Stormy Weather, Stormy Sales

Jonathan Lu and Kara Wong

12/13/2019

Introduction

While we can't control the weather, we can control how we react to it.

Based on a survey conducted by a mobile testing firm, SOASTA, checking the weather is the first thing 45% of participants do in the morning [1]. The weather is the second most commonly first-checked item in the morning, after checking emails. Weather plays an integral part of society, as it affects our daily lives. It determines how we dress, commute, eat, and feel. For example, if we see that a rainstorm is approaching, we would most likely choose to stay indoors rather than going out to run errands. This extreme influence that weather has on us is reflected in the changes in the U.S. gdp numbers. According to the Atomospheric and Environmental Research Center, thirty percent of U.S. gross domestic product is affected by weather one way or the other [2].

As part of an existing drive to be able to predict consumer behaviors, weather is just another factor which retailers want to consider. There are plenty of advantages to analyzing the effect weather has on a business. One of the largest issues which businesses face today are largely inventory based as floor space is expensive. By anticipating and reacting to weather-driven demand, businesses would be better able to maintian necessary stock during times of need, while other times, they would be able to use the space in the store for other, more profitable items. While this knowledge would be useful across all stores in the US, this would have a far larger effect in stores in metropolitan areas as the floor space is worth a lot more per square foot. Being able to anticipate trends in sales would also allow businesses to work around logistic issues such as transportation of goods. While this benefit ties in to the main one of understanding the necessary stock, optimizing logistic issues surounding transportation of goods between locations will also help businesses lower their operating costs and overhead.

While the idea behind an analysis such as this seem to be extremely useful, actual applications of such a model are extremely limited. When considering models for this data, there is a clear trade of between interpretability and accuracy, the less interpretable, the more accurate. Since this analysis is focused mostly on linear models, the interpretability of the coefficients the most useful aspect to this analysis. This is because while predicting how much you should've stocked is relatively useful information, using models such as the ones explored below to predict how much you should stock will be relatively fruitless. This is because in order to "predict" the future using these models, you will need to have access to accurate weather data in the future. Usually weather data is relatively accurate for one-week out forecasts [3], however, through experience, everyone knows that the weather forecast can change from day to day. This level of uncertainty leads to uncertainties in the model, which in turn, makes them less reliable.

The data used in this analysis was provided by Walmart as part of a recruiting event on Kaggle [4]. On Kaggle, they provided training and testing datasets. In addition to those files, they also provided a key to map stores to stations and a collection of relevant weather station data from the NOAA. The data spans 45 different Walmart locations, 20 different weather stations, and over 2 years of data. The data initially provided is relatively messy as the weather data has a lot of missing values The objective of this analysis is to predict the amount of units sold for each product with regards to major weather events.

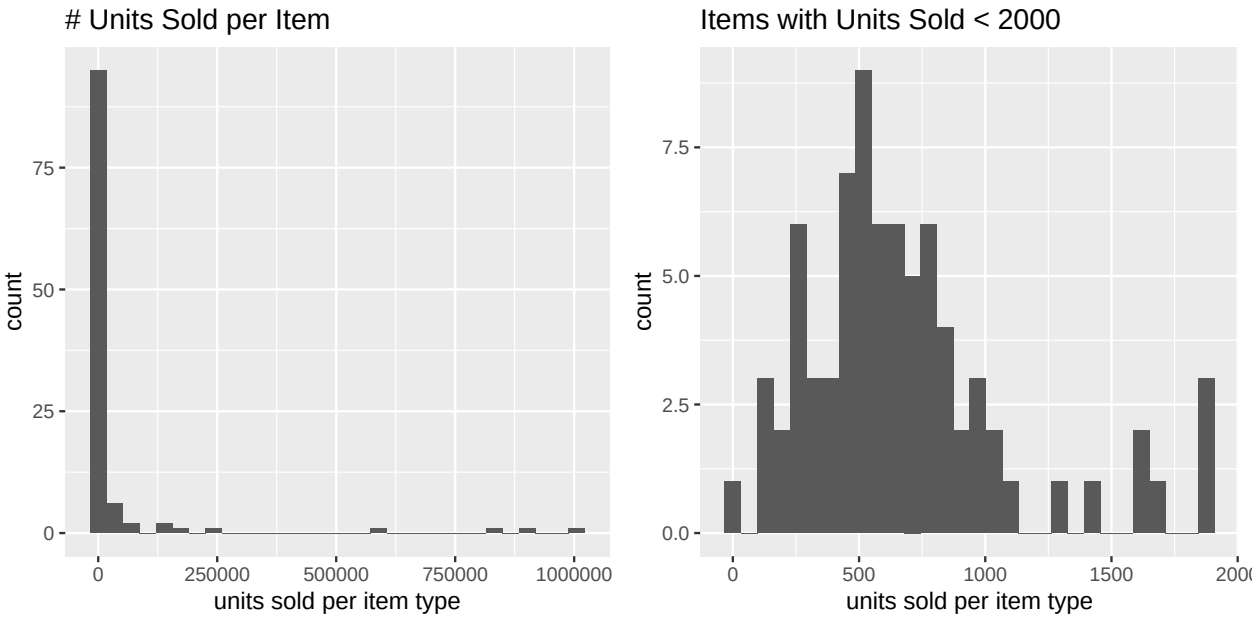
[1] Morning Routine: What's the First Thing You Do When You Wake Up?
[2] Retail and Supply Chain
[3] Weather Prediction Accuracy
[4] Kaggle: Walmart Recruiting II: Sales in Stormy WEather

Exploratory Data Analysis

Our first observation when looking into our dataset is that our dataset has a lot of items with low sales. In fact, out of the 111 items, 59% have less than 1,000 sales and 24% have less than 500 sales. On the other hand, the top-five highest selling items all have over 10,000 units sold. Below, we have provided summary statistics for the number of units sold per item type, as well as a graphical representation.

Table 1: Table: Summary of Number of Units Sold per Item Type

Minimum	First Quartile	Median	Mean	Third Quartile	Maximum
31	526	781	41054	2845	1005111

