CS 760

HOUSING MARKETS PREDICTING WILDFIRES

Noah Cohen Kalafut
Computer Science Doctoral Student
University of Wisconsin-Madison
nkalafut@wisc.edu
https://github.com/Oafish1/CSC-714

December 13, 2020

1 Abstract

In this project, rental agreements and housing arrangements are used as predictors for the wildfire risk in a certain area. Of course, changing the terms of a rental agreement will not directly cause or prevent wildfires. However, housing markets are good representatives of urban development and economic status. An analysis of this nature could shed some light on the relationship between business, governmental, and environmental changes.

2 Introduction

In 2020, wildfires have become a real problem, especially for California. In fact, 2020 has been the largest recorded wildfire season for California [1]. Having long been the wildfire capital of the US, California is also home to some of the most diverse incomes in all of the country [2]. These factors make California a good subject for studies on urban development's impact on the local environment.

We are reaching a critical point in determining the future of our planet. Although it may not directly point to the most influential variables at play, an analysis tying housing markets to environmental consequences might be able to point future research and policy in the right direction. The aim of this project is to tie these two factors together and hopefully extract some useful insights into their relationship.

3 Related Work

Both [3] and [4] review the effects of environmental policy on the housing market. Interestingly, both cite the lack of research within this area. This is true even today, despite both papers being 15 years old.

There seem to be no pieces of literature examining the exact relationship of this project. This means that the paper will either be revolutionary or we will find out why nobody is examining this exact relationship.

4 Dataset

This project merges two datasets. One concerns California wildfires from 2013 until the end of 2019. The other scrapes home rental data from Craigslist. In merging them, we can see housing statistics along with wildfire prevalence.

4.1 Housing

The Kaggle dataset [5] consists of $384,977$ housing listings pulled from Craigslist.

Cat or dog allowance, smoking rules, wheelchair accessibility, electric car charging stations, and furnishings are all tracked in a binary format. Region, housing type, laundry services, and parking are all stored as categorical variables.

Lastly, beds, baths, square footage, and rent are stored in their original numerical formats. All of these variables will be used for our analysis.

There are also links to the original listings, but it seems that Craigslist has since had some renovations, meaning that the links are unusable. It is worth noting that there are some duplicate entries – likely caused by the re-listing of a home once a renter moves out or after the home has been on the market too long. The dataset was last updated in January of 2020 and appears to contain all listings that were visible on the site at the time. For our purposes, we want to use listings from California.

4.2 Wildfires

[6] contains information about $1,636$ California wildfires from the beginning of 2013 to the end of 2019. The data is pulled from the California Department of Fire Protection's archive [1].

For this project, we're focusing on the average number of wildfires per county, per year, as an estimation of wildfire risk. However, the dataset also includes numbers on land damaged, resources (including helicopters, fire engines, bulldozers, etc.) deployed, deaths and injuries, and infrastructure damage (in number of buildings) in numerical format. The primary managerial body, public statements, and timestamps are also included as categorical, text, and date variables, respectively.

5 Approach

All of the code used in this project was created for use in this class. Most are repurposed from earlier homework assignments and have been modified to fit our application. The lecture notes from *Daniel L. Pimentel-Alarcón* concerning CS 760 are heavily used in sections 5.2.1 and 5.3.1.

5.1 Preprocessing

Before anything else, we need to merge our two datasets into one. This may seem quite simple, as both of our datasets contain location data. However, our location data is in very different formats for each. So, we need to manually map each Craigslist region from [5] to a corresponding county/wildfire statistic in [6]. Luckily, we have enough data such that we can create this correspondence. For California, there are $33,085$ entries. Two regions in [5] had no equivalent, however. These regions were *Imperial County* and *SF Bay Area*. The former was excluded due to no wildfire data being available, despite wildfires occurring in the area during the time in question. The latter was excluded for containing too many regions to reliably attribute to one county. The two datasets could then be joined by county, leaving $30,666$ datapoints.

Next, there is a mix of numeric, categorical, ordinal, and binary variables in [5]. Numeric and binary variables can be kept as they are, numerical values modified only to fit with the schema discussed later in this section. Categorical and ordinal variables are stored as text in our dataset. So, a key can be created that links each possible value to an index. For example, we have a variable *laundry_options* that has indexes from 0-5, representing no listing, no laundry services, on-site laundry, in-building laundry, washer-dryer hookups, and washer-dryer preinstalled, respectively.

