Using recipes in tidymodels

Jaime Davila

3/1/2022

notes: we did prediction and classification x1, x2, x… –> predict y (y is continuous) one of the most fundamental ways to do prediction is through a linear model. equation is linear combination (ax + a1x + …) the best line minimizes error (misclassification rate) "residual error" = RSS = sum of (predicted - actual) note: RSS = n*MSE TSS = sum of(predicted - actual_mean)ˆ2 = (*nvariance) rˆ2 = (TSS - RSS) / TSS

lm() knn3() knnreg() I think this is what these 3 are called. see how to use these 3.

Using recipes in `tidymodels`

In today's class we will explore how to create better models by creating new variables from old ones. This process is sometimes called *feature engineering* and we will learn how to do using `recipes` from `tidymodels`. In this worksheet we will be covering most of the material from https://www.tmwr.org/recipes.html

Using the ames dataset

Let's start by loading the appropriate libraries, datasets, and converting our variable `price` so that is in on log-scale. Also let's make sure that we create our testing and training dataset

```
library(tidymodels)
tidymodels_prefer()

library(modeldata)
data(ames)
ames <- ames %>%
  mutate(Sale_Price=log10(Sale_Price))

set.seed(12345)
ames.split <- initial_split(ames, prop=0.8)
ames.train <- training(ames.split)
ames.test <- testing(ames.split)
```

Initial modeling

We are interested in predicting the price of the property adding the following variables:

- `Neighborhood`
- `Gr_Liv_Area`, corresponding to the gross above-grade living area.
- `Year_built`
- `Bldg_type` corresponding to the building type

1. What is the type of each of these four variables? If a variable is categorical how many different values (levels) it has

```
## [1] "factor"
```

```
## [1] "integer"
```

```
## [1] "integer"
```

```
## [1] "factor"
```

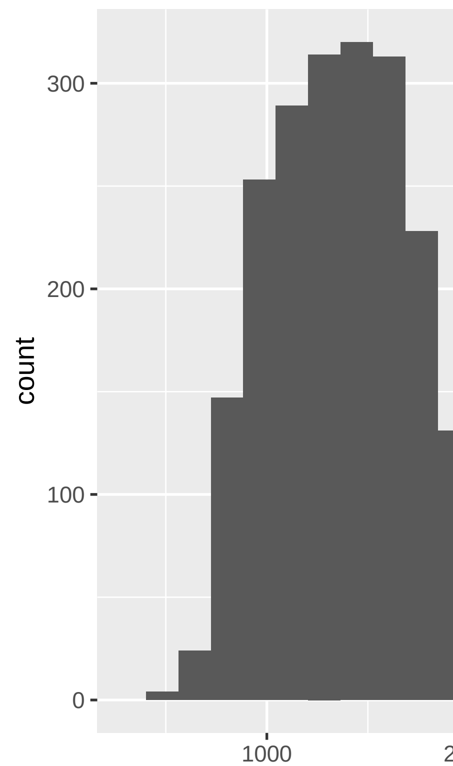
```
##  [1] "North_Ames"
##  [2] "College_Creek"
##  [3] "Old_Town"
##  [4] "Edwards"
##  [5] "Somerset"
##  [6] "Northridge_Heights"
##  [7] "Gilbert"
##  [8] "Sawyer"
##  [9] "Northwest_Ames"
## [10] "Sawyer_West"
## [11] "Mitchell"
## [12] "Brookside"
## [13] "Crawford"
## [14] "Iowa_DOT_and_Rail_Road"
## [15] "Timberland"
## [16] "Northridge"
## [17] "Stone_Brook"
## [18] "South_and_West_of_Iowa_State_University"
## [19] "Clear_Creek"
## [20] "Meadow_Village"
## [21] "Briardale"
## [22] "Bloomington_Heights"
## [23] "Veenker"
## [24] "Northpark_Villa"
## [25] "Blueste"
## [26] "Greens"
## [27] "Green_Hills"
## [28] "Landmark"
## [29] "Hayden_Lake"
```

```
## [1] "OneFam"   "TwoFmCon" "Duplex"   "Twnhs"    "TwnhsE"
```