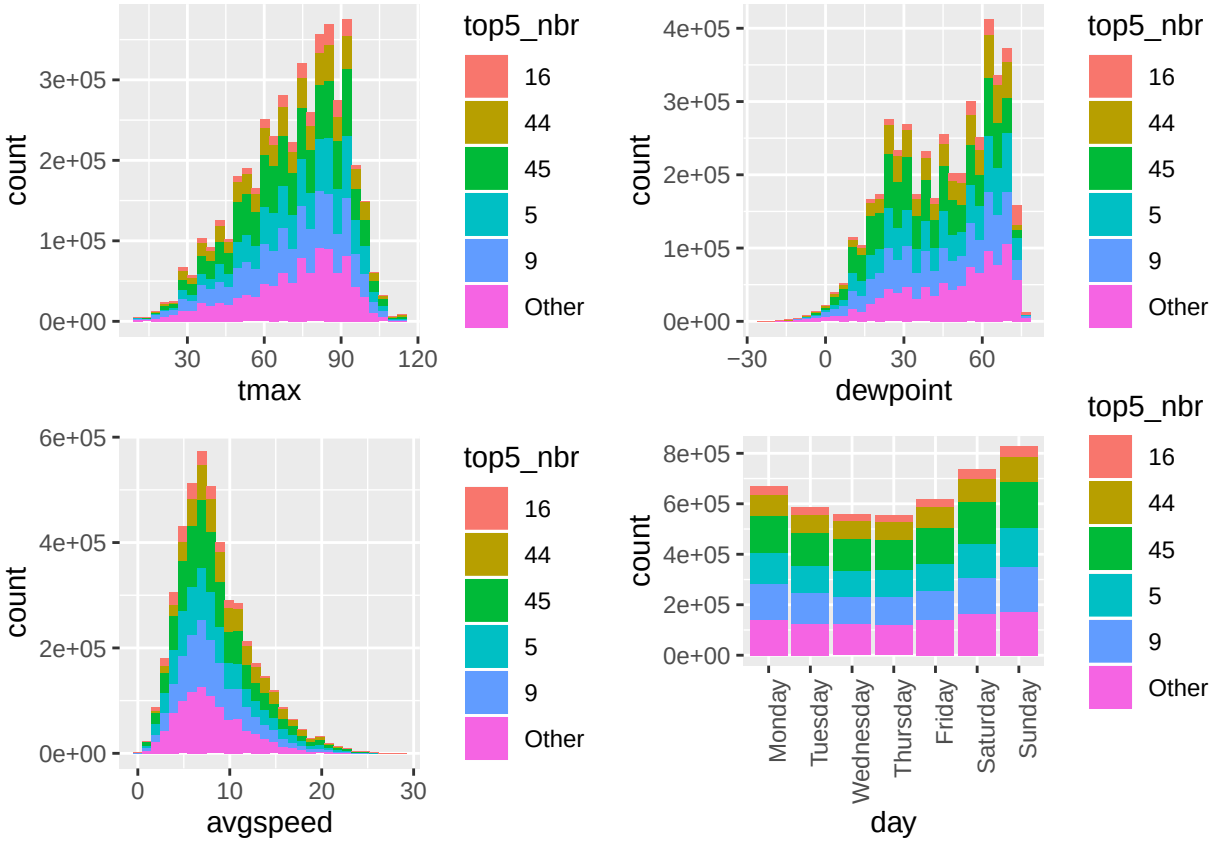


From the graphs above, we can see how drastic the difference of sales between items truly is. This discrepancy was briefly touched on in the inital Kaggle posting when Walmart said that some of these items might be everyday objects such as milk. In order to get a better idea on the actual distribution of the number of items sold, if we look at the distributions of items with under 2,000 sold, we can see that most of the items sold are between 500 and 700 units sold. This is relatively interesting as this averages to a little bit under one unit sold a day. The most important thing to get from these graphs however, is still the discrepance between items. Before even fitting models, it's clear that there are going to be many influential "outliers" because the item that sold the most sold 1005111 items which is approximately 1400 units per day across all stores. When compared to the 1 unit per day of most observations, it becomes clear that modeling all of the variables at once is going to be challenging.

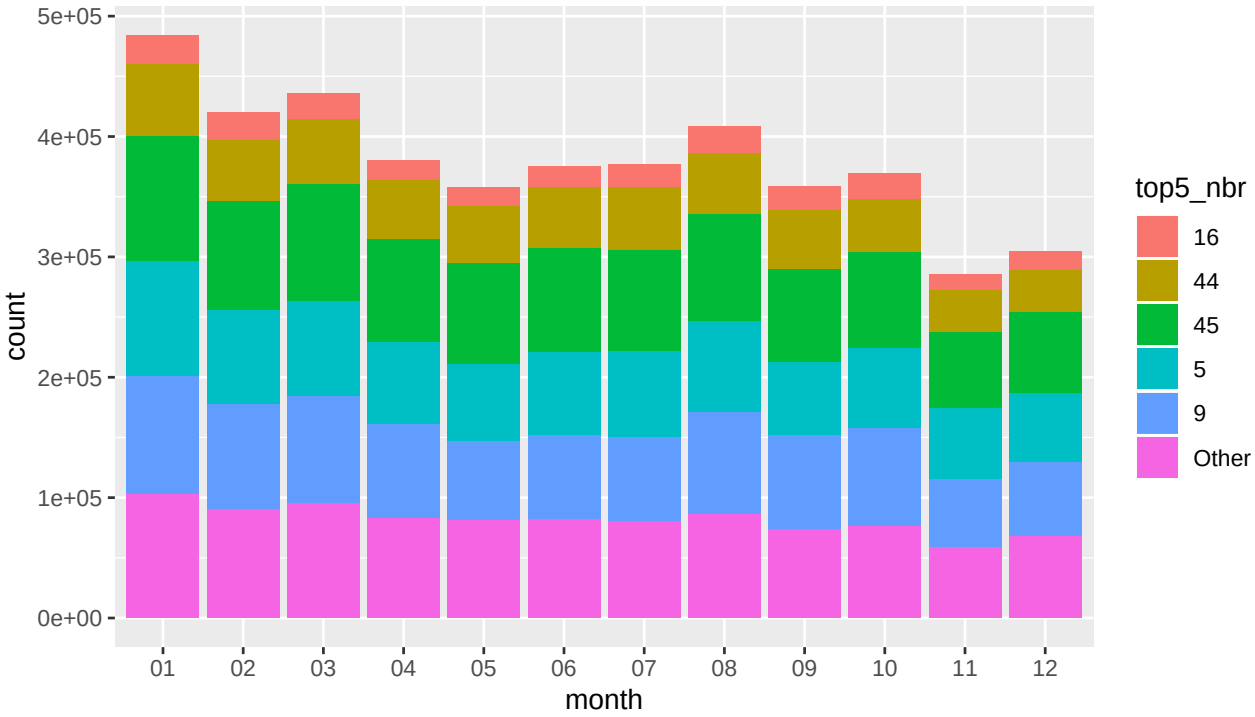
An important thing to note about our dataset is that it does not capture the discrepancies between inventory and demand. The number of units sold may show 0, but this may not necessarily mean that the item was not in demand, as it could just simply have been out of stock or discontinued. This may be an issue when it comes to making predictions.

