What makes a Halloween Candy Desirable?

Anav Vora and Lavanya Kudli

10/28/2024

Introduction

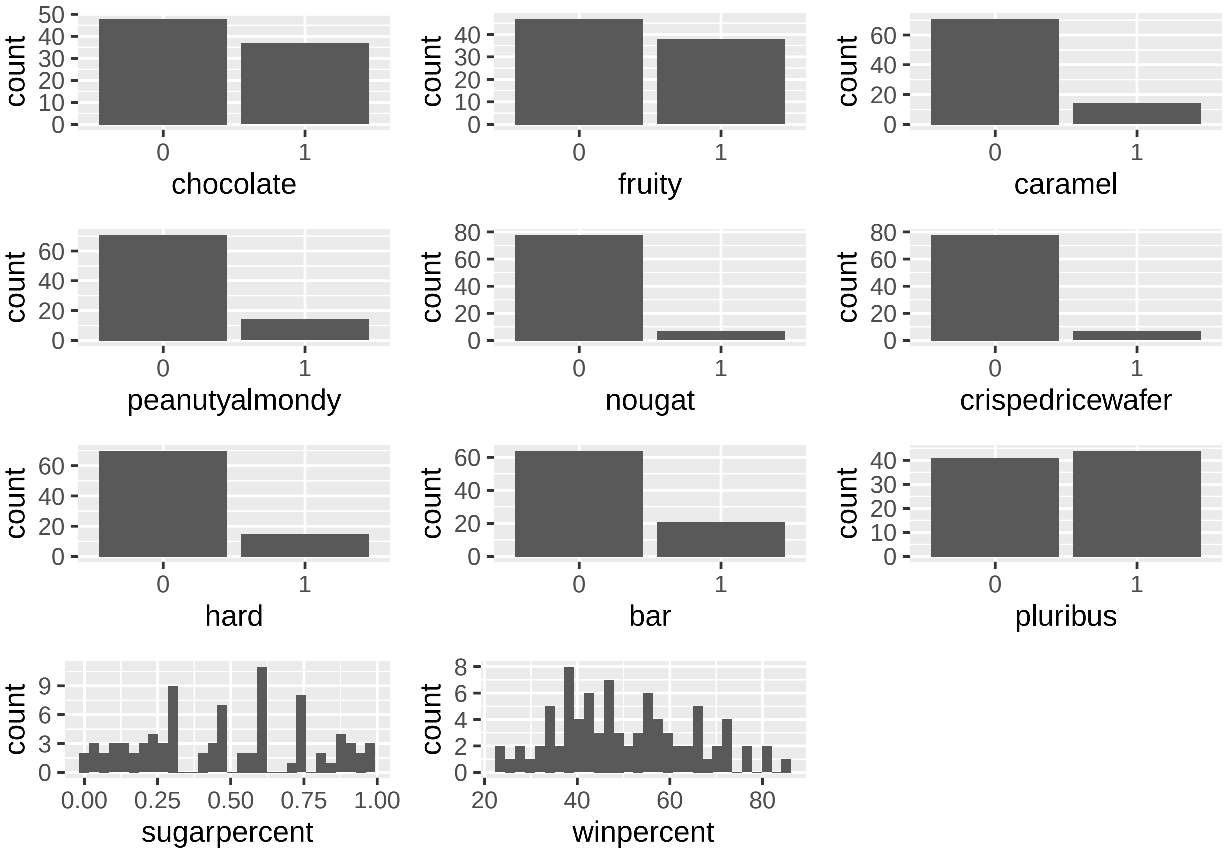
Halloween candy is a beloved treat enjoyed by both children and adults alike making it a fascinating subject for statistical analysis. To understand drivers of consumer preferences with respect to Halloween candy,FiveThirtyEight conducted a survey by generating an extensive dataset through 269,000 match-ups between various candies, garnering 8,371 survey responses. This dataset also includes a computed win percentage (referred to as winpercent) for each candy, reflecting the proportion of wins it achieved in these match-ups. Corresponding to the win percentages, the presence of ten relevant variables in the candy, such as, chocolate, fruity, caramel, peanutalmondy, nougat, crispedricewafer, hard, bar, pluribus, and sugarpercent are also available at our disposal.

We are motivated to identify and understand the specific characteristics that make a Halloween candy desirable. Using the winpercent as the response and the 10 variables as the predictors we will utilize statistical data analysis techniques for a multiple linear regression framework to identify characteristics that are most strongly associated with consumer preference for Halloween candy.For all our analysis we selected 0.05 as the alpha.

Data Description

After thoroughly verifying the data to ensure absence of missing values we begin the necessary data exploration which is an essential step before conducting statistical modelling.

To conduct exploratory data analysis, we construct histograms to understand the distribution of qualities that might be influencing the desirability of Halloween candy.



We see that very few of the candies in the dataset include caramel, peanuts/almonds, nougat, and crisped rice/wafer in the ingredient list. Further, there are very few hard candies and candy bars in the dataset. Thus, we should use caution when interpreting the influence of the above mentioned ingredients and qualities on candy desirability, as there is lack of data in our sample recording candy desirability in the presence of caramel,peanuts/almonds,nougat,crisped rice/wafer, hard and bar type candy.

Model Fitting

We fit a multiple linear regression model to examine the relationship between candy qualities and desirability computed using the winpercent.

```
Candy.mlr = lm(winpercent~.,data=Candy_Data_Subset)
summary(Candy.mlr)
```

```
##
## Call:
## lm(formula = winpercent ~ ., data = Candy_Data_Subset)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -20.7320  -6.1438   0.4359   6.2276  24.2048
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)         33.262      4.159   7.997 1.31e-11 ***
## chocolate1          19.216      3.871   4.964 4.30e-06 ***
## fruity1              9.587      3.764   2.547  0.01294 *
## caramel1             1.982      3.654   0.542  0.58918
## peanutyalmondy1      9.454      3.574   2.645  0.00996 **
## nougat1              1.918      5.628   0.341  0.73422
## crispedricewafer1    8.637      5.267   1.640  0.10529
## hard1               -5.866      3.448  -1.701  0.09306 .
## bar1                -1.233      4.821  -0.256  0.79879
## pluribus1           -1.147      3.031  -0.378  0.70629
## sugarpercent         7.490      4.421   1.694  0.09444 .
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## Residual standard error: 10.71 on 74 degrees of freedom
## Multiple R-squared:  0.5329, Adjusted R-squared:  0.4698
## F-statistic: 8.443 on 10 and 74 DF,  p-value: 5.252e-09
```

The MLR model can be written as: winpercent = 33.262 + 19.216chocolate + 9.587fruity + 1.982caramel + 9.454peanutyalmondy + 1.918nougat + 8.637crispedricewafer - 5.866hard - 1.233bar - 1.147pluribus + 7.490sugarpercent

Before making interpretations from the model, we must perform diagnostics and check if there are any unusual observations in the data.