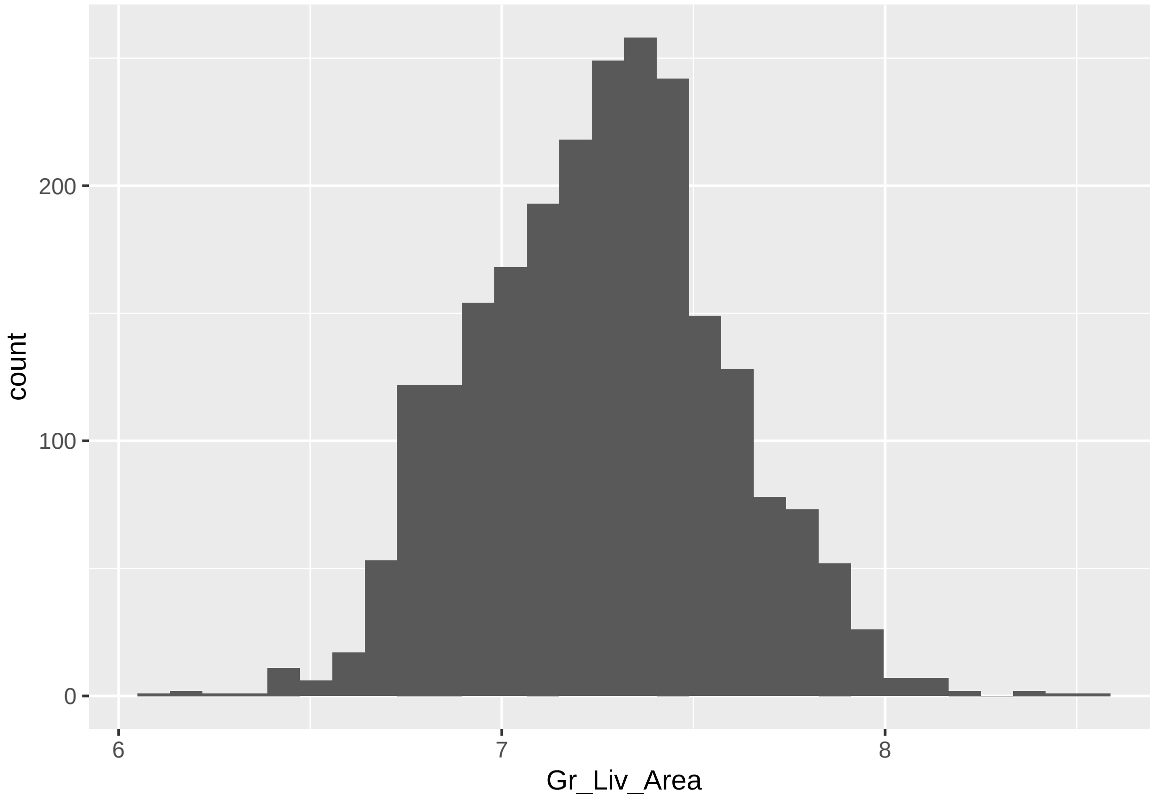
```
##  [1] Edwards
##  [2] Northridge_Heights
##  [3] Old_Town
##  [4] Brookside
##  [5] North_Ames
##  [6] Sawyer
##  [7] College_Creek
##  [8] Timberland
##  [9] Somerset
## [10] Sawyer_West
## [11] Gilbert
## [12] Crawford
## [13] Blueste
## [14] Mitchell
```

```
## [15] Northwest_Ames
## [16] South_and_West_of_Iowa_State_University
## [17] Northpark_Villa
## [18] Northridge
## [19] Greens
## [20] Clear_Creek
## [21] Meadow_Village
## [22] Stone_Brook
## [23] Iowa_DOT_and_Rail_Road
## [24] Landmark
## [25] Veenker
## [26] Briardale
## [27] Bloomington_Heights
## [28] Green_Hills
## 29 Levels: North_Ames College_Creek Old_Town Edwards ... Hayden_Lake
```

factor int int factor typeof() just calls them all ints. (levels are stored as integers)



2. Do a histogram of `Gr_Liv_Area`. How does this histogram looks using a log scale?



Log scale histogram looks more normally distributed

Creating a recipe

A recipe is a collection of steps for preprocessing a dataset. Our initial recipe will include the following steps:

