1.1 Feature Selection

- 1.1.0 Feature Scaling
- 1.1.1 Factor Analysis
- 1.1.2 Mutual Information
- 1.1.3 Decision Trees

Code hiding script from Damian Kao (http://blog.nextgenetics.net/?e=102).

Out[1]: Code hidden for easier reading: toggle on/off.

Load train/test data. Recall the date ranges (inclusive):

- Training Data: Sept 2011 - Apr 2014
- Test Data: Sept 2014 - Oct 2014

```
Train: N = 17075, P(CriticalFound|X) = 0.141
Test:  N = 1637, P(CriticalFound|X) = 0.158
```

Load the logistic regression model used by the City. This confusion matrix serves as a baseline for comparison to other models.

Note: Unless stated otherwise, all models are fitted to the training data and evaluated on the test data.

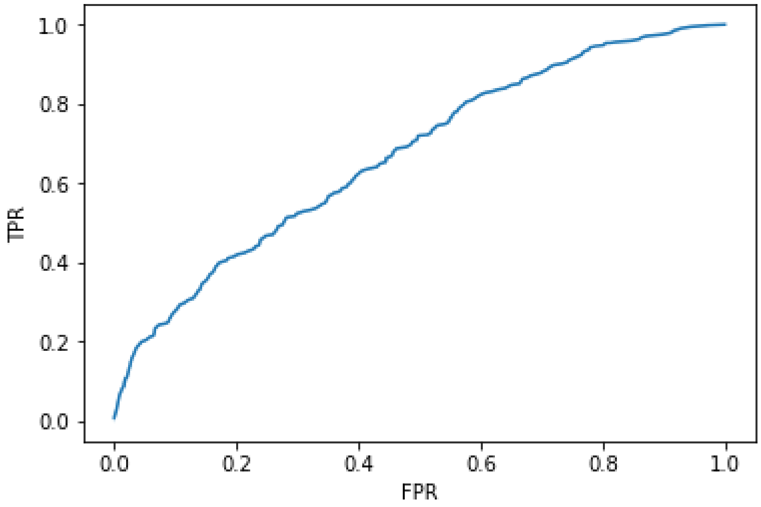
Table 1. Confusion matrix for the logistic regression model used by the City.

```
Logistic Regression
------------------
F1 Score = 0.08664
Precision = 0.63158
Recall = 0.04651
```

Out[4]:

	Predicted +	**Predicted -**
Actual +	12	246
Actual -	7	1372

```
AUROC = 0.672
```