We then decided to analyze the amount of units sold of these top-five selling items based on the weather-based metrics provided (maximum/min temperature, dewpoint, etc.). The five highest selling items from highest to lowest are items: 45, 9, 5, 44, and 16. From the following plots, we can see that more purchases are made when it's hot and humid, when it's not too windy, and on weekdays. The most important takeaway from these plots however, is that the top five selling items all seem to follow a similar distribution when it comes to units sold. This means that if we could hypothetically accurately predict items sold from the weather data, we should be able to accurately predict units sold across most of the items. From the rightmost plot,

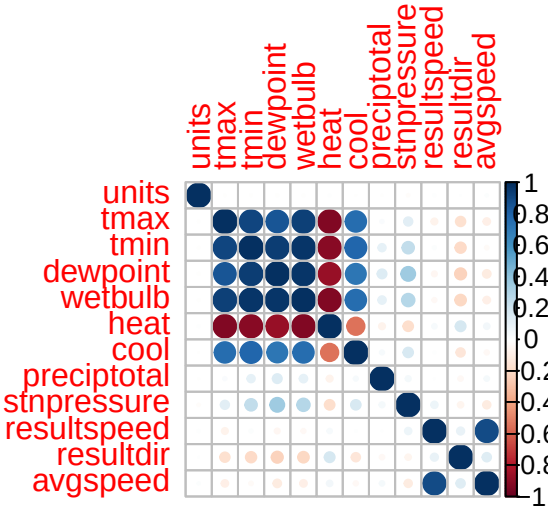
we can see that the same idea of the weather data holds as well for days in that the items follow a similar distribution across the days of the week. This graph also confirms our initial ideas that the weekend has more sales that the other days of the week, thus leading us to create a categorical variable to represent weekend vs weekday. From the plots below, we can also see that there isn't one item type that is driving sales based on these weather metrics. More of these plots can be found in the Appendix.



In addition to comparing the amount of sales on weekdays versus weekends, we also looked to see if there were any seasonal differences that could be driving sales. We extracted the month from `date` and provided a graph below that shows how there seems to be a downward trend in sales starting from January. This is a bit suprising considering that during the winter months, the temperature is approximately the same. While the graph below hint that seasons and months might not be a strong predictor of units sold, we still believe that months might be relevant as we wanted to create a variable that might capture the data represented by temperature and time since we are planning to removing date. Season on the other hand, seems to be completely irrelevant as their isn't a hint of consistency within each season as shown below.



From the correlation plot below, we can see that all of the temperature data is highly correlated with each other. From this, we can anticipate that if we ran variable selection on this dataset, we would end up removing multiple variables. The rain and wind data seems more promising as they are very loosely correlated with all the other variables in this data set. While the correlation between the max temperature and the min temperature in a day is expected, we'd hoped for there to be more of a deviation. However, we still decided to remove the average temperature instead of the minimum and maximum temperature as we think that the more extreme the temperature might be, the better a predictor it will be.



Before we could build our first linear model, we had to stitch the weather data to the original dataset provided by Walmart. The main bulk of the work when it came to this process was cleaning the weather data to a point where it could be added together.

At an initial glance, we could see that the weather data was extremely incomplete *Fig.1*. When aggregated to a station level, every single station had some sort of missing data. The variables that were missing the most data were depart, sunrise, sunset, and snowfall. Each of these variables had over 50% of their total

data across all stations missing. With this many missing observations, we decided that it would be extremely unreasonable to try and impute data for that many missing observations as we were neither knowledgeable enough about weather, nor knowledgeable enough about the stations themselves to be able to classify these missing observations as anything other than "structurally missing". Therefore, since there was nothing we could do on our own to make these variables complete, we chose to remove them from the data. A variable which was close to the cusp in terms of being "un-imputable" was sea level. However, we quickly noted that sea level was pretty much identical to stnpressure (station pressure) as they both measured approximately the same thing. Since these two variables are pretty much perfectly collinear, we chose to remove sea level as there were far more observations for station pressure. Another variable we removed due to multicollinearity is tavg, or the average temperature of the day. When we looked into it, the value is calculated as the average of the minimum temperature and the maximum temperature. We chose to remove the average because we believed that the more extreme weather experienced, which would be captured better by min and max temperature, the more likely people are to buy weather related gear.

While removing a couple of variables definitely improved on the amount of missing data remaining, almost every single variable and station combination had some number of missing data. However, the amount of data missing at this point was extremely trivial ranging from $\sim 0.5\%$ of each station's data. Given that very few observations were missing, we felt relatively comfortable imputing a four day moving average for the missing observations. While this method cleaned up most of the variables for all of the stations except one. Once we aggregated the data to a station level, it became clear that station five had an un-imputable amount of data. Usually in these circumstances, we would remove the station completely, however, since we needed to have matching data for every store in Walmart's initial dataset we had to come up with a way to impute multiple values. To do this, we aggregated the data to a day level and imputed the mean of all the other station's observations at that day.

Before we finish with our weather data, we decided to create additional variables off of the weather data. When looking over the variables provided, one really stood out and that was date. We thought that it would be incredibly naive to think that the actual value of the date would have some predictive power over the number of units sold so instead; we constructed two categorical variables to suplant the use of the date. We decided that whether or not the day was a weekend and what season the day would fall under would be far more predictive as it narrows down the date to certain features related to it. While we thought that seasons might be really interesting as it sort of has to do with the weather, we were a bit worried about whether or not the change in seasons would already be reflected in the temperature data. However, through our EDA process, it became very clear that there wasn't a clear trend represented by the different seasons, thus leading us to remove it from the first model. We also changed codeSum from a set of strings representing weather events to a categorical variable representing whether or not some weather event happened on the day.

Linear Regression Model

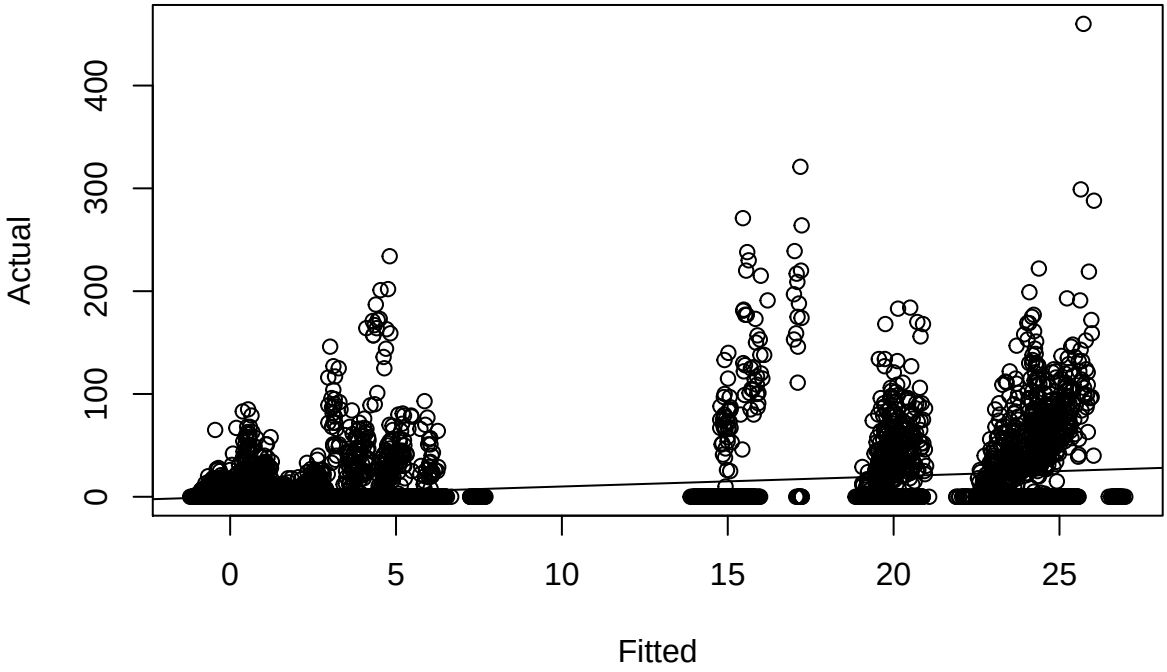
Joining the weather data to the Walmart data was extremely straightforward as all it required was two left joins. The first was to match a station to each store using the provided key. The last join was on the station number and the date, thus giving us a fully merged data set.

