**Details**

Here I (re)estimated a series of models for low-risk fires, fire size, injuries and deaths.

Predictors used in the model were organized into subgroups to simplify model selection for most models. The groups are summarized below.

**Time**: Year, with 2014 being year zero, was included in the analysis to allow for risk to vary over time.

**Base**: The base variables were population, number of males, and total number of housing units.

**Race**: includes a number of racial and ethnic categories.

**Age**: includes the age distribution of the population.

**House**: Includes a set of housing characteristics, including number of vacant houses, number of renters, number of households with more people than rooms, *number of 1 and two unit residences (“single-family homes”),* number of units that are part of a 10-unit or more block, number of units built before 1980, and number of mobile homes.

**Personal**: A list of personal and household characteristics: median household income, Social Vulnerability Index, *number of married people, number of unemployed, number of those not in the labor force,* and percent of adult smokers (determined at both the state and county level).

**Fuel**: Type of fuel used for heating.

***Data***

I defined LOW\_RISK fires as discussed in our earlier emails. That primarily means single-family residences (type 419 from NFIRS). The data used for this analysis is generated using a series of queries which are included as Appendix I.

Geocoding is incomplete, and varies by department and year. This model adjusts for geocoding percentage at the department × year level, by including the geocoding percentage of all reported incidents (by department and year) as an offset to the model.

As discussed, I used the data from 2007 to 2013 to estimate the model. The following filters were also used. Precise definitions of the filters are included as Appendix II.

*base*

I excluded any tract with any of the following characteristics:

* SVI < 0
* No reported median income
* No reported department size
* No reported County smoking data

*small.x*

The definition of *small* varied depending on the model. Specifics of the various *small* filters is listed in Table 1. All models excluded department × years where the department reported responding to fewer than 25 *incidents* that year. Since there was some discussion regarding what the appropriate *small* filter should be, I ran several different versions of it. I ran models where the floor was 50, 100, 150, 200, 250, 500, and 1000 incidents that year. I also filtered out “outlier” years. “Outlier” years were defined as those years which fell at least 2 standard deviations below the mean number of incidents for the department. A specific definition is included in Appendix II. Note that the “two standard deviations” standard is based on at most 7 years per department of data reported as part of the study.

Table 1: *Small* Filters: Specific Definitions.

|  |  |  |
| --- | --- | --- |
| Filter | Floor | Outlier |
| small.0 | 25 |  |
| small.1 | 50 |  |
| small.2 | 100 |  |
| small.3 | 150 |  |
| small.4 | 200 |  |
| small.5 | 250 |  |
| small.6 | 500 |  |
| small.7 | 1000 |  |
| small.0a | 25 | x |
| small.1a | 50 | x |
| small.2a | 100 | x |
| small.3a | 150 | x |
| small.4a | 200 | x |
| small.5a | 250 | x |
| small.6a | 500 | x |
| small.7a | 1000 | x |

*giants*

The definitions of the *giants* filter is unchanged from before. Specific definition is included as part of Appendix II.

*random\_subset*

One third of the tracts are set aside to serve as a test set, while the remaining tracts are used as the training set.

***Department Size × Region***

As before, I excluded all departments serving fewer than 10,000 people (sizes 0 – 2). All departments nationwide serving 1 million or more people were analyzed together. For some models (more details below) departments in the Northeast in size range 8 were combined with those in the size range 7.

***Models***

All models were estimated using the techniques of generalized linear models (glm), and had the following basic form:

|  |  |  |
| --- | --- | --- |
|  |  | (1) |

Where *yij* is the dependent variable being analyzed for the *i*th tract served by the *j*th department in year *t*, *g* is a linking function used in the glm analysis, *aijt* is a *known* offset value for the model, *xijt* are the predictors used for that observation, *βt* is the change per year, *β* are the parameter estimates, *ηj* is the department effect.

It is further assumed that

|  |  |  |
| --- | --- | --- |
|  |  | (2) |

And what is reported in the model is . Specific values for the individual departments can be provided.

For the constant model, the department dummies were not included. For the department effects model, only the department effects (and a constant) were included.

It is assumed that the number of fires, injuries or deaths in a tract follow a (overdispersed) Poisson process. An offset term is used to reflect the proportion of incidents that were geolocated by department. That should convert these estimates, based on geolocated fires, injuries or deaths, into a model of total fires, injuries or deaths.

For fire size, it is assumed that they follow a binomial model. No offset was used for the fire size models because the number of relevant fires was already included as part of the model.

A large number of different models were run. They fall into two main classes. In the first group, I effectively estimated separate models for each region × department size group of departments. This is the same as I did before. In the second group, I just included dummy variables representing department size and region. This tested the effectiveness of a more parsimonious model.

The setup and functions used in computing and analyzing the models are described in Appendix III.

***Results***

All models are estimated against the training set. Then the estimated model is used to predict number of fires (or percent of fires for fire size, or injuries or deaths, depending on the model) for each tract in the test set. Then for each model the Mean Square Error (MSE) of the predicted value is calculated for the test set. Note that all models (except the constant model) reported here include a department dummy as part of the model estimate. All “Dummies” models also include dummies for department size and region, in addition to the variables listed. There are a few tracts in the test set associated with departments which do not appear in the training set and, therefore, for which no department dummy could be estimated. In those cases, the department dummy for that department was arbitrarily assigned a value of zero.

For comparison, I include three naïve models that estimated number of fires for the test set. The first is a constant model. It assumes that all tracts have the same (average) expected value per year. The second is a pure department-effect model. It assumes that all tracts served by a particular department have the same expected value per year. The third is a tract model that predicts that the value for a tract would be the same as the value for that tract experienced in the previous year. Since my data set did not include data for 2006, no estimate was made for 2007 for the tract model. Again, specific definition of the tract model is included in Appendix III.

In all cases below, I effectively restricted the test set to departments serving 100,000 people or more. Since much of the model testing below involved different definitions of *small*, I wanted a test set that used a criterion different from the various *small* filters. The restriction based on department size seemed the best way of determining what effect the various *small* filters had on predictions for the departments that will most likely be using the FireCARES system (at least at first). Note that the test set against which MSE was computed stayed the same regardless of the definition of the *small* filter used. That enabled me to compare “apples to apples,” when comparing the effectiveness of the various models.

Results are reported in Table 2 through Table 6, below.

Comparing the results to the averages per tract in the test set indicates that the models for fires, injuries and deaths are still over-dispersed relative to a Poisson model.

Overfitting of the models is apparent for all models except those for fires. It becomes severe for deaths.

For the most part, the models compare well to the naïve models. As a rule, about half the reduction in variance is attributable to the department effect. The TRACT naïve model beats the model for fires, but does poorly for all the other models. Remember that the tract ID contains a lot of information, including department ID and all the census data. That is why it performs so well against the fire models. However, once the event being predicted becomes rare (e.g., ~ 88 % of the time there are zero injuries in a tract), then the tract label contains only a little information about occurrence. So, pooling information about similar tracts (which is essentially what the models do) improves the prediction.

Table 2: Mean Square Errors of the Fire models tested.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | **Small Filter** | | **Mean Fires** | **MSE** | |
| **Model Run** | **Predictors** | **Floor** | **Outlier** | **Separate** | **Dummies** |
| Constant |  | 25 | X | 2.1228 | 2.9559 | |
| dept.effect |  | 25 | X | 2.1228 | 2.8447 | |
| tract |  |  |  | 2.1228 | 2.5570 | |
| lasso.min | time, base, race, age, house, personal, fuel | 25 | X | 2.1228 | 2.6180 | |
| lasso.1se | time, base, race, age, house, personal, fuel | 25 | X | 2.1228 | 2.6526 | |
| rForest | time, base, race, age, house, personal, fuel | 25 | X | 2.1228 | **2.3859** | |
| base | time, base, race, age, house, personal, fuel | 25 | X | 2.1228 | 2.6641 | 2.6191 |
| M101111 | time, base, age, house, personal, fuel | 25 | X | 2.1228 | 2.6383 | 2.6244 |
| M100111 | time, base, house, personal, fuel | 25 | X | 2.1228 | **2.6046** | 2.6176 |
| M101011 | time, base, age, personal, fuel | 25 | X | 2.1228 | 2.7534 | 2.7888 |
| M101101 | time, base, age, house, fuel | 25 | X | 2.1228 | 2.6115 | 2.6290 |
| M101110 | time, base, age, house, personal | 25 | X | 2.1228 | 2.6392 | 2.6186 |
| M110111 | time, base, race, house, personal, fuel | 25 | X | 2.1228 | 2.6322 | 2.6271 |
| M110011 | time, base, race, personal, fuel | 25 | X | 2.1228 | 2.7421 | 2.7856 |
| M110101 | time, base, race, house, fuel | 25 | X | 2.1228 | 2.6474 | 2.6435 |
| M110110 | time, base, race, house, personal | 25 | X | 2.1228 | 2.6353 | 2.6189 |
| M111011 | time, base, race, age, personal, fuel | 25 | X | 2.1228 | 2.7735 | 2.7504 |
| M111001 | time, base, race, age, fuel | 25 | X | 2.1228 | 2.8366 | 2.8048 |
| M111010 | time, base, race, age, personal | 25 | X | 2.1228 | 2.7798 | 2.7541 |
| M111101 | time, base, race, age, house, fuel | 25 | X | 2.1228 | 2.6579 | 2.6322 |
| M111100 | time, base, race, age, house | 25 | X | 2.1228 | 2.6626 | 2.6242 |
| M111110 | time, base, race, age, house, personal | 25 | X | 2.1228 | 2.6635 | 2.6130 |
| f.050 | time, base, house, personal, fuel | 50 | X | 2.1228 | 2.6055 |  |
| f.100 | time, base, house, personal, fuel | 100 | X | 2.1228 | 2.6060 |  |
| f.150 | time, base, house, personal, fuel | 150 | X | 2.1228 | 2.6083 |  |
| f.200 | time, base, house, personal, fuel | 200 | X | 2.1228 | 2.6119 |  |
| f.250 | time, base, house, personal, fuel | 250 | X | 2.1228 | 2.6260 |  |
| f.500 | time, base, house, personal, fuel | 500 | X | 2.1228 | 2.6378 |  |
| f.000 | time, base, house, personal, fuel | 1000 | X | 2.1228 | 2.9886 |  |