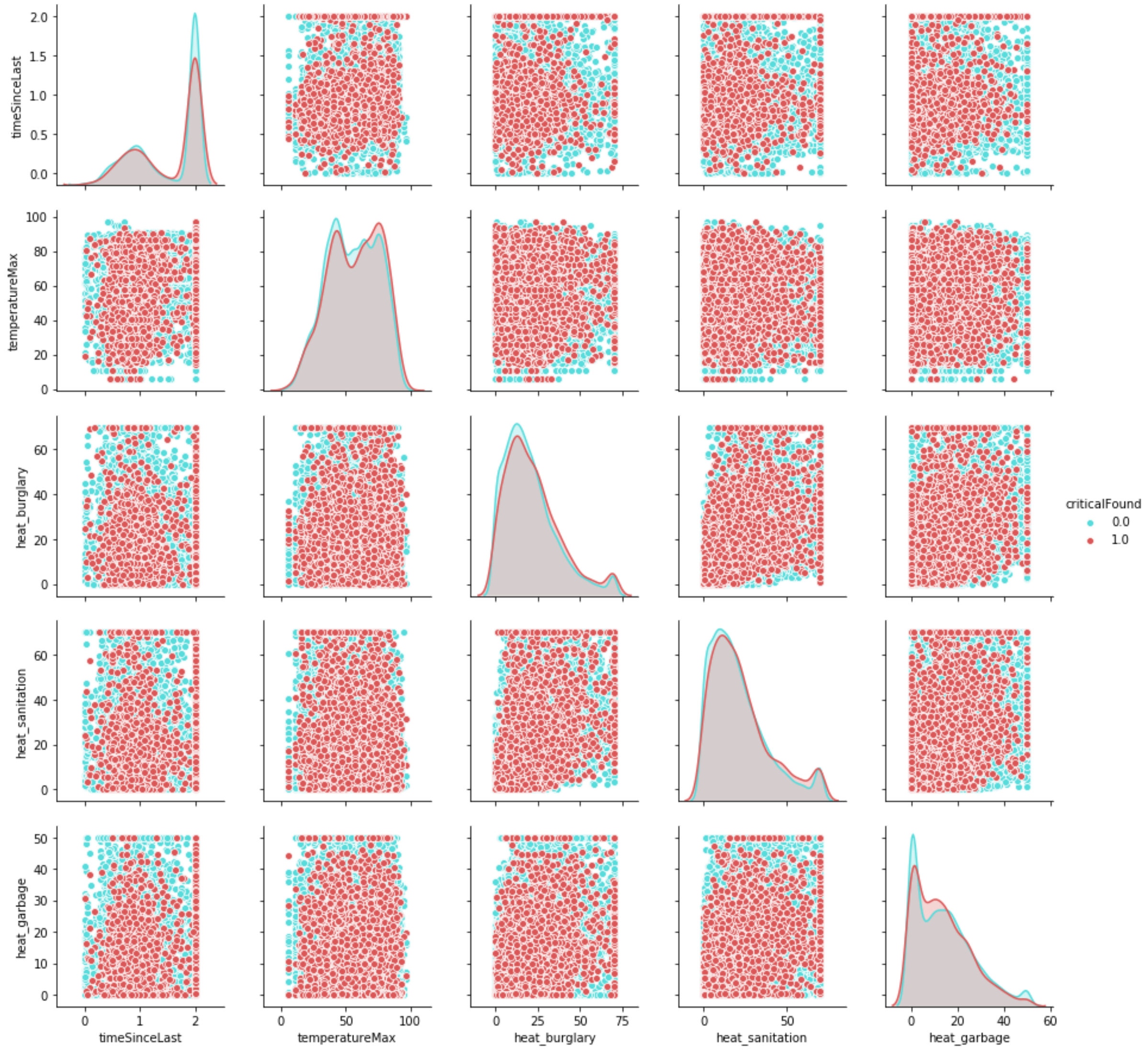


1.1.0 Feature Scaling

Figure 1 shows the scatter plot matrix. Hue indicates whether or not a critical violation was found (blue: none, red: at least one). This figure is not very helpful for binary features, so it only includes the relationships among the continuous features.

Note: In theory, `ageAtInspection` should be a continuous feature, but in this dataset, it appears to only have two distinct values.

Figure 1. Scatter plot matrix for the continuous features, with hue for the target variable.



The scatterplot matrix does not show any clear relationships between the continuous predictor variables.

I use `StandardScaler` from scikit-learn to z-score each continuous feature.

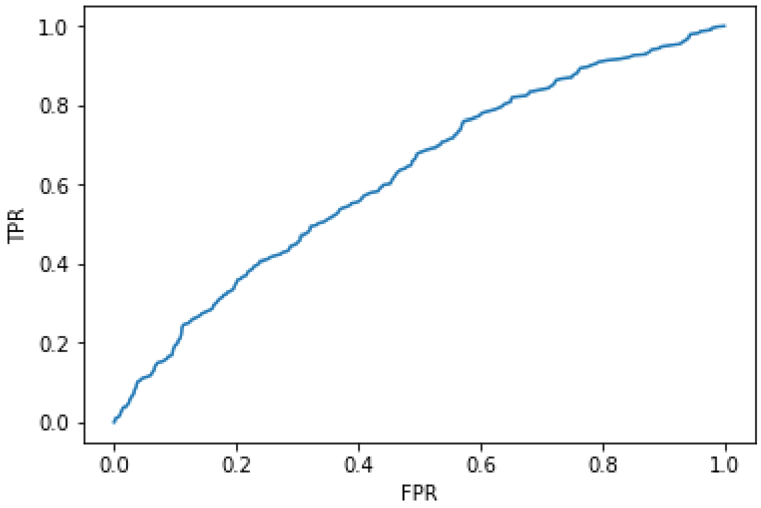
Scaling the continuous features (not including `ageAtInspection`) improves recall at the cost of precision, with a much better overall F1-score. However, AUC is worse compared to the City model.

```
Logistic Regression (Scaled Continuous Features)
------------------
F1 Score = 0.32959
Precision = 0.21728
Recall = 0.68217
```

```
Out[9]:
```