- We would like to make it explicit that we are modeling the `Sale_Price` (response variable) based on `Latitude` and `Longitude`, `Gr_Liv_area`, and `Bldg_type` (explanatory variables)
- We would like to use a log scale for `Gr_Liv_Area`
- We would like to transform all of our categorical variables into indicator variables.

```
ames.recipe <-
  recipe(Sale_Price ~ Longitude + Latitude + Gr_Liv_Area +
           Bldg_Type, data=ames.train) %>%
  step_log(Gr_Liv_Area, base=10) %>%
  step_dummy(all_nominal_predictors())#turn all nominal variables into factors?
ames.recipe
```

```
## Recipe
##
## Inputs:
##
##       role #variables
##    outcome          1
##  predictor          4
##
## Operations:
##
```

```
## Log transformation on Gr_Liv_Area
## Dummy variables from all_nominal_predictors()
```

Once we created the recipe we can use in conjunction with a linear model, add it to a workflow and fit our workflow using our training dataset

```
lm.model <- linear_reg() %>%
  set_engine("lm")

lm.wflow <- workflow() %>%
  add_recipe(ames.recipe) %>%
  add_model(lm.model)

lm.fit <- fit(lm.wflow, ames.train)
lm.fit
```

```
## == Workflow [trained] ==========================================================
## Preprocessor: Recipe
## Model: linear_reg()
##
## -- Preprocessor ----------------------------------------------------------------
## 2 Recipe Steps
##
## * step_log()
## * step_dummy()
##
## -- Model -----------------------------------------------------------------------
##
## Call:
## stats::lm(formula = ..y ~ ., data = data)
##
## Coefficients:
##       (Intercept)          Longitude           Latitude         Gr_Liv_Area
##        -169.33768           -1.22032            1.37084             0.84592
## Bldg_Type_TwoFmCon    Bldg_Type_Duplex    Bldg_Type_Twnhs    Bldg_Type_TwnhsE
##          -0.13283           -0.11533           -0.04138             0.05977
```

3.

a. What is the R^2 of the linear model you created? How do you interpret this value?

```
## # A tibble: 1 x 12
##   r.squared adj.r.squared sigma statistic p.value    df logLik    AIC    BIC
##       <dbl>         <dbl> <dbl>     <dbl>   <dbl> <dbl>  <dbl>  <dbl>  <dbl>
## 1     0.610         0.608 0.110      521.       0     7  1848. -3678. -3626.
## # ... with 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

r^2 = 0.610 R^2 = $(n * variance - sum(predicted - actual))/n * variance$ r^2 represents how well the model works compared with actual data. 1 is a perfect model. "This model accounts for 61% of the variance of this graph"

b. Interpret the coefficients corresponding to:

```
## # A tibble: 8 x 5
##   term                 estimate std.error statistic  p.value
##   <chr>                   <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)          -169.      10.3        -16.4  3.11e-57
## 2 Longitude              -1.22     0.0899     -13.6  2.01e-40
```

```
## 3 Latitude              1.37     0.128       10.7  5.32e-26
## 4 Gr_Liv_Area           0.846    0.0171      49.6  0
## 5 Bldg_Type_TwoFmCon   -0.133    0.0158      -8.39 8.47e-17
## 6 Bldg_Type_Duplex     -0.115    0.0123      -9.35 2.00e-20
## 7 Bldg_Type_Twnhs      -0.0414   0.0123      -3.36 7.85e- 4
## 8 Bldg_Type_TwnhsE      0.0598   0.00845      7.07 2.04e-12
```

- The living area of the house.

for every increase by log-scale in 1, there is increase in log-scale of .846 in log scale for price

- The type of building.

  log(price) decreases for 3 of building types by a log constant for each property (-.133, -.115, or -.0414) and increases for one: TwnhsE by 0.0598

4. Evaluate your model using the testing dataset. What is the MSE on this dataset?

