The Prediction of California Housing Prices

Hüseyin KAYAR

3/25/23

Supervised Learning: Regression Model

In this task, I am expected to use a regression model on a data set about "California House Prices" on Kaggle. I tried to predict "Median house value" feature in the data set using this model. While training the regression model, I used the features contained in the data set. These are as follows:

X1 : Longitude X2 : Latitude X3 : Housing median age X4 : Total rooms X5 : Total bedrooms X6 : Population X7 : Households X8 : Median income X9 : Ocean proximity Y : Median house value

X1 - X9 are my features to use in regression model. Y is the target to predict.

Packages

To use regression model and make prediction we have to use some packages like below;

```
install.packages("readr")
install.packages("car")
library(readr)
library(car)
```

Dataset

The data contains information from the 1990 California census. There are 9 different features and 1 target to predict in the data set.

```
housing <- read_csv("housing.csv")
```

```
Rows: 20640 Columns: 10
-- Column specification --------------------------------------------------------
Delimiter: ","
chr (1): ocean_proximity
dbl (9): longitude, latitude, housing_median_age, total_rooms, total_bedroom...

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

Variable Types

There are 10 different variables in the data set.

```r
str(housing)
```

```
spc_tbl_ [20,640 x 10] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
 $ longitude         : num [1:20640] -122 -122 -122 -122 -122 ...
 $ latitude          : num [1:20640] 37.9 37.9 37.9 37.9 37.9 ...
 $ housing_median_age: num [1:20640] 41 21 52 52 52 52 52 52 42 52 ...
 $ total_rooms       : num [1:20640] 880 7099 1467 1274 1627 ...
 $ total_bedrooms    : num [1:20640] 129 1106 190 235 280 ...
 $ population        : num [1:20640] 322 2401 496 558 565 ...
 $ households        : num [1:20640] 126 1138 177 219 259 ...
 $ median_income     : num [1:20640] 8.33 8.3 7.26 5.64 3.85 ...
 $ median_house_value: num [1:20640] 452600 358500 352100 341300 342200 ...
 $ ocean_proximity   : chr [1:20640] "NEAR BAY" "NEAR BAY" "NEAR BAY" "NEAR BAY" ...
 - attr(*, "spec")=
  .. cols(
  ..   longitude = col_double(),
  ..   latitude = col_double(),
  ..   housing_median_age = col_double(),
  ..   total_rooms = col_double(),
  ..   total_bedrooms = col_double(),
  ..   population = col_double(),
  ..   households = col_double(),
  ..   median_income = col_double(),
  ..   median_house_value = col_double(),
  ..   ocean_proximity = col_character()
  .. )
 - attr(*, "problems")=<externalptr>
```

1. Longitude:(Numeric) A measure of how far west a house is; a higher value is farther west
2. Latitude: (Numeric) A measure of how far north a house is; a higher value is farther north
3. Housing Median Age:(Numeric) Median age of a house within a block; a lower number is a newer building
4. Total Rooms: (Numeric) Total number of rooms within a block
5. Total Bedrooms: (Numeric) Total number of bedrooms within a block
6. Population: (Numeric) Total number of people residing within a block
7. Households: (Numeric)Total number of households, a group of people residing within a home unit, for a block
8. Median Income:(Numeric) Median income for households within a block of houses (measured in tens of thousands of US Dollars)
9. Median House Value: (Numeric) Median house value for households within a block (measured in US Dollars)
10. Ocean Proximity: (Character) Location of the house w.r.t ocean/sea

Dimension

I used the code below to find the number of rows and columns in the data set we have, this will give me information about how many different variables are in the data I have.

```
dim(housing)
```

```
[1] 20640    10
```

```
housing <- na.exclude(housing)
```

Traning

During the training phase, train and test sets are used and these sets should contain different data from each other. I did this by randomly splitting the original data set.

As first step, I created a train set. I set the train set to contain 80% of the entire data set. In addition, as the second stage, I used 20% of the test data set.

```
set.seed(123)
index <- sample(1:nrow(housing),round(nrow(housing)*0.80))
train <- housing[index, ]
test <- housing[-index, ]
```

This is the first 10 columns in the train set:

```
train
```

```
# A tibble: 16,346 x 10
   longitude latitude housing_median_age total_rooms total_bedrooms population
       <dbl>    <dbl>              <dbl>       <dbl>          <dbl>      <dbl>
 1     -122      38.4                 16        2509            366       1043
 2     -122.     38.3                  8        5092            988       1657
 3     -119.     35.3                 10        7011           1453       4163
 4     -124.     41.8                 11        3159            616       1343
 5     -118.     34.3                 41        1297            327        733
 6     -121.     38.8                 14        2028            255        781
 7     -118.     34.0                 44        1944            458        981
 8     -118.     34.1                 43        1716            402       1343
 9     -121.     37.9                 30        1061            230        851
10     -116.     33.4                 23        1586            448        338
# i 16,336 more rows
# i 4 more variables: households <dbl>, median_income <dbl>,
#   median_house_value <dbl>, ocean_proximity <chr>
```

This is the first 10 columns in the test set:

```
test
```

```
# A tibble: 4,087 x 10
   longitude latitude housing_median_age total_rooms total_bedrooms population
       <dbl>    <dbl>              <dbl>       <dbl>          <dbl>      <dbl>
 1     -122.     37.8                 52        1467            190        496
 2     -122.     37.8                 52        3104            687       1157
 3     -122.     37.8                 52        3503            752       1504
 4     -122.     37.8                 52        1688            337        853
 5     -122.     37.8                 49        1655            366        754
 6     -122.     37.8                 51        2665            574       1258
 7     -122.     37.8                 52        1470            330        689
 8     -122.     37.8                 43        1007            312        558
 9     -122.     37.8                 41        3221            853       1959
10     -122.     37.8                 52        1387            341       1074
# i 4,077 more rows
# i 4 more variables: households <dbl>, median_income <dbl>,
#   median_house_value <dbl>, ocean_proximity <chr>
```

After separating the original data set into train and test sets, I used the lm() function to train the linear regression model.