	Predicted +	**Predicted -**
Actual +	176	82
Actual -	634	745

```
AUROC = 0.623
```

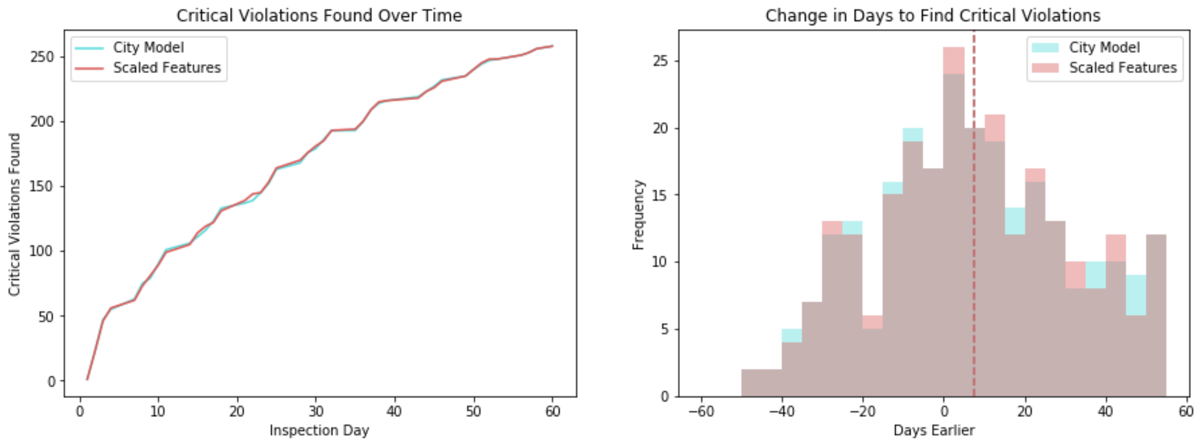


As expected, this scaling leads to changes in model coefficients. Percent change shows the difference as a percentage of the original coefficient.

```
Out[11]:
```

	City Model	Scaled Model	Percent Change	Sign Change
Inspector_blue	0.950	0.949	-0.052	False
Inspector_brown	-1.306	-1.631	24.919	False
Inspector_green	-0.244	-0.293	20.095	False
Inspector_orange	0.202	0.178	-11.832	False
Inspector_purple	1.555	1.572	1.121	False
Inspector_yellow	-0.697	-0.794	13.966	False
pastSerious	0.302	0.311	3.018	False
pastCritical	0.427	0.421	-1.599	False
timeSinceLast	0.097	0.059	-38.741	False
ageAtInspection	-0.164	-0.182	10.801	False
consumption_on_premises_incidental_activity	0.411	0.417	1.471	False
tobacco_retail_over_counter	0.171	0.210	23.363	False
temperatureMax	0.005	0.086	1652.557	False
heat_burglary	0.002	0.029	1078.894	False
heat_sanitation	0.002	0.031	1649.857	False
heat_garbage	-0.004	-0.043	957.249	False

Ultimately, scaling the continuous features does not seem to improve the metrics the city cares about.



Out[12]:

	Model	First Half	Mean Change	Std. Change
1	Scaled Features	0.69	7.457	24.677
0	City Model	0.69	7.438	25.156

1.1.1 Factor Analysis

The goal of factor analysis is to resolve multicollinearity issues between the predictors.

To choose the number of composite features from factor analysis, figure 2 show the proportion of variance explained as the number of factors grows. It takes five factors to account for all variance. I referenced this StackOverflow post (https://stackoverflow.com/questions/41388997/factor-analysis-in-sklearn-explained-variance) for the code to calculate explained variance.

Figure 2. Proportion of variance explained by factors.