We begin by checking if there are any high leverage points, i.e., if there are any data points that are very far from the center of the whole sample. To do so we perform the following steps:

1) We identify the leverages that are higher than the 2p/n threshold

2) For any leverages obtained, we classify these high leverage points between good or bad. Good high-leverage points are the ones for which the y value follows the pattern of the rest of the data, but with an xi value that is far away from the sample mean, whereas bad high-leverage points are the ones for which the y value does not follow the pattern suggested by the rest of the data, so the LS fitting might change a lot if we remove this point.

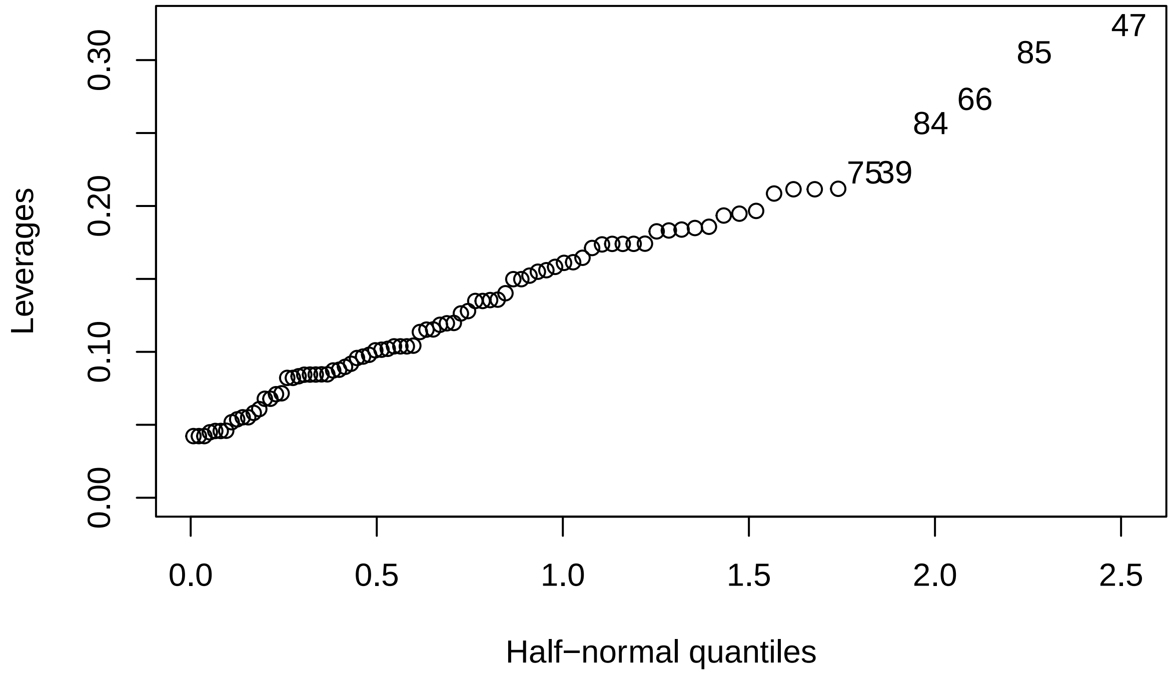
3) Then we estimate the IQR for our dependent variable, winpercent, in our original (full) data frame and use this metric to identify the high-leverage observations that don’t follow the pattern of the data.

```r
candy.leverages = lm.influence(Candy.mlr)$hat
n=dim(Candy_Data_Subset)[1]
p=length(variable.names(Candy.mlr))
candy.leverages.high = candy.leverages[candy.leverages>2*p/n]
candy.leverages.high
```

```
##        47        66        85
## 0.3241648 0.2733416 0.3053670
```

We observe 3 points with high leverage which must be classified into into good and bad.

```r
library(faraway)
halfnorm(candy.leverages, nlab=6, labs=as.character(1:length(candy.leverages)), ylab="Leverages")
```



In the plot above we see that none of the leverages seems to be unusually large.

Next we classify the leverages into good and bad

```
#Calculate the IQR for the dependent variable
IQR_win = IQR(Candy_Data_Subset$winpercent)

#Define a range with its lower limit being (Q1 - IQR) and upper limit being (Q3 + IQR)
QT1_win = quantile(Candy_Data_Subset$winpercent,0.25)
QT3_win = quantile(Candy_Data_Subset$winpercent,0.75)

lower_lim_win = QT1_win - IQR_win
upper_lim_win= QT3_win + IQR_win

vector_lim_win = c(lower_lim_win,upper_lim_win)

# Extract observations with high leverage points from the original data frame
candy.highlev = Candy_Data_Subset[candy.leverages>2*p/n,]

# Select only the observations with leverage points outside the range
candy.highlev_lower = candy.highlev[candy.highlev$winpercent < vector_lim_win[1], ]
candy.highlev_upper = candy.highlev[candy.highlev$winpercent > vector_lim_win[2], ]
candy.highlev2 = rbind(candy.highlev_lower,candy.highlev_upper)
candy.highlev2
```

```
##  [1] chocolate        fruity           caramel          peanutyalmondy
##  [5] nougat           crispedricewafer hard             bar
##  [9] pluribus         sugarpercent     winpercent
## <0 rows> (or 0-length row.names)
```

Based on our analysis we can conclude that none of our observations are bad leverage points.

Next, we examine using the outlier test if there are points which do not fit the model as well as others. For this we compare our studentized residual with bonferroni corrected values to be certain that the overall type I error rate is no greater than α. When doing so, each case would be tested at level α/n where n is total number of observations.

```
#Outlier detection
candy.resid = rstudent(Candy.mlr)
candy.resid.sorted = sort(abs(candy.resid), decreasing=TRUE)[1:10]
candy.resid.sorted
```

```
##       69        8       52       45       11       85       66       60
## 2.401679 2.156369 2.052663 2.030742 2.006484 1.935938 1.813454 1.811340
##       53       67
## 1.734393 1.728569
```

```
bonferroni_cv = qt(.05/(2*n), n-p-1)
bonferroni_cv
```

```
## [1] -3.593956
```

From our analysis, we can conclude that there are no outliers in the dataset as even the largest studentized resiudal, which in our case is 2.402, is not larger than abs(bonferroni cv) which is 3.594.

