Housing Prices report

Yoan Bidart

2/25/2018

Introduction

Here is a report for solving the House Prices prediction competition on Kaggle. Feel free to fork and comment if you want to contribute!

Exploratory analysis

```
library(caret)
library(dplyr)
library(ModelMetrics)
library(ggplot2)
```

Here is our dataset.

```
training <- read.csv(file="train.csv", stringsAsFactors = FALSE)
str(training)
```

```
## 'data.frame':    1460 obs. of  81 variables:
##  $ Id           : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ MSSubClass   : int  60 20 60 70 60 50 20 60 50 190 ...
##  $ MSZoning     : chr  "RL" "RL" "RL" "RL" ...
##  $ LotFrontage  : int  65 80 68 60 84 85 75 NA 51 50 ...
##  $ LotArea      : int  8450 9600 11250 9550 14260 14115 10084 10382 6120 7420 ...
##  $ Street       : chr  "Pave" "Pave" "Pave" "Pave" ...
##  $ Alley        : chr  NA NA NA NA ...
##  $ LotShape     : chr  "Reg" "Reg" "IR1" "IR1" ...
##  $ LandContour  : chr  "Lvl" "Lvl" "Lvl" "Lvl" ...
##  $ Utilities    : chr  "AllPub" "AllPub" "AllPub" "AllPub" ...
##  $ LotConfig    : chr  "Inside" "FR2" "Inside" "Corner" ...
##  $ LandSlope    : chr  "Gtl" "Gtl" "Gtl" "Gtl" ...
##  $ Neighborhood : chr  "CollgCr" "Veenker" "CollgCr" "Crawfor" ...
##  $ Condition1   : chr  "Norm" "Feedr" "Norm" "Norm" ...
##  $ Condition2   : chr  "Norm" "Norm" "Norm" "Norm" ...
##  $ BldgType     : chr  "1Fam" "1Fam" "1Fam" "1Fam" ...
##  $ HouseStyle   : chr  "2Story" "1Story" "2Story" "2Story" ...
##  $ OverallQual  : int  7 6 7 7 8 5 8 7 7 5 ...
##  $ OverallCond  : int  5 8 5 5 5 5 5 6 5 6 ...
##  $ YearBuilt    : int  2003 1976 2001 1915 2000 1993 2004 1973 1931 1939 ...
##  $ YearRemodAdd : int  2003 1976 2002 1970 2000 1995 2005 1973 1950 1950 ...
##  $ RoofStyle    : chr  "Gable" "Gable" "Gable" "Gable" ...
##  $ RoofMatl     : chr  "CompShg" "CompShg" "CompShg" "CompShg" ...
##  $ Exterior1st  : chr  "VinylSd" "MetalSd" "VinylSd" "Wd Sdng" ...
##  $ Exterior2nd  : chr  "VinylSd" "MetalSd" "VinylSd" "Wd Shng" ...
##  $ MasVnrType   : chr  "BrkFace" "None" "BrkFace" "None" ...
##  $ MasVnrArea   : int  196 0 162 0 350 0 186 240 0 0 ...
##  $ ExterQual    : chr  "Gd" "TA" "Gd" "TA" ...
```

```
##  $ ExterCond    : chr  "TA" "TA" "TA" "TA" ...
##  $ Foundation   : chr  "PConc" "CBlock" "PConc" "BrkTil" ...
##  $ BsmtQual     : chr  "Gd" "Gd" "Gd" "TA" ...
##  $ BsmtCond     : chr  "TA" "TA" "TA" "Gd" ...
##  $ BsmtExposure : chr  "No" "Gd" "Mn" "No" ...
##  $ BsmtFinType1 : chr  "GLQ" "ALQ" "GLQ" "ALQ" ...
##  $ BsmtFinSF1   : int  706 978 486 216 655 732 1369 859 0 851 ...
##  $ BsmtFinType2 : chr  "Unf" "Unf" "Unf" "Unf" ...