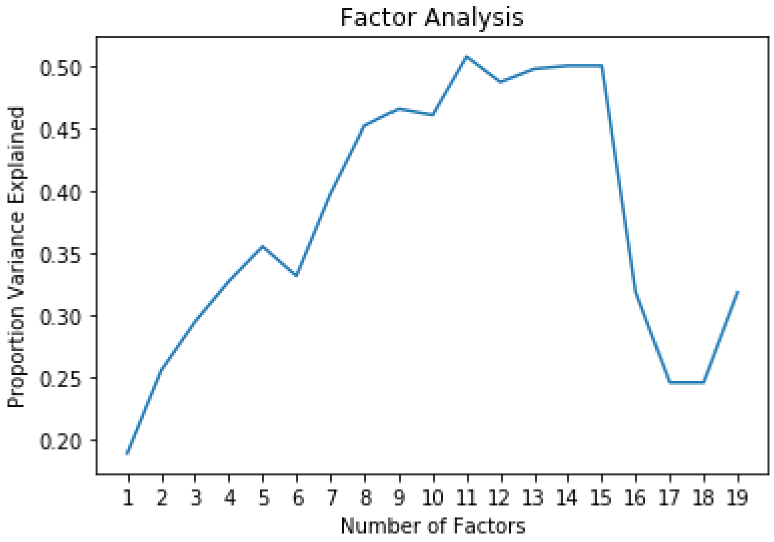


Figure 3 shows how each of the original features contributes to the factors.

Help: The loading matrix weights seem to be influenced by the scale of values. Should the continuous features be scaled before factor analysis? If so, what kind of scaling? What is the appropriate way to treat binary variables for factor analysis?

Figure 3. Loading matrix values for each composite feature.

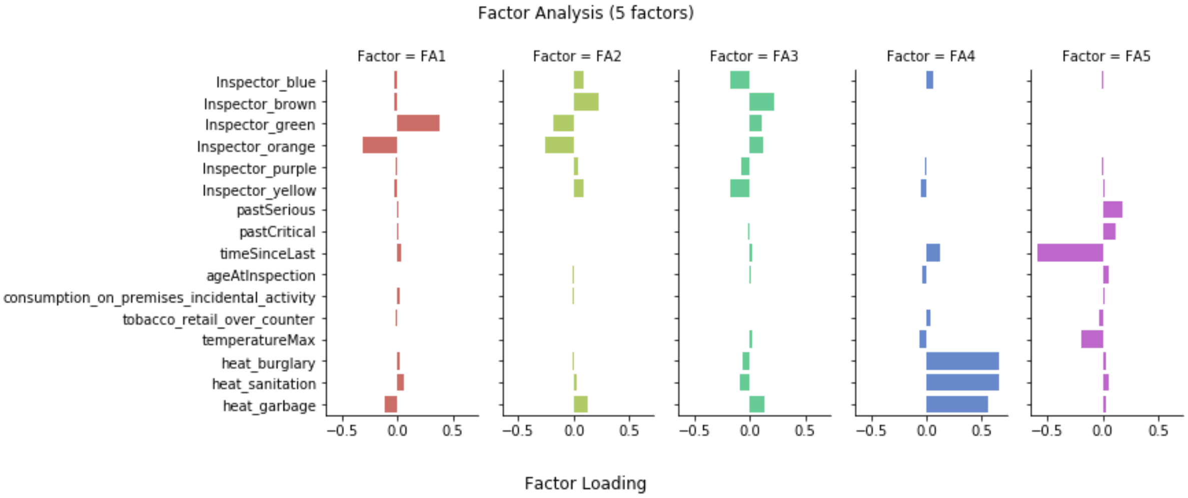


Figure 4. Inspections plotted according to the first two composite features, with hue for the target variable.

