Machine Learning Engineer Nanodegree Capstone Report

Drew Lehe

I. Definition

Neighborhood Clustering and Prediction

This project uses a dataset of housing sales from Melbourne, Australia, from 2016 through 2017. Melbourne experienced a large property bubble from the late 2000s until 2018, leaving many Australians distraught over rapidly-rising house prices (which are now deflating). With this project I hope to answer some questions: What creates a high-value neighborhood? Is it close to the city center and densely-populated, or is it remote and sparse? What kind of housing is built there?

Answering these questions will be useful to both demographers and real estate investors. Demographers are interested in tracking the change of neighborhoods over time. Real estate developers have a particularly tricky goal: they want to invest in a low-value neighborhood, and change it to a high-value neighborhood by building the right combination of features. Developers also don't want their neighborhood to suddenly drop in price, and individuals don't want their neighborhood to suddenly increase (causing displacement). If we know what creates both a high-value and a low-value neighborhood, we can help avoid bubbles and displacement. Through clustering and examining each cluster's features, we can find the attributes of valuable neighborhoods.

Predicting a neighborhood is a notoriously hard problem, just as a "neighborhood" is notoriously hard to define. For that reason, it's most useful to examine different *levels* of municipalities. In Australian parlance, the "suburb" refers to what Americans call a "neighborhood," the "Council Area" is the municipality, and the "City" refers to the central city itself (in this case Melbourne).

For demographers, a common problem is working with sparse data from public websites. If we can predict, with a fair degree of accuracy, a feature about housing data, we can help them perform better research in the future. Therefore, I'll also attempt to predict which Council Area a house is in, based on certain features.

Metrics

For clustering, I'll use the silhouette score to examine quality of the clusters. The silhouette score is a measure of how similar the observations in a cluster are. It ranges from -1 (observations are very dissimilar) to 1 (observations are completely alike).

For Council Area prediction, I'll use the F1-score. The F1-score is ideal for a classification model because it weighs the "precision" (number of true positives over the total number of positives) and "recall" (number of true positives over the total number of observations) of your model, which is a smarter metric than simple accuracy. Scikit-learn also has a built-in feature called score_report, which tells me how well I predicted each class (council area).