Table 3: Mean Square Errors of the Size-2 models tested.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | **Small Filter** | | **Mean % Size 2** | **MSE** | |
| **Model Run** | **Predictors** | **Floor** | **Outlier** | **Separate** | **Dummies** |
| Constant |  | 25 | X | 0.4247 | 0.4132 | |
| dept.effect |  | 25 | X | 0.4247 | 0.3909 | |
| tract |  |  |  | 0.4247 | 0.5095 | |
| base | time, base, race, age, house, personal, fuel | 25 | X | 0.4247 | 0.3901 | **0.3895** |
| M101111 | time, base, age, house, personal, fuel | 25 | X | 0.4247 | 0.3908 | 0.3896 |
| M100111 | time, base, house, personal, fuel | 25 | X | 0.4247 | 0.3901 | 0.3895 |
| M101011 | time, base, age, personal, fuel | 25 | X | 0.4247 | 0.3910 | 0.3899 |
| M101101 | time, base, age, house, fuel | 25 | X | 0.4247 | 0.3909 | 0.3898 |
| M101110 | time, base, age, house, personal | 25 | X | 0.4247 | 0.3908 | 0.3897 |
| M110111 | time, base, race, house, personal, fuel | 25 | X | 0.4247 | 0.3901 | 0.3895 |
| M110011 | time, base, race, personal, fuel | 25 | X | 0.4247 | 0.3903 | 0.3900 |
| M110101 | time, base, race, house, fuel | 25 | X | 0.4247 | 0.3901 | 0.3897 |
| M110110 | time, base, race, house, personal | 25 | X | 0.4247 | 0.3901 | 0.3896 |
| M111011 | time, base, race, age, personal, fuel | 25 | X | 0.4247 | 0.3910 | 0.3899 |
| M111001 | time, base, race, age, fuel | 25 | X | 0.4247 | 0.3911 | 0.3903 |
| M111010 | time, base, race, age, personal | 25 | X | 0.4247 | 0.3911 | 0.3902 |
| M111101 | time, base, race, age, house, fuel | 25 | X | 0.4247 | 0.3909 | 0.3898 |
| M111100 | time, base, race, age, house | 25 | X | 0.4247 | 0.3909 | 0.3899 |
| M111110 | time, base, race, age, house, personal | 25 | X | 0.4247 | 0.3909 | 0.3896 |
| f.050 | time, base, house, personal, fuel | 50 | X | 0.4247 |  |  |
| f.100 | time, base, house, personal, fuel | 100 | X | 0.4247 |  |  |
| f.150 | time, base, house, personal, fuel | 150 | X | 0.4247 |  |  |
| f.200 | time, base, house, personal, fuel | 200 | X | 0.4247 |  |  |
| f.250 | time, base, house, personal, fuel | 250 | X | 0.4247 |  |  |
| f.500 | time, base, house, personal, fuel | 500 | X | 0.4247 |  |  |
| f.000 | time, base, house, personal, fuel | 1000 | X | 0.4247 |  |  |

Table 4: Mean Square Errors of the Size-3 models tested.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | **Small Filter** | | **Mean % Size 3** | **MSE** | |
| **Model Run** | **Predictors** | **Floor** | **Outlier** | **Separate** | **Dummies** |
| Constant |  | 25 | X | 0.1082 | 0.2710 | |
| dept.effect |  | 25 | X | 0.1082 | 0.2704 | |
| tract |  |  |  | 0.1082 | 0.3487 | |
| base | time, base, race, age, house, personal, fuel | 25 | X | 0.1082 | 0.2708 | **0.2697** |
| M101111 | time, base, age, house, personal, fuel | 25 | X | 0.1082 | 0.2722 | 0.2698 |
| M100111 | time, base, house, personal, fuel | 25 | X | 0.1082 | 0.2708 | 0.2697 |
| M101011 | time, base, age, personal, fuel | 25 | X | 0.1082 | 0.2717 | 0.2699 |
| M101101 | time, base, age, house, fuel | 25 | X | 0.1082 | 0.2716 | 0.2697 |
| M101110 | time, base, age, house, personal | 25 | X | 0.1082 | 0.2722 | 0.2702 |
| M110111 | time, base, race, house, personal, fuel | 25 | X | 0.1082 | 0.2706 | 0.2698 |
| M110011 | time, base, race, personal, fuel | 25 | X | 0.1082 | 0.2702 | 0.2699 |
| M110101 | time, base, race, house, fuel | 25 | X | 0.1082 | 0.2705 | 0.2698 |
| M110110 | time, base, race, house, personal | 25 | X | 0.1082 | 0.2708 | 0.2703 |
| M111011 | time, base, race, age, personal, fuel | 25 | X | 0.1082 | 0.2715 | 0.2699 |
| M111001 | time, base, race, age, fuel | 25 | X | 0.1082 | 0.2710 | 0.2700 |
| M111010 | time, base, race, age, personal | 25 | X | 0.1082 | 0.2715 | 0.2705 |
| M111101 | time, base, race, age, house, fuel | 25 | X | 0.1082 | 0.2715 | 0.2699 |
| M111100 | time, base, race, age, house | 25 | X | 0.1082 | 0.2715 | 0.2703 |
| M111110 | time, base, race, age, house, personal | 25 | X | 0.1082 | 0.2719 | 0.2703 |
| f.050 | time, base, house, personal, fuel | 50 | X | 0.1082 |  |  |
| f.100 | time, base, house, personal, fuel | 100 | X | 0.1082 |  |  |
| f.150 | time, base, house, personal, fuel | 150 | X | 0.1082 |  |  |
| f.200 | time, base, house, personal, fuel | 200 | X | 0.1082 |  |  |
| f.250 | time, base, house, personal, fuel | 250 | X | 0.1082 |  |  |
| f.500 | time, base, house, personal, fuel | 500 | X | 0.1082 |  |  |
| f.000 | time, base, house, personal, fuel | 1000 | X | 0.1082 |  |  |

