Project 4: Can you predict that?

Author: Gabriel Sanahuano

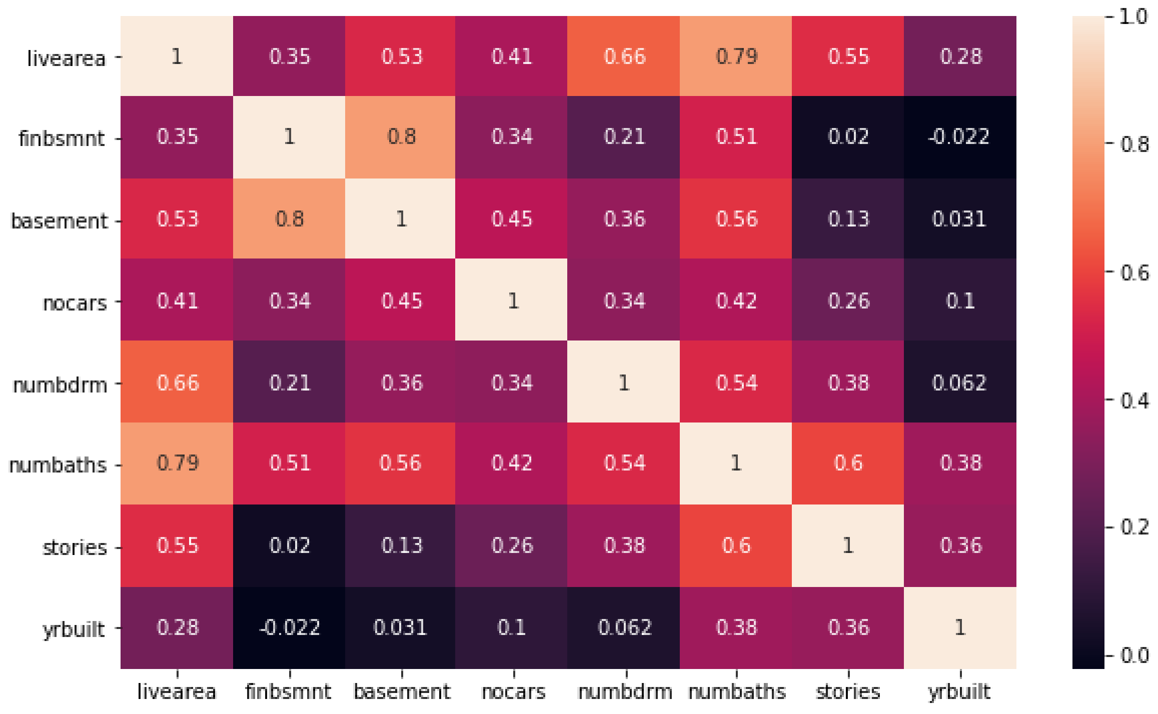
Elevator pitch

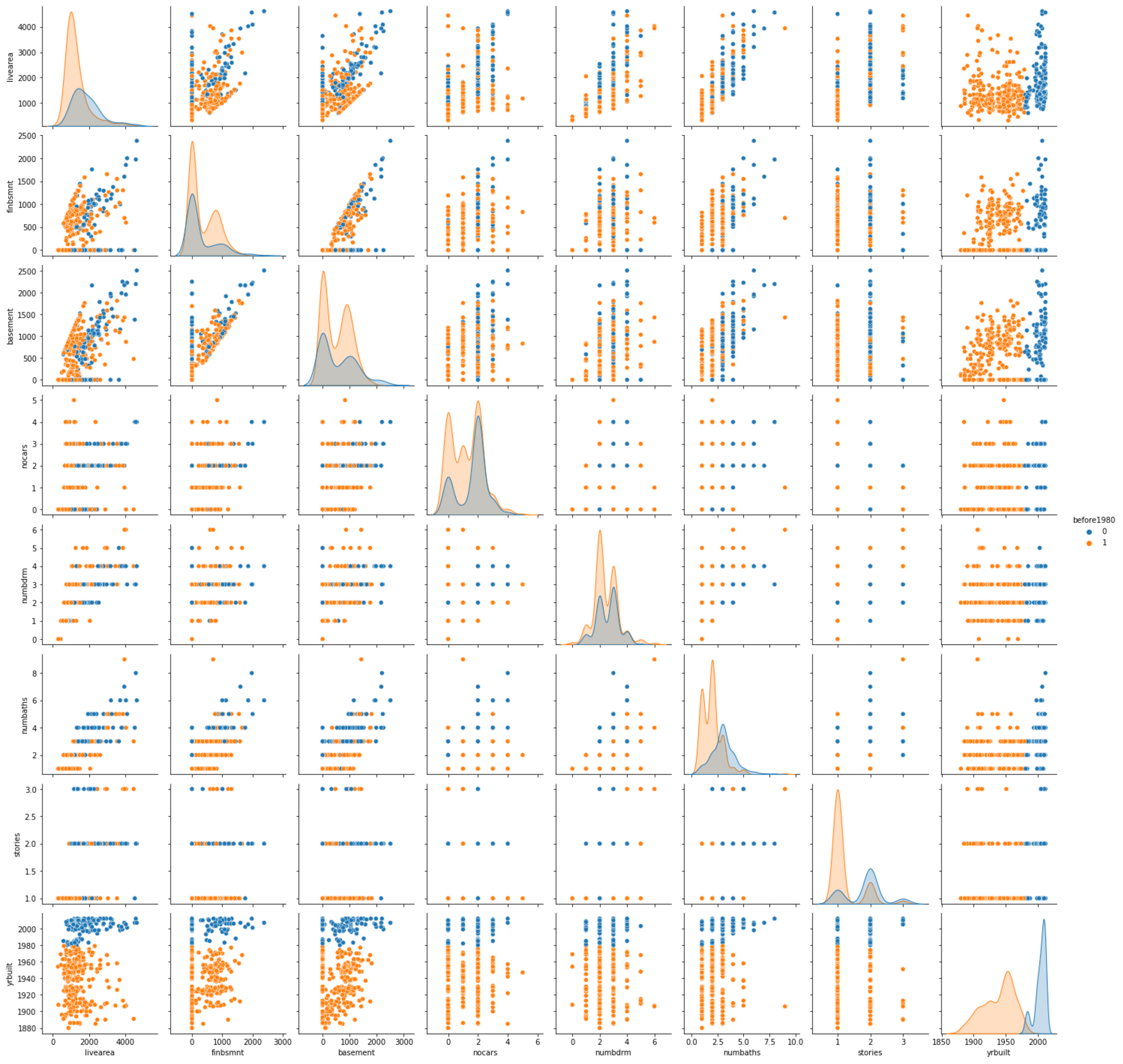
I was assigned a project in which I had to apply the most basic principles of machine learning. This project consisted of building a predictive model that could classify if a house was built before 1980. I was given some CSV files with important information that would help me build this model. Some of the tools I used were some such as machine learning libraries, visualization charts, feature scaling, confusion matrix, among others. This report answers some of the most important questions about the development and implementation of my machine learning project.

Technical Details

Grand Question #1

- Create 2-3 charts that evaluate potential relationships between the home variables and before1980.





Grand Question #2

- Can you build a classification model (before or after 1980) that has at least 90% accuracy for the state of Colorado to use (explain your model choice and which models you tried)?

I was able to generate a model using two machine learning techniques to develop ML models, such as a Decision Tree Classifier and a KNeighbors Classifier. Before doing this, I did a quick check of the different features and the target of the project. I used different functions like head (), describe (), isnull (), and sum () to be able to analyze the information that was contained on the files for this project. During this process, I also inspected the data to see if there were some missing values, and for this, I used the isnull () function. Fortunately, there was no missing value.

Also during this data preprocessing state, I generated a pair plot chart using Seaborn to be able to analyze correlations between some special features and the target, which is the 'before 1980' column. Another visualization that is also very useful to analyze these patterns is the heatmap chart that helps to find correlations between features and the target. As the last step, I decided to apply an important technique which is feature scaling, and as is well known there are different ways to do this, but in this situation, I decided that it was better to apply the Min-Max Scaler to fix ranges of data in the different features that were not balanced.