We could also perform similar processing with [6], but we only cared to extract a single numerical value, average wildfires per year by county – which is obtained by counting the total number of stored wildfires for a particular county and dividing by 7, which is the number of years for which the dataset has information.

To summarize, this preprocessing leaves us with $30,666$ datapoints on 14 variables

Wildfires per Year	Numeric	Dogs Allowed	Binary
Price	Numeric	Smoking Allowed	Binary
Housing Type	Categorical	Wheelchair Accessible	Binary
Square Footage	Numeric	Electric Vehicle Charging	Binary
Beds	Numeric	Pre-furnished	Binary
Baths	Numeric	Laundry Services	Ordinal
Cats Allowed	Binary	Parking Services	Categorical

5.2 Naive Bayes

There are a few large limitations to the accuracy of the classifier. For one, Naive Bayes is a classifier. Because of this, predicting continuous values is outside its scope, meaning that our dataset has to have the proper resolution of output figures while also having enough samples for any **exact** output value. For instance, if one county had 4 wildfires a year, while another had 3.99, the predictions for $Y = 4$ and $Y = 3.99$ would use completely different subsections of the data.

In our case, we have a significant number of data points. Additionally, we have several distinct Y. So, we can approach with regular Naive Bayes rather than performing any modifications that might necessitate combining different data for Y – as might be done for continuous Y. However, if we perform the analysis in this manner, we might not be testing for wildfire prevalence so much as identification of a county by renter's agreement. So, we can instead group our wildfire data into high, medium, and low risk, split evenly by percentage rather than percentile so that jumps in risk are intuitive. To see why this is done, imagine one area has 20 wildfires per year and the next highest has 10. We would not want these in the same group because their values are so different, despite the fact that they are effectively as close as possible in percentile.

Upon seeing the process of creating a Naive Bayes classifier, you may conclude that it is not a good choice for predicting our outcome variable, due to its underlying assumption of independence between input variables – which makes it an unreliable technique for predicting/formulating aggregate measures, such as urbanization and economic development. However, this does not mean it is useless. Based on the quality of the Naive Bayes classifier on our dataset, we might be able to gain a better understanding of the independence, or lack thereof, of our variables when comparing to another classifier. For example, if a linear model were to classify our data well, but Naive Bayes was to fail, we could say that our most important variables are almost certainly not independent with respect to our outcome.

For our purposes, we can say that binary variables follow a Bernoulli distribution, numeric variables follow a Gaussian distribution, and categorical/ordinal variables follow a Multinomial distribution. In our case, this looks as follows:

Price	Gaussian
Housing Type	Multinomial
Square Footage	Gaussian
Beds	Gaussian
Baths	Gaussian
Cats Allowed	Bernoulli
Dogs Allowed	Bernoulli
Smoking Allowed	Bernoulli
Wheelchair Accessible	Bernoulli
Electric Vehicle Charging	Bernoulli
Pre-furnished	Bernoulli
Laundry Services	Multinomial
Parking Services	Multinomial

5.2.1 Theory

Naive Bayes starts with trying to maximize

$$\mathbb{P}(y|x)$$

for y.

Using Bayes' law, we can see

$$\begin{aligned} &\arg\max_y \quad \mathbb{P}(y|x) \\ =&\arg\max_y \quad \frac{\mathbb{P}(x \cap y)}{\mathbb{P}(x)} \\ =&\arg\max_y \quad \mathbb{P}(x|y)\mathbb{P}(y) \end{aligned}$$

Now we assume independence for features x_i and x_j where $i \neq j$

$$\arg\max_y \quad \mathbb{P}(y) \prod_i \mathbb{P}(x_i|y)$$

$\mathbb{P}(y)$ is straightforward to estimate for any given dataset. We can simply count the number of y corresponding to the given value and divide it by the total number of samples.

Calculating $\mathbb{P}(x_i|y)$ is more involved.

5.2.2 Bernoulli Variables

For Bernoulli variables, by Bayes' law, we can say

$$\mathbb{P}(x_i|y) = \frac{\mathbb{P}(x_i \cap y)}{\mathbb{P}(y)}$$

We can determine $P(x_i|y)$ by simply counting the instances where x_i and y match our given values and dividing by the instances where y matches our given value.