Table 5: Mean Square Errors of the injury models tested.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | **Small Filter** | | **Mean Injuries** | **MSE** | |
| **Model Run** | **Predictors** | **Floor** | **Outlier** | **Separate** | **Dummies** |
| Constant |  | 25 | X | 0.1805 | 0.6442 | |
| dept.effect |  | 25 | X | 0.1805 | 0.6331 | |
| tract |  |  |  | 0.1805 | 0.8746 | |
| lasso.min | time, base, race, age, house, personal, fuel | 25 | X | 0.1805 | 0.6327 | |
| lasso.1se | time, base, race, age, house, personal, fuel | 25 | X | 0.1805 | 0.6340 | |
| rForest | time, base, race, age, house, personal, fuel | 25 | X | 0.1805 | 0.6402 | |
| base | time, base, race, age, house, personal, fuel | 25 | X | 0.1805 | 0.6448 | 0.6263 |
| M101111 | time, base, age, house, personal, fuel | 25 | X | 0.1805 | 0.6479 | 0.6257 |
| M100111 | time, base, house, personal, fuel | 25 | X | 0.1805 | 0.6448 | 0.6263 |
| M101011 | time, base, age, personal, fuel | 25 | X | 0.1805 | 0.6333 | 0.6260 |
| M101101 | time, base, age, house, fuel | 25 | X | 0.1805 | 0.6469 | 0.6273 |
| M101110 | time, base, age, house, personal | 25 | X | 0.1805 | 0.6449 | 0.6258 |
| M110111 | time, base, race, house, personal, fuel | 25 | X | 0.1805 | 0.6478 | 0.6264 |
| M110011 | time, base, race, personal, fuel | 25 | X | 0.1805 | 0.6313 | 0.6275 |
| M110101 | time, base, race, house, fuel | 25 | X | 0.1805 | 0.6478 | 0.6281 |
| M110110 | time, base, race, house, personal | 25 | X | 0.1805 | 0.6415 | 0.6264 |
| M111011 | time, base, race, age, personal, fuel | 25 | X | 0.1805 | 0.6395 | 0.6260 |
| M111001 | time, base, race, age, fuel | 25 | X | 0.1805 | 0.6411 | 0.6313 |
| M111010 | time, base, race, age, personal | 25 | X | 0.1805 | 0.6361 | 0.6261 |
| M111101 | time, base, race, age, house, fuel | 25 | X | 0.1805 | 0.6505 | 0.6276 |
| M111100 | time, base, race, age, house | 25 | X | 0.1805 | 0.6469 | 0.6275 |
| M111110 | time, base, race, age, house, personal | 25 | X | 0.1805 | 0.6475 | **0.6254** |
| f.050 | time, base, house, personal, fuel | 50 | X | 0.1805 |  |  |
| f.100 | time, base, house, personal, fuel | 100 | X | 0.1805 |  |  |
| f.150 | time, base, house, personal, fuel | 150 | X | 0.1805 |  |  |
| f.200 | time, base, house, personal, fuel | 200 | X | 0.1805 |  |  |
| f.250 | time, base, house, personal, fuel | 250 | X | 0.1805 |  |  |
| f.500 | time, base, house, personal, fuel | 500 | X | 0.1805 |  |  |
| f.000 | time, base, house, personal, fuel | 1000 | X | 0.1805 |  |  |

Table 6: Mean Square Errors of the fire-death models tested.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | **Small Filter** | | **Mean Deaths** | **MSE** | |
| **Model Run** | **Predictors** | **Floor** | **Outlier** | **Separate** | **Dummies** |
| Constant |  | 25 | X | 0.0184 | 0.1772 | |
| dept.effect |  | 25 | X | 0.0184 | 0.1772 | |
| tract |  |  |  | 0.0184 | 0.2538 | |
| lasso.min | time, base, race, age, house, personal, fuel | 25 | X | 0.0184 | 0.1768 | |
| lasso.1se | time, base, race, age, house, personal, fuel | 25 | X | 0.0184 | 0.1770 | |
| rForest | time, base, race, age, house, personal, fuel | 25 | X | 0.0184 | 0.1789 | |
| base | time, base, race, age, house, personal, fuel | 25 | X | 0.0184 | 5.3604 | 0.1769 |
| M101111 | time, base, age, house, personal, fuel | 25 | X | 0.0184 | 6.3266 | **0.1769** |
| M100111 | time, base, house, personal, fuel | 25 | X | 0.0184 | 5.3604 | 0.1769 |
| M101011 | time, base, age, personal, fuel | 25 | X | 0.0184 | 0.1793 | 0.1769 |
| M101101 | time, base, age, house, fuel | 25 | X | 0.0184 | 2.5591 | 0.1769 |
| M101110 | time, base, age, house, personal | 25 | X | 0.0184 | 4.0317 | 0.1769 |
| M110111 | time, base, race, house, personal, fuel | 25 | X | 0.0184 | 7.0284 | 0.1769 |
| M110011 | time, base, race, personal, fuel | 25 | X | 0.0184 | 0.1778 | 0.1770 |
| M110101 | time, base, race, house, fuel | 25 | X | 0.0184 | 2.9679 | 0.1769 |
| M110110 | time, base, race, house, personal | 25 | X | 0.0184 | 2.9177 | 0.1769 |
| M111011 | time, base, race, age, personal, fuel | 25 | X | 0.0184 | 0.1794 | 0.1769 |
| M111001 | time, base, race, age, fuel | 25 | X | 0.0184 | 0.1784 | 0.1770 |
| M111010 | time, base, race, age, personal | 25 | X | 0.0184 | 0.1780 | 0.1770 |
| M111101 | time, base, race, age, house, fuel | 25 | X | 0.0184 | 3.5768 | 0.1769 |
| M111100 | time, base, race, age, house | 25 | X | 0.0184 | 2.2835 | 0.1769 |
| M111110 | time, base, race, age, house, personal | 25 | X | 0.0184 | 5.5102 | 0.1769 |
| f.050 | time, base, house, personal, fuel | 50 | X | 0.0184 |  |  |
| f.100 | time, base, house, personal, fuel | 100 | X | 0.0184 |  |  |
| f.150 | time, base, house, personal, fuel | 150 | X | 0.0184 |  |  |
| f.200 | time, base, house, personal, fuel | 200 | X | 0.0184 |  |  |
| f.250 | time, base, house, personal, fuel | 250 | X | 0.0184 |  |  |
| f.500 | time, base, house, personal, fuel | 500 | X | 0.0184 |  |  |
| f.000 | time, base, house, personal, fuel | 1000 | X | 0.0184 |  |  |

**Appendix I**

This first query creates a dept\_incidents intermediate table. Its purpose is to provide me the data I need to correct for geolocation errors. Note that it is determined at the department level rather than the tract level. Its structure is listed below.

|  |  |  |
| --- | --- | --- |
| **Column** | **Type** | **Description** |
| state | text | State from NFIRS |
| fdid | text | Department ID from NFIRS |
| year | integer | Year |
| incidents | bigint | Total number of incidents reported for the department and year |
| incidents\_loc | bigint | Number of incidents that were geolocated for the department and year |
| v5\_incidents | bigint | Total number of incidents reported in “Version 5” for the department and year |
| v5\_incidents\_loc | bigint | Number of incidents that were geolocated in “Version 5” for the department and year |
| fires | bigint | Total number of fires reported for the department and year |
| fires\_loc | bigint | Number of fires that were geolocated for the department and year |
| mod\_fires | bigint | Total number of fires with the fire module filled out reported for the department and year |
| mod\_fires\_loc | bigint | Number of fires with the fire module filled out that were geolocated for the department and year |
| struc\_fires | bigint | Total number of structural fires reported for the department and year |
| struc\_fires\_loc | bigint | Number of structural fires that were geolocated for the department and year |
| res\_fires | bigint | Total number of residential structure fires reported for the department and year |
| res\_fires\_loc | bigint | Number of residential structure fires that were geolocated for the department and year |
| lr\_fires | bigint | Total number of low-risk structure fires reported for the department and year |
| lr\_fires\_loc | bigint | Number of low-risk fires structure that were geolocated for the department and year |
| injuries | bigint | Total number of injuries reported for the department and year |
| injuries\_loc | bigint | Number of injuries that were geolocated for the department and year |
| deaths | bigint | Total number of deaths reported for the department and year |
| deaths\_loc | bigint | Number of deaths that were geolocated for the department and year |

It is created with the following query. As an aside, the ‘geog’ column is a POSTGIS Geography column that I generated by turning the latitude / longitude pairs into a point geography. In this query its only purpose is to identify the records that were successfully geocoded. The query would work just as well by substituting latitude or longitude.

WITH t AS (

SELECT b.state, b.fdid, b.inc\_date % 10000 AS year,

CASE

WHEN a.geog IS NOT NULL THEN 1

ELSE 0

END AS located,

CASE

WHEN b.inc\_type LIKE '1%' OR f.state IS NOT NULL THEN 1

ELSE 0

END AS fire,

CASE

WHEN b.version::numeric(4,1) = 5.0::numeric(4,1) THEN 1

ELSE 0

END AS version5,

CASE

WHEN b.aid IN ( '3', '4' ) THEN 0

ELSE 1

END AS aid,

CASE

WHEN

( b.inc\_date % 10000 > 2001 AND b.inc\_type IN ( '111', '120', '121', '122', '123' ) OR

b.inc\_date % 10000 > 2001 AND b.inc\_date % 10000 < 2008 AND b.inc\_type = '112'

