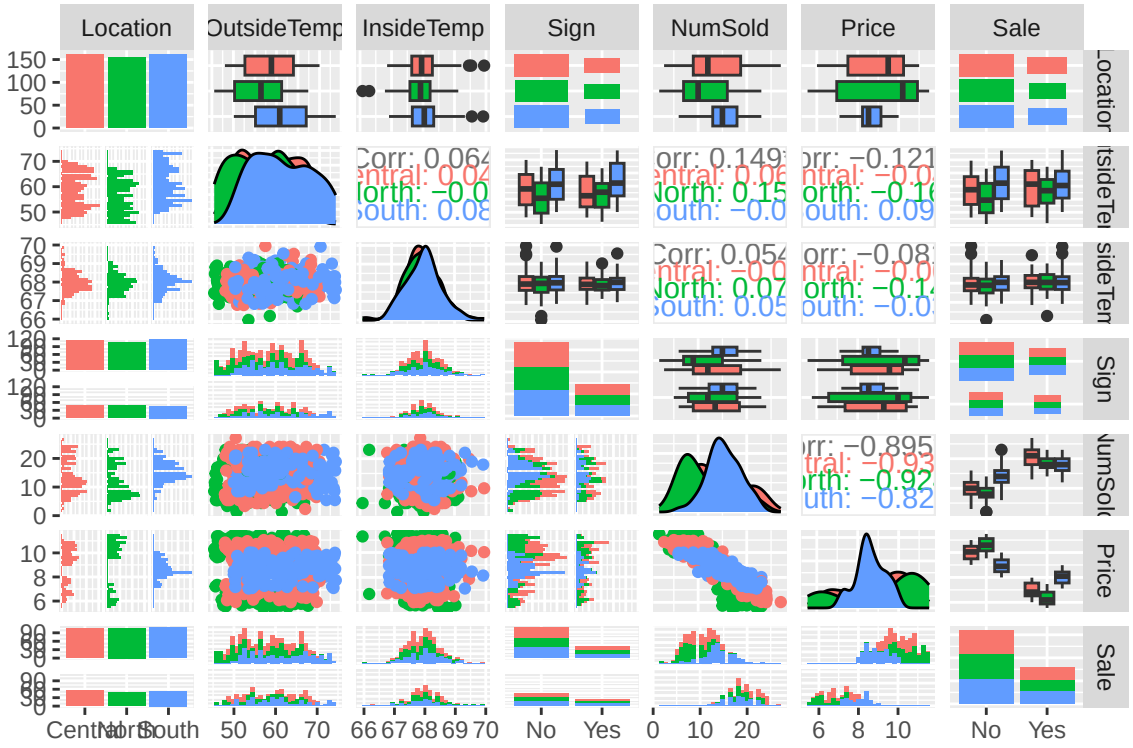
```
##   rowname freqRatio percentUnique zeroVar  nzv
## 1 Weather   78.8333      0.417537   FALSE TRUE
```

As we suspected earlier in our exploration, `Weather` has near zero variance in the training set, so we will not be including it in further analysis.

```
# take out weather
widgets_train <- widgets_train %>% select(-Weather)
ggpairs(select(widgets, -Weather), aes(color = Location))
```




```
ggpairs(widgets_train, aes(color = Location)) #Scatterplot matrix
```



We took a look at the scatterplot matrix of the training set and compared it to the matrix of the full set and found them to be very similar; the training set appears to be a good representation of the full dataset. As there does not appear to be anything of concern, we move on to constructing models to predict `NumSold`.

Methods

In order to create a model that will better predict unseen or future data, we split the dataset into two parts - the training set and the test set. We use the training set to build our model, tune any hyperparameters, and compare different models we generate. Then, after choosing our final model, we use the test set to test our model's performance. We decided on a 80%/20% split in order to have enough data in the training set to generate a good model without hindering our assessment of the model by making the test set too small.