5.2.3 Gaussian Variables

For Gaussian variables we can calculate $P(x_i|y)$ using the PDF of the normal distribution

$$P(x_i|y) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(\mathbf{x}-\mu)^2}{2\sigma^2}}$$

for a given $\mathbf{x}$ and μ, σ^2 where y matches our given value.

5.2.4 Multinomial Variables

Finally, for Multinomial variables, we can approximate $P(x_i|y)$ using

$$\mathbb{P}(x_i|y) = \frac{\mathbb{P}(x_i \cap y)}{\mathbb{P}(y)} \approx \frac{1 + \sum \mathbb{1}_{x=\mathbf{x},y=\mathbf{y}}}{K + \sum \mathbb{1}_{y=\mathbf{y}}}$$

Notice that we add 1 to the numerator and K, the number of possible values for x_i, to the denominator. This is known as Laplace smoothing and is used to prevent our product $\mathbb{P}(y|x)$ from going to 0 if our dataset contains no information on a certain variable while also having a minimal effect on the accuracy of the approximation.

5.3 Forest

Much like our Naive Bayes implementation, a decision tree is a classifier. So, our same restrictions will apply. We will group our wildfire risk assessments into groups of high, medium, and low according to percentage. In addition, we will split our data into exclusively binary variables to simplify the problem. All numeric variables will be split by their mean, binary variables require no change, and ordinal/categorical variables will be changed to availability. For example, *Laundry Services* will be 1 if a washer/dryer is available on-site. The one exception to this will be *Housing Type*, which will document whether or not the home is an apartment.

Price	$> 2,893$	Smoking Allowed	$> .5$
Housing Type	Apartment	Wheelchair Accessible	$> .5$
Square Footage	$> 1,067$	Electric Vehicle Charging	$> .5$
Beds	> 1	Pre-furnished	$> .5$
Baths	> 1	Laundry Services	On-site
Cats Allowed	$> .5$	Parking Services	On-site
Dogs Allowed	$> .5$		

5.3.1 Theory

Given our data, we can calculate entropy/new information as

$$\begin{aligned} H(x_i) &= \mathbb{E}\left(\frac{1}{\log_2 \mathbb{P}(x_i)}\right) \\ &= \sum_{\mathbf{x}} \mathbb{P}(x_i = \mathbf{x}) \left(\frac{1}{\log_2 \mathbb{P}(x_i = \mathbf{x})}\right) \end{aligned}$$

We can calculate conditional entropy as

$$\begin{aligned} H(x_i|y) &= \mathbb{E}\left(\frac{1}{\log_2 \mathbb{P}(x_i|y)}\right) \\ &= \sum_{\mathbf{y}} \sum_{\mathbf{x}} \mathbb{P}(x_i = \mathbf{x}, y = \mathbf{y}) \left(\frac{1}{\log_2 \mathbb{P}(x_i = \mathbf{x}|y = \mathbf{y})}\right) \\ &= \sum_{\mathbf{y}} \mathbb{P}(y = \mathbf{y}) \sum_{\mathbf{x}} \mathbb{P}(x_i = \mathbf{x}|y = \mathbf{y}) \left(\frac{1}{\log_2 \mathbb{P}(x_i = \mathbf{x}|y = \mathbf{y})}\right) \\ &= \sum_{\mathbf{y}} \mathbb{P}(y = \mathbf{y}) H(x_i|y = \mathbf{y}) \end{aligned}$$

If we want to calculate the amount of information shared by variables x_i, y, we can designate

$$I(x_i, y) = H(x_i) - H(x_i|y)$$

In the context of decision trees, this is known as information gain. We can use $I(x_i, y)$ to decide which variable x_i has the most information concerning our target variable y. Splitting on this variable is the basis for creating a decision tree.

In practice, determining $\mathbb{P}(x_i|y)$ for some x_i, y can be done through counting, as was done for the Naive Bayes method.

For the actual creation of the tree, information gain $H(x_i, y)$ is calculated for each unused variable x_i with above some predefined number of samples. Then, the variable with the highest information gain, if above a certain threshold, is used to split the data into two subsets. This process repeats until the tree can split no longer. This is performed on all subsets of the data.

At this time, the majority value of the output at each leaf is said to be its prediction. This completes the creation of a decision tree.