) AND f.struc\_type IN ( '1', '2' ) OR

( b.inc\_type IN ( '113', '114', '115', '116', '117', '118' ) OR

b.inc\_type = '110' AND b.inc\_date % 10000 < 2009

) AND ( f.struc\_type IN ( '1', '2' ) OR f.struc\_type IS NULL ) THEN 1

ELSE 0

END AS struc,

CASE

WHEN f.state IS NOT NULL THEN 1

ELSE 0

END AS module,

CASE

WHEN ( f.not\_res = 'N' OR b.prop\_use LIKE '4%' ) THEN 1

ELSE 0

END AS res,

CASE

WHEN b.prop\_use = '419' OR b.prop\_use LIKE '9%' THEN 1

ELSE 0

END AS lr,

b.ff\_inj, b.oth\_inj, b.ff\_death, b.oth\_death

FROM basicincident b

LEFT JOIN coded\_addresses a ON ( b.state = a.state AND b.fdid = a.fdid AND

b.inc\_date = a.inc\_date AND b.inc\_no = a.inc\_no AND

b.exp\_no = a.exp\_no )

LEFT JOIN fireincident f ON ( b.state = f.state AND b.fdid = f.fdid AND

b.inc\_date = f.inc\_date AND b.inc\_no = f.inc\_no AND

b.exp\_no = f.exp\_no )

)

SELECT

t.state,

t.fdid,

t.year,

sum(t.aid) AS incidents,

sum(t.aid \* t.located) AS incidents\_loc,

sum(t.aid \* t.version5) AS v5\_incidents,

sum(t.aid \* t.version5 \* t.located) AS v5\_incidents\_loc,

sum(t.aid \* t.fire) AS fires,

sum(t.aid \* t.located \* t.fire) AS fires\_loc,

sum(t.aid \* t.module) AS mod\_fires,

sum(t.aid \* t.located \* t.module) AS mod\_fires\_loc,

sum(t.aid \* t.struc) AS struc\_fires,

sum(t.aid \* t.located \* t.struc) AS struc\_fires\_loc,

sum(t.aid \* t.res \* t.struc) AS res\_fires,

sum(t.aid \* t.located \* t.res \* t.struc) AS res\_fires\_loc,

sum(t.aid \* t.lr \* t.struc) AS lr\_fires,

sum(t.aid \* t.located \* t.lr \* t.struc) AS lr\_fires\_loc,

sum(t.ff\_inj + t.oth\_inj \* t.aid) AS injuries,

sum(t.located \* (t.ff\_inj + t.oth\_inj \* t.aid)) AS injuries\_loc,

sum(t.ff\_death + t.oth\_death \* t.aid) AS deaths,

sum(t.located \* (t.ff\_death + t.oth\_death \* t.aid)) AS deaths\_loc

INTO dept\_incidents

FROM t

GROUP BY t.state, t.fdid, t.year;

This query creates a coded\_fires intermediate table. Its collects all the relevant NFIRS data for each fire into a single table. I should point out that this table is all fires, not residential fires. It has also been revised since I originally used it to include fires that I originally excluded. The structure of the table is:

|  |  |  |
| --- | --- | --- |
| **Column** | **Type** | **Description** |
| state | text | State from NFIRS |
| fdid | text | Department ID from NFIRS |
| inc\_date | integer | from NFIRS |
| inc\_no | text | from NFIRS |
| exp\_no | integer | from NFIRS |
| year | integer | Year, modified from inc\_date |
| version | Numeric( 4, 1 ) | NFIRS Version number. |
| struc | text | Whether this is a structure fire incident. |
| risk | text | ‘High’, ‘Med’ or ‘Low’ risk. |
| inc\_type | text | From the basicincident table |
| not\_res | text | From the fireincident table |
| fire\_sprd | text | From the basicincident table |
| prop\_use | text | From the basicincident table |
| struc\_type | text | From the fireincident table |
| struc\_stat | text | From the fireincident table |
| bldg\_above | integer | From the fireincident table |
| alarm\_time | Timestamp w/o tz | This is the ALARM field from the basicincident table, that I had previously converted to a TIMESTAMP field |
| arrival\_time | timestamp w/o tz | This is the ARRIVAL field from the basicincident table, that I had previously converted to a TIMESTAMP field |
| travel\_time | interval | ARRIVAL – ALARM (previously calculated) |
| ff\_death | integer | From the basicincident table |
| oth\_death | integer | From the basicincident table |
| ff\_inj | integer | From the basicincident table |
| oth\_inj | integer | From the basicincident table |
| geoid\_00 | text | 2000 Census Tract, modified to match a standard complete census format Tract label. |
| geoid | text | 2010 Census Tract, modified to match a standard complete census format Tract label. |
| geog | geography | This is a POSTGIS Point Geography field that captures latitude and longitude |

The table is created with the following query:

WITH t AS (

SELECT b.state, b.fdid, b.inc\_date, b.inc\_no, b.exp\_no,

b.version::numeric(4,1) as version, b.dept\_sta, b.inc\_type, b.add\_wild,

b.aid, b.alarm, b.arrival, b.inc\_cont, b.lu\_clear, b.shift, b.alarms,

b.district, b.act\_tak1, b.act\_tak2, b.act\_tak3, b.app\_mod, b.sup\_app,

b.ems\_app, b.oth\_app, b.sup\_per, b.ems\_per, b.oth\_per, b.resou\_aid,

b.prop\_loss, b.cont\_loss, b.prop\_val, b.cont\_val, b.ff\_death,

b.oth\_death, b.ff\_inj, b.oth\_inj, b.det\_alert, b.haz\_rel, b.mixed\_use,

b.prop\_use, b.census, b.loc\_type, b.num\_mile, b.street\_pre,

b.streetname, b.streettype, b.streetsuf, b.apt\_no, b.city,

b.state\_id, b.zip5, b.zip4, b.x\_street, b.inc\_date2, b.alarm\_time,

b.arrival\_time, b.travel\_time, b.id, a.tr00\_fid, a.tr10\_fid, a.geog

FROM basicincident b LEFT JOIN coded\_addresses a

ON ( b.state = a.state AND b.fdid = a.fdid AND

b.inc\_date = a.inc\_date AND b.inc\_no = a.inc\_no AND

b.exp\_no = a.exp\_no )

WHERE b.aid NOT IN ( '3', '4' )

)

SELECT t.state, t.fdid, t.inc\_date, t.inc\_no, t.exp\_no,

t.inc\_date % 10000 AS year, t.version,

CASE

WHEN

( t.inc\_date % 10000 > 2001 AND t.inc\_type IN ('111','120','121','122','123') OR

t.inc\_date % 10000 > 2001 AND t.inc\_date % 10000 < 2008 AND t.inc\_type = '112'

) AND f.struc\_type IN ( '1', '2' ) OR

( t.inc\_type IN( '113', '114', '115', '116', '117', '118' ) OR

t.inc\_type = '110' AND t.inc\_date % 10000 < 2009

) AND ( f.struc\_type IN( '1', '2' ) OR f.struc\_type IS NULL ) THEN 'Y'::text

ELSE 'N'::text

END AS struc,

CASE

WHEN t.prop\_use = '419' OR t.prop\_use LIKE '9%' THEN 'Low Risk'

WHEN t.prop\_use NOT IN ( '419', '644', '645' ) AND

substring(t.prop\_use, 1, 1) IN ( '4', '5', '6', '7', '8' ) AND

( f.bldg\_above IS NULL OR f.bldg\_above < 7 ) THEN 'Med Risk'

ELSE 'High Risk'

END AS risk,

t.inc\_type, f.not\_res, f.fire\_sprd, t.prop\_use,

f.struc\_type, f.struc\_stat, f.bldg\_above, t.alarm\_time, t.arrival\_time,

t.travel\_time, t.ff\_death, t.oth\_death, t.ff\_inj, t.oth\_inj,

'14000US' || t.tr00\_fid AS geoid\_00, '14000US' || t.tr10\_fid AS geoid, t.geog

INTO coded\_fires

FROM t LEFT JOIN fireincident f

ON ( t.state = f.state AND t.fdid = f.fdid AND

t.inc\_date = f.inc\_date AND t.inc\_no = f.inc\_no AND

t.exp\_no = f.exp\_no )

WHERE t.inc\_type LIKE '1%' OR f.state IS NOT NULL;

This query creates a casualties\_fire intermediate table. Its merges the casualties from the ffcasualty and civiliancasualty tables, and collects some extra NFIRS data for each fire-casualty that I use for either filtering or summarizing. The structure of the table is:

|  |  |  |
| --- | --- | --- |
| **Column** | **Type** | **Description** |
| state | text | From NFIRS |
| fdid | text | From NFIRS |
| inc\_date | integer | From NFIRS |
| inc\_no | text | From NFIRS |
| exp\_no | integer | From NFIRS |
| type | text | Either ‘ff’ or ‘civ’ |
| seq\_no | integer | From the ffcasualty or civiliancasualty table |
| year | integer | Year of the incident. |
| aid\_flag | text | Flag identifying whether this as a mutual aid record. |
| gender | text | From NFIRS |
| age | double precision | From NFIRS |
| race | text | From NFIRS. Note that it is only recorded for civilian casualties |
| ethnicity | text | From NFIRS. Note that it is only recorded for civilian casualties |
| sev | text | Injury severity. Since the two tables use different definitions for severity, the ffcasualty portion of the query includes a rough translation to make it consistent with the civiliancasualty definitions. |
| res | text | Residential. Based on the PROP\_USE field |
| struc | text | ‘Whether the casualty was associated with a structure fire incident. |
| module | text | ‘Y’ means that the fire module has been filled out. |
| risk | text | ‘High’, ‘Med’ or ‘Low’ risk. |
| geoid | text | 2010 Census tract, using the Census standard format |
| geog | geography | Latitude / Longitude in a POSTGIS Geography field |