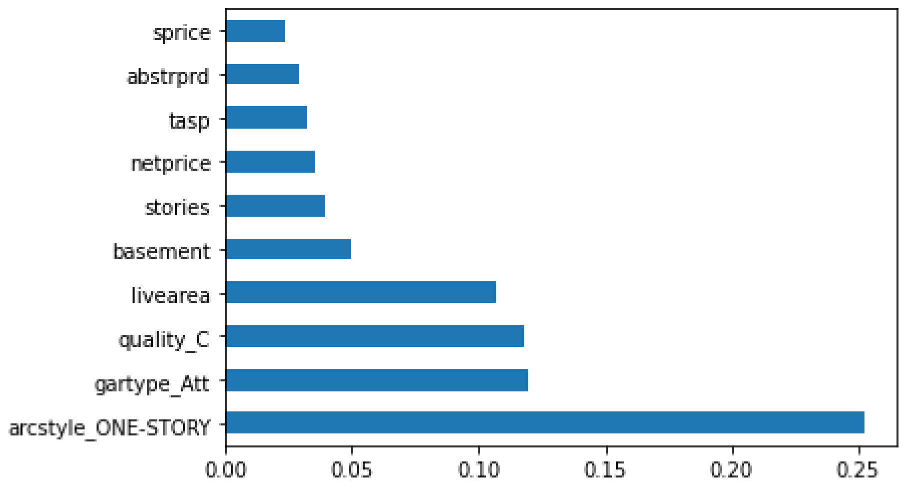
To create my model, I divided the data into training and test data. This to be able to have information with which to feed my model and train it several times. After this, I used the test data to compare and see results, in this case, to analyze how well my model was able to respond if a house was built before 1980. Now, using the Decision Tree Classifier model, I was able to have an accuracy of 0.9111, which is a very good result. But I also wanted to try and see how the results would come out using a different type of model. For that, I used a model using KNeighborsClassifier, but I did not get the same results. The accuracy it generated was only 0.8684. This confirms that using the decision tree classifier can produce better accuracy for my model.

In response to the grand question, I was indeed able to generate a model that has an accuracy of better than 90%.

Grand Question #3

- Will you justify your classification model by detailing the most important features in your model (a chart and a description are a must)?

I did a little research to find the best tool to demonstrate the importance of the features that have the most impact on my model. I found that there is an inbuilt class called feature_importances of tree base classifier, whose objective is to give a score to the features that are most important and that have a greater effect on the model. Also, I generated a visualization demonstrating the most important features.



Grand Question #4

- Can you describe the quality of your classification model using 2-3 evaluation metrics?
  You need to explain how to interpret each evaluation metric when you provide the value.

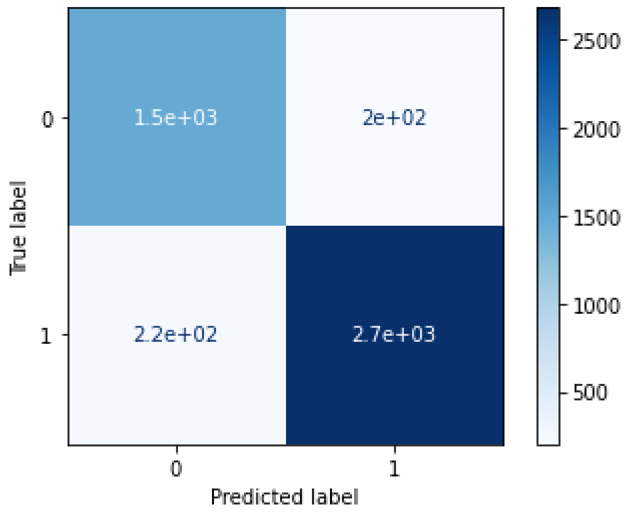
Use the following evaluation metrics to measure the quality of my classification model:

- Confusion Matrix

We know how often the model was correct, but we don't know when it was wrong or why. This is where the confusion matrix plays an important role.

We had a quantity of 4583 samples on our test data to test with our model. As you can see in the graph below, we visually project the results of the confusion matrix and we were able to obtain very good results from the metric error analysis. It can be said that our model was correct a total of 4159 times, this being 91%, and it was wrong 424 times that they make 9%

array([1477, 201], [223, 2682])



- F1 Score

F1 Score is an evaluation metric for the classification algorithm. It combines precision and recalls in a weighted average of them. F1 score reaches its best value at 1 and worst at 0.

The result of the F1 score applied to the Decision Tree Classifier model was 92%. Which is a very good result and our model gets a good evaluation.

F1 Score: 0.9267449896337249

Appendix A (Python Script)

https://gist.github.com/gabecastri/4a39f68c45f0bf27451d6cd27f5f51e7