final_notes

Zach White

December 5, 2016

She mentions location, location, location

```
lm.anova = lm(price~Neighborhood,data = ames_train)
summary(lm.anova)
```

```
##
## Call:
## lm(formula = price ~ Neighborhood, data = ames_train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -160647  -24832   -4462   18716  281353
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)          198634.55   15842.07  12.538  < 2e-16 ***
## NeighborhoodBlueste  -72834.55   34222.78  -2.128 0.033567 *
## NeighborhoodBrDale   -99704.55   22957.34  -4.343 1.55e-05 ***
## NeighborhoodBrkSide  -76161.98   17841.10  -4.269 2.16e-05 ***
## NeighborhoodClearCr   -5480.70   21525.14  -0.255 0.799072
## NeighborhoodCollgCr   -1683.69   16835.97  -0.100 0.920361
## NeighborhoodCrawfor    5562.01   18605.57   0.299 0.765047
## NeighborhoodEdwards  -62312.53   17233.18  -3.616 0.000315 ***
## NeighborhoodGilbert   -5306.53   17530.31  -0.303 0.762179
## NeighborhoodGreens      -72.05   30678.04  -0.002 0.998127
## NeighborhoodGrnHill   81365.45   40389.51   2.015 0.044230 *
## NeighborhoodIDOTRR  -101013.86   18161.71  -5.562 3.45e-08 ***
## NeighborhoodMeadowV -105687.67   20579.45  -5.136 3.40e-07 ***
## NeighborhoodMitchel  -33620.91   17711.97  -1.898 0.057965 .
## NeighborhoodNAmes    -57278.57   16394.57  -3.494 0.000498 ***
## NeighborhoodNoRidge   97210.10   18696.71   5.199 2.44e-07 ***
## NeighborhoodNPkVill  -59359.55   30678.04  -1.935 0.053290 .
## NeighborhoodNridgHt  135012.19   17303.30   7.803 1.56e-14 ***
## NeighborhoodNWAmes    -4540.64   17841.10  -0.255 0.799160
## NeighborhoodOldTown  -78408.94   17025.10  -4.605 4.66e-06 ***
## NeighborhoodSawyer   -59321.84   17211.28  -3.447 0.000592 ***
## NeighborhoodSawyerW  -15533.55   17634.80  -0.881 0.378619
## NeighborhoodSomerst   35961.33   16978.74   2.118 0.034426 *
## NeighborhoodStoneBr  140681.50   19723.22   7.133 1.92e-12 ***
## NeighborhoodSWISU    -68015.05   21932.35  -3.101 0.001983 **
## NeighborhoodTimber    66557.61   19906.54   3.344 0.000859 ***
## NeighborhoodVeenker   35015.45   22957.34   1.525 0.127524
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 52540 on 973 degrees of freedom
## Multiple R-squared:  0.5992, Adjusted R-squared:  0.5885
## F-statistic: 55.96 on 26 and 973 DF,  p-value: < 2.2e-16
```

```r
anova(lm.anova)
```

```
## Analysis of Variance Table
##
## Response: price
##               Df     Sum Sq    Mean Sq F value    Pr(>F)
## Neighborhood  26 4.0164e+12 1.5448e+11  55.956 < 2.2e-16 ***
## Residuals    973 2.6861e+12 2.7607e+09
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
#predict.lm(lm.anova,ames_test)

# Build Hierarchical Model with location
ames_train %>% group_by(Neighborhood) %>%
  summarise(min_price = min(price),q1 = quantile(price,.25),med.price = quantile(price,.5),mean.price =
```

```
## # A tibble: 27 × 7
##    Neighborhood min_price       q1 med.price mean.price       q3 max.price
##          <fctr>     <int>    <dbl>     <dbl>      <dbl>    <dbl>     <int>
## 1       Blmngtn    172500 182112.5    191000   198634.5 205000.0    246990
## 2       Blueste    116500 120200.0    123900   125800.0 130450.0    137000
## 3        BrDale     83000  88250.0     98750    98930.0 104975.0    125500
## 4       BrkSide     39300  93000.0    124000   122472.6 135000.0    207000
## 5       ClearCr    107500 164000.0    185000   193153.8 240000.0    277000
## 6       CollgCr    110000 160000.0    195800   196950.9 218836.0    475000
## 7       Crawfor     96500 154900.0    205000   204196.6 235000.0    392500
## 8       Edwards     61500 112250.0    127250   136322.0 148000.0    415000
## 9       Gilbert    133000 171500.0    183500   193328.0 199500.0    377500
## 10       Greens    155000 197375.0    212625   198562.5 213812.5    214000
## # ... with 17 more rows
```

```r
# Levels of Neighborhood
levels.train = levels(ames_train$Neighborhood)
levels.test = levels(ames_test$Neighborhood)

levels.train == levels.test
```