The table is created with the following query:

WITH c AS (

SELECT ffcasualty.state, ffcasualty.fdid, ffcasualty.inc\_date,

ffcasualty.inc\_no, ffcasualty.exp\_no, 'ff' AS type,

ffcasualty.ff\_seq\_no AS seq\_no, ffcasualty.gender,

ffcasualty.age, NULL AS race, NULL AS ethnicity,

CASE

WHEN ffcasualty.severity IN ( '2', '3' ) THEN '1'

WHEN ffcasualty.severity = '4' THEN '2'

WHEN ffcasualty.severity = '5' THEN '3'

WHEN ffcasualty.severity = '6' THEN '4'

WHEN ffcasualty.severity = '7' THEN '5'

ELSE ffcasualty.severity

END AS sev

FROM ffcasualty

WHERE ffcasualty.severity <> '1'

UNION

SELECT civiliancasualty.state, civiliancasualty.fdid,

civiliancasualty.inc\_date, civiliancasualty.inc\_no,

civiliancasualty.exp\_no, 'civ' AS type,

civiliancasualty.seq\_number AS seq\_no,

civiliancasualty.gender, civiliancasualty.age,

civiliancasualty.race, civiliancasualty.ethnicity,

civiliancasualty.sev

FROM civiliancasualty

)

SELECT c.state, c.fdid, c.inc\_date, c.inc\_no, c.exp\_no, c.type,

c.seq\_no, c.inc\_date % 10000 AS year,

CASE

WHEN b.aid IN( '3', '4' ) THEN 'Y'::text

ELSE 'N'::text

END AS aid\_flag,

c.gender, c.age, c.race, c.ethnicity, c.sev,

CASE

WHEN b.prop\_use LIKE '4%' THEN 'Y'

ELSE 'N'

END AS res,

CASE

WHEN

( b.inc\_date % 10000 > 2001 AND b.inc\_type IN ('111','120','121','122','123') OR

b.inc\_date % 10000 > 2001 AND b.inc\_date % 10000 < 2008 AND b.inc\_type = '112'

) AND f.struc\_type IN ( '1', '2' ) OR

( b.inc\_type IN( '113', '114', '115', '116', '117', '118' ) OR

b.inc\_type = '110' AND b.inc\_date % 10000 < 2009

) AND ( f.struc\_type IN( '1', '2' ) OR f.struc\_type IS NULL ) THEN 'Y'::text

ELSE 'N'::text

END AS struc,

CASE

WHEN f.state IS NULL THEN 'N'

ELSE 'Y'

END AS module,

CASE

WHEN b.prop\_use = '419' OR b.prop\_use LIKE '9%' THEN 'Low Risk'

WHEN b.prop\_use NOT IN ( '419', '644', '645' ) AND

substring(b.prop\_use, 1, 1) IN ( '4', '5', '6', '7', '8' ) AND

( f.bldg\_above IS NULL OR f.bldg\_above < 7 ) THEN 'Med Risk'

ELSE 'High Risk'

END AS risk,

'14000US' || a.tr10\_fid AS geoid, a.geog

INTO casualties\_fire

FROM c JOIN basicincident b

ON ( c.state = b.state AND c.fdid = b.fdid AND

c.inc\_date = b.inc\_date AND c.inc\_no = b.inc\_no AND

c.exp\_no = b.exp\_no )

LEFT JOIN coded\_addresses a

ON ( c.state = a.state AND c.fdid = a.fdid AND

c.inc\_date = a.inc\_date AND c.inc\_no = a.inc\_no AND

c.exp\_no = a.exp\_no )

LEFT JOIN fireincident f

ON ( c.state = f.state AND c.fdid = f.fdid AND

c.inc\_date = f.inc\_date AND c.inc\_no = f.inc\_no AND

c.exp\_no = f.exp\_no )

WHERE inc\_type like '1%' or f.state is not null;

This monstrous query constructs the data table I actually use from the constituent queries. The LEFT JOINs below are probably an unnecessary complication, since I will effectively filter out any records with null values.

WITH f AS (

SELECT cf.year, cf.geoid, count(\*) AS tot\_fires,

sum(

CASE

WHEN cf.prop\_use LIKE '4%' AND cf.geoid IS NOT NULL THEN 1

ELSE 0

END ) AS res\_all,

sum(

CASE

WHEN cf.struc = 'Y' AND risk='Low Risk' AND cf.geoid IS NOT NULL THEN 1

ELSE 0

END) AS low\_risk,

sum(

CASE

WHEN cf.struc = 'Y'::text AND cf.risk = 'Low Risk'::text AND cf.geoid IS NOT NULL

AND cf.fire\_sprd IS NOT NULL THEN 1

ELSE 0

END) AS res\_1,

sum(

CASE

WHEN cf.struc = 'Y' AND risk='Low Risk' AND cf.geoid IS NOT NULL AND

cf.fire\_sprd IN ( '3', '4', '5') THEN 1

ELSE 0

END) AS res\_2,

sum(

CASE

WHEN cf.struc = 'Y' AND risk='Low Risk' AND cf.geoid IS NOT NULL AND

cf.fire\_sprd = '5' THEN 1

ELSE 0

END) AS res\_3,

sum(

CASE

WHEN cf.struc = 'Y'::text AND cf.risk = 'Low Risk'::text AND

cf.geoid IS NOT NULL THEN cf.ff\_inj + cf.oth\_inj

ELSE 0

END) AS injuries,

sum(

CASE

WHEN cf.struc = 'Y'::text AND cf.risk = 'Low Risk'::text AND

cf.geoid IS NOT NULL THEN cf.ff\_death + cf.oth\_death

ELSE 0

END) AS deaths

FROM coded\_fires cf

WHERE cf.year > 2006 AND cf.year < 2014 AND cf.version = 5.0 AND

NOT ( cf.inc\_type = '112' AND cf.year > 2007 )

GROUP BY cf.year, cf.geoid

), d AS (

SELECT i.year, d.firecares\_id AS fd\_id, d.fd\_size,

sum(i.incidents) AS incidents, sum(i.incidents\_loc) AS located,

sum(i.fires) AS dept\_fires, sum(i.lr\_fires) AS dept\_lr

FROM dept\_incidents i JOIN gov\_units d ON i.state = d.state AND i.fdid = d.fdid

WHERE i.year > 2006 AND i.year < 2014

GROUP BY i.year, d.firecares\_id, d.fd\_size

) , c AS (

SELECT casualties\_fire.geoid, casualties\_fire.year,

sum(

CASE

WHEN casualties\_fire.sev <> '5'::text AND

(casualties\_fire.type = 'ff'::text OR casualties\_fire.aid\_flag = 'N'::text ) THEN 1

ELSE 0

END) AS injuries,

sum(

CASE

WHEN casualties\_fire.sev = '5'::text AND

( casualties\_fire.type = 'ff'::text OR casualties\_fire.aid\_flag = 'N'::text )

THEN 1

ELSE 0

END) AS deaths

FROM casualties\_fire

GROUP BY casualties\_fire.geoid, casualties\_fire.year

)

SELECT tr.year, tr.geoid, tr.region, tr.state, tr.fc\_dept\_id AS fd\_id,

d.fd\_size, d.incidents AS dept\_incidents, d.dept\_fires, d.dept\_lr,

CASE

WHEN d.incidents > 0 THEN d.located::double precision / d.incidents::double precision

WHEN d.incidents = 0 AND d.located > 0 THEN 'Infinity'::double precision

ELSE 'NaN'::double precision

END AS f\_located,

f.res\_all, f.low\_risk, f.res\_1, f.res\_2, f.res\_3, c.injuries, c.deaths,

CASE

WHEN acs."B25002\_002E" > 0 THEN acs."B01001\_001E"::double precision / acs."B25002\_002E"::double precision

WHEN acs."B25002\_002E" = 0 AND acs."B01001\_001E" > 0 THEN 'Infinity'::double precision