When it comes to building the first model, we wanted to keep in as many variables as we could, however, we decided to remove date and station number from model. We chose to remove date because we believed that the actual date would have little to no impact on whether or not someone buys something and instead chose to use our created variables. We also chose to remove station number because the dummy variable that are created to represent each station is perfectly collinear with a certain set of store numbers that the station maps to. The code for the model can be found below.

Since in this model, we treated store number and item number as a categorical variable, there are 178 total variables not including the intercept in this model. When looking at the summary output for this model, it becomes clear that matching store numbers is one of the best ways to predict the amount of unit sales. However, like all categorical variables with an extreme number of factors, not all of the levels are significant. Item number also follows a similar trend in that there are some levels which have coefficients indistinguishable from zero, but most of the levels are irrelevant. We think that this is because most of the items sell less than one a day. Both of these issues will be explored further in the next section of this report.

The initial Kaggle score for the initial model with all of the variables is *.74047* which was good for 475th on the leaderboard.

Taking a deeper dive into the summary of all the stores, there were 5 stores that had a siginificant coefficient. This is particularly interesting because when looking at the coefficients for each of these stores, all of them were overwhelmingly positive, this means that those stores sold far more than the average number of items when compare to other stores. The positive nature of these coefficients also makes sense when the intercept is taken into consideration. The intercept for this linear model was 1.622 which is pretty interesting because it was insignificant while having a decently large "coefficient". The intercept in this scenario is confounded with the first store and first item which makes its coefficient almost impossible to understand on its own.



When looking at the plot of actual vs fitted values above, we can see that the intial model massively underpredicts for a lot of the items. We think that this is probably due to extreme dichotomy of the number of units sold between the stores. We can see that the store which sold the most units in total tops out

at approximately 300,000 units while the stores that sold the least only sold less than 10,000 units. This dichotomy is deadly to a linear model because it ends up fitting close to the middle of the two which in turn means that you simultaneously overpredict and underpredict. Methods of which to help fix this issue will be explored in the next section.

The thoughts that we expressed in the previous paragraph regarding the dichotomy of the observations is also reflected in the plot of leverages vs cook's distance *Fig.5*. This plot is extremely intersting as it suggests that most of the observations are extremely influential. This again confirms our suspicions about the nature of the data.

Apart from the heavily categorical variables, all the other variables in the model were isignificant except for dewpoint (at an alpha of .1). This hints that the weather data isn't predictive at all for the number of units sold which is a bit disheartening.

Improvements

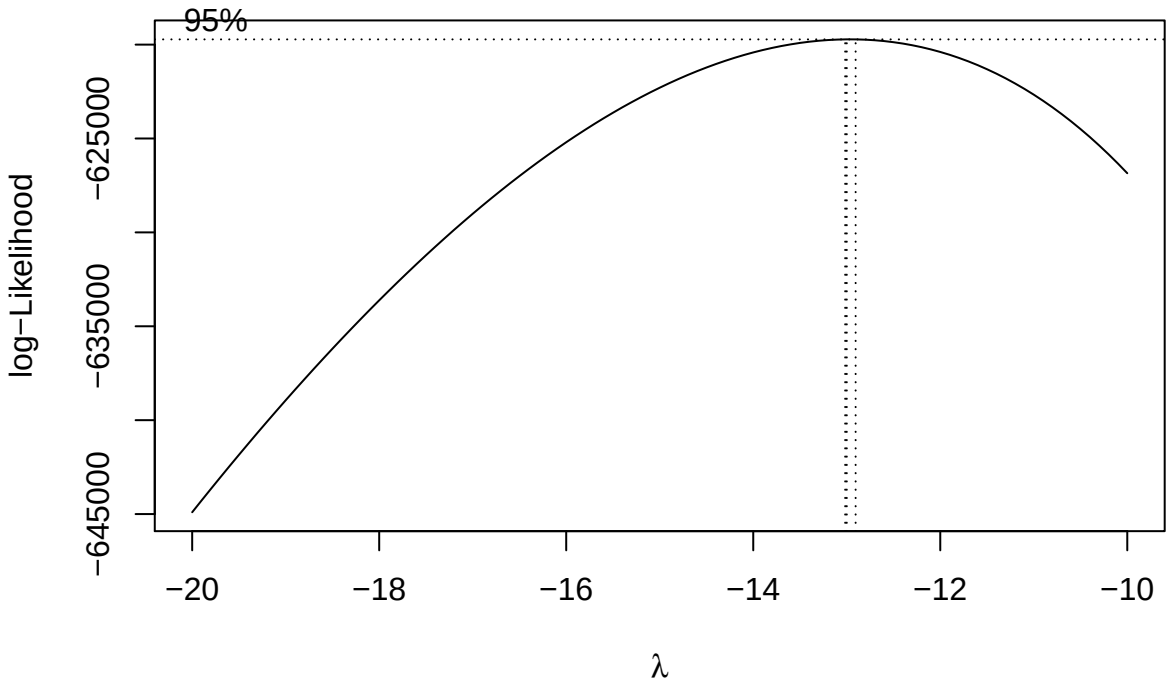
Before we began exploring these models further, we chose to validate the results by creating a validation split of 20% of the training data. By doing so, we managed to avoid overloading the kaggle submissions and having to spend minutes waiting for results.

When looking at the summary of the first created model, we could see that all of the weather data was insignificant. To further verify this suspicion, we decided to start the improvements section by regressing the number of units sold on the store number alone. The reason why we used this as a starting point is because from the EDA, it became very clear that the stores differed extremely drastically from each other in terms of units sold. While this "initial" model performed much worse (which was expected) as we removed all the other variables besides store number. A lot of the stores where shown to be significant which means that the stores are significantly different from eachother.