Finally, we check if there are any observations whose removal greatly affects the regression analysis i.e., influential points

```
#Influential observations
candy.cooks = cooks.distance(Candy.mlr)
sort(candy.cooks, decreasing = TRUE)[1:10]
```

```
##         85         66         11          8         52         80         53
## 0.14441837 0.10908569 0.09429055 0.08496175 0.07227392 0.05354287 0.04881193
##         84         10         29
## 0.04562158 0.04447935 0.04272190
```

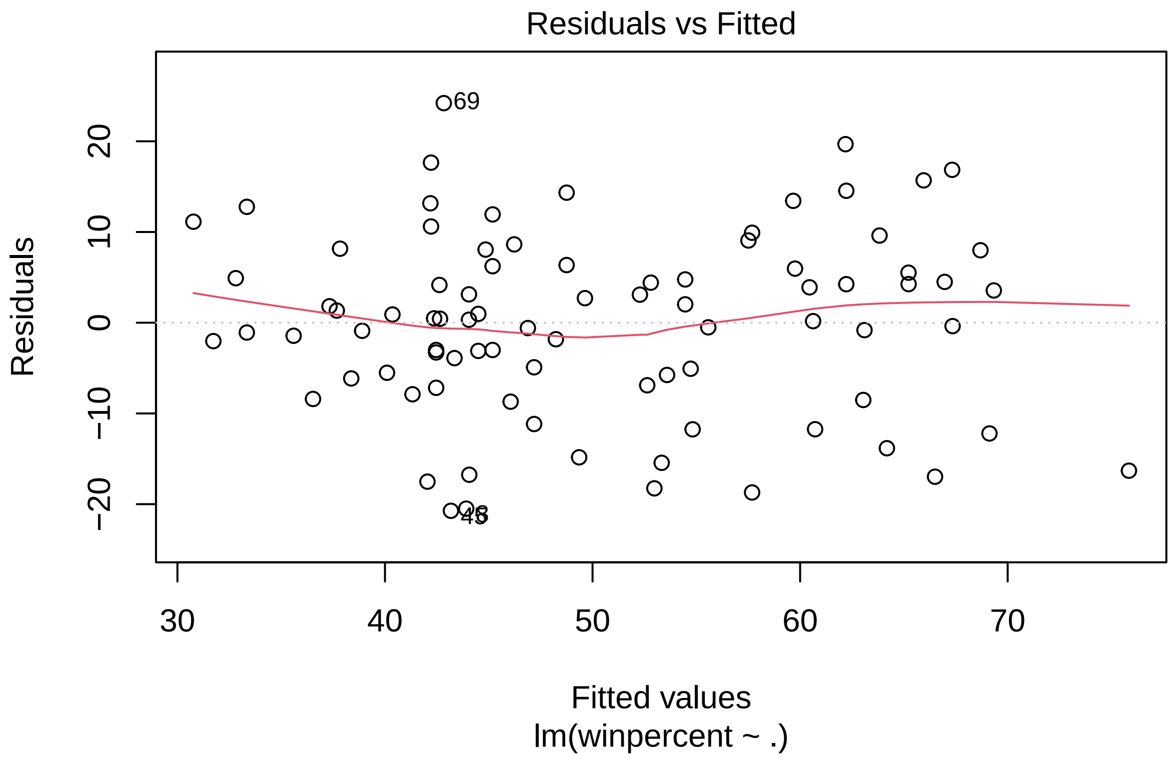
Since none of the cooks distance is above 1, therefore we can say that there are no influential points in the data.

Overall, from all our analyses we do not observe any bad high leverage points, outliers, and influential observations.

Therefore,we can proceed to model assumption, i.e., check if any of the assumptions of MLR are violated.

We begin by examining if the variance of the residuals is constant. To do so we will plot residuals versus fitted values. We will also verify this using the BP test.

```r
plot(Candy.mlr,which=1) #Shows variance is constant
```



```r
library(lmtest)
bptest(Candy.mlr)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  Candy.mlr
## BP = 14.988, df = 10, p-value = 0.1325
```

Since the residuals look like a football-shaped cloud, we can say that the variance is constant. So we do not need to do variance stabilization. To verify this we also do the Breusch-Pagan Test.