##  $ BsmtFinSF2   : int  0 0 0 0 0 0 0 32 0 0 ...
##  $ BsmtUnfSF    : int  150 284 434 540 490 64 317 216 952 140 ...
##  $ TotalBsmtSF  : int  856 1262 920 756 1145 796 1686 1107 952 991 ...
##  $ Heating      : chr  "GasA" "GasA" "GasA" "GasA" ...
##  $ HeatingQC    : chr  "Ex" "Ex" "Ex" "Gd" ...
##  $ CentralAir   : chr  "Y" "Y" "Y" "Y" ...
##  $ Electrical   : chr  "SBrkr" "SBrkr" "SBrkr" "SBrkr" ...
##  $ X1stFlrSF    : int  856 1262 920 961 1145 796 1694 1107 1022 1077 ...
##  $ X2ndFlrSF    : int  854 0 866 756 1053 566 0 983 752 0 ...
##  $ LowQualFinSF : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ GrLivArea    : int  1710 1262 1786 1717 2198 1362 1694 2090 1774 1077 ...
##  $ BsmtFullBath : int  1 0 1 1 1 1 1 1 0 1 ...
##  $ BsmtHalfBath : int  0 1 0 0 0 0 0 0 0 0 ...
##  $ FullBath     : int  2 2 2 1 2 1 2 2 2 1 ...
##  $ HalfBath     : int  1 0 1 0 1 1 0 1 0 0 ...
##  $ BedroomAbvGr : int  3 3 3 3 4 1 3 3 2 2 ...
##  $ KitchenAbvGr : int  1 1 1 1 1 1 1 1 2 2 ...
##  $ KitchenQual  : chr  "Gd" "TA" "Gd" "Gd" ...
##  $ TotRmsAbvGrd : int  8 6 6 7 9 5 7 7 8 5 ...
##  $ Functional   : chr  "Typ" "Typ" "Typ" "Typ" ...
##  $ Fireplaces   : int  0 1 1 1 1 0 1 2 2 2 ...
##  $ FireplaceQu  : chr  NA "TA" "TA" "Gd" ...
##  $ GarageType   : chr  "Attchd" "Attchd" "Attchd" "Detchd" ...
##  $ GarageYrBlt  : int  2003 1976 2001 1998 2000 1993 2004 1973 1931 1939 ...
##  $ GarageFinish : chr  "RFn" "RFn" "RFn" "Unf" ...
##  $ GarageCars   : int  2 2 2 3 3 2 2 2 2 1 ...
##  $ GarageArea   : int  548 460 608 642 836 480 636 484 468 205 ...
##  $ GarageQual   : chr  "TA" "TA" "TA" "TA" ...
##  $ GarageCond   : chr  "TA" "TA" "TA" "TA" ...
##  $ PavedDrive   : chr  "Y" "Y" "Y" "Y" ...
##  $ WoodDeckSF   : int  0 298 0 0 192 40 255 235 90 0 ...
##  $ OpenPorchSF  : int  61 0 42 35 84 30 57 204 0 4 ...
##  $ EnclosedPorch: int  0 0 0 272 0 0 0 228 205 0 ...
##  $ X3SsnPorch   : int  0 0 0 0 0 320 0 0 0 0 ...
##  $ ScreenPorch  : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ PoolArea     : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ PoolQC       : chr  NA NA NA NA ...
##  $ Fence        : chr  NA NA NA NA ...
##  $ MiscFeature  : chr  NA NA NA NA ...
##  $ MiscVal      : int  0 0 0 0 0 700 0 350 0 0 ...
##  $ MoSold       : int  2 5 9 2 12 10 8 11 4 1 ...
##  $ YrSold       : int  2008 2007 2008 2006 2008 2009 2007 2009 2008 2008 ...
##  $ SaleType     : chr  "WD" "WD" "WD" "WD" ...
##  $ SaleCondition: chr  "Normal" "Normal" "Normal" "Abnorml" ...
##  $ SalePrice    : int  208500 181500 223500 140000 250000 143000 307000 200000 129900 118000 ...
```