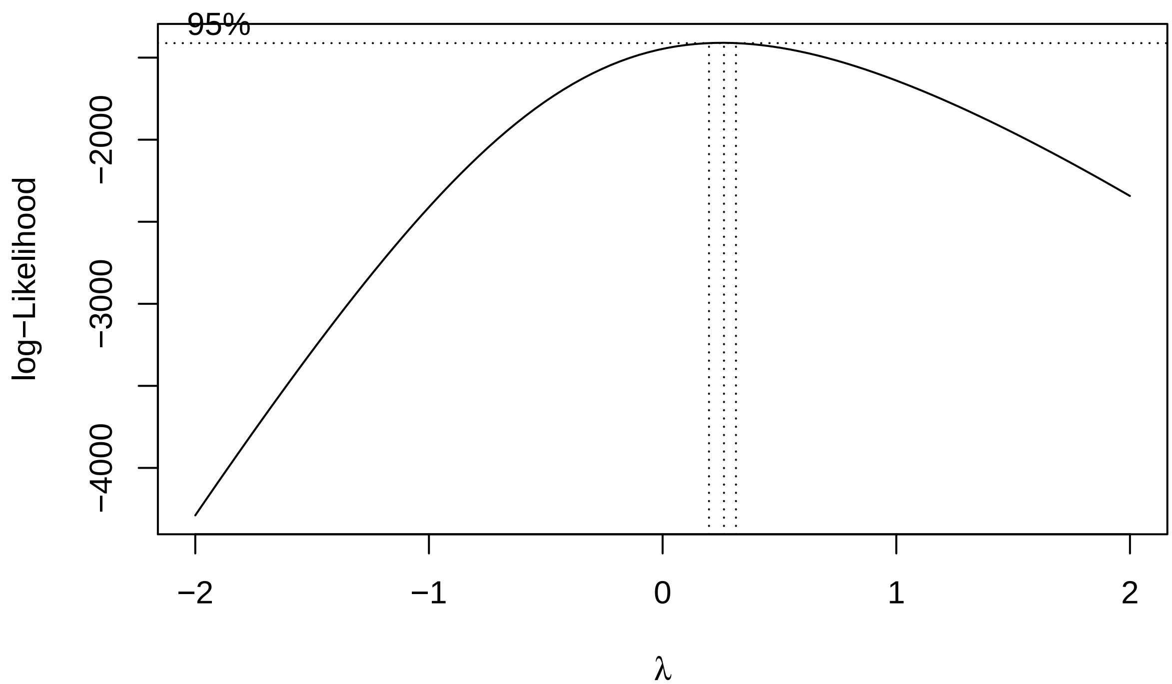
```
##  [1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [15] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
```

I have an idea where we could use a hierarchical model with Neighborhood as the first level, and some of the more quantitative variables on the second.