The hypothesis for the BP test is,

H0 :the variance is constant

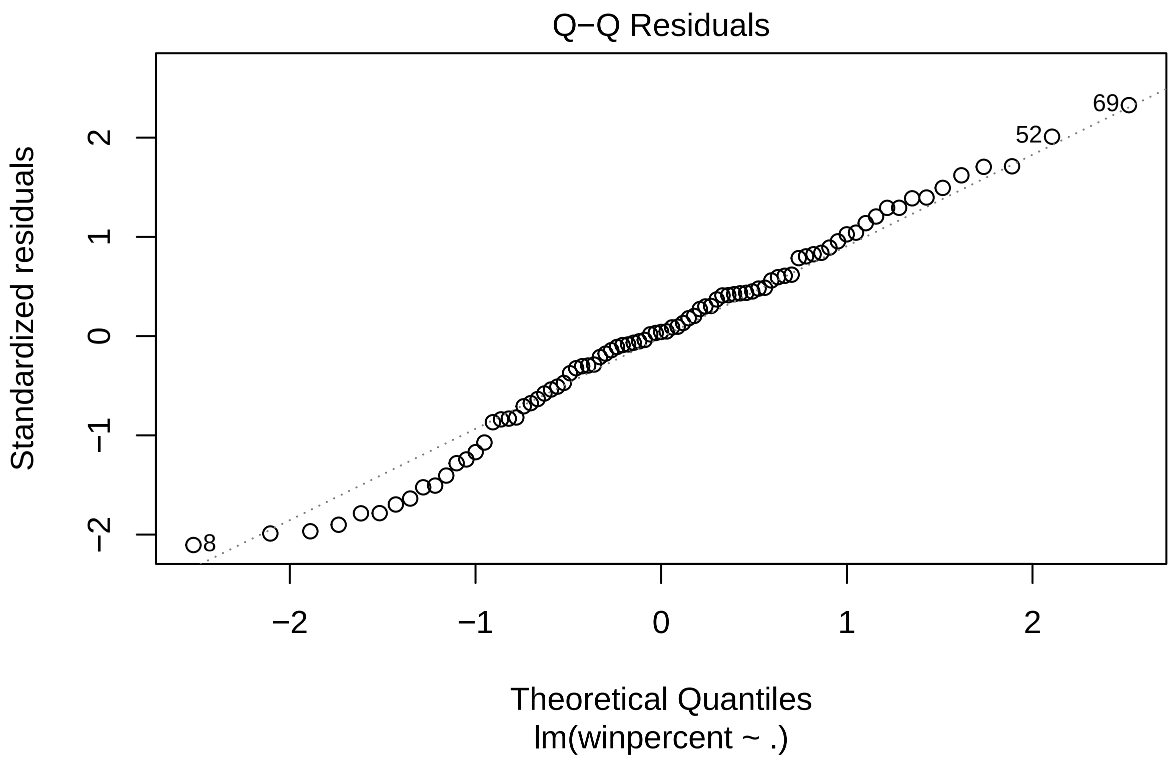
Hα: the variance is not constant

Since the p-value is greater than 0.05, we fail to reject the null and conclude that the variance is constant.

Next, we check if the error terms are normally distributed by using the following two approaches:-

1) a Q-Q plot
2) a KS test since our n>50

```
plot(Candy.mlr,which=2)
```



```
ks.test(Candy.mlr$residuals,"pnorm") #Normality violated
```

```
##
##  Exact one-sample Kolmogorov-Smirnov test
##
## data:  Candy.mlr$residuals
## D = 0.42013, p-value = 3.997e-14
## alternative hypothesis: two-sided
```

From the Q-Q test we see that there is a slight departure from normality at the lower end.

We also do the KS test with the below hypothesis:

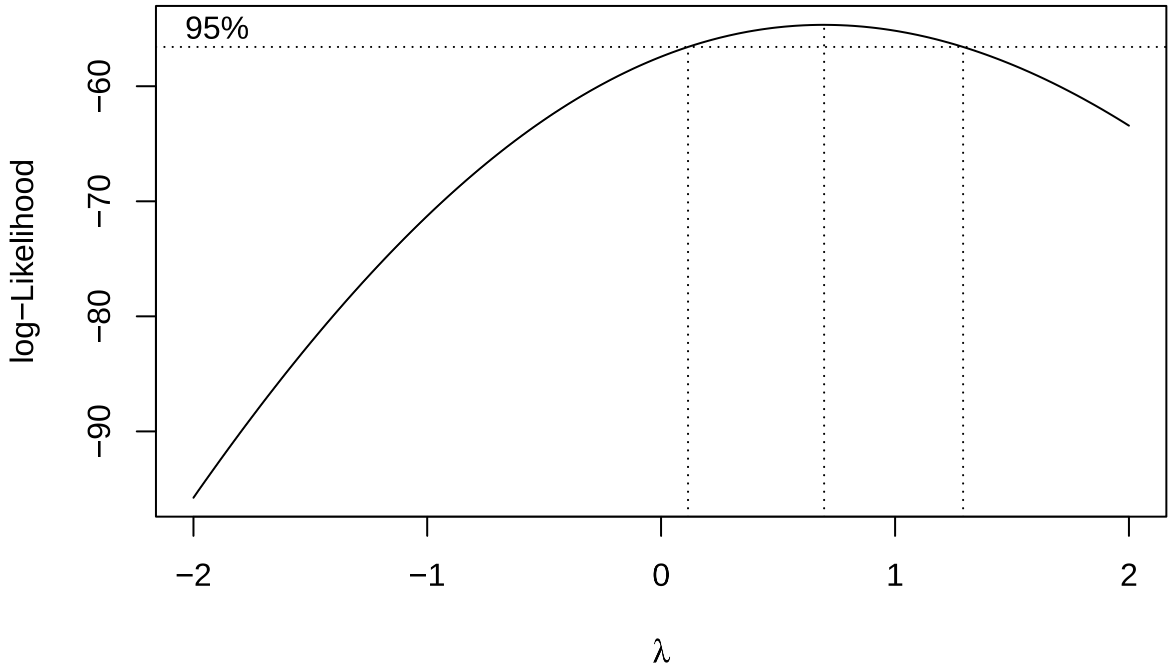
H0: the distribution is normal

Hα: the distribution is not normal

The p-value of 3.997e-14 is less than 0.05. So, we reject the null hypotheses of normality and conclude that the normality assumption is not satisfied.

As the normality assumption is violated we try a box-cox transformation.

```
library(MASS)
Candy.transformation = boxcox(Candy.mlr, lambda=seq(-2,2, length=400))
```