We can see we have a lot of features, and some work to do in feature engineering, which is the heart of this challenge.

Feature engineering

We have some variables with a lot of NAs we will get rid of. Some others that we will replace according to the meaning of these variables, accordingly to the data description.

```
cleanNa <- function(x) {
        #replace LotFrontage with the median value
        x$LotFrontage[is.na(x$LotFrontage)] = median(training$LotFrontage, na.rm=TRUE)
        #remove Alley
        x <- select(x, -Alley)
        #For some variables Na is actually meaning none
        x$MasVnrType[is.na(x$MasVnrType)] = as.factor("None")
        x$MasVnrArea[is.na(x$MasVnrArea)] = 0
        x$BsmtQual[is.na(x$BsmtQual)] = "None"
        x$BsmtCond[is.na(x$BsmtCond)] = "None"
        x$BsmtExposure[is.na(x$BsmtExposure)] = "None"
        x$BsmtFinType1[is.na(x$BsmtFinType1)] = "None"
        x$BsmtFinType2[is.na(x$BsmtFinType2)] = "None"
        #replace electrical by the most represented type
        x$Electrical[is.na(x$Electrical)] = "SBrkr"
        x$FireplaceQu[is.na(x$FireplaceQu)] = "None"
        x$GarageType[is.na(x$GarageType)] = "None"
        x$GarageYrBlt = as.numeric(x$GarageYrBlt)
        x$GarageYrBlt[is.na(x$GarageYrBlt)] = 0
        x$GarageFinish[is.na(x$GarageFinish)] = "None"
        x$GarageQual[is.na(x$GarageQual)] = "None"
        x$GarageCond[is.na(x$GarageCond)] = "None"
        x$PoolQC[is.na(x$PoolQC)] = "None"
        x$Fence[is.na(x$Fence)] = "None"
        x$MiscFeature[is.na(x$MiscFeature)] = "None"
        x$GarageYrBlt = as.numeric(x$GarageYrBlt)
        x
}
trainC <- cleanNa(training)
#replace caracters variables by factors
for (i in c(1:80)) {
        x <- is.character(trainC[,i])
        if (x==TRUE) {
                trainC[,i] <- as.factor(trainC[,i])
        }
}
```

We are done with feature engineering.

Machine Learning

Create train and test set

```
set.seed(7)
inTrain <- createDataPartition(trainC$Id, p=0.7, list=FALSE)
train1 <- trainC[inTrain, ]
test1 <- trainC[-inTrain, ]
out1 <- test1$SalePrice
test1 <- select(test1, -SalePrice)
```