As mentioned briefly in the Data section, we stratified by `Sale` when splitting the data because we wanted to control the training and test sets to have similar distributions. During data exploration, we noted that `Sale` and `Sign` had unequal distributions between levels. Since there appeared to be a more evident relationship between `Sale` and `NumSold` than `Sign` and `NumSold`, we decided to stratify based on `Sale`. This ensured the training and test sets had similar proportions of observations with `Sale` = Yes and observations with `Sale` = No.

Our model selection process involves the use of backwards elimination. Starting with a "kitchen sink" approach, we generated Model 1 using all the variables other than `Weather`, which we identified not to be useful in the Data section, to predict `NumSold`. Note that we also include the interaction between `Location` and `Price` in Model 1, based on our findings in the Data section. The process of backwards elimination involves taking out the least statistically significant terms in the model in order to improve model performance, which in this instance we will measure with Akaike's Information Criterion (AIC). In other words, we drop terms one by one until we can no longer decrease the model AIC, as we are aiming for a lower AIC. This process gives us Model 2.

We decided to include a third model, Model 3, which was made from adjusting Model 2 by adding a term that was previously dropped. One drawback of backwards elimination is that it is possible for a model with a lower AIC to be overlooked because a term was dropped earlier on in the process. Because we noted a relationship between `Sale` and `NumSold` in the exploratory data analysis, we decided to test an additional model to see if Model 2 could be improved by adding back the term `Sale`.

We chose the method of k-fold cross validation to compare models generated with the training data. In k-fold cross validation, the "fold" refers to groups of observations created by resampling from the training set. With $k = 5$ folds, we resampled the data by randomly forming them into 5 groups, fit the model on 4 of those groups, and finally used the remaining group to compute model performance. This process of fitting and testing is repeated $k - 1$, or 4 more times, so that each group has a turn as the test group we assess the model against. We average the k test errors to get our overall k-fold CV estimate. We used k-fold CV on Models 1, 2, and 3, and compared their mean performance scores to decide which model is the best. Performance measures we looked for were a low RMSE and high R^2.

According to *Hands-on Machine Learning with R*, generally $k = 5$ or $k = 10$ is used, but a larger k will get us closer to the true model performance at the cost of greater computational burden (Boehmke & Greenwell, 2020). With 600 observations, we decided that $k = 10$ would be reasonable. The text also states that for smaller data sets with fewer than 10,000 observations, its beneficial to employ repeated k-fold cross validation in order to improve accuracy. We decided to repeat the CV process 10 times.

Results

```
Model1 <- lm(NumSold ~ Location + OutsideTemp + InsideTemp + Sign + Sale +
                       Price + Price:Location, data = widgets_train)

msummary(Model1)
```

```
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)           57.3614    11.9159    4.81  2.0e-06 ***
## LocationNorth        -10.3838     1.2257   -8.47  3.1e-16 ***
## LocationSouth          4.8873     2.1986    2.22    0.027 *
## OutsideTemp            0.0101     0.0142    0.71    0.475
## InsideTemp            -0.2086     0.1740   -1.20    0.231
## SignYes               -0.0394     0.2027   -0.19    0.846
## SaleYes                0.4291     0.3674    1.17    0.243
## Price                 -3.3673     0.1471  -22.90  < 2e-16 ***
## LocationNorth:Price    1.0051     0.1330    7.56  2.2e-13 ***
## LocationSouth:Price   -0.5138     0.2527   -2.03    0.043 *
##
## Residual standard error: 2.02 on 469 degrees of freedom
## Multiple R-squared:  0.851,  Adjusted R-squared:  0.849
## F-statistic:  299 on 9 and 469 DF,  p-value: <2e-16
```

We fit the first model being considered above. Model 1 is a multiple linear regression model that predicts `NumSold` from all the variables we decided to keep for our analysis: `Location`, `OutsideTemp`, `InsideTemp`, `Sign`, `Sale` and `Price` *with* the additional term of the interaction between `Price` and `Location`.

```
# backwards elimination
step(Model1, direction = "backward")
```

```
# saving model found through backward elimination
Model2 <- lm(NumSold ~ Location + Price + Location:Price,
             data = widgets_train)
```

Next, we performed backwards elimination starting from Model 1. The first term dropped was `Sign`, followed by `OutsideTemp`, `Sale`, and lastly, `InsideTemp`. We save the reduced model as Model2, which predicts `NumSold` from `Price`, `Location`, and the interaction of `Price` and `Location`.