Next, we decided to try and clean up the weather data a bit. From the plot of all the correlations shown in the EDA, we can see that the weather aspect of the data were all highly correlated with eachother. While it's important to note that high multicollinearity doesn't decrease the predictive power of the model, multicollinearity could lead to overfitting of the model. The best way to explore this is through the variance inflation factors. In *Figure.5* we can see that the VIF's are way over the "cutoff" of 5. To remedy this, we removed wetbulb, heat, and the minimum temperature. We also noted that we initially missed that there were two highly collinear wind speed terms, average wind speed and results wind speed. We decided to remove resultspeed because avgerage windspeed was far more intuitive to us. This model actually performed better than the initial model. However it must be noted that the difference was almost negligible as the RMSE improved by 2.8×10^{-4}.

The next thing that we wanted to explore were interaction terms. The very first term we wanted to explore was between item and store number. The reason why we thought that this would be relevant was because it would allow the model to account for the relationship between store and item which we could see in *Figure.7* and *Figure.8*. This relationship should theoretically be extremely relevant for some combinations as we can see that the vast majority of combinations never sold any units. However, there are very clear issues when trying to fit a model of this size as just using this interaction would create a model with 4997 variables. To try and work around this, we decided that we wanted to explore possibly reducing the numeber of factors that are present in the item numbers and store numbers. In order to do this, we conducted Tukey's HSD test on items and store number.

When we conducted Tukey's test on each of the two variables. When it came to item numbers, many of the items were insignificant from eachother, this isn't really suprising because in the EDA, we found out that there were many items which barely any units at all per day across each store. This was also reflected in our analysis with Tukey as the number of unique items was reduced to 11. We used 0 to represent all of the items which were determined to "not" be different. We conducted the exact same analysis on store numbers. However, it turns out that a far larger percent of stores differed from eachother which left us with 34. We then tried fitting a model with interactions between item number and store number, however, we were sorely dissapointed as the model performed worse in both our validation subset and the kaggle grader and ended up with a kaggle score of *.748*.



Extra models

```
## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading
```

Appendix

Fig.1 Initial state of weather data

Table 2: M, NA, and VNA represent the count of M's, -'s, and NA's respectively

tmax	tmin	tavg	depart	dewpoint	wetbulb	heat	cool	sunrise	sunset	codesum	snowfall	preciptotal	stnpressure	sealevel	resultspeed	resultdir	avgspeed
2 NA values	2 NA values	6 M	1035 M	6 M	11 M	6 M	6 M	1035 VNA	1035 VNA	510 NA values	1035 M	5 M	4 M	8 M	4 M	4 M	3 M
1 NA values	1 NA values	12 M	43 M	6 M	16 M	12 M	12 M	31 VNA	31 VNA	510 NA values	11 M	1 M	6 M	11 M	5 M	5 M	4 M
2 NA values	2 NA values	7 M	38 M	OK	3 M	7 M	7 M	31 VNA	31 VNA	638 NA values	4 M	2 M	3 M	3 M	OK	OK	2 M
2 NA values	2 NA values	10 M	41 M	8 M	15 M	10 M	10 M	31 VNA	31 VNA	629 NA values	5 M	2 M	4 M	19 M	7 M	7 M	2 M
822 NA values	822 NA values	822 M	822 M	446 M	852 M	822 M	822 M	31 VNA	31 VNA	852 NA values	822 M	822 M	852 M	447 M	446 M	446 M	822 M
2 NA values	2 NA values	6 M	37 M	OK	3 M	6 M	6 M	31 VNA	31 VNA	522 NA values	9 M	2 M	4 M	3 M	OK	OK	2 M
35 NA values	34 NA values	41 M	1035 M	37 M	42 M	41 M	41 M	1035 VNA	1035 VNA	546 NA values	57 M	1 M	1 M	39 M	3 M	3 M	3 M
14 NA values	12 NA values	468 M	1035 M	50 M	64 M	468 M	468 M	1035 VNA	1035 VNA	337 NA values	1030 M	5 M	10 M	1035 M	32 M	32 M	5 M
6 NA values	7 NA values	19 M	1035 M	16 M	64 M	19 M	19 M	1035 VNA	1035 VNA	497 NA values	1017 M	5 M	6 M	14 M	3 M	3 M	5 M
OK	OK	8 M	1035 M	OK	3 M	8 M	8 M	31 VNA	31 VNA	499 NA values	1035 M	1 M	1 M	2 M	OK	OK	1 M
1 NA values	1 NA values	4 M	35 M	60 M	63 M	4 M	4 M	31 VNA	31 VNA	446 NA values	3 M	1 M	5 M	60 M	60 M	60 M	3 M
1 NA values	1 NA values	4 M	1035 M	1 M	4 M	4 M	4 M	1035 VNA	1035 VNA	423 NA values	33 M	1 M	2 M	4 M	1 M	1 M	2 M
1 NA values	1 NA values	14 M	1035 M	OK	3 M	14 M	14 M	1035 VNA	1035 VNA	647 NA values	1035 M	1 M	3 M	3 M	OK	OK	1 M
2 NA values	2 NA values	6 M	37 M	3 M	9 M	6 M	6 M	31 VNA	31 VNA	649 NA values	2 M	2 M	6 M	15 M	2 M	2 M	2 M
1 NA values	3 NA values	5 M	36 M	7 M	9 M	5 M	5 M	31 VNA	31 VNA	421 NA values	4 M	1 M	2 M	10 M	6 M	6 M	3 M
7 NA values	7 NA values	13 M	1035 M	8 M	11 M	13 M	13 M	1035 VNA	1035 VNA	499 NA values	1035 M	2 M	2 M	9 M	7 M	7 M	8 M
1 NA values	1 NA values	5 M	1035 M	1 M	9 M	5 M	5 M	1035 VNA	1035 VNA	585 NA values	55 M	1 M	2 M	11 M	4 M	4 M	1 M
2 NA values	2 NA values	4 M	34 M	OK	3 M	4 M	4 M	31 VNA	31 VNA	688 NA values	5 M	2 M	5 M	2 M	OK	OK	2 M
2 NA values	2 NA values	8 M	38 M	11 M	49 M	8 M	8 M	31 VNA	31 VNA	487 NA values	2 M	2 M	3 M	13 M	OK	OK	2 M
2 NA values	4 NA values	7 M	1035 M	6 M	19 M	7 M	7 M	1035 VNA	1035 VNA	666 NA values	25 M	1 M	8 M	16 M	9 M	9 M	2 M