ELSE 'NaN'::double precision

END AS ave\_hh\_sz,

acs."B01001\_001E" AS pop, acs."B02001\_003E" AS black,

acs."B02001\_004E" AS amer\_es,

acs."B02001\_005E" + acs."B02001\_006E" + acs."B02001\_007E" + acs."B02001\_008E" AS other,

acs."B03003\_003E" AS hispanic, acs."B01001\_002E" AS males,

acs."B01001\_003E" + acs."B01001\_027E" AS age\_under5,

acs."B01001\_004E" + acs."B01001\_028E" AS age\_5\_9,

acs."B01001\_005E" + acs."B01001\_029E" AS age\_10\_14,

acs."B01001\_006E" + acs."B01001\_007E" + acs."B01001\_030E" + acs."B01001\_031E" AS age\_15\_19,

acs."B01001\_008E" + acs."B01001\_009E" + acs."B01001\_010E" + acs."B01001\_032E" + acs."B01001\_033E" + acs."B01001\_034E" AS age\_20\_24,

acs."B01001\_011E" + acs."B01001\_012E" + acs."B01001\_035E" + acs."B01001\_036E" AS age\_25\_34,

acs."B01001\_013E" + acs."B01001\_014E" + acs."B01001\_037E" + acs."B01001\_038E" AS age\_35\_44,

acs."B01001\_015E" + acs."B01001\_016E" + acs."B01001\_039E" + acs."B01001\_040E" AS age\_45\_54,

acs."B01001\_017E" + acs."B01001\_018E" + acs."B01001\_019E" + acs."B01001\_041E" + acs."B01001\_042E" + acs."B01001\_043E" AS age\_55\_64,

acs."B01001\_020E" + acs."B01001\_021E" + acs."B01001\_022E" + acs."B01001\_044E" + acs."B01001\_045E" + acs."B01001\_046E" AS age\_65\_74,

acs."B01001\_023E" + acs."B01001\_024E" + acs."B01001\_047E" + acs."B01001\_048E" AS age\_75\_84,

acs."B01001\_025E" + acs."B01001\_049E" AS age\_85\_up,

acs."B25002\_001E" AS hse\_units, acs."B25002\_003E" AS vacant,

acs."B25014\_008E" AS renter\_occ,

acs."B25014\_005E" + acs."B25014\_006E" + acs."B25014\_007E" + acs."B25014\_011E" + acs."B25014\_012E" + acs."B25014\_013E" AS crowded,

acs."B25024\_002E" + acs."B25024\_003E" + acs."B25024\_004E" AS sfr,

acs."B25024\_007E" + acs."B25024\_008E" + acs."B25024\_009E" AS units\_10,

acs."B25024\_010E" AS mh,

acs."B25034\_006E" + acs."B25034\_007E" + acs."B25034\_008E" + acs."B25034\_009E" + acs."B25034\_010E" AS older,

acs."B19013\_001E" AS inc\_hh,

svi.r\_pl\_themes AS svi,

acs."B12001\_001" - (acs."B12001\_003" + acs."B12001\_012") AS married,

acs."B23025\_005" AS unemployed, acs."B12001\_007" AS nilf,

sm.adult\_smoke AS smoke\_st, sc.smoking\_pct AS smoke\_cty,

acs."B25040\_002E" AS fuel\_gas,

acs."B25040\_003E" AS fuel\_tank, acs."B25040\_005E" AS fuel\_oil,

acs."B25040\_006E" AS fuel\_coal, acs."B25040\_007E" AS fuel\_wood,

acs."B25040\_008E" AS fuel\_solar, acs."B25040\_009E" AS fuel\_other,

acs."B25040\_010E" AS fuel\_none

FROM tract\_years tr LEFT JOIN f ON tr.geoid = f.geoid AND tr.year = f.year

LEFT JOIN d ON tr.fc\_dept\_id = d.fd\_id AND tr.year = d.year

LEFT JOIN svi2010 svi ON tr.geoid = ('14000US' || svi.fips)

LEFT JOIN acs\_est acs ON tr.geoid = acs.geoid AND

CASE

WHEN tr.year < 2008 THEN 2008

WHEN tr.year > 2012 THEN 2012

ELSE tr.year

END::double precision = (acs.year - 2::double precision)

LEFT JOIN sins sm ON tr.state = sm.postal\_code AND sm.year = 2010

LEFT JOIN sins\_county sc ON "substring"(tr.geoid, 8, 5) = sc.fips

LEFT JOIN c ON tr.geoid::text = c.geoid AND tr.year = c.year

WHERE tr.year > 2006 AND tr.year < 2014;

The following extra tables appear:

‘**tract\_years’** is an intermediate table that is the Cartesian product of tracts and years of the study. It also includes some additional information for the tracts, allowing me to reduce the number of tables referenced.

|  |  |  |
| --- | --- | --- |
| **Column** | **Type** | **Description** |
| geoid | character varying | Census geographic ID for the tract. |
| year | integer | year |
| fc\_dept\_id | numeric(10,0) | FireCARES id for the department associated with the tract |
| region | character varying | Census region |
| state | character varying | State |

**gov\_units** was provided (in a slightly modified form) by Tyler. This table serves two functions. First it serves as a translation table between the old NFIRS identifiers and the firecares id. Second, it contains the department size information used in the model. Its structure is:

|  |  |  |
| --- | --- | --- |
| **Column** | **Type** | **Description** |
| gid | integer | Internal ID |
| fd\_populat | numeric(10,0) | Department Population |
| fd\_unit\_ty | character varying | Type of Department (Incorpor., Unincorpor., Reserve, County, m.civ, Nat.Am.) |
| firecares\_id | numeric(10,0) | FireCARES ID |
| mean\_fd\_po | numeric | Another Department Population Field |
| gov\_un\_id | numeric(10,0) | ID number for the government unit associated with the department |
| gov\_un\_nam | character varying | Government Unit Name |
| fd\_name | character varying | Fire Department Name |
| count\_fd\_n | numeric(10,0) |  |
| shape\_leng | numeric |  |
| shape\_area | numeric |  |
| geom | geometry | Geometry field representing the boundaries of the government unit |
| state | text | State |
| fdid | text | Old NFIRS department ID |
| fd\_size | text | Department Size Category |

**acs\_est** is the data from the American Community Survey.

|  |  |  |
| --- | --- | --- |
|  |  |  |
| year | double precision | ACS Year. This is the last year in the five-year range for the table. |
| LOGRECNO | integer | ACS Logical record number |
| geoid | text | Census geographic ID for the tract |
| state | text | State |
| county | text | County |
| tract | text | Tract |
| B01001\_001E | integer | Population |
| B01001\_002E | integer | Number of Males |
| … | … | There are more than 100 more census data fields in the table. |

**svi2010** contains the Social Vulnerability Index used in the model and can be obtained from <http://svi.cdc.gov>.

**sins** table contains alcohol and tobacco consumption data by state and year. It was developed by one of my coworkers, who in turn generated it from the Tobacco Use Supplement to the Current Population Survey, performed by the BLS and Census. Here is the structure.

|  |  |  |
| --- | --- | --- |
| **Column** | **Type** | **Description** |
| postal\_code | character varying | Two-Character State Postal Code |
| year | integer | Year |
| beer | double precision | Beer consumption |
| wine | double precision | Wine Consumption |
| spirits | double precision | Spirits Consumption |
| all\_alcohol | double precision | All-Alcohol Consumption |
| adult\_smoke | double precision | Percent of adults in the state who smoke. |

**sins\_county** table contains tobacco consumption data by state and county. It is from the University of Wisconsin County Health Rankings (<http://www.countyhealthrankings.org/>). They, in turn, got their data from the Behavioral Risk Factor Surveillance System survey, regularly performed by the CDC. This data set is based on surveys for the years 2006 to 2012.

|  |  |  |
| --- | --- | --- |
| **Column** | **Type** | **Description** |
| fips | text | State / County FIPS code |
| state | text | State |
| county | text | County |
| sample\_size | integer | Number of respondents in the survey for the county |
| smoking\_pct | double precision | Percentage of smokers |
| lcl95 | double precision | Upper 95% confidence level |
| ucl95 | double precision | Lower 95% confidence level |
| quartile | text | Quartile the county falls in nationwide for smokers |

**Appendix II**

Most of the filters below have changed from what was reported below.

First I get the results of the main query into R. Note that ‘lr\_table’ is the result of the main query listed above.

> fires <- dbGetQuery( conn, "select \* from lr\_table" )

Next, I set all the fire results to zero where the query above returns a Null value.

> for( i in c( "res\_all","res\_stc","res\_unc","res\_2","res\_3","res\_inj","res\_death" ) )   
fires[[ i ]][ is.na( fires[[ i ]] ) ] <- 0

The following commands set a few values that I need for the analysis in R.

> fires$region[ fires$state == "PR" ] <- "Puerto Rico"

> fires$region[ is.na( fires$region ) ] <- "West"

> fires$inc\_hh <- as.numeric( fires$inc\_hh )

> fires$inc\_hh <- log( fires$inc\_hh )

> for( i in c( "region", "state", "fd\_id", "fd\_size" ) ) fires[[ i ]] <- factor( fires[[ i ]] )

> fires$region <- relevel( fires$region, "West" )

The next command creates an empty set of filters that will be used in the model.