```
## # A tibble: 586 x 1
##    .pred
##    <dbl>
##  1  5.33
##  2  5.13
##  3  5.18
##  4  5.39
##  5  5.31
##  6  5.41
##  7  5.24
##  8  5.24
##  9  5.26
## 10  5.43
## # ... with 576 more rows
```

```
## [1] 0.0112031
```

[1] 0.0112031

prof: new.ames.test <- lm.fit %>% augment(new_data = ames.test) rmse_vec(new.ames.test$Sale_Price, new.ames.test$pred)^2 0.112031

5. Add `Year_Built` as an input variable in your existing recipe. What is the R^2 of your model? What is the MSE on the testing dataset?

```
ames.recipe.no5 <-
  recipe(Sale_Price ~ Longitude + Latitude + Gr_Liv_Area +
           Bldg_Type + Year_Built, data=ames.train) %>%
  step_log(Gr_Liv_Area, base=10) %>%
  step_dummy(all_nominal_predictors())#turn all nominal variables into factors?
ames.recipe.no5
```

```
## Recipe
##
## Inputs:
##
##       role #variables
##    outcome          1
##  predictor          5
##
## Operations:
##
```

```
## Log transformation on Gr_Liv_Area
## Dummy variables from all_nominal_predictors()
```

```
lm.model.no5 <- linear_reg() %>%
  set_engine("lm")

lm.wflow.no5 <- workflow() %>%
  add_recipe(ames.recipe.no5) %>%
  add_model(lm.model.no5)

lm.fit.no5 <- fit(lm.wflow.no5, ames.train)
lm.fit.no5
```

```
## == Workflow [trained] ==========================================================
## Preprocessor: Recipe
## Model: linear_reg()
##
## -- Preprocessor ----------------------------------------------------------------
## 2 Recipe Steps
##
## * step_log()
## * step_dummy()
##
## -- Model -----------------------------------------------------------------------
##
## Call:
## stats::lm(formula = ..y ~ ., data = data)
##
## Coefficients:
##        (Intercept)           Longitude            Latitude         Gr_Liv_Area
##         -19.597028            0.028403            0.476537            0.733723
##         Year_Built  Bldg_Type_TwoFmCon    Bldg_Type_Duplex     Bldg_Type_Twnhs
##           0.002609           -0.048382           -0.116547           -0.100736
##   Bldg_Type_TwnhsE
##          -0.010134
```

```
## # A tibble: 1 x 12
##   r.squared adj.r.squared  sigma statistic p.value    df logLik    AIC    BIC
##       <dbl>         <dbl>  <dbl>     <dbl>   <dbl> <dbl>  <dbl>  <dbl>  <dbl>
## 1     0.733         0.732 0.0911      803.       0     8  2295. -4569. -4512.
## # ... with 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

R^2 = 0.733

```
## # A tibble: 586 x 1
##    .pred
##    <dbl>
##  1  5.34
##  2  5.15
##  3  5.15
##  4  5.44
##  5  5.37
##  6  5.39
##  7  5.31
##  8  5.26
##  9  5.32
```

```
## 10  5.46
## # ... with 576 more rows
```

```
## [1] 0.007296907
```