Fig.2 Sample of plot of residuals of first model

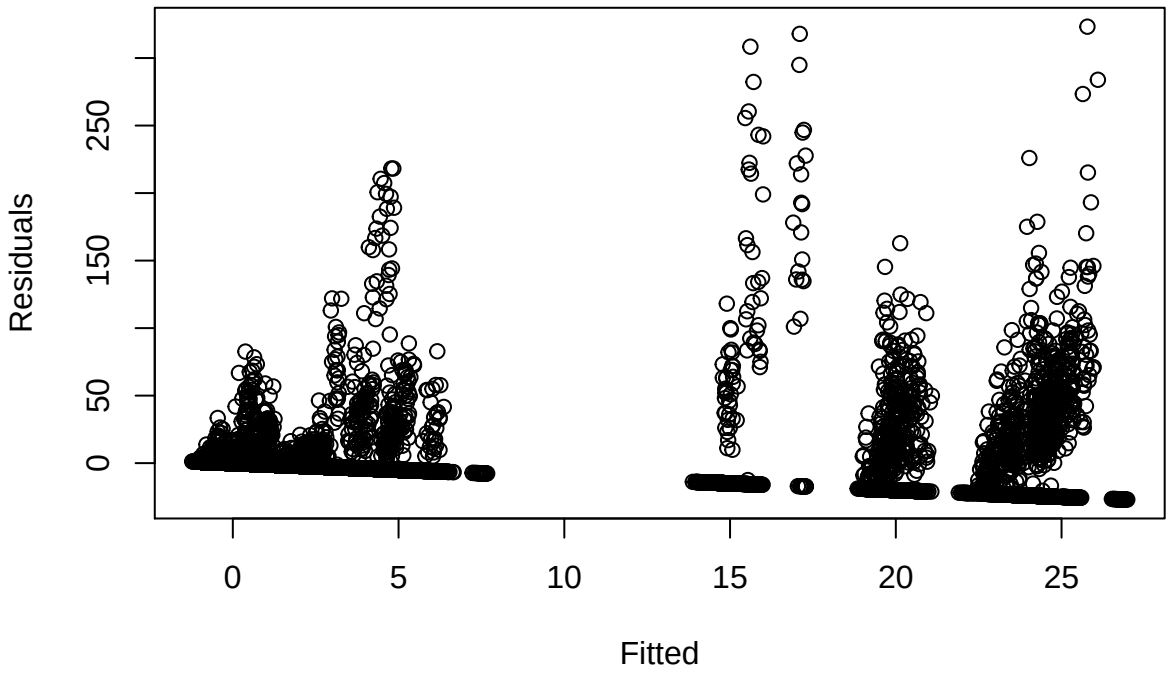


Fig.3 Distributions of sales by store

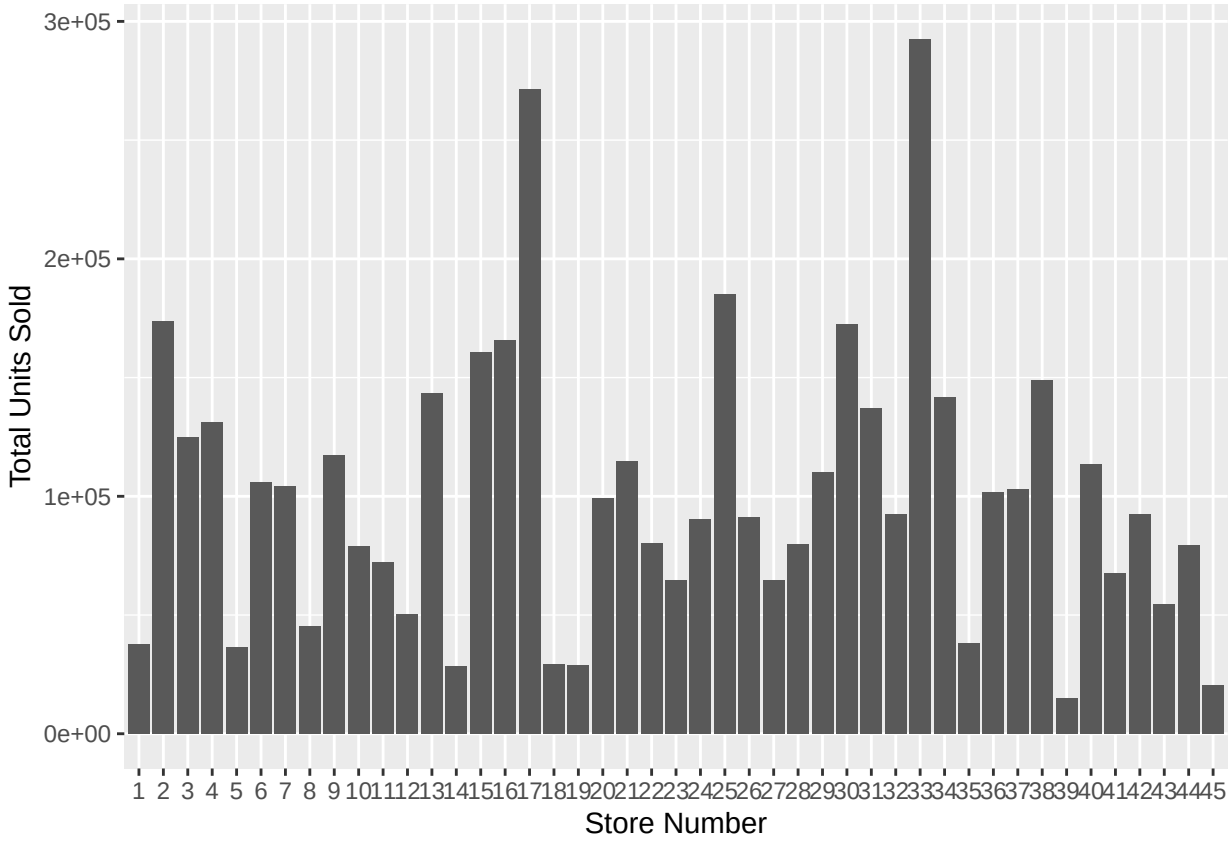


Fig.4 Plot of first linear model

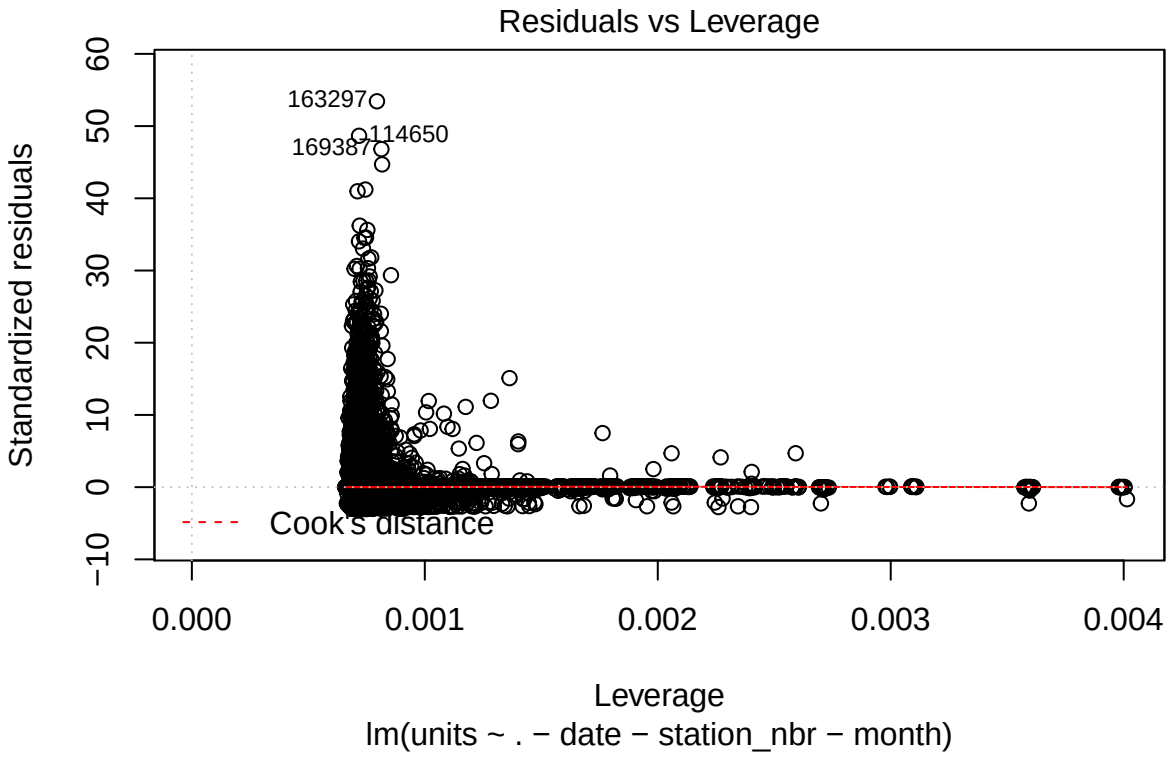


Fig.5 Initial VIFs

```