A forest is several decision trees merged together, where the most common answer among the trees is taken as the final result. An element of randomness/change needs to be present during the training of each tree. Otherwise, the forest would contain the same tree multiple times. Two methods of generating this randomness will be used in this project. One involves training each tree on a random subset of the data. The other involves training each tree with one variable excluded. The former will henceforth be referred to as a **random forest**, the latter will be an **elimination forest**.

The methodology by which these trees are created requires more explanation. In both cases, 80 percent of the source data is randomly selected to train a forest. The other 20 percent is used to assess its accuracy. Then, the forest with the maximal validation accuracy is selected. The accuracy statistic used from now on will represent the accuracy of the best forest on the whole dataset.

6 Results

6.1 Naive Bayes

Testing on the whole dataset, the Naive Bayes classifier has very poor performance. Although it does trend toward the proper assessment, the classification accuracy is only high for specific outcomes. The following table demonstrates this fact.

Statistic	Low Risk	Medium Risk	High Risk
Average Output	0.9998	0.9993	1.0003
Classification Accuracy	0.0007	0.9987	0.0008

(1)

If we take an equal number of samples for each $\mathbf{y}$ – as is outlined in the next section, we obtain

Statistic	Low Risk	Medium Risk	High Risk
Average Output	1.6893	1.6797	1.8337
Classification Accuracy	0.0083	0.3177	0.8373

This can lead to one or both of two conclusions. It could be that our input data contains little information about wildfire risk. It could also be the case that our assumption of independence between variables is fatally flawed. In fact, we know that our assumption is untrue – higher prices generally correlate with greater square footage. This could be detrimental to the accuracy of the classification, causing 'double counting' and providing lower probabilities than may be realistic.

We can pull random examples for low and high risk agreements,

Statistic	Low Risk	High Risk
Price	$4,500$	$1,475$
Housing Type	House	Apartment
Square Footage	$6,600$	916
Beds	5	2
Baths	6	5
Cats Allowed	0	1
Dogs Allowed	0	1
Smoking Allowed	0	1
Wheelchair Accessible	1	0
Electric Vehicle Charging	0	0
Pre-furnished	1	0
Laundry Services	W/D Hookup	On-site
Parking Services	Attached Garage	Carport

These are both outliers in their own right. From these examples, it might appear that dense housing is more likely to be in a fire-prone area while larger housing is more likely to be more secure. It is likely that the high-risk listing is counting public amenities.

6.2 Forest

This section will be split into analyses of the random forest and elimination forest implementations.

6.2.1 Random Forest

Upon testing the random forest on the whole dataset, the accuracy was almost .7!. However, this seemed very suspicious. The prevalence of 0s within the original data could have skewed the measurement. With the following MATLAB code, 9000 random samples can be taken, 3000 for each y

```
% Random Subset
Y_unique=unique(Y);sub=[];
for i=1:size(Y_unique,2)
    f=find(Y==Y_unique(i));
    sub=[sub f(randperm(size(f,2),3000))];
end
% Shuffle
sub(randperm(size(sub,2),size(sub,2)))=sub;
% Implement
Y=Y(sub);
X=X(:,sub);
```

Training and evaluating on this newly balanced dataset allows us to evaluate the forest while giving equal weight to each variable in the measurement.

After this adjustment, an accuracy of .45 could be obtained – which is better than random chance, but worse than desirable. Additionally, the trees were in desperate need of pruning. While maintaining a reasonable degree of accuracy, with $\frac{1}{9}$ as the minimum amount of data remaining to split and .02 as the minimal information gain, the following forest was produced with the accuracy calculated on the new subset of the data[1]

```
SMO(ELE(DOG(1,SQU(1,1)),0),LAU(BAT(1,2),DOG(BED(0,1),SQU(ELE(2,0),1))))
SMO(CAT(1,PRI(SQU(1,1),0)),LAU(BAT(1,2),DOG(BAT(1,1),SQU(ELE(1,0),1))))
SMO(PRI(PAR(0,WHE(CAT(1,1),1)),0),LAU(PRI(PAR(BAT(1,2),0),2),DOG(BAT(1,1),
    SQU(ELE(PRI(1,0),0),WHE(1,1)))))
SMO(CAT(1,FUR(WHE(1,1),2)),LAU(BAT(1,2),DOG(BED(1,1),
    SQU(ELE(PRI(2,0),0),1))))
SMO(PRI(PAR(0,WHE(CAT(1,1),1)),0),BED(ELE(PRI(1,0),0),LAU(2,DOG(1,
    SQU(BAT(1,2),BAT(0,1))))))
Accuracy: 0.3990
```

Above 3, the number of decision trees did not affect the accuracy a significant amount.