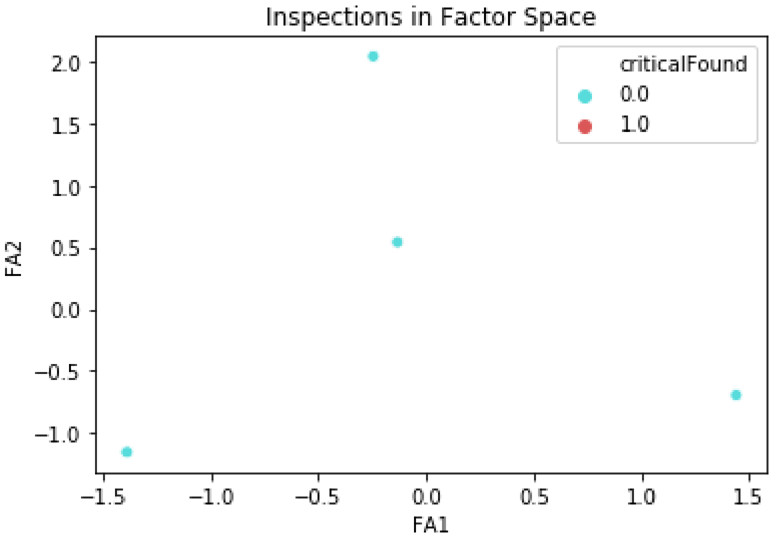
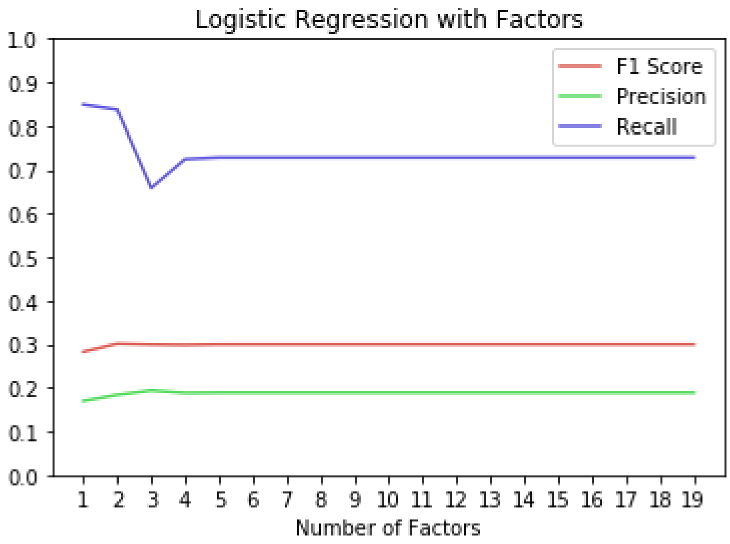


Figure 5. Evaluation scores for logistic regression using different numbers of composite features.



Based on figure 5, it seems best to use the first two composite features. F1 score and precission stay roughly the same, but recall starts to drop after two factors.

Table 2. Confusion matrix for logistic regression using two composite features from factor analysis.

```
F1 Score = 0.30105
Precision = 0.18352
Recall = 0.83721
```

Out[18]:

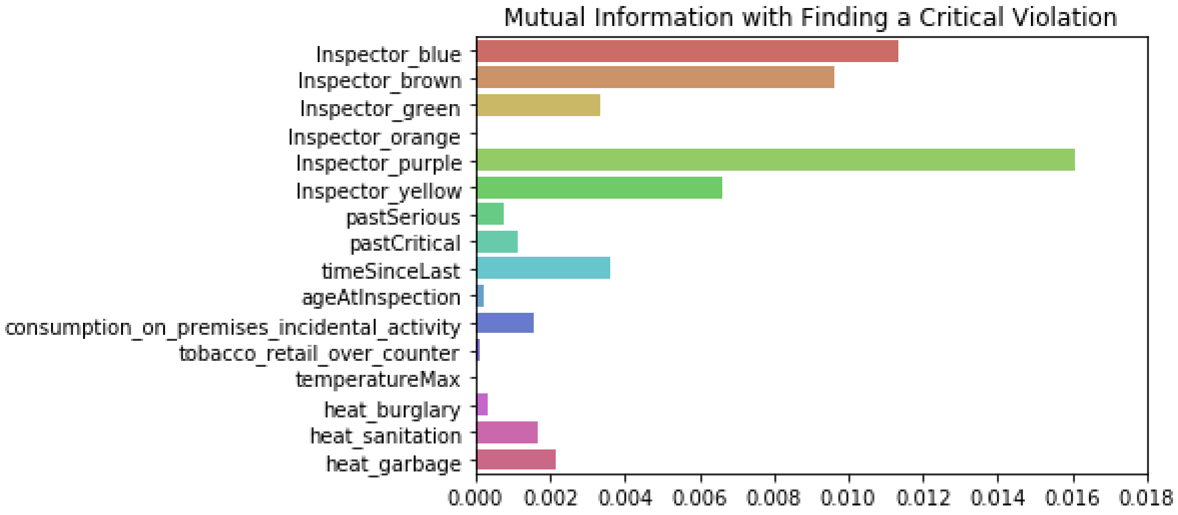
	Predicted +	Predicted -
Actual +	216	42
Actual -	961	418