##      tmax      tmin  dewpoint   wetbulb      heat      cool
##  52.86188  60.27932  41.45919 134.41767  78.28949  27.28412
```

Fig.6 Reduced VIFs

```
##     tmax dewpoint     cool
## 4.367154 3.859928 2.515915
```

Fig.7 Distribution of Units Sold by Store and Item

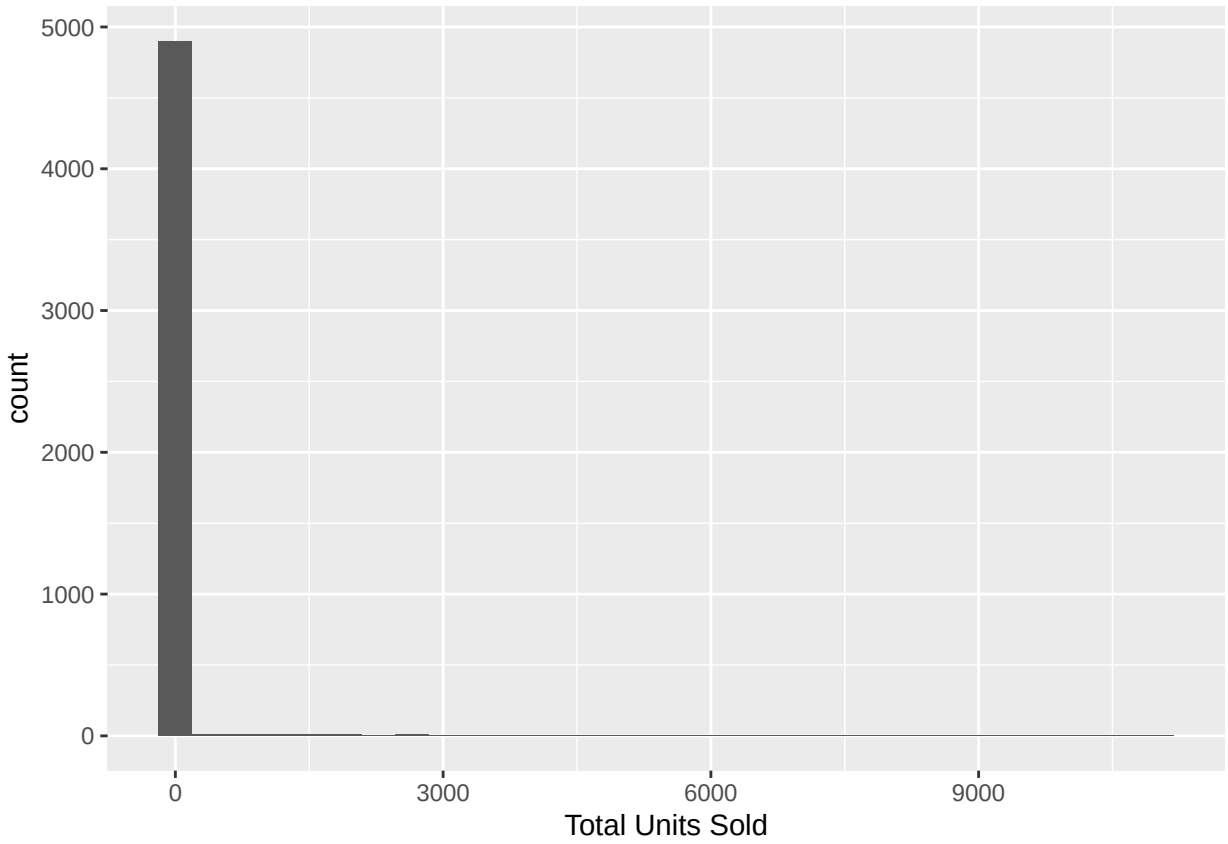
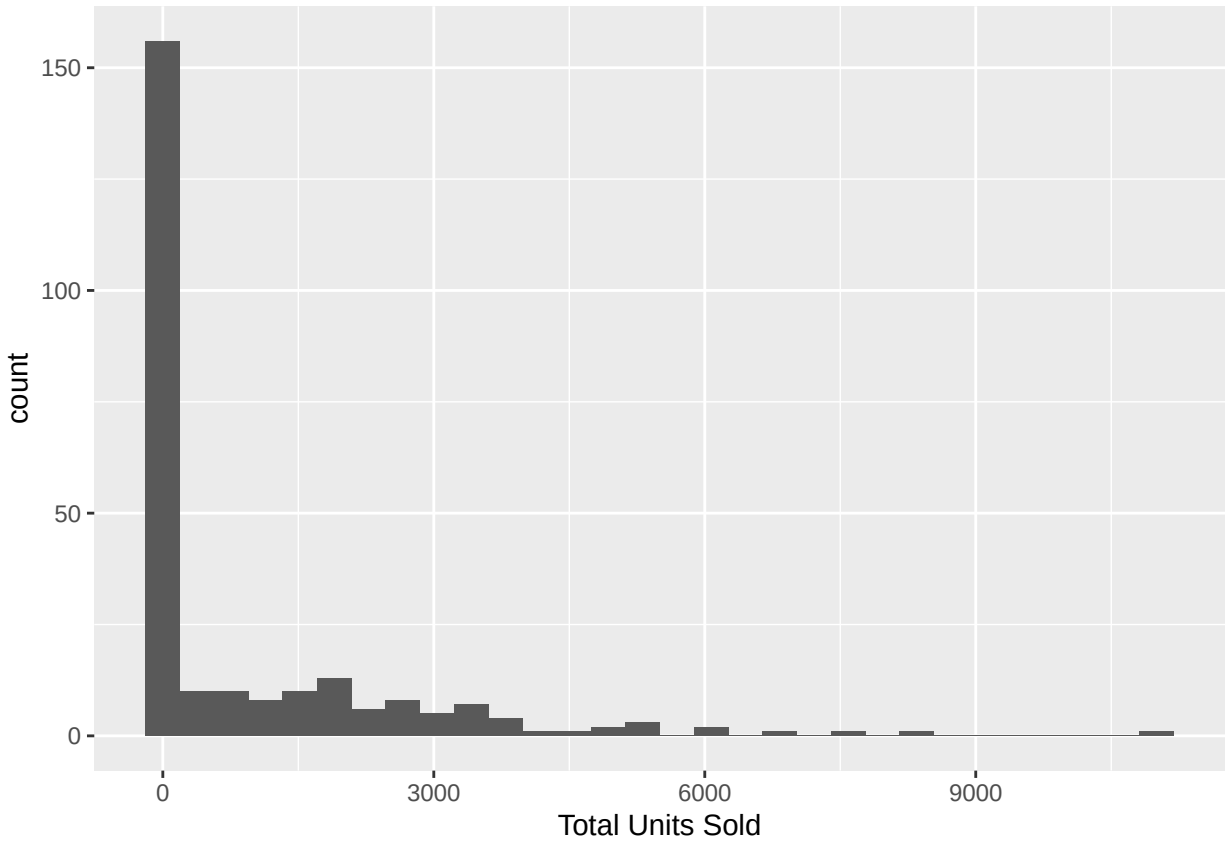