> fires$no.fire <- fires$giants <- fires$small <- fires$base <- fires$include <- TRUE

Then the *base* filter is created.

> fires$base <- with( fires, fd\_size %in% paste( "size\_", 3:9, sep="" ) & ! ( is.na( fd\_id ) | is.na( fd\_size ) ) )

> fires$base <- fires$base & ! is.na( fires$inc\_hh )

> fires$base <- fires$base & fires$f\_located > 0

> fires$base <- fires$base & ! is.na( fires$smoke\_cty )

And then the *giants* filter is created.

> u <- with( fires[ fires$base, ], list( pop=geoid[ pop > quantile( pop, .999 ) ],

hse.units=geoid[hse\_units > quantile(hse\_units, .999)],

males=geoid[males > quantile(males, .999)],

age\_45\_54=geoid[age\_45\_54 > quantile(age\_45\_54, .999)]))

> v <- NULL

> for( i in names( u ) ) v <- union( v, u[[ i ]] )

> fires$giants <- ! fires$geoid %in% v

> rm( i, u, v )

And finally the *small* filter is created.

> fires$small <- fires$dept\_incidents > 25 & ! is.na( fires$dept\_incidents )

The three filters are then combined into a single filter column.

> fires$include <- with( fires, base & small & giants )

The following commands define the “outlier” (here called “lcl”) filter. It will be used in some cases in the models that are run.

> dept <- fires[, c( 'geoid', 'year', 'fd\_id', 'dept\_incidents' ) ]

> ddd <- aggregate( fires$dept\_incidents, list( fd\_id=fires$fd\_id ),   
 function( x ) c( mean( x, na.rm=TRUE ), sd( x, na.rm=TRUE ) ))

> ddd$m <- ddd$x[,1]

> ddd$sd <- ddd$x[,2]

> dept$x <- NULL

> dept$m <- ddd$m[ match( dept$fd\_id, ddd$fd\_id ) ]

> dept$sd <- ddd$sd[ match( dept$fd\_id, ddd$fd\_id ) ]

> dept$lg <- ! ( is.na( dept$dept\_incidents ) | dept$dept\_incidents < dept$m – 2 \* dept$sd )

> fires$lcl <- dept$lg

> rm( dept )

Finally, this set of commands partitions the data set into thirds, and identifies one of the thirds as the test set. In spite of the names given, the “training” and “validation” subsets are combined together in the analysis and used as the training set, in spite of the names given.

> geoid <- unique( fires$geoid )

> geoid <- data.frame( geoid=geoid, v=floor( runif( length( geoid ) ) \* 3 ), set="",   
 stringsAsFactors=FALSE )

> geoid$set[ geoid$v == 0 ] <- "training"

> geoid$set[ geoid$v == 1 ] <- "validation"

> geoid$set[ geoid$v == 2 ] <- "test"

> geoid$set <- factor( geoid$set )

> fires$set <- geoid$set[ match( fires$geoid, geoid$geoid ) ]

> rm( geoid )

**Appendix III**

As before I create a master object that directs the running of each model. Its structure is described below.

“models”

input

“runs”

“model 1”

“model 2”

The ‘run’ tree has one of the following two structures. The tree can be of unlimited length, but it can only be two layers deep.

The first tree is this. This directs the system to run a separate model for each combination of department size and region.

“runs”

“size\_3”

“West”

fd\_size == “size\_3” & region == “West”

“Midwest”

fd\_size == “size\_3” & region == “Midwest”

“size\_9”

fd\_size == “size\_9”

…

“South”

fd\_size == “size\_3” & region == “South”

“Northeast”

fd\_size == “size\_3” & region == “Northeast”

The second tree is this. This directs the system to run a single model for the entire data set.

“runs”

“all”

TRUE

Similarly, the “models” subtree can have unlimited length. Each model in “models” has the following structure:

“models”

“fn”

c( library=”lme4”, ff = “glmer” )

“inputs”

“formula”

“data”

low\_risk ~ I( year – 2014 ) + pop …

fires

The “fn” node of the tree enables me to run different statistical functions (and dynamically load the relevant library) using the same routine. The “inputs” subtree is a named list of all the formal arguments that would be used to call the statistical function.

ONE of the model input structures is listed here:

> inputs$models$f.A\_ <- list( fn=c( library=”lme4”, ff=glmer” ),  
 inputs=list( formula=low\_risk ~ I(year - 2014) + pop + males +   
 hse\_units + black + amer\_es + other + hispanic   
 + age\_under5 + age\_5\_9 + age\_10\_14 + age\_15\_19   
 + age\_20\_24 + age\_25\_34 + age\_35\_44 +   
 age\_45\_54 + age\_55\_64 + age\_65\_74 + age\_75\_84   
 + age\_85\_up + vacant + renter\_occ + crowded +   
 units\_10 + mh + older + inc\_hh + svi +   
 smoke\_st + smoke\_cty + (1 | fd\_id),  
 data=quote( fires ), subset=quote( include & lcl ),  
 nAGQ=0, family=quote( poisson ),   
 offset=quote( log( f\_located ) ) ) )

The three formal arguments that are assumed to exist in all model calls are “formula”, “data”, and “subset.”

The R function that runs the analysis based on this structure is listed here. Note that while it is very similar to the version I sent before, it has changed some. It is still designed to perform bootstrapping when requested. However, I have not done that this round. If you call it with parameter n=0 (as I have consistently done for this round of analysis) it will simply run the model, produce results, and not bother with bootstrapping. Also note that the limitation of the analysis to the “training” set (again, strictly speaking those tracts tagged with either a “training” or “validation” label) is carried out transparently by the function. I have commented out several lines that make this more user-friendly when run from the command line. If you do run it from the command line, you might want to uncomment those lines.

Note that his has changed slightly from the previous version in order to improve error handling.

> fn.run

function( sets, n=0, sink=NULL )

{

require( boot )

require( utils )

# u <- Sys.time()

out <<- list()

if( ! is.null( sink ) )

{

if( is.character( sink ) ) ff <- file( sink, "w" )

else

{

warning( "the 'sink' term must be a character" )

ff <- NULL

}

}

else ff <- NULL

for( k in names( sets$models ) )

{

if( tolower( sets$models[[k]]$fn[ 'library' ] ) == "null" ) next

out[[k]] <<- list()

require( sets$models[[k]]$fn[ 'library' ], character.only=TRUE )

fn <- sets$models[[k]]$fn[ 'ff' ]

aa <- a <- sets$models[[k]]$inputs

subset.a <- a$subset

a$subset <- NULL

data <- a$data

for( i in names( sets$runs ) )

{

out[[k]][[i]] <<- list()

if( is.list( sets$runs[[i]] ) )

{

for( j in names( sets$runs[[i]] ) )

{

# u[1] <- Sys.time()

out[[k]][[i]][[j]] <<- list()

# cat( "Evaluating ", k, format( " model: ", width=16 - nchar( k ) ), i, " ", j, format( ":", width=11 - nchar( j ) ), sep="" )

aa$subset <- substitute( u & v & set %in% c( "training", "validation" ), list( u=sets$runs[[i]][[j]], v=subset.a ) )

if( ! is.null( ff ) ) sink( ff, type="message", append=TRUE )

tryCatch(

out[[k]][[i]][[j]]$model <<- do.call( fn, aa ),

error =function( e ) cat( "ERROR in Model: ", k, ", run ", i, "-", j, ". Message: ", e$message, "\n", sep="", file=stderr() ),

message=function( e ) cat( "MESSAGE in Model: ", k, ", run ", i, "-", j, ". Message: ", e$message, "\n", sep="", file=stderr() )

)

if( ! is.null( ff ) ) sink( type="message" )

if( n > 0 )

{

dta <- do.call( "subset", list( x=data, subset=aa$subset ) )

# pb <- winProgressBar( title=paste( "Bootstrapping ", k, " model: ", i, " ", j, ": ", n, " iterations", sep="" ), label="0", max=n )

out[[k]][[i]][[j]]$boot <<- boot( dta, bbb, R=n, strata=dta$fd\_id, a=a, ff=ff, fn=fn, pb=pb, nme=names( fixef( out[[k]][[i]][[j]]$model ) ) )

