12/18/2022

SHRAVAN SOBHANI

Ssobhani@kent.edu

MACHINE LEARNING REPORT

HOUSING DATA

## **Objective**

To find out the houses which are near to the coastal area so we have given the ocean approximately and the longitude latitude of the houses based on the analysis we can define how many houses are near to the ocean

And we are going to let the ocean proximity into binary columns because most machine learning algorithms handle categorical in single columns.

In this data set we are going to use random forest like 1000 decision trees able to predict the median price of the houses based on the location.

## **Analysis**

 additional categorical attribute called ocean proximity was added, indicating whether each block group is near the ocean, near the Bay area, inland or on an island.

Dataset consists of20640 observations and 14 variables

Diagram

Description automatically generated with medium confidence

Longitudinal and latitude of the house, what’s the average age in a house, population in each house, income, and house value are the variables in our dataset called housing.

1. Longitude and latitude-is a factor which shows the longitude value in the latitude of a particular house.
2. House median age-it is the value which shows the age of a house (age from the day of built)
3. Total rooms- it is a value which shows how many rooms each house has
4. Total bedrooms- no of bedroom in a house
5. Population-every house consists of people and its value is stored in population factor.
6. Households- number of workers in a house.
7. Median income-it is the average income of a house
8. Median house - value it is the average house value
9. ocean proximity-it is the distance between your house to ocean

## **So, from that summary we can see a few things we need to do before running algorithms.**

1)NAs in total bedrooms need to be addressed. These must be given a value

2)We will split the ocean proximity into binary columns. Most machine learning algorithms in R can handle categorical in a single column, but we will cater to the lowest common denominator and do the splitting.

3)Make the total bedrooms and total rooms into a mean\_number\_bedrooms and mean\_number\_rooms columns as there are likely more accurate depictions of the houses in each group.

Timeline

Description automatically generated

1)There are some housing blocks with old age homes in them.

2)The median house value has some weird cap applied to it causing there to be a blip at the rightmost point on the hist. There are most definitely houses in the bay area worth more than 500,000... even in the 90s when this data was collected!

3)We should standardize the scale of the data for any non-tree based methods. As some of the variables range from 0-10, while others go up to 500,000

4)We need to think about how the cap on housing prices can affect our prediction... may be worth removing the capped values and only working with the data we are confident in.

## **Cleaning the data**

film medium for total bedrooms which is the only column with missing values the median is used instead off mean because it is less influenced by extreme outliers

Table

Description automatically generated

after cleaning the data

turn categorical into Booleans:

1)Get a list of all the categories in the ‘ocean proximity’ column

2)Make a new empty data frame of all 0s, where each category is its own column

3)Use a for loop to populate the appropriate columns of the data frame

4)Drop the original column from the data frame.

Table

Description automatically generated

Scale the numerical variables Note here I scale every one of the numerical except for ‘median\_house\_value’ as this is what we will be working to predict. The x values are scaled so that coefficients in things like support vector machines are given equal weight, but the y value scale doesn’t affect the learning algorithms in the same way (and we would just need to re-scale the predictions at the end which is another hassle).

Graphical user interface, text

Description automatically generated

Test set:

Note that the train data below has all the columns we want, and also that the index is jumbled (so we did take a random sample). The second check makes sure that the length of the train and test dataframes equals the length of the dataframe they were split from, which shows we didn’t lose data or make any up by accident.

Table

Description automatically generated

Test some predictive models simple linear model using 3 of the available predictors. Median income, total rooms, and population. This serves as an entry point to introduce the topic of cross validation and a basic model.

So here we do cross validation to test the model using the training data itself. Our K is 5, what this means is that the training data is split into 5 equal portions. One of the 5 folds is put to the side (as a mini test data set) and then the model is trained using the other 4 portions. After that the predictions are made on the folds that was withheld, and the process is repeated for each of the 5 folds and the average predictions produced from the iterations of the model is taken. This gives us a rough understanding of how well the model predicts on external data.

Graphical user interface, application, Word

Description automatically generated

The first component is the raw cross-validation estimate of prediction error. The second component is the adjusted cross-validation estimate.

## **Using Algorithm (Random Forest)**

So here we do cross validation to test the model using the training data itself

Text

Description automatically generated

## **The out-of-bag (oob) error estimate**

In random forests, there is no need for cross-validation or a separate test set to get an unbiased estimate of the test set error. It is estimated internally, during the run, as follows:

Each tree is constructed using a different bootstrap sample from the original data. About one-third of the cases are left out of the bootstrap sample and not used in the construction of the kth tree.

Text

Description automatically generated with medium confidence

## [1] 49126.22

So even using a random forest of only 1000 decision trees we are able to predict the median price of a house in a given district to within $49,000 of the actual median house price. This can serve as our bechmark moving forward and trying other models.

How well does the model predict on the test data?

Graphical user interface, text

Description automatically generated

## **Conclusion**

Well, that looks great! Our model scored roughly the same on the training and testing data, suggesting that it is not overfit and that it makes good predictions.

## **Reference :**

<https://econpapers.repec.org/scripts/search.pf?ft=california+housing>