Because there appeared to be some kind of relationship between `Sale` and `NumSold`, we're curious to see if the reduced model might actually benefit from adding back `Sale`. We develop a third model, Model 3, that predicts `NumSold` from `Price`, `Location`, `Sale`, and the interaction of `Price` and `Location`.

Below, we assess the performance of our three models though 10 fold cross validation repeated 10 times.

```
# Model1 model CV
set.seed(240)
cv_model1 <- train(
  NumSold ~ Location + OutsideTemp + InsideTemp + Sign + Sale + Price +
            Price:Location,
  data = widgets_train,
  method = "lm",
  trControl = trainControl(method = "repeatedcv", number = 10, repeats = 10)
)

# Model2 model CV
set.seed(240)
cv_model2 <- train(
```

```
  NumSold ~ Location + Price + Price:Location,
  data = widgets_train,
  method = "lm",
  trControl = trainControl(method = "repeatedcv", number = 10, repeats = 10)
)

# Model3 model CV
set.seed(240)
cv_model3 <- train(
  NumSold ~ Location + Sale + Price + Price:Location,
  data = widgets_train,
  method = "lm",
  trControl = trainControl(method = "repeatedcv", number = 10, repeats = 10)
)
# Extract out of sample performance measures
summary(resamples(list(
  model1 = cv_model1,
  model2 = cv_model2,
  model3 = cv_model3
)))
```

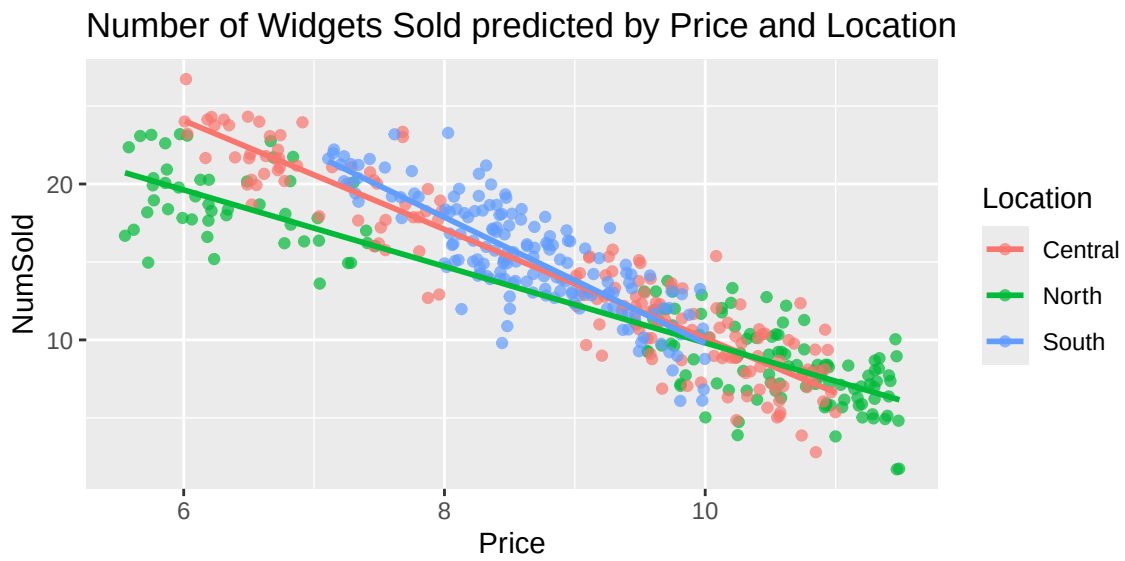
```
##
## Call:
## summary.resamples(object = resamples(list(model1 = cv_model1, model2
##  = cv_model2, model3 = cv_model3)))
##
## Models: model1, model2, model3
## Number of resamples: 100
##
## MAE
##           Min. 1st Qu.  Median    Mean 3rd Qu.    Max. NA's
## model1 1.21598 1.47329 1.60727 1.62703 1.75723 2.20560    0
## model2 1.15978 1.45793 1.58901 1.61537 1.75431 2.19402    0
## model3 1.16164 1.45593 1.59249 1.61440 1.74467 2.21083    0
##
## RMSE
##           Min. 1st Qu.  Median    Mean 3rd Qu.    Max. NA's
## model1 1.47465 1.86952 2.01417 2.03065 2.19357 2.62335    0
## model2 1.47086 1.85222 2.01105 2.01673 2.17708 2.62198    0
## model3 1.48977 1.84693 1.99590 2.01906 2.18519 2.62860    0
##
## Rsquared
##            Min.  1st Qu.   Median     Mean  3rd Qu.     Max. NA's
## model1 0.743199 0.830393 0.854114 0.849800 0.874251 0.923807    0
## model2 0.740365 0.834250 0.854829 0.851920 0.876286 0.926939    0
## model3 0.742686 0.834390 0.855447 0.851465 0.875994 0.926253    0
```