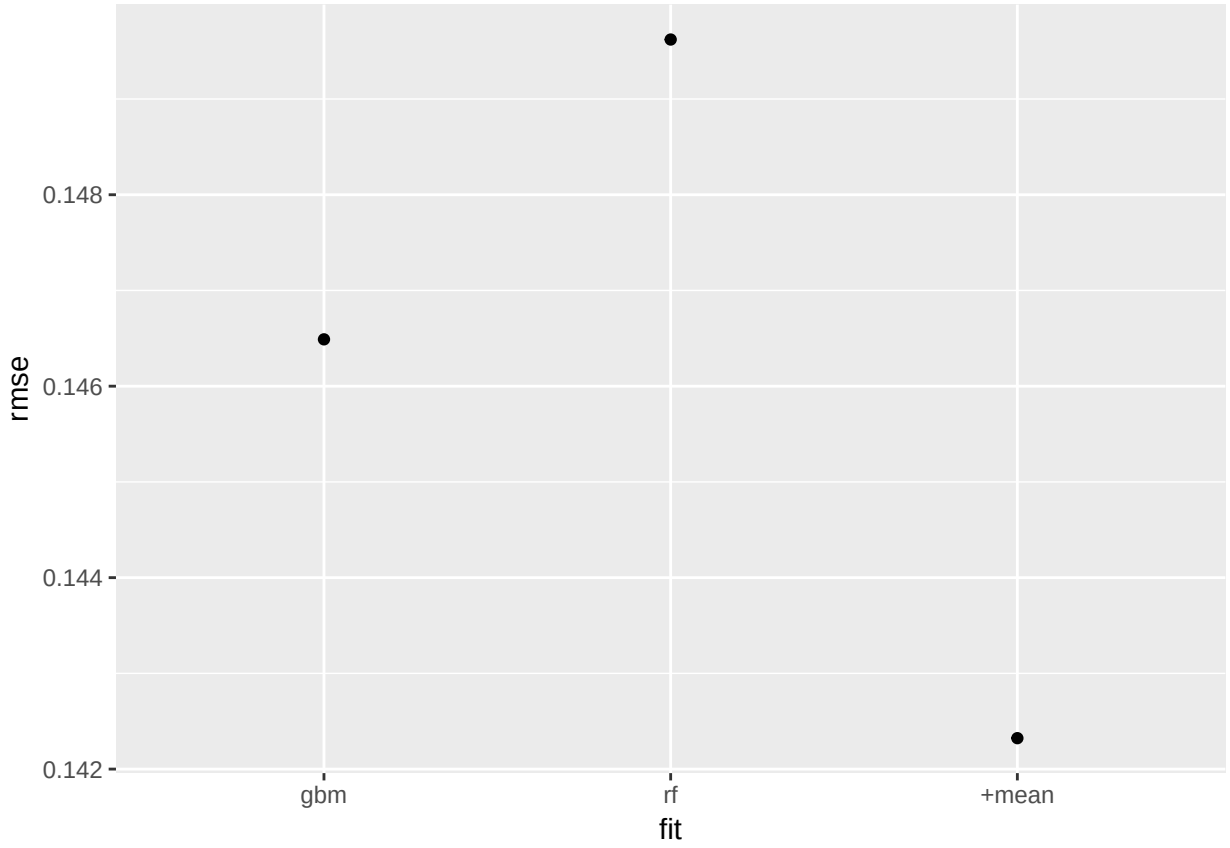
Model fitting

After fitting some models we found the best accuracy to combine Gradient Boosting Machine (gbm) and Random Forest (rf).

```
fit1 <- train(SalePrice~., data=train1, method="gbm", verbose=FALSE)
fit2 <- train(SalePrice~., data=train1, method="rf")
```

Predicting and evaluating model

```
pred1 <- predict(fit1, test1)
accu <- data.frame(fit = "gbm", rmse = rmse(log(out1), log(pred1)))
pred2 <- predict(fit2, test1)
accu <- rbind(accu, data.frame(fit="rf", rmse=rmse(log(out1), log(pred2))))
#compute the mean of the two predictions
input3 <- as.data.frame(cbind(pred1, pred2))
pred3 <- apply(input3, 1, mean )
accu <- rbind(accu, data.frame(fit="+mean", rmse=rmse(log(out1), log(pred3))))
qplot(fit, rmse, data=accu)
```



We can see that our Root Mean Squared Error is lower when combining the two models.

Creating output for the competition

```
testing <- read.csv(file="test.csv", stringsAsFactors = FALSE)
#preprocessing
testC <- cleanNa(testing)
for (i in c(1:79)) {
        x <- is.character(testC[,i])
        if (x==TRUE) {
                testC[,i] <- as.factor(testC[,i])
        }
}
```

Some NA values where present in the test set provided, we created a function to replace them with median or most represented values.

```
testNa <- function(test1) {
        i <- c(456, 757, 791)
        test1[i,3] = "RM"
        test1[1445, 3] = "RL"

        test1$Utilities[is.na(test1$Utilities)] <- "AllPub"
        test1$Exterior1st[is.na(test1$Exterior1st)] <- "VinylSd"
        test1$Exterior2nd[is.na(test1$Exterior2nd)] <- "VinylSd"
        test1$BsmtFinSF1[is.na(test1$BsmtFinSF1)] <- 0
```

```
        test1$BsmtFinSF2[is.na(test1$BsmtFinSF2)] <- 0
        test1$BsmtUnfSF[is.na(test1$BsmtUnfSF)] <- 0
        test1$TotalBsmtSF[is.na(test1$TotalBsmtSF)] <- 0
        test1$BsmtHalfBath[is.na(test1$BsmtHalfBath)] <- 0
        test1$KitchenQual[is.na(test1$KitchenQual)] <- "TA"
        test1$Functional[is.na(test1$Functional)] <- "Typ"
        test1$GarageCars[is.na(test1$GarageCars)] <- median(test1$GarageCars, na.rm=TRUE)
        test1$GarageArea[is.na(test1$GarageArea)] <- median(test1$GarageArea, na.rm=TRUE)
        test1$SaleType[is.na(test1$SaleType)] <- "WD"
        test1$BsmtFullBath[is.na(test1$BsmtFullBath)] <- 0

        test1
}
test1 <- testNa(testC)
```

Prediction time !

```
pred1 <- predict(fit1, test1)
pred2 <- predict(fit2, test1)
input3 <- as.data.frame(cbind(pred1, pred2))
pred4 <- apply(input3, 1, mean )
submission <- cbind.data.frame(test1$Id, pred4)
names(submission) <- c("Id", "SalePrice")
write.csv(submission, file="sub.csv", row.names=FALSE)
```

That's it !! As you can see this competition was really about data cleaning and feature engineering.