Data Cleaning and Preprocessing

Zach Weber

March 27, 2019

Contents

1 Purpose

To prepare our training data for model creation through pruning and feature engineering.

2 Loading in Libraries and Data

First, lets load in some libraries that may be useful for our purposes

```
library(dplyr, quietly=TRUE)
library(ggplot2, quietly=TRUE)
```

Next we can load in our training data

```
# generate path to training data
data.dir <- "datasets/"
train.fpath <- paste0(data.dir,"train.csv")

# load in training data
train.dat <- read.table(train.fpath, sep=",", header=TRUE)
```

3 Exploring Features

Lets take a look at what features we have. We know that our goal is to predict the price of the home which is encoded in the “SalePrice” variable, but what predictors do we have acceess to and what type of variable are they?

```
# get the number of predictors
length(colnames(train.dat))-1 # columns minus SalePrice
```

```
## [1] 80
```

```
# get the colnames of the predictors
featnames <- colnames(train.dat)[which(colnames(train.dat)!="SalePrice")]
featnames
```

```
##  [1] "Id"            "MSSubClass"    "MSZoning"      "LotFrontage"
##  [5] "LotArea"       "Street"        "Alley"         "LotShape"
##  [9] "LandContour"   "Utilities"     "LotConfig"     "LandSlope"
## [13] "Neighborhood"  "Condition1"    "Condition2"    "BldgType"
## [17] "HouseStyle"    "OverallQual"   "OverallCond"   "YearBuilt"
## [21] "YearRemodAdd"  "RoofStyle"     "RoofMatl"      "Exterior1st"
## [25] "Exterior2nd"   "MasVnrType"    "MasVnrArea"    "ExterQual"
## [29] "ExterCond"     "Foundation"    "BsmtQual"      "BsmtCond"
## [33] "BsmtExposure"  "BsmtFinType1"  "BsmtFinSF1"    "BsmtFinType2"
## [37] "BsmtFinSF2"    "BsmtUnfSF"     "TotalBsmtSF"   "Heating"
## [41] "HeatingQC"     "CentralAir"    "Electrical"    "X1stFlrSF"
## [45] "X2ndFlrSF"     "LowQualFinSF"  "GrLivArea"     "BsmtFullBath"
## [49] "BsmtHalfBath"  "FullBath"      "HalfBath"      "BedroomAbvGr"
## [53] "KitchenAbvGr"  "KitchenQual"   "TotRmsAbvGrd"  "Functional"
## [57] "Fireplaces"    "FireplaceQu"   "GarageType"    "GarageYrBlt"
## [61] "GarageFinish"  "GarageCars"    "GarageArea"    "GarageQual"
## [65] "GarageCond"    "PavedDrive"    "WoodDeckSF"    "OpenPorchSF"
## [69] "EnclosedPorch" "X3SsnPorch"    "ScreenPorch"   "PoolArea"
## [73] "PoolQC"        "Fence"         "MiscFeature"   "MiscVal"
## [77] "MoSold"        "YrSold"        "SaleType"      "SaleCondition"
```

It might be helpful to see which predictors are continuous and which predictors are categorical in this list. We can get a rough split by querying the data type of the column

```r
# split features into numeric and categorical (roughly)
numericFeats <- featnames[unlist(lapply(train.dat[,featnames], is.numeric))]
categoricalFeats <- featnames[!featnames %in% numericFeats]

# display numeric features
numericFeats
```

```
##  [1] "Id"            "MSSubClass"    "LotFrontage"   "LotArea"
##  [5] "OverallQual"   "OverallCond"   "YearBuilt"     "YearRemodAdd"
##  [9] "MasVnrArea"    "BsmtFinSF1"    "BsmtFinSF2"    "BsmtUnfSF"
## [13] "TotalBsmtSF"   "X1stFlrSF"     "X2ndFlrSF"     "LowQualFinSF"
## [17] "GrLivArea"     "BsmtFullBath"  "BsmtHalfBath"  "FullBath"
## [21] "HalfBath"      "BedroomAbvGr"  "KitchenAbvGr"  "TotRmsAbvGrd"
## [25] "Fireplaces"    "GarageYrBlt"   "GarageCars"    "GarageArea"
## [29] "WoodDeckSF"    "OpenPorchSF"   "EnclosedPorch" "X3SsnPorch"
## [33] "ScreenPorch"   "PoolArea"      "MiscVal"       "MoSold"
## [37] "YrSold"
```

```r
# display categorical features
categoricalFeats
```

```
##  [1] "MSZoning"      "Street"        "Alley"         "LotShape"
##  [5] "LandContour"   "Utilities"     "LotConfig"     "LandSlope"
##  [9] "Neighborhood"  "Condition1"    "Condition2"    "BldgType"
## [13] "HouseStyle"    "RoofStyle"     "RoofMatl"      "Exterior1st"
## [17] "Exterior2nd"   "MasVnrType"    "ExterQual"     "ExterCond"
## [21] "Foundation"    "BsmtQual"      "BsmtCond"      "BsmtExposure"
## [25] "BsmtFinType1"  "BsmtFinType2"  "Heating"       "HeatingQC"
## [29] "CentralAir"    "Electrical"    "KitchenQual"   "Functional"
## [33] "FireplaceQu"   "GarageType"    "GarageFinish"  "GarageQual"
## [37] "GarageCond"    "PavedDrive"    "PoolQC"        "Fence"
## [41] "MiscFeature"   "SaleType"      "SaleCondition"
```

3.1 Pruning Predictors

We also should be interested in which features have missing data in both the numeric and categorical type. Because missing data imputation is not totally within the scope of this class we would probably remove all sets of predictors with missing data. We can start by generating filters for columns with missing data.

```
# generate filters for missing data
num.NA.filt <- unlist(lapply(train.dat[,numericFeats],
                             function(x) any(is.na(x))))
cat.NA.filt <- unlist(lapply(train.dat[,categoricalFeats],
                             function(x) any(is.na(x))))
```

Lets summarize the findings when we apply the filter (print statements not shown)

```
## Number of Numeric Features: 37
```

```
## Number of Numeric Features w/o NAs: 34
```

```
## Features with NAs:  LotFrontage MasVnrArea GarageYrBlt
```

```
## Number of Categorical Features: 37
```

```
## number of Categorical Features w/o NAs: 27
```

```
## Features with NAs
## :
```

```
##  [1] "Alley"        "MasVnrType"   "BsmtQual"     "BsmtCond"
##  [5] "BsmtExposure" "BsmtFinType1" "BsmtFinType2" "Electrical"
##  [9] "FireplaceQu"  "GarageType"   "GarageFinish" "GarageQual"
## [13] "GarageCond"   "PoolQC"       "Fence"        "MiscFeature"
```

Now that we've identified those features with missing data, we can remove them from the data frame to produce a new pruned dataset. We should have 34 + 27 = 61 features remaining on which to predict.

```
# get the column names to be removed
numeric.to.remove <- numericFeats[num.NA.filt]
categor.to.remove <- numericFeats[cat.NA.filt]
pruned.data <- train.dat[,!colnames(train.dat) %in% c(numeric.to.remove,categor.to.remove)]
dim(pruned.data)
```

```
## [1] 1460   67
```