mse = 0.007296907

6. Add `Neighborhood` as an input variable recipe to your model from 5. What is the R^2 of your model? What is the MSE on the testing dataset?

```
ames.recipe.no6 <-
  recipe(Sale_Price ~ Longitude + Latitude + Gr_Liv_Area +
           Bldg_Type + Year_Built + Neighborhood, data=ames.train) %>%
  step_log(Gr_Liv_Area, base=10) %>%
  step_dummy(all_nominal_predictors())#turn all nominal variables into factors?
ames.recipe.no6
```

```
## Recipe
##
## Inputs:
##
##       role #variables
##    outcome          1
##  predictor          6
##
## Operations:
##
## Log transformation on Gr_Liv_Area
## Dummy variables from all_nominal_predictors()
```

```
lm.model.no6 <- linear_reg() %>%
  set_engine("lm")

lm.wflow.no6 <- workflow() %>%
  add_recipe(ames.recipe.no6) %>%
  add_model(lm.model.no6)

lm.fit.no6 <- fit(lm.wflow.no6, ames.train)
lm.fit.no6
```

```
## == Workflow [trained] ==========================================================
## Preprocessor: Recipe
## Model: linear_reg()
##
## -- Preprocessor ----------------------------------------------------------------
## 2 Recipe Steps
##
## * step_log()
## * step_dummy()
##
## -- Model -----------------------------------------------------------------------
##
## Call:
## stats::lm(formula = ..y ~ ., data = data)
##
## Coefficients:
##                                       (Intercept)
```

```
##                                               -63.559511
##                                                Longitude
##                                                -0.112870
##                                                 Latitude
##                                                 1.241635
##                                              Gr_Liv_Area
##                                                 0.626999
##                                               Year_Built
##                                                 0.002047
##                                        Bldg_Type_TwoFmCon
##                                                -0.031654
##                                          Bldg_Type_Duplex
##                                                -0.092194
##                                           Bldg_Type_Twnhs
##                                                -0.100407
##                                          Bldg_Type_TwnhsE
##                                                -0.038940
##                                Neighborhood_College_Creek
##                                                 0.033822
##                                     Neighborhood_Old_Town
##                                                -0.015720
##                                      Neighborhood_Edwards
##                                                -0.029868
##                                     Neighborhood_Somerset
##                                                 0.040568
##                          Neighborhood_Northridge_Heights
##                                                 0.111114
##                                      Neighborhood_Gilbert
##                                                -0.056205
##                                       Neighborhood_Sawyer
##                                                 0.002816
##                               Neighborhood_Northwest_Ames
##                                                -0.012576
##                                  Neighborhood_Sawyer_West
##                                                -0.011588
##                                     Neighborhood_Mitchell
##                                                 0.068331
##                                    Neighborhood_Brookside
##                                                 0.004856
##                                     Neighborhood_Crawford
##                                                 0.129778
##                      Neighborhood_Iowa_DOT_and_Rail_Road
##                                                -0.059162
##                                   Neighborhood_Timberland
##
## ...
## and 30 more lines.
```

```
## # A tibble: 1 x 12
##   r.squared adj.r.squared  sigma statistic p.value    df logLik    AIC    BIC
##       <dbl>         <dbl>  <dbl>     <dbl>   <dbl> <dbl>  <dbl>  <dbl>  <dbl>
## 1     0.801         0.798 0.0792      265.       0    35  2636. -5199. -4986.
## # ... with 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

R^2 = 0.801

```
## # A tibble: 586 x 1
##    .pred
##    <dbl>
##  1  5.30
##  2  5.16
##  3  5.14
##  4  5.52
##  5  5.46
##  6  5.44
##  7  5.24
##  8  5.23
##  9  5.25
## 10  5.49
## # ... with 576 more rows
```

```
## [1] 0.005853077
```

MSE = 0.005853077

7.

a. Summarize and sort the number of observations in each neighborhood. How many many neighborhoods have less than 20 observations?

```
## # A tibble: 4 x 2
##   Neighborhood observations
##   <fct>               <int>
## 1 Blueste                10
## 2 Greens                  8
## 3 Green_Hills             2
## 4 Landmark                1
```

There are four such neighborhoods. Blueste, Greens, Green_Hills, and Landmark.

b. Consult the documentation for `step_other` and add a step to your recipe where you collapse neighborhoods with less than 1% of your data. Make sure to add this step before the `step_dummy` command.

```
ames.recipe.no7 <-
  recipe(Sale_Price ~ Longitude + Latitude + Gr_Liv_Area +
           Bldg_Type + Year_Built + Neighborhood, data=ames.train) %>%
  step_log(Gr_Liv_Area, base=10) %>%
  step_other(Neighborhood, threshold = 0.01) %>%
  step_dummy(all_nominal_predictors())#turn all nominal variables into factors?
ames.recipe.no7
```

```
## Recipe
##
## Inputs:
##
##       role #variables
##    outcome          1
##  predictor          6
##
## Operations:
##
## Log transformation on Gr_Liv_Area
## Collapsing factor levels for Neighborhood
## Dummy variables from all_nominal_predictors()
```