First step. Get the data in a usable format

```r
# Practical subones
y = ames_train$price
indices = c(2,6,7,18,20,21,22,23,30,31,33,34,37,39,40,41,42,43,44,46,47,49,50:56,58,59,63,64,65,66,78,79
X = ames_train[,indices]

X$Bldg.Type  = as.character(X$Bldg.Type)
X$Exter.Cond = as.character(X$Exter.Cond)
X$Exter.Qual = as.character(X$Exter.Qual)
X$Central.Air = as.character(X$Central.Air)
X$Bsmt.Cond = as.character(X$Bsmt.Cond)
X$Bsmt.Qual = as.character(X$Bsmt.Qual)
```

```r
X$Heating = as.character(X$Heating)
X$Heating.QC = as.character(X$Heating.QC)
X$Kitchen.Qual = as.character(X$Kitchen.Qual)
X$Fireplace.Qu = as.character(X$Fireplace.Qu)
X$Garage.Qual = as.character(X$Garage.Qual)
X$Garage.Cond = as.character(X$Garage.Cond)

## This is the key
X[X == "Ex"] = 5
X[X == "Gd"] = 4
X[X == "TA"] = 3
X[X == "Fa"] = 2
X[X == "Po"] = 1
X$Heating[is.na(X$Heating)] = 0
X$Central.Air[is.na(X$Central.Air)] = 0
X[is.na(X)] = 0

X$Bldg.Type  = as.factor(X$Bldg.Type)
X$Exter.Cond = as.factor(X$Exter.Cond)
X$Exter.Qual = as.factor(X$Exter.Qual)
X$Bsmt.Cond = as.factor(X$Bsmt.Cond)
X$Bsmt.Qual = as.factor(X$Bsmt.Qual)
X$Central.Air = as.factor(X$Central.Air)
X$Heating = as.factor(X$Heating)
X$Heating.QC = as.factor(X$Heating.QC)
X$Kitchen.Qual = as.factor(X$Kitchen.Qual)
X$Fireplace.Qu = as.factor(X$Fireplace.Qu)
X$Garage.Qual = as.factor(X$Garage.Qual)
X$Garage.Cond = as.factor(X$Garage.Cond)

#BOX COX on y
modelling.data = as.data.frame(cbind(y,X))
initial.model = lm(y~.,data = modelling.data)
bc = boxcox(initial.model)
```




```
lambda = bc$x[which.max(bc$y)]
bc.mat = modelling.data %>% mutate(bc.y = (y^lambda-1) / lambda) %>% dplyr::select(-y)

#Area
log.area = log(bc.mat$area)
bc.mat$log.area = log.area
index.area = which(colnames(bc.mat) == "area")
bc.mat = bc.mat[,-index.area]
# X1st.Flr.SF
log.X1st.Flr.SF = log(bc.mat$X1st.Flr.SF)
bc.mat$log.X1st.Flr.SF = log.X1st.Flr.SF
index.X1st = which(colnames(bc.mat) == "X1st.Flr.SF")
bc.mat=bc.mat[,-index.X1st]
# X2nd.Flr.SF

# Total.Bsmt.SF

# Lot Area
log.lot.area = log(bc.mat$Lot.Area)
bc.mat$log.log.area = log.lot.area
index.lot = which(colnames(bc.mat) == "Lot.Area")
bc.mat = bc.mat[,-index.lot]

transformed.data = as.data.frame(bc.mat)
transform.model = lm(bc.y~.,data = transformed.data)
transform.mat = model.matrix(transform.model)
```

```r
#Continuous pre-processing
cont.index = which(sapply(transformed.data,class) == "numeric")
integer.index = which(sapply(transformed.data,class) == "integer")
factor.index = which(sapply(transformed.data,class) == "factor")

scaled.mat = as.data.frame(scale(transformed.data[,cont.index]))
names.cont = names(scaled.mat)
for(i in 1:ncol(scaled.mat)){
  index = which(names(transformed.data) == names.cont[i])
  transformed.data[,index] = scaled.mat[,i]
}
# So the transformations.  I did a boxcox.  Did some logs.  Then I scaled the continuous ones?
y.index = which(names(transformed.data) == "bc.y")
```