6.2.2 Elimination Forest

The elimination trees performed much the same. The number of trees was a problem, as evidenced by an extreme number of duplicate trees when the thresholds were set even slightly too high. Take this example of a tree with $\frac{1}{9}$ as the minimum amount of data remaining to split and .0155 as the minimal information gain

```
SMO(1,1)
DOG(1,FUR(BED(1,2),CAT(BED(1,1),PRI(1,0))))
FUR(LAU(TYP(PRI(1,0),2),1),DOG(BAT(1,1),PRI(1,0)))
DOG(1,FUR(BAT(1,2),CAT(1,PRI(BED(WHE(1,0),BAT(1,SMO(LAU(1,1),1))),0))))
```

[1] The formatting is such that `A(a,b)` indicates a split on variable A. If $A = 0$, a will be returned/is at the leaf. Otherwise, b will be returned.

```
SMO(1,FUR(BAT(1,2),DOG(BED(1,1),PRI(BED(1,WHE(BAT(1,LAU(1,1)),1)),0))))
DOG(1,FUR(BED(1,2),CAT(SQU(1,1),SQU(WHE(1,0),SMO(LAU(1,BED(1,1)),1)))))
SMO(1,LAU(BAT(1,2),DOG(BED(1,1),PRI(BED(ELE(1,0),WHE(BAT(1,1),1)),0))))
DOG(1,FUR(BED(1,2),CAT(SQU(1,1),PRI(SQU(WHE(1,0),SMO(BED(1,LAU(1,1)),1)),0))))x5
DOG(1,FUR(LAU(TYP(PRI(1,0),2),1),CAT(BED(1,1),PRI(BED(WHE(1,0),SMO(LAU(1,1),1)),0))))
Accuracy: 0.3827
```

The duplicates suggest that some of our input variables are fairly unhelpful in predicting our output.

7 Conclusions and Future Work

Although our classifiers were not as high-accuracy as might be desired, we can still extract useful information from these runs. When forests would have thresholds set too high, two variables would often appear: *Smoking* and *dogs*. *Pre-furnished* also appeared a decent amount, but without as much frequency. In particular, *smoking* and *dogs* tended to be associated with a greater risk of wildfire, while *pre-furnished* had the opposite relation.

Empirically, *smoking* and *dogs* were also more divisive in the Naive Bayes classifier, often producing probabilities for each classification $\mathbb{P}(x_i, y = \mathbf{y})$ that could vary by up to 200 percent for any given $\mathbf{y}$.

Combining the slightly greater accuracy in our forests with our outcome for Naive Bayes, we can also say our input variables are probably inter-related, but that it does not affect our results in a major way. The difference in accuracy can be attributed to the lack of assuming independence.

Running a single tree on the data with no thresholding should provide the maximal accuracy for the dataset. In this case, only $.51$ accuracy is obtained. This points to a lack of information contained within the dataset itself. It is worth noting that transforming the set into all binary values almost certainly also has something to do with the loss in accuracy. However, $.51$ is very low, even with the lost nuance.

In future implementations, it would be interesting to see how a tree working with the raw data would perform. On a similar vein, future implementations should use larger datasets containing more variables about the topic at hand. There may be a link between housing economies and wildfires, but no definitive, substantial link can be totally confirmed by this project.

All code and the modified dataset can be found on GitHub.

References

[1] California Department of Forestry and Fire Protection. 2020 incident archive. `https://www.fire.ca.gov/incidents/2020/`.

[2] U.S. Census Bureau. Gini index of income inequality (b19083).

[3] Katherine A. Kiel. Environmental regulations and the housing market: A review of the literature. *Cityscape*, 8(1):187–207, 2005.

[4] David L. Sunding. The economics of environmental regulations of housing development. *Housing and Society*, 32:23–38, 2005.

[5] Austin Reese. Usa housing listings. `https://www.kaggle.com/austinreese/usa-housing-listings/version/3`.

[6] ARES. California wildfire incidents (2013-2020). `https://www.kaggle.com/ananthu017/california-wildfire-incidents-20132020/version/1`.