Housing data analysis - Women Who Code workshop

Yournamehere

13/03/2018

Basic preamble - live coding in parallel with Darya:

- variables
- data types

The link about assignment operators

Task 1

Load the data in from csv.

```
trainH <- read.csv("train.csv")
testH <- read.csv("test.csv")
```

Task 2

1. What features are there in the data?
2. What are the dimensions of the data?
3. What are the column headers?

Use the summary() and str() functions to explore…

What does the distribution of sale price look like?

Task 3

1. Is the sale price (the variable we're interested in prediting) normally distributed?
2. Plot a histogram of the distribution using ggplot2.
3. Find its mean, standard deviation

Task 4

1.Plot a quantile-quantile plot (QQ plot) to "assess" normality.

```
Note: This plot compares the data we have (Sample Quantiles) with a theoretical sample coming from a no
```

A standard way of transforming the data to be better approximated by a normal distribution is by using the log-transform?

Task 5

1. Carry out this transformation
2. Use a histogram and QQ plot to see whether it works…

Missing data

Task 6

What happens if we only use complete data? How much data is missing?

Topics used here (but not explored): Subsetting data frames The apply family

We need to combine the datasets for imputation, so that we don't have NAs in the test data as well!

Task 7

Combine the testing and training data.

How do we impute the missing data?

Task 8

Explore the data using the table() function (variable by variable).

Read the metadata file and see that many of the NAs should be recoded as None since these features are lacking in the house.

Task 9

Recode the NA values that should be None using mutate() and fct_explicit_na().

Task 10

For the GarageYrBlt - set NA values using replace_na() to zero.

Task 11

For Lot frontage - set it to be the median for the neighborhood using group_by() and mutate().

```
# as a hint - use group_by() and mutate()
# will also need ifelse() function
```

Now split data again

Task 12

Split back into training (trainHC) and test (testHC) sets (because kaggle training set had prices, test didn't).

```
# no comment
```

Basic exploratory data analysis of training data

Task 13

1. How does the sale price depend on living area: X1stFlrSF, X2ndFlrSF, TotalBsmtSF? (use a scatterplot to visualise this)
2. Create a variable TotalSqFt which is a combination of these

3. Does it better predict the house price? (again, just using scatterplot at this point)

Task 14

Identify and remove outliers with a high total square foot, but low price.

Useful reference for dplyr

Does having more bedrooms increase sale price?

Task 15

Use a geom_boxplot() to explore this

Task 16

Visualise both number of bedrooms (as a factor) and TotalSqFt as a scatterplot to see if a trend is visible.

Are newer or more recently renovated properties more expensive?

Task 17

1. Investigate this generally and then
2. … specifically for 2 - 4 bedroom properties.

Lets convert kitchen quality to numeric (we'll see why we need this later):

From the metadata we know it can be:

- Ex Excellent
- Gd Good
- TA Typical/Average
- Fa Fair
- Po Poor

Task 18

Recode this to numeric values using mutate() and recode().

Task 19

Convert Bldgtype to numeric

What variables are correlated with each other and with price?

Task 20

1. Plot a correlation plot using corrplot() for all numeric variables and
2. … those that show the top correlation with LogSalePrice.

Task 21

Use the createDataPartition() function to separate the training data into a training and testing subset. Allocate 50% of the data to each class. Run set.seed(12) before this.

```r
set.seed(12)
```

Task 22

Fit a linear model considering the "top 10"" correlated (top 9, ignore LogSalePrice for obvious reasons). Code the variables (column names) manually.

Task 23

1. Use predict() to predict house prices using our top10 model on the "test" portion of the training dataset.
2. Use rmse to assess the root mean square error (our metric of accuracy).

Task 24

1. Use randomForest() to train a random forest model on all of the variables.
2. Use predict() and rmse() to make the prediction and assess the accuracy respectively.
3. Was a linear (on 9 features) or random forest model more accurate?

```
# randFor <- randomForest(LogSalePrice ~ ., data=trainHCtrain)
# # Predict using the test set
# prediction_rf <- predict(randFor, trainHCtest)
# trainHCtest$randFor <- prediction_rf
# # rmse?
# rmse(trainHCtest$LogSalePrice, trainHCtest$randFor)
```

Task 25

1. Use xgboost to predict house prices from numeric features of training dataset.
2. Use xgb.plot.importance() to assess which variables are most important for predicting house prices.

```
# trainHCtrainNum <- as(as.matrix(trainHCtrain[ , sapply(trainHCtrain, is.numeric)]), "sparseMatrix")
# trainHCtestNum <-  as(as.matrix(trainHCtest[ , sapply(trainHCtest, is.numeric)]), "sparseMatrix")
#
# trainD <- xgb.DMatrix(data = trainHCtrainNum, label = trainHCtrainNum[,"LogSalePrice"])
#
# #Cross validate the model
# cv.sparse <- xgb.cv(data = trainD,
#                      nrounds = 600,
#                      min_child_weight = 0,
#                      max_depth = 10,
#                      eta = 0.02,
#                      subsample = .7,
#                      colsample_bytree = .7,
#                      booster = "gbtree",
#                      eval_metric = "rmse",
#                      verbose = TRUE,
#                      print_every_n = 50,
#                      nfold = 4,
#                      nthread = 2,
#                      objective="reg:linear")
#
# #Train the model
# #Choose the parameters for the model
# param <- list(colsample_bytree = .7,
#               subsample = .7,
#               booster = "gbtree",
#               max_depth = 10,
#               eta = 0.02,
#               eval_metric = "rmse",
#               objective="reg:linear")
#
#
# #Train the model using those parameters
# bstSparse <-
#   xgb.train(params = param,
#              data = trainD,
#              nrounds = 600,
#              watchlist = list(train = trainD),
#              verbose = TRUE,
#              print_every_n = 50,
#              nthread = 2)
#
# testD <- xgb.DMatrix(data = trainHCtestNum)
#
# prediction <- predict(bstSparse, testD) #Make the prediction based on the half of the training data s
#
# #Put testing prediction and test dataset all together
#
# prediction <- as.data.frame(as.matrix(prediction))
# colnames(prediction) <- "xgboost"
# trainHCtest$xgboost <- prediction$xgboost
#
#
# #Test with RMSE
# rmse(trainHCtest$LogSalePrice, trainHCtest$xgboost)
#
# # Feature importance
# importance_matrix <- xgb.importance(dimnames(trainD)[[2]], model = bstSparse)
# xgb.plot.importance(importance_matrix[1:10])
```

Task 26

1.Use the glmnet library to train a ridge regression model. 2. Is it more or less accurate than XGBoost?

```
# trainHCtrainNumMatrix <- as.matrix(trainHCtrain[ , sapply(trainHCtrain, is.numeric)])
# trainHCtestNumMatrix  <-  as.matrix(trainHCtest[ , sapply(trainHCtest, is.numeric)])
# # cross validation for glmnet
# glm.cv.ridge <- cv.glmnet(trainHCtrainNum[,c(1:38,40)], trainHCtrainNum[,"LogSalePrice"], alpha = 0)
# penalty.ridge <- glm.cv.ridge$lambda.min
# glm.ridge <- glmnet(x = trainHCtrainNum[,c(1:38,40)], y = trainHCtrainNum[,"LogSalePrice"], alpha = 0
# y_pred.ridge <- as.numeric(predict(glm.ridge, trainHCtestNum[,c(1:38,40)] ))
# rmse(trainHCtest$LogSalePrice, y_pred.ridge)
```