```r
lm.model.no7 <- linear_reg() %>%
  set_engine("lm")

lm.wflow.no7 <- workflow() %>%
  add_recipe(ames.recipe.no7) %>%
  add_model(lm.model.no7)

lm.fit.no7 <- fit(lm.wflow.no7, ames.train)
lm.fit.no7
```

```
## == Workflow [trained] ==========================================================
## Preprocessor: Recipe
## Model: linear_reg()
##
## -- Preprocessor ------------------------------------------------------------------
## 3 Recipe Steps
##
## * step_log()
## * step_other()
## * step_dummy()
##
## -- Model -------------------------------------------------------------------------
##
## Call:
## stats::lm(formula = ..y ~ ., data = data)
##
## Coefficients:
##                                          (Intercept)
##                                           -66.244294
##                                            Longitude
##                                            -0.454376
##                                             Latitude
##                                             0.544911
##                                          Gr_Liv_Area
##                                             0.625078
##                                           Year_Built
##                                             0.002053
##                                   Bldg_Type_TwoFmCon
##                                            -0.031414
##                                     Bldg_Type_Duplex
##                                            -0.092101
##                                      Bldg_Type_Twnhs
##                                            -0.104979
##                                     Bldg_Type_TwnhsE
##                                            -0.038985
##                             Neighborhood_College_Creek
##                                            -0.006108
##                                Neighborhood_Old_Town
##                                            -0.022344
##                                 Neighborhood_Edwards
##                                            -0.060798
##                                Neighborhood_Somerset
##                                             0.037239
##                       Neighborhood_Northridge_Heights
```

```
##                                                  0.111473
##                                      Neighborhood_Gilbert
##                                                 -0.051534
##                                       Neighborhood_Sawyer
##                                                 -0.021770
##                               Neighborhood_Northwest_Ames
##                                                 -0.013180
##                                  Neighborhood_Sawyer_West
##                                                 -0.040612
##                                     Neighborhood_Mitchell
##                                                  0.037583
##                                    Neighborhood_Brookside
##                                                 -0.004497
##                                     Neighborhood_Crawford
##                                                  0.103378
##                       Neighborhood_Iowa_DOT_and_Rail_Road
##                                                 -0.074474
##                                   Neighborhood_Timberland
##
## ...
## and 18 more lines.
```

c. Rerun your workflow and your model. How do you interpret the coefficient of the model associated with the collapsed set of neighborhoods? What is the MSE of this new model?

```
## # A tibble: 31 x 5
##    term                                estimate std.error statistic   p.value
##    <chr>                                  <dbl>     <dbl>     <dbl>     <dbl>
##  1 (Intercept)                         -6.62e+1 32.1          -2.06  3.91e-  2
##  2 Longitude                           -4.54e-1  0.306        -1.49  1.37e-  1
##  3 Latitude                             5.45e-1  0.395         1.38  1.68e-  1
##  4 Gr_Liv_Area                          6.25e-1  0.0145       43.2   2.87e-299
##  5 Year_Built                           2.05e-3  0.000121     16.9   1.16e- 60
##  6 Bldg_Type_TwoFmCon                  -3.14e-2  0.0118       -2.66  7.84e-  3
##  7 Bldg_Type_Duplex                    -9.21e-2  0.00920     -10.0   4.10e- 23
##  8 Bldg_Type_Twnhs                     -1.05e-1  0.0118       -8.92  9.55e- 19
##  9 Bldg_Type_TwnhsE                    -3.90e-2  0.00766      -5.09  3.92e-  7
## 10 Neighborhood_College_Creek          -6.11e-3  0.0245       -0.250 8.03e-  1
## 11 Neighborhood_Old_Town               -2.23e-2  0.00959      -2.33  1.99e-  2
## 12 Neighborhood_Edwards                -6.08e-2  0.0192       -3.17  1.55e-  3
## 13 Neighborhood_Somerset                3.72e-2  0.0127        2.94  3.33e-  3
## 14 Neighborhood_Northridge_Heights      1.11e-1  0.0160        6.99  3.61e- 12
## 15 Neighborhood_Gilbert                -5.15e-2  0.0130       -3.97  7.43e-  5
## 16 Neighborhood_Sawyer                 -2.18e-2  0.0187       -1.16  2.44e-  1
## 17 Neighborhood_Northwest_Ames         -1.32e-2  0.0104       -1.27  2.04e-  1
## 18 Neighborhood_Sawyer_West            -4.06e-2  0.0230       -1.76  7.81e-  2
## 19 Neighborhood_Mitchell                3.76e-2  0.0224        1.68  9.35e-  2
## 20 Neighborhood_Brookside              -4.50e-3  0.0115       -0.393 6.95e-  1
## 21 Neighborhood_Crawford                1.03e-1  0.0167        6.20  6.85e- 10
## 22 Neighborhood_Iowa_DOT_and_Rail_Road -7.45e-2  0.0137       -5.44  5.99e-  8
## 23 Neighborhood_Timberland              7.47e-2  0.0247        3.03  2.48e-  3
## 24 Neighborhood_Northridge              6.61e-2  0.0166        3.99  6.71e-  5
## 25 Neighborhood_Stone_Brook             1.39e-1  0.0163        8.55  2.21e- 17
## 26 Neighborhood_South_and_West_of_Iowa_S~ -3.67e-2  0.0204    -1.80  7.16e-  2
## 27 Neighborhood_Clear_Creek             4.58e-2  0.0233        1.97  4.95e-  2
```