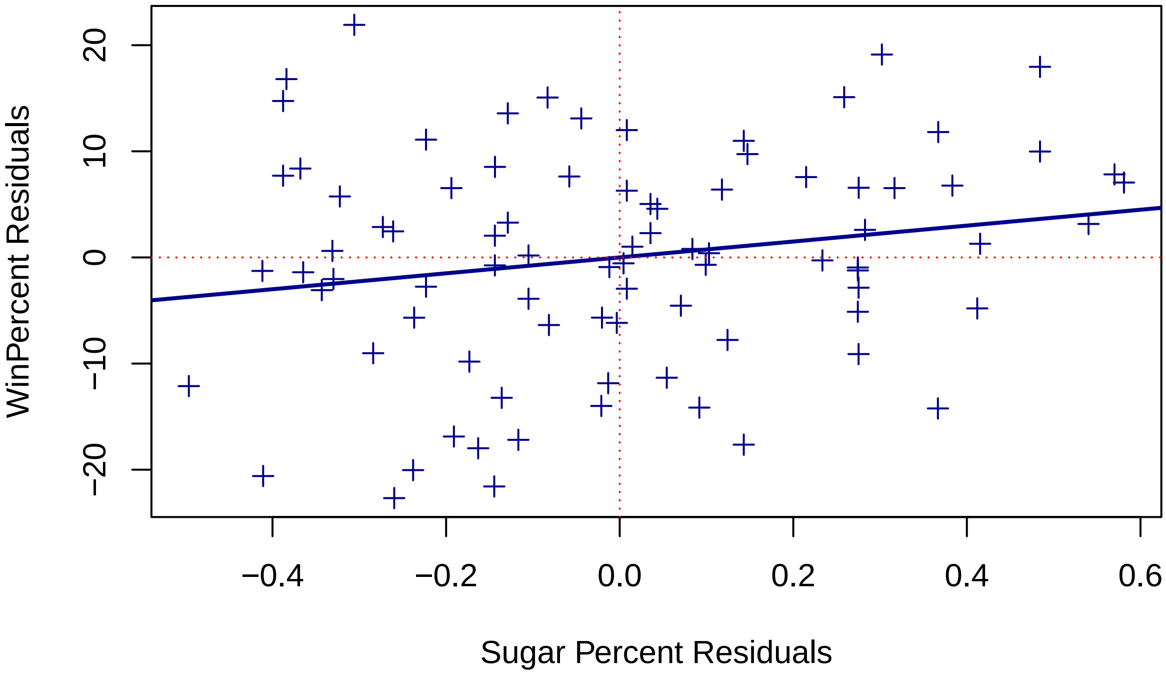


From the box-cox results, we see that the optimal lambda is close to 1 and that 1 is in the 95% interval. So we can proceed without any transformation.

We do not need to check for independence of error terms (i.e., correlation) as the data we have is not collected in an ordered sequence.

We next check if the assumption of a linear dependence is satisfied. We only do this for continuous data. Our sample includes only one continuously distributed predictor - sugar percentage.

```
y.sugarpercent = lm(winpercent~chocolate+fruity+caramel+peanutyalmondy+nougat+
                      crispedricewafer+hard+bar+pluribus,data=Candy_Data_Subset)$residuals
x.sugarpercent = lm(sugarpercent~chocolate+fruity+caramel+peanutyalmondy+nougat+
                      crispedricewafer+hard+bar+pluribus,data=Candy_Data_Subset)$residuals
plot(x.sugarpercent, y.sugarpercent, xlab="Sugar Percent Residuals", ylab="WinPercent Residuals",
     col='Darkblue', pch=3, size=3)
abline(lm(y.sugarpercent ~ x.sugarpercent), col='Darkblue', lwd=2)
abline(v = 0, col="red", lty=3)
abline(h = 0, col="red", lty=3)
```



Since the points appear to be randomly scattered around the fitted regression line, so the linearity assumption is satisfied here.

We do not need to check for collinearity in the predictors as we only have one continuously distributed predictor in the design matrix.

As none of the assumptions for MLR are violated, we can start fine-tuning the model to retain only statistically significant predictors. We take a backward selection approach, i.e., we start dropping variables from the full model in order of decreasing p-value (of t-tests in the full model). After each drop of variables, we perform a partial anova test, and compare the reduced models with the full model (that includes all the variables).

We begin by dropping the predictor "bar" which has the highest p-value of 0.799 in the full model (through a t-test for individual betas)

We perform partial F-tests for dropping each predictor. We use the following generic form of the hypothesis test for all fine tuning.

Hypotheses:

H0: $\beta_i = \beta_j = ... = \beta_k = 0$

Hα: Atleast one of β_i, β_j, . . . β_k is not equal to zero

where i, j,. . . , k are the predictors being dropped together. One or more predictors can be dropped together from the full model.

Decision rule: If the p-value is less than 0.05 then reject the null hypothesis.

```r
#Dropping bar
candy.red1 = lm(winpercent ~ chocolate + fruity + caramel + peanutyalmondy + nougat +
                  crispedricewafer + hard + pluribus + sugarpercent,
                data=Candy_Data_Subset)
anova(candy.red1,Candy.mlr)
```

```
## Analysis of Variance Table
##
## Model 1: winpercent ~ chocolate + fruity + caramel + peanutyalmondy +
##     nougat + crispedricewafer + hard + pluribus + sugarpercent
## Model 2: winpercent ~ chocolate + fruity + caramel + peanutyalmondy +
##     nougat + crispedricewafer + hard + bar + pluribus + sugarpercent
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1     75 8502.2
## 2     74 8494.7  1    7.5136 0.0655 0.7988
```

The p-value obtained is 0.799 (> 0.05), therefore we fail to reject the null that $\beta_{bar} = 0$, hence we can drop the predictor bar. We can now examine if we can drop nougat also along with bar

```r
#Dropping nougat (next largest p value) and bar together
candy.red2 = lm(winpercent ~ chocolate + fruity + caramel + peanutyalmondy +
                  crispedricewafer + hard + pluribus + sugarpercent ,
                data=Candy_Data_Subset)
anova(candy.red2,Candy.mlr)
```

```
## Analysis of Variance Table
##
## Model 1: winpercent ~ chocolate + fruity + caramel + peanutyalmondy +
##     crispedricewafer + hard + pluribus + sugarpercent
## Model 2: winpercent ~ chocolate + fruity + caramel + peanutyalmondy +
##     nougat + crispedricewafer + hard + bar + pluribus + sugarpercent
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1     76 8509.2
## 2     74 8494.7  2    14.496 0.0631 0.9389
```

The p-value obtained is 0.939 (> 0.05), therefore we fail to reject the null that $\beta_{bar} = \beta_{nougat} = 0$, hence we can drop the predictors bar and nougat.

We can now examine if we can drop pluribus along with nougat and bar

```r
#Now dropping pluribus also (next largest p value)
candy.red3 = lm(winpercent ~ chocolate + fruity + caramel + peanutyalmondy +
                  crispedricewafer + hard + sugarpercent
                ,data=Candy_Data_Subset)
anova(candy.red3,Candy.mlr)
```

```
## Analysis of Variance Table
##
## Model 1: winpercent ~ chocolate + fruity + caramel + peanutyalmondy +
##     crispedricewafer + hard + sugarpercent
## Model 2: winpercent ~ chocolate + fruity + caramel + peanutyalmondy +
##     nougat + crispedricewafer + hard + bar + pluribus + sugarpercent
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1     77 8524.5
## 2     74 8494.7  3    29.808 0.0866 0.9672
```

The p-value obtained is 0.967 (> 0.05), therefore we fail to reject the null that $\beta_{bar} = \beta_{nougat} = \beta_{pluribus} = 0$, hence we can drop the predictors bar, nougat, and pluribus.