Model2, which used `Location`, `Price`, and the interaction of `Location` and `Price` to predict NumSold, performed the best out of the three models. Model 2 had the lowest RMSE on average, as well as the highest R^2. However, we note that all three models perform similarly, with Model 1 being slightly worse than Model 2 or Model 3. In any case, adding `Sale` to Model 2 did not make any appreciable improvements to the model, therefore we prefer Model 2, the simpler model.

The scatterplot below illustrates our final model, highlighting the interaction between `Price` and `Location`:

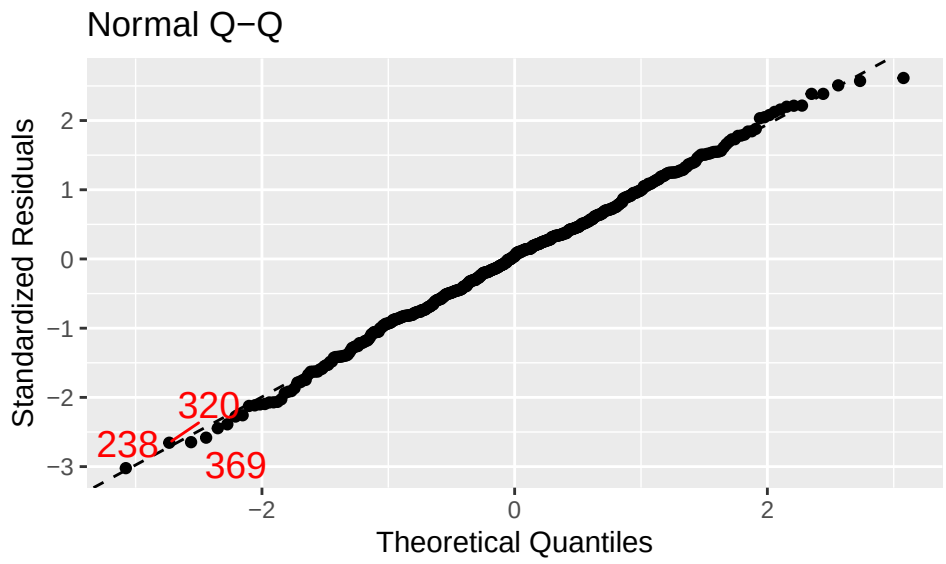
```
ggplot(data = widgets_train, aes(x = Price, y = NumSold, color = Location)) +
  geom_point(position = "jitter", size = 1.5, alpha = 0.7) +
  geom_smooth(method = "lm", se = FALSE) +
  labs(title = "Number of Widgets Sold predicted by Price and Location")
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



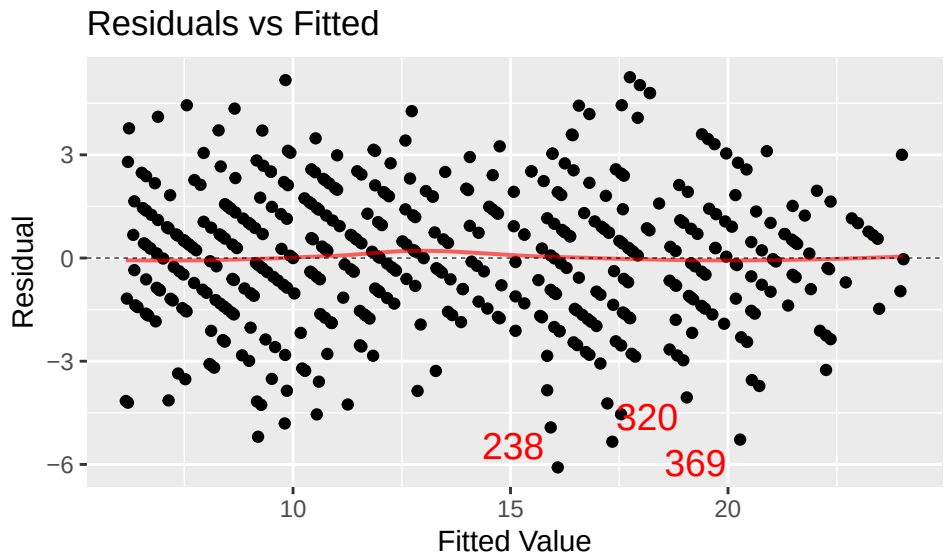
We see that the effect of `Price` on `NumSold` changes for each `Location`: Central, North, or South.

```
# checking conditions
mplot(Model2, which = 2)
```




```
mplot(Model2, which = 1)
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



The QQ plot shows that the residuals are more or less normally distributed. In the residuals vs. fitted plot, the residuals are scattered and centered at 0. The diagonal pattern of the residuals can be attributed to the fact that the response variable is count data, or non-negative integers, so we are not concerned by the appearance of the plot. The conditions for regression appear to be satisfied for Model 2.