```
## 28 Neighborhood_Meadow_Village           -5.44e-2  0.0259      -2.10  3.61e-  2
## 29 Neighborhood_Briardale                -5.38e-2  0.0204      -2.64  8.37e-  3
## 30 Neighborhood_Bloomington_Heights       1.36e-2  0.0216       0.629 5.29e-  1
## 31 Neighborhood_other                     5.94e-2  0.0152       3.92  9.14e-  5
```

Neighborhoods with fewer listings tend to have a higher sale price of log-inverse of 5.94e-2.

```
## # A tibble: 586 x 1
##    .pred
##    <dbl>
##  1  5.30
##  2  5.16
##  3  5.18
##  4  5.52
##  5  5.46
##  6  5.44
##  7  5.24
##  8  5.23
##  9  5.25
## 10  5.49
## # ... with 576 more rows
```

```
## [1] 0.005911335
```

mse = 0.005911335

8.

a. What two features are you planning to use for your first challenge?

An idea: x_1 = abs(q1 - q4) x_2 = abs(q2 - q3)

b. Using the MNIST dataset select a couple of instances of the two digits assigned to your group. Calculate the two features on those instances. Are the two features similar across the two different types of digits?

The numbers of my group are 0 and 7. 0 is symmetric so we should measure with differences between halves or quarters.

I am copy-pasting my work from the project document:

```
## List of 2
##  $ train:List of 2
##   ..$ images: int [1:60000, 1:784] 0 0 0 0 0 0 0 0 0 0 ...
##   ..$ labels: int [1:60000] 5 0 4 1 9 2 1 3 1 4 ...
##  $ test :List of 2
##   ..$ images: int [1:10000, 1:784] 0 0 0 0 0 0 0 0 0 0 ...
##   ..$ labels: int [1:10000] 7 2 1 0 4 1 4 9 5 9 ...
```

```
##   [1]   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
##  [19]   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
##  [37]   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
##  [55]   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
##  [73]   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
##  [91]   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
## [109]   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
## [127]   0   0  86 255 226 170  29   0   0   0   0   0   0   0   0   0   0   0
## [145]   0   0   0   0   0   0   0   0   0   0   0  86 255 170 170 255 226 114
## [163]   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
## [181]   0   0   0 255 226   0   0   0 170 255  57   0   0   0   0   0   0   0
## [199]   0   0   0   0   0   0   0   0 114 255 198  29   0 255  57   0   0   0
```