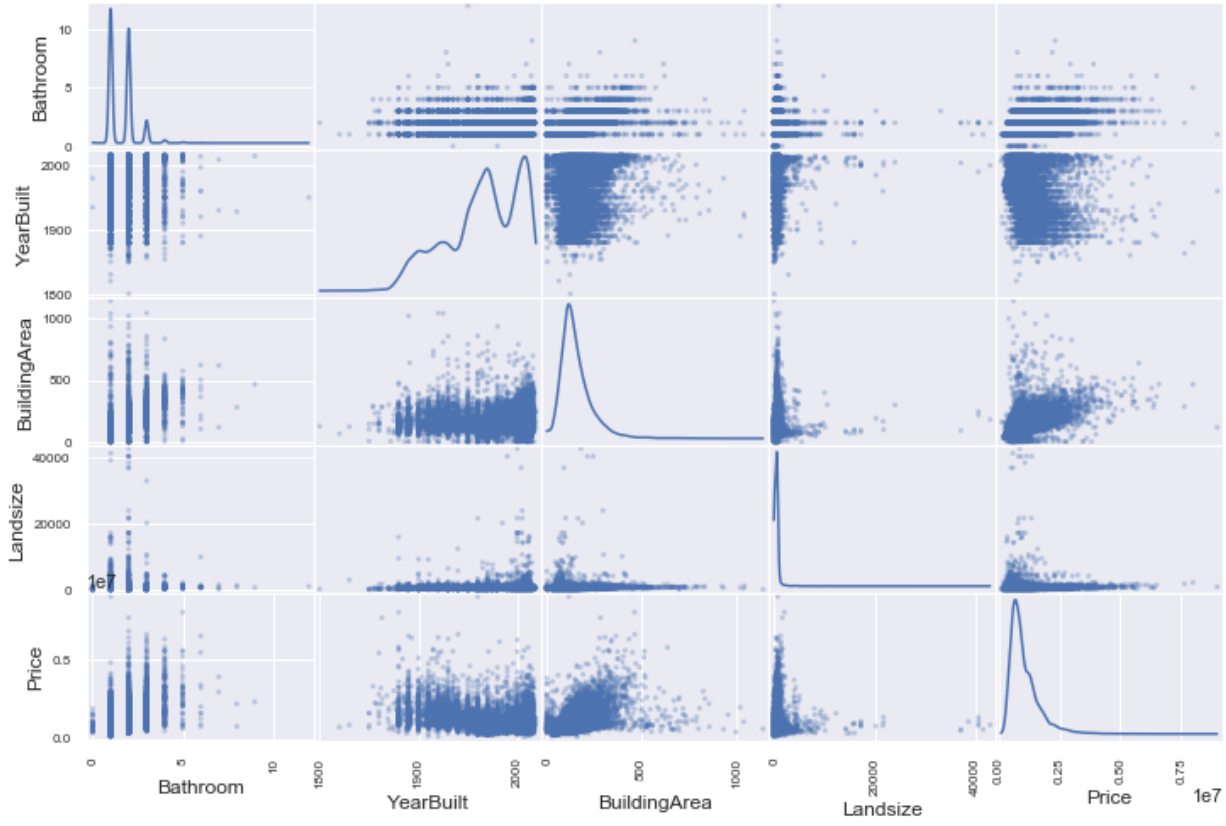
II. Analysis

Data Exploration

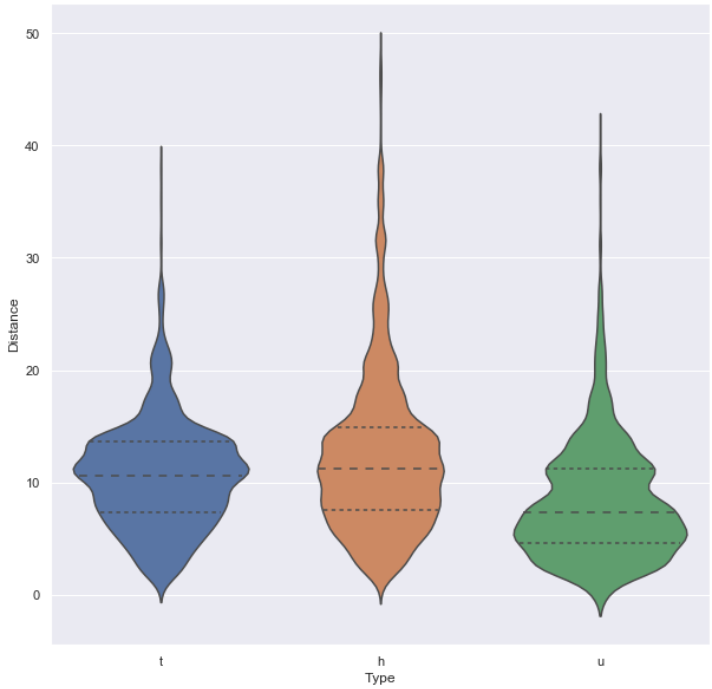
The data is the Melbourne Housing Market dataset from Kaggle. Some observations had erroneously-entered values, which I cleaned out in Excel. Some suburbs only had one observation but I felt that omitting them would make the problem unrealistically easy and the dataset unrealistic.

Exploratory Visualization

Right away I created a few correlograms to help me identify outliers and show the relationship between multiple variables. This is an essential part of data cleaning because abnormal points will show up on the graph. Then, as you delete them, new versions of your graph will start to show the "real" pattern you were looking for. It may also tell you that certain features need be log-transformed.



Seaborn's violinplot() function basically combines a boxplot with kernel density estimate, and helps me clearly see the distribution of different classes of the same feature. Incredibly useful for any data scientist doing EDA.



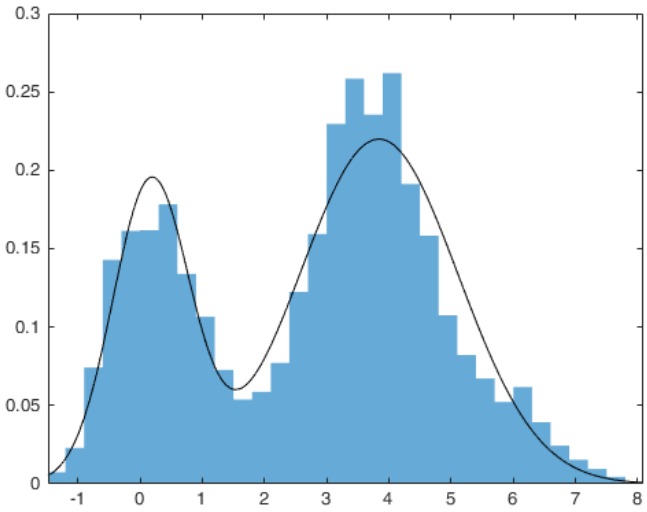
I. Apartment units were generally the most affordable. Detached houses exhibited the most variation.

Algorithms and Techniques

I benchmarked my clusters with Hierarchical Clustering. This was a great benchmark because Hierarchical Clustering is better with datasets with a small number of observations (I had a little over 300 suburbs listed). However, I'm interested in the Gaussian Mixture Model because it allows us to fit data to a variety of shapes and gives a greater degree of flexibility and interpretation.

The Gaussian Mixture Model assume multiple normal distributions are generating the data. It's typically used when your data presents more than one "hump" in kernel density. For this data, I noticed a lot of outliers, but there were too many for me to omit. So I assumed there was a 'double-peaked' distribution for a lot of these features: one distribution for the common observations at the bottom, and another for the rarer observations at the top of the range.

One oft-overlooked advantage of mixture models is that it can, in some cases, help you avoid log-transforming a feature. I think many beginning data scientists will see a group of outliers and think, "I should omit them," or "I should log-transform this column," but, in reality, they're just part of another distribution that was generating the data.