# close( pb )

}

# u[2] <- Sys.time()

# cat( "Elapsed time:", format( u[2] - u[1] ), "\n" )

}

}

else

{

# u[1] <- Sys.time()

# cat( "Evaluating ", k, format( " model: ", width=16 - nchar( k ) ), i, " all models:", sep="" )

aa$subset <- substitute( u & v & set %in% c( "training", "validation" ), list( u=sets$runs[[i]], v=subset.a ) )

if( ! is.null( ff ) ) sink( ff, type="message", append=TRUE )

tryCatch(

out[[k]][[i]]$model <<- do.call( fn, aa ),

error =function(e) cat( "ERROR in Model: ",k,", run ",i,"-All. Message: ",e$message,"\n",sep="",file=stderr()),

message=function(e) cat("MESSAGE in Model: ",k,", run ",i,"-All. Message: ",e$message,"\n",sep="",file=stderr() )

)

if( ! is.null( ff ) ) sink( type="message" )

if( n > 0 )

{

dta <- do.call( "subset", list( x=data, subset=aa$subset ) )

# pb <- winProgressBar( title=paste( "Bootstrapping ", k, " model: ", i, " all models: ", n, " iterations", sep="" ), label="0", max=n )

out[[k]][[i]]$boot <<- boot( dta, bbb, R=n, strata=dta$fd\_id, a=a, ff=ff, fn=fn, pb=pb, nme=names( fixef( out[[k]][[i]]$model ) ) )

# close( pb )

}

# u[2] <- Sys.time()

# cat( "Elapsed time:", format( u[2] - u[1] ), "\n" )

}

}

}

if( ! is.null( ff ) ) close( ff )

}

The next function takes the output from the fn.run function and produces MSE estimates for the test set. Its definition is listed here.

This function, too, has changed. This allows it to handle binomial models.

> fn.test

function( input, output, subset=NULL )

{

# Test to see if the data and dependent variables are all identical.

# If not, throw an error

x <- unlist( lapply( input$models, function( x ) as.character( x$inputs$data ) ) )

dta <- x[1]

if( ! all( dta == x ) ) stop( "data are not all identical. Try breaking up the input and output files." )

*x <- unlist( lapply( input$models, function( x ) as.character( x$inputs$formula[ 2 ] ) )* )

y <- x[1]

if( ! all( y == x ) ) stop( "Dependent variables are not all identical. Try breaking up the input and output files." )

rm( x )

if( is.null( subset ) )

{

x <- unlist( lapply( input$models, function( x ) as.character( x$inputs$subset ) ) )

if( ! all( x[1] == x ) ) warning( "The subsets are not all identical. Using the first. Try specifying the subset you want.")

rm( x )

subset <- input$models[[ 1 ]]$inputs$subset

}

if( is.list( subset ) )

{

old.res <- subset

subset <- old.res$subset

results <- old.res$results

if( y != old.res$lhs ) stop( "When ‘subset’ is the old results list, then the dependent variables must match." )

}

new.data <- do.call( "subset", list( x=get( dta ), subset=substitute( a & set == "test", list( a=subset ) ) ) )

if( ! exists( "old.res" ) )

{

results <- new.data[ , c( "year", "geoid", "state", "region”, "fd\_id", "fd\_size" ) ]

results$dept.new <- as.character( NA )

*if( deparse( input$models[[1]]$inputs$family ) == "binomial" )*

*{*

*tmp.y <- eval( parse( text=y ), envir=new.data )*

*results[[ y ]] <- tmp.y[,1] / ( tmp.y[,1] + tmp.y[,2] )*

*}*

*else results[[ y ]] <- eval( parse( text=y ), envir=new.data )*

}

vars <- NULL

for( k in names( input$models ) )

{

if( tolower( input$models[[k]]$fn["library"] ) == "null" ) next

vars <- c( vars, k )

require( input$models[[k]]$fn["library"], character.only=TRUE )

results[[ k ]] <- as.numeric( NA )

for( i in names( input$runs ) )

{

if( is.list( input$runs[[i]] ) )

{

for( j in names( input$runs[[i]] ) )

{

x <- eval( input$runs[[i]][[j]], envir=new.data )

if( any( x ) )

{

if( is.null( output[[k]][[i]][[j]]$model ) )

{

z <- as.numeric( NA )

warning( paste( "WARNING: Model ", k, " run ", i, "-", j, ": No model results were found.", sep="" ) )

}

else tryCatch(

{

if( input$models[[k]]$fn[“library”] == “lme4” )

{

z <- predict( output[[k]][[i]][[j]]$model,newdata=new.data[x,],type="response",allow.new.levels=TRUE )

x1 <- results$fd\_id %in% row.names( ranef( output[[k]][[i]][[j]]$model )$fd\_id )

results$dept.new[ x & ! x1 ] <- paste( results$dept.new[ x & ! x1 ], k, sep=";" )

}

else if( input$models[[k]]$fn["library"] == "glmnet" )

{

d.f <- model.frame( formula=input$models[[k]]$inputs$formula, data=new.data[x,], na.action=na.pass )

off <- eval( input$models[[k]]$inputs$offset, new.data[x,] )

z <- predict( output[[k]][[i]][[j]]$model, newx=model.matrix( terms( d.f ), d.f ), offset=off )

}

else

{

z <- predict( output[[k]][[i]][[j]]$model, newdata=new.data[x,], type="response" )

}

},

error=function( e ) stop( paste( "ERROR: Model ", k, " run ", i, "-All: ", e$message, sep="" ) )

)

results[[ k ]][x] <- z

}

}

}

else

{

x <- eval( input$runs[[i]], envir=new.data )

if( any( x ) )

{

if( is.null( output[[k]][[i]]$model ) )

{

z <- as.numeric( NA )

warning( paste( "WARNING: Model ", k, " run ", i, "-All: No model results were found.", sep="" ) )

}

else tryCatch(

{

if( input$models[[k]]$fn["library"] == "lme4" )

{

z <- predict( output[[k]][[i]]$model, newdata=new.data[x,], type=”response”, allow.new.levels=TRUE )

x1 <- results$fd\_id %in% row.names( ranef( output[[k]][[i]]$model )$fd\_id )

results$dept.new[ x & ! x1 ] <- paste( results$dept.new[ x & ! x1 ], k, sep=";" )

}

else if( input$models[[k]]$fn["library"] == "glmnet" )

{

d.f <- model.frame( formula=input$models[[k]]$inputs$formula, data=new.data[x,], na.action=na.pass )

off <- eval( input$models[[k]]$inputs$offset, new.data[x,] )

z <- predict( output[[k]][[i]]$model, newx=model.matrix( terms( d.f ), d.f ), offset=off )

}

else

{

z <- predict( output[[k]][[i]]$model, newdata=new.data[x,], type="response" )

}

},

error=function( e ) stop( paste( "ERROR: Model ", k, " run ", i, "-All: ", e$message, sep="" ) )

)

results[[ k ]][x] <- z

}

}

}

}

results$dept.new <- sub( "^NA;", "", results$dept.new )

# results$dept.new[ is.na( results$dept.new ) ] <- ""

s <- results[ , vars ]

s <- ( s – results[[ y ]] ) ^ 2

se <- sqrt( colSums( s, na.rm=TRUE ) / apply( s, 2, function( x ) length( x[ ! is.na( x ) ] ) ) )

if( exists( "old.res" ) ) list( lhs=y, subset=subset, se=c( old.res$se, se ), results=results )

else list( lhs=y, subset=subset, se=se, results=results )

}

In calling the fn.test function, I used two versions of the “subset” formal argument. When I did not include “lcl” (i.e., the “outlier” filter) as a filter on the model runs, I used the following subset for the test data:

> test.n <- fn.test( input, output, subset=quote( include & fd\_size %in% c( “size\_6”, “size\_7”,   
 “size\_8”, “size\_9”)))

When I did include “lcl” as a filter on the model runs, I used the following subset for the test data:

> test.n <- fn.test( input, output, subset=quote( lcl & include & fd\_size %in% c( “size\_6”,   
 “size\_7”, “size\_8”, “size\_9”)))

The naïve model was implemented using the following function.

> naive

function( test )

{

if( ! is.list( test ) ) stop( "this is not the output of the fn.test function" )

if( any( names( test ) != c( "lhs", "subset", "se", "results" ) ) ) stop( "this is not the output of the fn.test function" )

x <- test$results[, c( "year", "geoid", test$lhs ) ]

x$ndx <- paste( x$geoid, x$year, sep="." )

x$match <- paste( x$geoid, x$year - 1, sep="." )

x$naive <- x[[ test$lhs ]][ match( x$match, x$ndx ) ]

if( "f\_located" %in% names( test$results ) )

{

x$f\_located <- test$results$f\_located

x$naive <- x$naive \* x$f\_located / x$f\_located[ match( x$match, x$ndx ) ]

}

test$results$naive <- x$naive

s <- ( test$results[[ test$lhs ]] - test$results$naive ) ^ 2

test$se <- c( test$se, naive=sqrt( sum( s, na.rm=TRUE ) / length( s[ ! is.na( s ) ] ) ))

test

}