```
## [217]   0 255 170   0   0   0   0   0   0   0   0   0   0   0   0   0  57 198
## [235] 255 198 255 226   0 255  86   0   0   0   0 170 226   0   0   0   0   0
## [253]   0   0   0   0   0   0   0   0 255 255  86   0 198 255  29  86 170   0
## [271]   0   0   0 141 255  29   0   0   0   0   0   0   0   0   0   0   0  29
## [289] 255 226   0   0  57 255  86   0   0   0   0   0   0  29 255  86   0   0
## [307]   0   0   0   0   0   0   0   0   0 170 255 114   0   0  57 255  86   0
## [325]   0   0   0   0   0   0 255  86   0   0   0   0   0   0   0   0   0   0
## [343]   0 226 255  57   0   0 141 170   0   0   0   0   0   0   0  29 255  86
## [361]   0   0   0   0   0   0   0   0   0   0  57 255 198   0   0   0 226  57
## [379]   0   0   0   0   0   0   0  86 255   0   0   0   0   0   0   0   0   0
## [397]   0   0 170 255 114   0   0   0   0   0   0   0   0   0   0   0   0 141
## [415] 255   0   0   0   0   0   0   0   0   0   0   0 170 255  86   0   0   0
## [433]   0   0   0   0   0   0   0   0   0 170 255   0   0   0   0   0   0   0
## [451]   0   0   0   0 170 255  86   0   0   0   0   0   0   0   0   0   0   0
## [469]   0 226 141   0   0   0   0   0   0   0   0   0   0   0 170 255  86   0
## [487]   0   0   0   0   0   0   0   0   0   0  86 255  86   0   0   0   0   0
## [505]   0   0   0   0   0   0 114 255 141   0   0   0   0   0   0   0   0   0
## [523]   0   0 198 226   0   0   0   0   0   0   0   0   0   0   0   0   0 255
## [541] 255  57   0   0   0   0   0   0   0   0  29 170 255  86   0   0   0   0
## [559]   0   0   0   0   0   0   0   0   0 170 255 226  29   0   0   0   0   0
## [577]   0  57 226 255 198  29   0   0   0   0   0   0   0   0   0   0   0   0
## [595]   0  29 226 255 198 114  86  86 141 226 255 255 255 114   0   0   0   0
## [613]   0   0   0   0   0   0   0   0   0   0   0   0  29 226 255 255 255 255
## [631] 255 255 198 114  29   0   0   0   0   0   0   0   0   0   0   0   0   0
## [649]   0   0   0   0   0   0 114 198 226 170 141  29   0   0   0   0   0   0
## [667]   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
## [685]   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
## [703]   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
## [721]   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
## [739]   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
## [757]   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
## [775]   0   0   0   0   0   0   0   0   0   0
```

```
## [1] 0
```

```
## # A tibble: 200 x 3
##    value image[,1]  [,2]  [,3]  [,4]  [,5]  [,6]  [,7]  [,8]  [,9] [,10] label
##    <int>     <int> <int> <int> <int> <int> <int> <int> <int> <int> <int> <int>
##  1 12127         0     0     0     0     0     0     0     0     0     0     7
##  2 23071         0     0     0     0     0     0     0     0     0     0     7
##  3 51405         0     0     0     0     0     0     0     0     0     0     0
##  4 14096         0     0     0     0     0     0     0     0     0     0     7
##  5 44097         0     0     0     0     0     0     0     0     0     0     0
##  6 42749         0     0     0     0     0     0     0     0     0     0     7
##  7 59647         0     0     0     0     0     0     0     0     0     0     7
##  8 54098         0     0     0     0     0     0     0     0     0     0     7
##  9 11333         0     0     0     0     0     0     0     0     0     0     7
## 10 33871         0     0     0     0     0     0     0     0     0     0     7
## # ... with 190 more rows, and 1 more variable: image[11:784] <int>
```

First, take a single vector of 0: (will also do this process for 7 as a comparison: tester$label[1])

```
## [1] 6588
```

```
## [1] 8456
```

```
## [1] 7982
```

```
## [1] 8562
```

```
## [1] 2448
```

q1 = 6588; q2 = 8456; q3 = 7982; q4 = 8562 x_1 = abs(q1 - q4) = 1974 x_2 = abs(q2 - q3) = 474

Now, try this process above for 7:

```
## [1] 4821
```

```
## [1] 1757
```

```
## [1] 7921
```

```
## [1] 4487
```

```
## [1] 6498
```

q1 = 4821; q2 = 1757; q3 = 7921; q4 = 4487 x_1 = abs(q1 - q4) = 334 x_2 = abs(q2 - q3) = 6164

The results for the two numbers are fairly different. (plotting a bunch of data points would be a better indicator to guage if x_1 and x_2 are reasonable variables to use to classify 0 and 7.)