Comparing table 1 and table 2, using the composite features improves F1 score and recall at the expense of precision.

1.1.2 Mutual Information

Mutual information measures how much each predictor variable reduces entropy in the target variable.

Figure 6. Mutual information scores between each predictor and finding a critical violation.



The six inspector cluster variables have the highest mutual information with the target variable.

Recall the relative order of each inspector cluster according to their hit rate, shown in table 3.

Table 3. Inspector clusters ordered by hit rate for finding critical violations. `Count` indicates the number of inspections each cluster of sanitarians conducted in the training set. `Coefficient` shows the coefficient for that cluster in the model used by City.

```
Out[20]:
```

	Hit Rate	Count	Coefficient
Inspector_purple	0.406	1174	1.555
Inspector_blue	0.265	2897	0.950
Inspector_orange	0.136	3769	0.202
Inspector_green	0.095	4595	-0.244
Inspector_yellow	0.058	2762	-0.697
Inspector_brown	0.024	1878	-1.306

Help: How should I interpret these mutual information values?

The highest value for mutual information is below `0.018`. What is the typical scale for mutual information?

In this case, I did not treat `ageAtInspection` as a discrete variable. Based on the scatter plot matrix from figure 1, should it have been grouped with the discrete variables?

Why does the order of the inspector clusters according to mutual information not match the order according to critical violation hit rate?

I expected that `inspector_brown` would have the highest mutual information. For example, if we knew only that an inspection was conducted by a sanitarian from the brown cluster, then the proportion of passing inspections would be `1 - 0.024 = 0.976`. So, if we always guessed that an inspection conducted by cluster brown would pass, then we would be correct 97.6% of the time.

However, it makes sense that `inspector_purple` has the highest mutual information. The hit rate for sanitarians in this cluster is closest to `0.5`, which is where entropy would be maximimized.

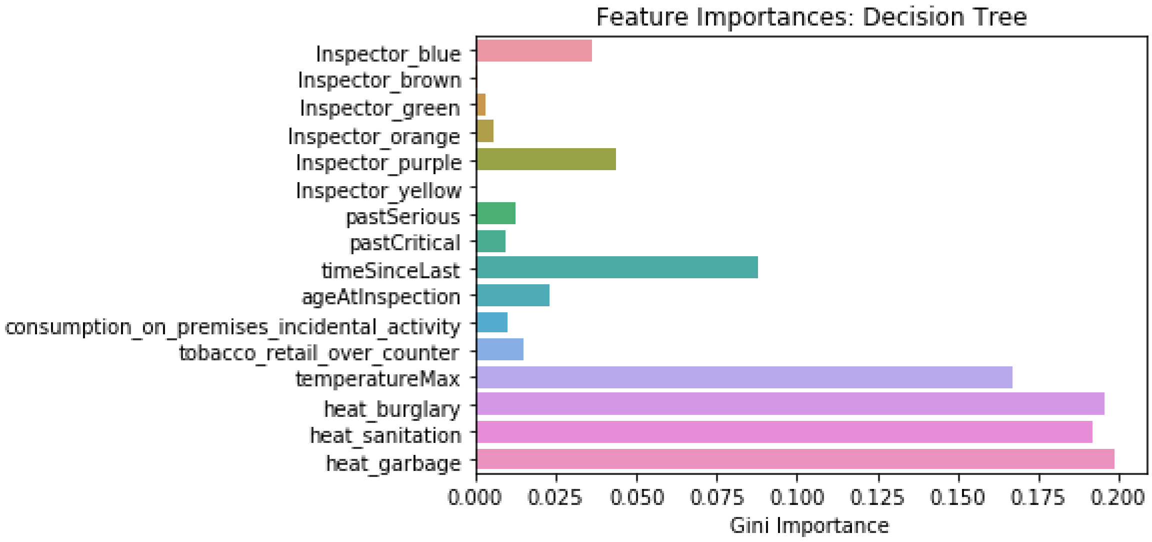
Why does `inspector_orange` have mutual information of zero?