Next we examine if we can drop caramel also

```r
#Now dropping caramel also (next largest p value)
candy.red4 = lm(winpercent ~ chocolate + fruity + peanutyalmondy +
                  crispedricewafer + hard + sugarpercent ,
                data=Candy_Data_Subset)
anova(candy.red4,Candy.mlr)
```

```
## Analysis of Variance Table
##
## Model 1: winpercent ~ chocolate + fruity + peanutyalmondy + crispedricewafer +
##     hard + sugarpercent
## Model 2: winpercent ~ chocolate + fruity + caramel + peanutyalmondy +
##     nougat + crispedricewafer + hard + bar + pluribus + sugarpercent
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1     78 8585.7
## 2     74 8494.7  4    90.972 0.1981 0.9386
```

The p-value obtained is 0.939 (> 0.05), therefore we fail to reject the null that $\beta_{bar} = \beta_{nougat} = \beta_{pluribus} = \beta_{caramel} = 0$, hence we can drop the predictors bar, nougat, pluribus, and caramel.

Checking now if crispedricewafer can also be dropped

```r
#Now dropping crispedricewafer also (next largest p value)
candy.red5 = lm(winpercent ~ chocolate + fruity + peanutyalmondy +
                  hard + sugarpercent
                ,data=Candy_Data_Subset)
anova(candy.red5,Candy.mlr)
```

```
## Analysis of Variance Table
##
## Model 1: winpercent ~ chocolate + fruity + peanutyalmondy + hard + sugarpercent
## Model 2: winpercent ~ chocolate + fruity + caramel + peanutyalmondy +
##     nougat + crispedricewafer + hard + bar + pluribus + sugarpercent
##   Res.Df    RSS Df Sum of Sq     F Pr(>F)
## 1     79 8970.5
## 2     74 8494.7  5    475.84 0.829 0.5331
```

The p-value obtained is 0.533 (> 0.05), therefore we fail to reject the null that $\beta_{bar} = \beta_{nougat} = \beta_{pluribus} = \beta_{caramel} = \beta_{crispedricewafer} = 0$, hence we can drop the predictors bar, nougat, pluribus, caramel, and crispedricewafer.

Now checking if we can drop sugarpercent as a predictor too

```r
#Now dropping sugarpercent also (next largest p value)
candy.red6 = lm(winpercent ~ chocolate + fruity + peanutyalmondy +
                  hard,data=Candy_Data_Subset)
anova(candy.red6,Candy.mlr)
```

```
## Analysis of Variance Table
##
## Model 1: winpercent ~ chocolate + fruity + peanutyalmondy + hard
## Model 2: winpercent ~ chocolate + fruity + caramel + peanutyalmondy +
##     nougat + crispedricewafer + hard + bar + pluribus + sugarpercent
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1     80 9428.2
## 2     74 8494.7  6    933.47 1.3553  0.244
```

The p-value obtained is 0.244 (> 0.05), therefore we fail to reject the null that $\beta_{bar} = \beta_{nougat} = \beta_{pluribus} = \beta_{caramel} = \beta_{crispedricewafer} = \beta_{sugarpercent} = 0$, hence we can drop the predictors bar, nougat, pluribus, caramel, crispedricewafer, and sugarpercent.

Testing if hard can be dropped as well

```
#Now dropping hard also (next largest p value)
candy.red7 = lm(winpercent ~ chocolate + fruity + peanutyalmondy ,data=Candy_Data_Subset)
anova(candy.red7,Candy.mlr)
```

```
## Analysis of Variance Table
##
## Model 1: winpercent ~ chocolate + fruity + peanutyalmondy
## Model 2: winpercent ~ chocolate + fruity + caramel + peanutyalmondy +
##     nougat + crispedricewafer + hard + bar + pluribus + sugarpercent
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1     81 9688.7
## 2     74 8494.7  7      1194 1.4859 0.1855
```

The p-value obtained is 0.185 (> 0.05), therefore we fail to reject the null that $\beta_{bar} = \beta_{nougat} = \beta_{pluribus} = \beta_{caramel} = \beta_{crispedricewafer} = \beta_{sugarpercent} = \beta_{hard} = 0$, hence we can drop the predictors bar, nougat, pluribus, caramel, crispedricewafer, sugarpercent, and hard.

Testing if we can drop fruity also next

```
#Now dropping fruity also (next largest p value)
candy.red8 = lm(winpercent ~ chocolate + peanutyalmondy, data=Candy_Data_Subset)
anova(candy.red8,Candy.mlr)
```

```
## Analysis of Variance Table
##
## Model 1: winpercent ~ chocolate + peanutyalmondy
## Model 2: winpercent ~ chocolate + fruity + caramel + peanutyalmondy +
##     nougat + crispedricewafer + hard + bar + pluribus + sugarpercent
##   Res.Df     RSS Df Sum of Sq      F  Pr(>F)
## 1     82 10236.0
## 2     74  8494.7  8    1741.3 1.8961 0.07325 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The p-value obtained is 0.073 (> 0.05), therefore we fail to reject the null that $\beta_{bar} = \beta_{nougat} = \beta_{pluribus} = \beta_{caramel} = \beta_{crispedricewafer} = \beta_{sugarpercent} = \beta_{hard} = \beta_{fruity} = 0$, hence we can drop the predictors bar, nougat, pluribus, caramel, crispedricewafer, sugarpercent,hard, and fruity.

Finally we check if we can drop peanutyalmondy also

```
#Now dropping peanutyalmondy also (next largest p value)
candy.red9 = lm(winpercent ~ chocolate, data=Candy_Data_Subset)
anova(candy.red9,Candy.mlr)
```

```
## Analysis of Variance Table
##
## Model 1: winpercent ~ chocolate
## Model 2: winpercent ~ chocolate + fruity + caramel + peanutyalmondy +
##     nougat + crispedricewafer + hard + bar + pluribus + sugarpercent
##   Res.Df     RSS Df Sum of Sq      F  Pr(>F)
## 1     83 10818.5
## 2     74  8494.7  9    2323.8 2.2493 0.02774 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We see that the p-value obtained is 0.027 (< 0.05), therefore we reject the null that $\beta_{bar} = \beta_{nougat} = \beta_{pluribus} = \beta_{caramel} = \beta_{crispedricewafer} = \beta_{sugarpercent} = \beta_{hard} = \beta_{fruity} = \beta_{peanutyalmondy} = 0$, and conclude that peanutyalmondy is an important predictor and can not be dropped.