```
lrmModel <- lm(median_house_value ~ ., data=train)
lrmModel
```

```

Call:
lm(formula = median_house_value ~ ., data = train)

Coefficients:
              (Intercept)                  longitude
               -2.292e+06                 -2.708e+04
                 latitude         housing_median_age
               -2.576e+04                  1.057e+03
              total_rooms             total_bedrooms
               -6.195e+00                  9.151e+01
               population                 households
               -3.734e+01                  5.891e+01
            median_income       ocean_proximityINLAND
                3.923e+04                 -3.781e+04
    ocean_proximityISLAND    ocean_proximityNEAR BAY
                1.709e+05                 -3.503e+03
ocean_proximityNEAR OCEAN
                4.828e+03
```

This is the summary of the regression model that I have trained. In here we can see a lot of information about the model like estimation of the feature,residuals etc.

```
summary(lrmModel)
```

```

Call:
lm(formula = median_house_value ~ ., data = train)

Residuals:
    Min      1Q  Median      3Q     Max
-555127  -42863  -10715   28683  756919

Coefficients:
```

```
                            Estimate Std. Error t value Pr(>|t|)
(Intercept)               -2.292e+06  9.858e+04 -23.254  < 2e-16 ***
longitude                 -2.708e+04  1.143e+03 -23.696  < 2e-16 ***
latitude                  -2.576e+04  1.127e+03 -22.849  < 2e-16 ***
housing_median_age         1.057e+03  4.940e+01  21.407  < 2e-16 ***
total_rooms               -6.195e+00  8.724e-01  -7.102 1.28e-12 ***
total_bedrooms             9.151e+01  7.551e+00  12.119  < 2e-16 ***
population                -3.734e+01  1.181e+00 -31.606  < 2e-16 ***
households                 5.891e+01  8.214e+00   7.172 7.71e-13 ***
median_income              3.923e+04  3.761e+02 104.308  < 2e-16 ***
ocean_proximityINLAND     -3.781e+04  1.955e+03 -19.339  < 2e-16 ***
ocean_proximityISLAND      1.709e+05  3.450e+04   4.953 7.38e-07 ***
ocean_proximityNEAR BAY   -3.503e+03  2.141e+03  -1.636    0.102
ocean_proximityNEAR OCEAN  4.828e+03  1.757e+03   2.748    0.006 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 68910 on 16333 degrees of freedom
Multiple R-squared:  0.6459,    Adjusted R-squared:  0.6457
F-statistic:  2483 on 12 and 16333 DF,  p-value: < 2.2e-16
```

Measuring Model Performance

In order to measure the performance of the trained model, we need to use the model with the test data and compare the results with the actual results.

For this, I first extracted the line to be predicted from the test data.

```
testWithoutTarget <- test[,-9]
```

Then I use lineer regression model to predict the median_house_value in the test set.

```
predicted <- predict(lrmModel,testWithoutTarget)
```

This is the top 10 data predicted by the regression model.

```
head(predicted)
```

```
       1        2        3        4        5        6
378962.7 255538.2 257098.7 188356.3 160631.5 222061.5
```

Error

I found the margin of error by subtracting the estimated values from the actual values in the test set.

```
error <- test$median_house_value - predicted
head(error)
```

```
         1          2          3          4          5          6 
 -26862.66  -14138.23  -15298.66  -88656.28  -55731.52 -112361.46 
```

There are some metrics that can be used to measure the performance of the model. These metrics are MSE (Mean Squared Error), RMSE (Root Mean Squared Error), MAE (Mean Absolute Error) respectively. In this task I have decided to use RMSE for certain resons. The Mean Squared Error (MSE) and Root Mean Square Error (RMSE) measures have a stronger penalty for larger prediction errors compared to the Mean Absolute Error (MAE). Of the two, RMSE is generally favored over MSE for evaluating the performance of regression models against other random models, since it shares the same units as the dependent variable (Y-axis). Also RMSE tells how well a regression model can predict the value of a response variable in absolute terms than others.

```
rmse <- sqrt(mean(error^2))
rmse
```

```
[1] 67676.82
```

Over-fitting/Under-fitting

We need to check whether there is over-fitting or under-fitting in the model we have created. Over-fitting happens if a model learns from train set too much. Under-fitting happens when there is a insufficient learning of a model from the train set. To check those problems we can use RMSE model that we have created before.

```
rmseTrain <- sqrt(mean((lrmModel$residuals)^2))
rmseTest <- rmse
```

In the result we can see train error and test error is nearly equal to each other. But it is a positive number so our train set is better then our test set. So there can not be over-fitting but there can be an under-fitting.

```
rmseTrain - rmseTest
```

```
[1] 1207.155
```

Adding New Observations

To test the model I trained, I will generate any home data that is not included in the data set and predict its price through the regression model.

```
temp <- data.frame(longitude=-123.76,
                   latitude = 37.28,
                   housing_median_age = 45,
                   total_rooms = 4983,
                   total_bedrooms =487,
                   population = 685,
                   households = 327,
                   median_income = 5.850,
                   ocean_proximity = "<1H OCEAN")
```

The data I just created was chosen between the min and max values of the rows in the data set because values out of the range can lead to incorrect predictions.

I use the regression model to predict the median_house_value for my own data.

```
tempPredictedPrice <- predict(lrmModel,temp)
```

This is the predicted median_house_value.

```
tempPredictedPrice
```

```
       1 
382874.2 
```