I benchmarked the city prediction with a K-Nearest Neighbors model. I'm particularly interested in KNN because I think of it as an intermediary step between clustering and predictive machine learning. Right away it tells us, "How good of a prediction can we make by just associating an observation with its closest neighbors in p-dimensional space?"

However, I used a Boosted Decision Tree classifier to improve over the benchmark. To improve my prediction quality, I tuned with GridSearchCV and tested a combination of different parameters (max_depth, learning_rate, gamma).

III. Methodology

Data Preprocessing

The dataset from Kaggle was fairly clean. However, some values were incorrectly entered. To find abnormal values, first I created a pairplot with pd.plotting.scatterplot(df['features']). Right away this showed bad distributions or outlying points. Often I could use df['feature'].max() or .min() to find a bad value.

Notably, since this data is publicly available from real estate sites, I could often directly find a house by entering its address or square footage in Google. Pulling up a webpage with photographs and details of the house was useful in deciding if the observation was relevant to analysis. If a house is listed with a 36-car garage, I can google it and find out it's not a house and should not be in the dataset. Some houses had lot sizes abnormally large for a residential neighborhood, so I could look them up and find out they were actually rural areas, too far from town to be considered a Melbourne neighborhood

Two features I thought to log-transform were Land Size and Building Area, but chose not to. 2500 observations, mostly of type "unit" (apartment/condominium) were given a Land Size of 0. This is unfortunately not realistic, but the 'Landsize' column proved influential in Principal Components Analysis and helpful in both clustering and decision tree modeling.

IV. Results

Model Evaluation and Validation

For Gaussian Mixture Model clustering, I achieved a silhouette score of 35% for 4 clusters. This was an improvement over the Hierarchical benchmark's score of 34% for 4 clusters.

Cluster Analysis:

Cluster 1 is overwhelmingly single-family detached housing. It's also the most affordable. Because of its longer distance from the city center, it appears to be semi-rural neighborhoods or what we'd call in America "exurbs". These are smaller, newer homes built on large lots. They probably contain more blue-collar families.

Cluster 2 is significantly closer to the city center and older than Cluster 1. It's majority single-family detached and appears to show a much more even mix of housing types, with the largest amount of "townhouse" units. I'm guessing this cluster shows what we call "inner-ring" suburbs in America. This mix appears to comprise the most desirable neighborhoods because this cluster

Cluster 3 is the closest to the city center. It contains by far the oldest and smallest units, and exhibits the most even distribution of housing types. No data was available on age of the residents, but I'd assume this cluster contains many college students, young blue-collar workers living together or single, with an occasional luxury high-rise unit, or detached house in-town by the beach. Notably, some of the most expensive units in the dataset were luxury houses close to the city center.

Cluster 4 only has 18 observations which are all single-family detached homes. These exhibit the features of rural, agricultural homes. The average lot size and house size is extremely large.

Cluster Conclusions:

It seems that the most valuable communities here have a mixture of apartments, houses and townhouses. It would probably be beneficial to a developer to build a mixture of housing types in a neighborhood.

In America, it's usually the case that older homes contain fewer bathrooms, but in Australia that doesn't seem to be the case. Bathroom did not show any linear trend with price or building age in my graphs above.

For Boosted Decision Tree prediction, I achieved an F1 score of 76.5%. This was a stark improvement over the K-Nearest Neighbors benchmark of 20.3%.

V. Conclusion

Personally, this project taught me the difference between "data science" and "statistics". I understood a lot of the models involved, but reporting on *why* and *how* I did everything made me rethink some methods.

My most important conclusion from this project was that we should examine a lot of features you might see in the most desirable neighborhoods, and rebuild them in other neighborhoods. This would take pressure off the center city and even demand across the metro area. A more even mix of housing types may help people avoid speculation, but, as a side benefit, would encourage walkability and non-car-dependent modes of transport. It would make infrastructure, like transit, feasible to run to more neighborhoods, and avoid one neighborhood spiking up in value while another plummets.

Improvement

Anyone could delete the suburbs here with only one observation and instantly see an increase in performance of every model. I chose to leave them in because I wanted to simulate real-world data that you might find from a municipal site (such as an OpenData program).

Additionally, the Gaussian Mixture Model has a host of interesting visualization and built-in output functions that can relay more information about one's results. I chose not to use these and use silhouette scoring, because these metrics aren't available for other types of models (such as KMeans or Hierarchical Clustering). I wouldn't be able to compare to a benchmark.

Importantly, my model produced 4 clusters, but only 3 were meaningful (the 3rd was only 9 observations large). However, I chose to retain this model because the values of the silhouette scores were good and the clusters were similar.

Python (especially scikit-learn) is limited in its graph capabilities. R and MATLAB both have extensive built-in graphing tools that far exceed Python's, and in MATLAB you can easily create 3D graphs. 3D graphs are especially useful for PCA and clustering, because a lot of clusters are not visible in 2 dimensions. Such was the case in my data, where the 2-dimensional graphs didn't show real clusters, even though silhouette scores told us clusters emerged.