We find that a model with both chocolate and peanutyalmondy as predictors is required. We re-check all the assumptions and perform diagnostics for the selected reduced model:

Beginning with checking leverages:

```
candy.final = lm(winpercent ~ chocolate + peanutyalmondy, data=Candy_Data_Subset)
candy.final.leverages = lm.influence(candy.final)$hat
n=dim(Candy_Data_Subset)[1]
p2=length(variable.names(candy.final))
candy.final.leverages.high = candy.final.leverages[candy.final.leverages>2*p2/n]
candy.final.leverages.high
```

```
##          6          7          8         33         41         43         47
## 0.07256796 0.07256796 0.11244664 0.07256796 0.07256796 0.07256796 0.11244664
##         48         52         53         54         55         65         66
## 0.07256796 0.07256796 0.07256796 0.07256796 0.07256796 0.07256796 0.07256796
```

We find 14 observations with high leverage. We must check if these are good or bad high leverage points.

```
# Extract observations with high leverage points from the original data frame
candy.final.highlev = Candy_Data_Subset[candy.final.leverages>2*p2/n,]

# Select only the observations with leverage points outside the range
candy.final.highlev_lower = candy.final.highlev[candy.final.highlev$winpercent < vector_lim_win[1], ]
candy.final.highlev_upper = candy.final.highlev[candy.final.highlev$winpercent > vector_lim_win[2], ]
candy.final.highlev2 = rbind(candy.final.highlev_lower,candy.final.highlev_upper)
candy.final.highlev2
```

```
##    chocolate fruity caramel peanutyalmondy nougat crispedricewafer hard bar
## 52         1      0       0              1      0                0    0   0
## 53         1      0       0              1      0                0    0   0
##    pluribus sugarpercent winpercent
## 52        0        0.034   81.86626
## 53        0        0.720   84.18029
```

We find two observations with bad high leverage (observations 52 and 53).

Next we check for outliers:

```
candy.final.resid = rstudent(candy.final)
candy.final.resid.sorted = sort(abs(candy.final.resid), decreasing=TRUE)[1:10]
candy.final.resid.sorted
```

```
##        8       69       60       80       61       63       11       45 
## 2.554929 2.342306 2.215680 2.162937 1.956068 1.905784 1.800803 1.775908 
##       53       29 
## 1.702336 1.690120
```

```
bonferroni_cv = qt(.05/(2*n), n-p2-1)
bonferroni_cv
```

```
## [1] -3.577889
```

Even the largest studentized residual (at 2.55) is smaller than the bonferronu cv (3.55). Hence, no outliers are detected.

Next, we test for influential observations:

```
candy.final.cooks = cooks.distance(candy.final)
sort(candy.final.cooks, decreasing = TRUE)[1:10]
```

```
##          8         53         60         80         52          6         63 
## 0.25825986 0.07387455 0.06088827 0.05818038 0.05620019 0.05570571 0.04572695 
##         11         69         29 
## 0.04101652 0.03720543 0.03629592
```

None of the cook's distances are greater than 1, and hence, we don't have any influential observations.

Next, we perform model diagnostics to detect deviations from MLR assumptions, beginning with checking the variance of the residuals. We conduct the Breusch-Pagan (BP) Test.

The hypothesis for the BP test is,

H0 :the variance is constant

Hα: the variance is not constant

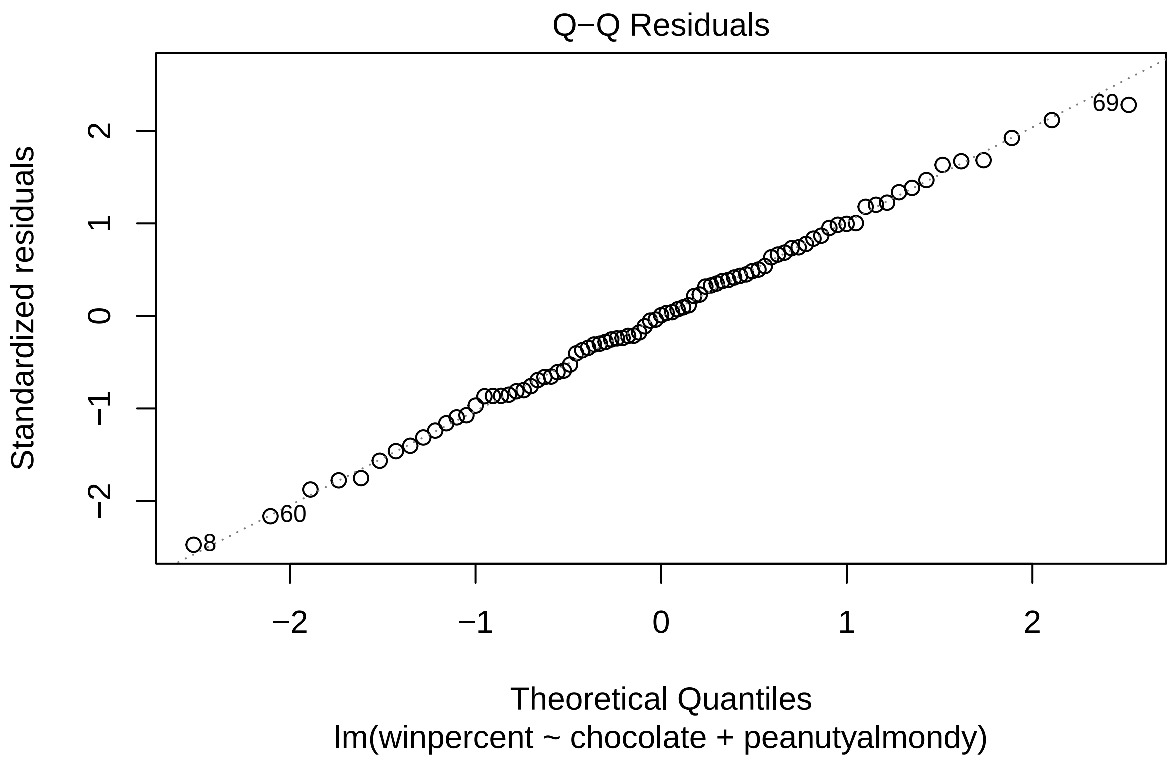
```
bptest(candy.final)
```

```
## 
##  studentized Breusch-Pagan test
## 
## data:  candy.final
## BP = 0.76774, df = 2, p-value = 0.6812
```

Since the p-value is 0.68 (> 0.05), we fail to reject the null and conclude that the variance is constant.