After choosing our model and testing conditions, we can finally assess our model against the test set.

```
# fit model to training data
train_pred <- predict(Model2, newdata = widgets_train)
train_mse <- mean((widgets_train$NumSold - train_pred)^2)
train_mse
```

```
## [1] 4.03211
```

```
# getting test mse
test_pred <- predict(Model2, newdata = widgets_test)
test_mse <- mean((widgets_test$NumSold - test_pred)^2)
test_mse
```

```
## [1] 5.16522
```

We found the MSE of the training set to be 4.032 and the MSE of the test set to be 5.165.

Conclusion

```
msummary(Model2)
```

Through our analysis, we found an appropriate model to help WidgetsRUs achieve their goal of predicting daily widget sales. We developed the following model:

$NumSold = 45.01 - 10.69LocationNorth + 5.2LocationSouth - 3.49Price + 1.04LocationNorth * Price - 0.553LocationSouth * Price$,

which predicts the daily number of widgets sold from the price of the widget and the store location that it is sold from. It appears to perform well with an R^2 of 0.85, with conditions reasonably satisfied. Of the models tested in this analysis it also had the lowest RMSE found through repeated 10-fold cross validation.

The test MSE was 5.16, which suggests that the response values on future data would be off by an average of about 2.27 units, or about 2-3 widgets. With this analysis, and the model we have constructed from it, WidgetsRUs can project future daily widget sales and make informed decisions about the quantity of widgets they need to produce and stock with reasonable accuracy.