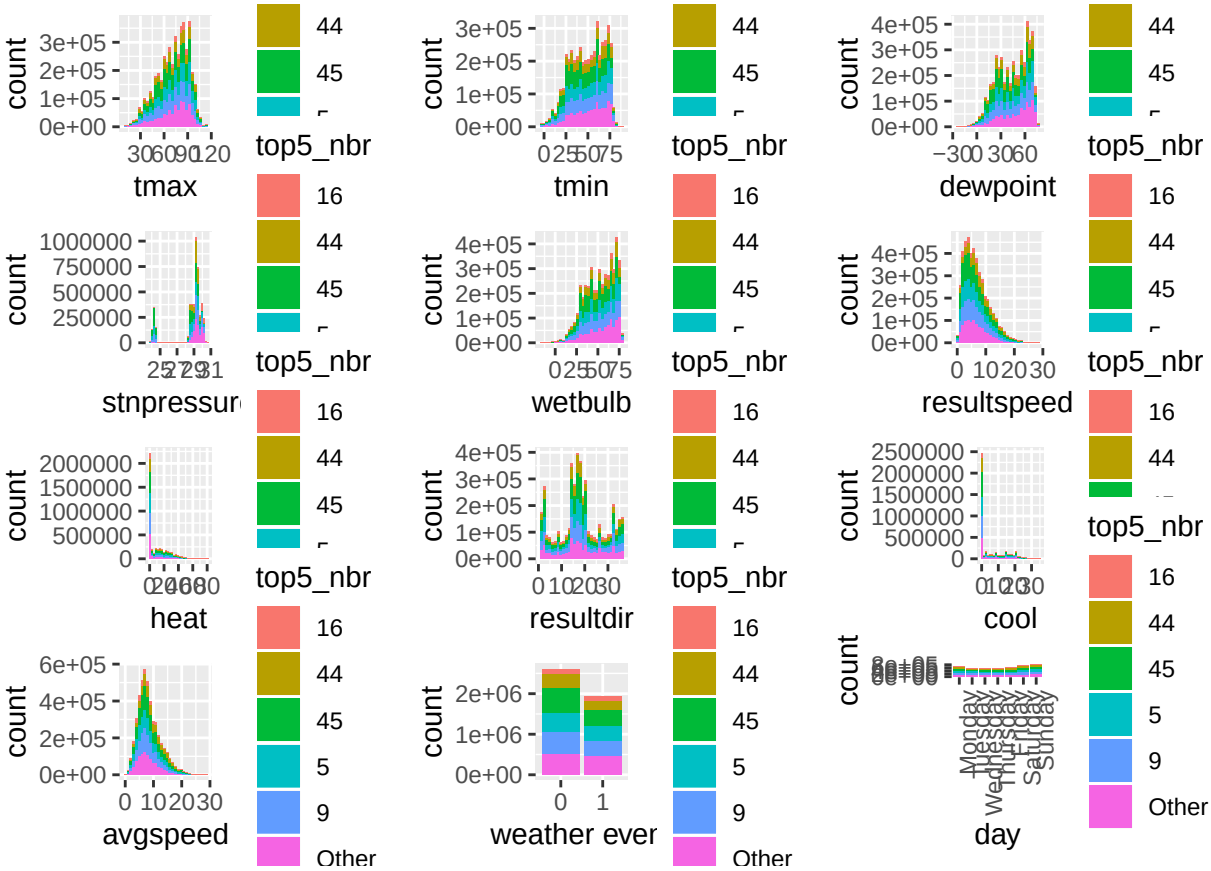
Fig.8 Distribution of Units Sold by Store and Item without 0



Data Dictionary

- `item_nbr`: categorical variable for each of the 111 types of store items
- `nbr_unit`: integer value of how many units of the corresponding item were sold
- `tmax`: maximum temperature (degrees Fahrenheit)
- `tmin`: minimum temperature (degrees Fahrenheit)
- `dewpoint`: average dew point; the higher the dew point, the greater the amount of moisture in the air
- `wetbulb`: average wet bulb temperature; lowest temperature to which air can be cooled by the evaporation of water into the air at constant pressure
- `heat`: heating degree day (HDD) with season beginning with July; measures the demand for energy needed to heat a building; number of degrees that a day's average temperature is below 65°F
- `cool`: cooling degree day (CDD) with season beginning with January; measures the demand for energy needed to cool buildings; number of degrees that a day's average temperature is above 65°F
- `codesum`: binary: 1 if a weather event occurred, else 0.
- `stnpressure`: average station pressure
- `resultspeed`: resultant wind speed; average of all wind speeds at a given place for a certain period
- `resultdir`: resultant direction; average of all wind directions at a given place for a certain period
- `avgspeed`: average wind speed

EDA


count
tmax
tmin
dewpoint
stnpressur
wetbulb
resultspeed
heat
resultdir
cool
avgspeed
weather ever
day
top5_nbr
16
44
45
5
9
Other
Monday
Tuesday
Wednesday
Thursday
Friday
Saturday
Sunday

References