Next we check if the normality of residuals assumption is violated via the QQ plot and the KS test.

```
plot(candy.final,which=2)
```

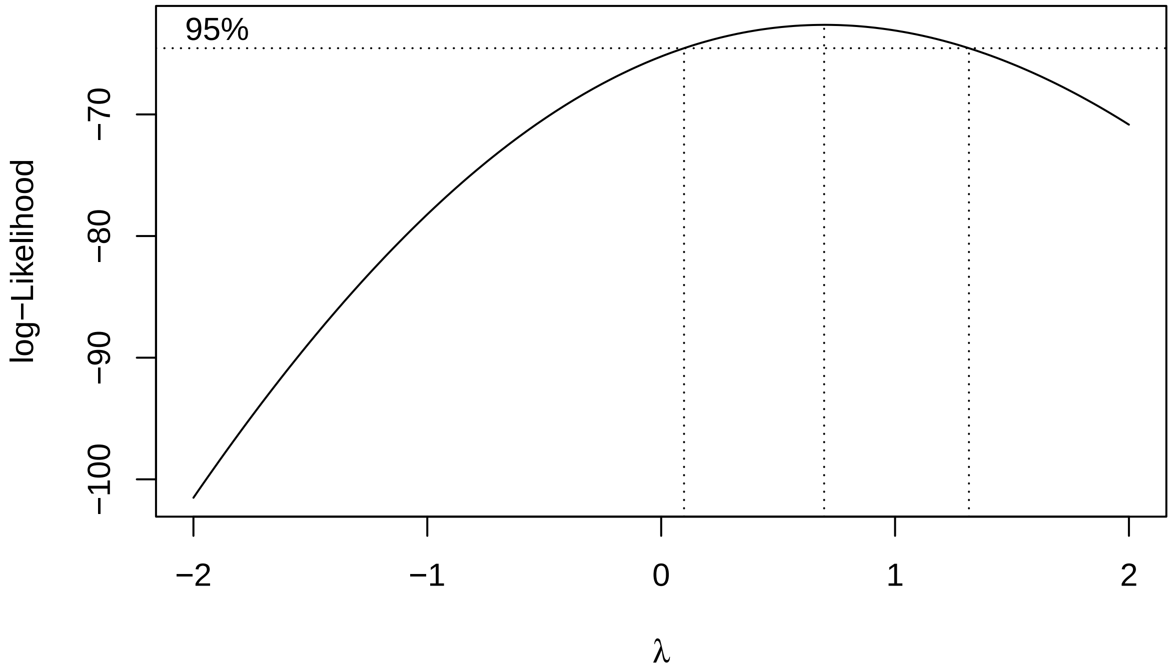


```
ks.test(candy.final$residuals,"pnorm")
```

```
##
##  Exact one-sample Kolmogorov-Smirnov test
##
## data:  candy.final$residuals
## D = 0.43802, p-value = 2.22e-15
## alternative hypothesis: two-sided
```

The QQ plot does not indicate departure from normality as the points fall on a stratigh line, however the p-value less than 0.05 from the KS test suggests a departure from normality. So we attempt a tranformation to check if box-cox can fix normality.

```
Candy.final.transformation = boxcox(candy.final, lambda=seq(-2,2, length=400))
```



As 1 fall within the 95% confidence interval for lambda, we proceed without any transformations.

We do not need to check for independence of residuals as the data is not collected in an ordered sequence. We also need not check for linearity as both chocolate and peanutyalmondy are categorical predictors. Due to both predictors being categorical, we also need not check for collinearity in the predictors.

Model Application

Our checks reveal that the selected reduced model can be used for estimation and prediction. We use estimation (confidence interval) here and not prediction interval, because the candies we select are a part of the training dataset

```
KitKat = Candy_Data_Subset[which(Candy_Data$competitorname=="Kit Kat"),]
CandyCorn = Candy_Data_Subset[which(Candy_Data$competitorname=="Candy Corn"),]
AlmondJoy = Candy_Data_Subset[which(Candy_Data$competitorname=="Almond Joy"),]
predict.lm(candy.final,KitKat,interval = "confidence")
```

```
##         fit      lwr      upr
## 29 58.44926 54.14408 62.75445
```

```
predict.lm(candy.final,CandyCorn,interval = "confidence")
```

```
##        fit      lwr    upr
## 9 41.82464 38.60328 45.046
```

```
predict.lm(candy.final,AlmondJoy,interval = "confidence")
```

```
##        fit      lwr      upr
## 6 66.07208 60.08473 72.05943
```

We find that the win percent of Candy Corn (41.82%) which does not have chocoloat or peanuts/almonds is the least amongst the three selected candies. Next, Kit Kat which contains chocolate but not almonds, has a higher winpercent (58.45%) compared to Candy Corn. The candy with the highest win percent is Almond Joy (66.07%) which contains both chocolate and almonds/peanuts.

Model Interpretation

To beta values of an MLR model reveal the influence of different predictors on the response. Hence, checking the beta for chocolate and peanutyalmondy in the selected model

```
summary(candy.final)
```

```
## 
## Call:
## lm(formula = winpercent ~ chocolate + peanutyalmondy, data = Candy_Data_Subset)
## 
## Residuals:
##      Min       1Q   Median       3Q      Max 
## -26.0296  -7.6657   0.0797   7.3629  25.2130 
## 
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)    
## (Intercept)       41.825      1.619  25.828  < 2e-16 ***
## chocolate1        16.625      2.640   6.297 1.42e-08 ***
## peanutyalmondy1    7.623      3.529   2.160   0.0337 *  
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 11.17 on 82 degrees of freedom
## Multiple R-squared:  0.4372, Adjusted R-squared:  0.4235 
## F-statistic: 31.85 on 2 and 82 DF,  p-value: 5.822e-11
```

The result shows that the difference in the mean win percentage for two candies with and without the presence of chocolate as an ingredient is 16.625. Similarly, the difference in the mean win percentage for two candies with and without the presence of peanuts/almonds as an ingredient is 7.62. Hence, an ideal candy should have chocolate and peanuts/almonds as ingredients.

Conclusion

In our quest of identifying the best features in a halloween candy we find that it should contain both chocolate and peanuts/almonds.Our finds were based on crucial steps including exploratory data analysis, model fitting, model diagonostics and model testing.