1.1.3 Decision Trees

The goal of building decision trees for this task is to analyze how useful each feature is for splitting the data.

Gini importance measures the average decrease in entropy due to each feature.

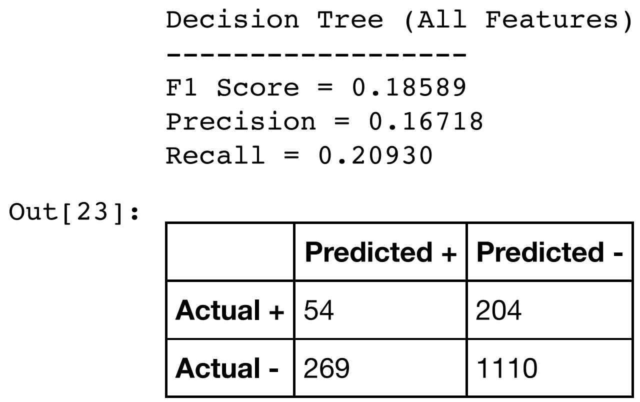
Figure 7. Gini importance for each feature when fitting a decision tree with no stopping condition.



Help: How should I interpret the feature importance scores?

Why do the inspector variables have lower Gini importance compared to `temperatureMax`, `head_burglary`, `heat_sanitation`, and `heat_garbage`?

Table 4. Confusion matrix for decision tree using all predictors and no stopping condition.

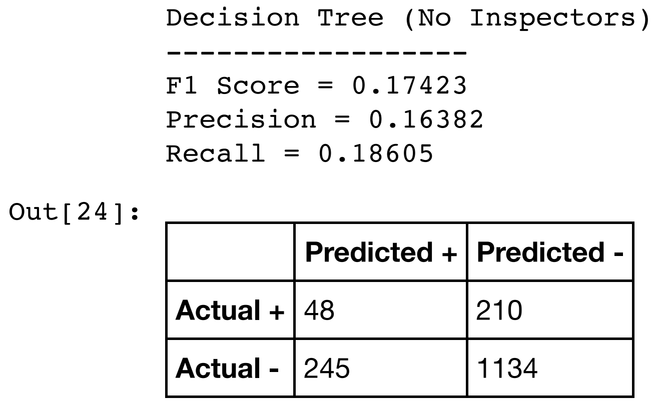


```
Decision Tree (All Features)
------------------
F1 Score = 0.18589
Precision = 0.16718
Recall = 0.20930
```

Out[23]:

	Predicted +	Predicted -
Actual +	54	204
Actual -	269	1110

Table 5. Confusion matrix for decision tree excluding inspector features and using no stopping condition.



```
Decision Tree (No Inspectors)
------------------
F1 Score = 0.17423
Precision = 0.16382
Recall = 0.18605
```

Out[24]:

	Predicted +	Predicted -
Actual +	48	210
Actual -	245	1134

Comparing tables 4 and 5 to table 1 (City model), both decision trees show higher scores for F1 and recall and lower scores for precision. Comparing tables 4 and 5 to table 2 (logistic regression with factor analysis), both decision trees have higher scores for precision but lower scores for F1 and recall.

Comparing table 4 to table 5, the decision tree that excludes the inspector variables appears to have higher precision and equivalent recall, leading to a higher overall F1 score.

Figures 8 through 11 show the evaluation metrics for varying the decision tree stopping conditions.

Figure 8. Evaluation metrics for decision trees with varying maximum depth. Inclusive range: min=1, max=50, step=1. Scores appear to stay level for depth values above 50, up to `max_depth=500`.

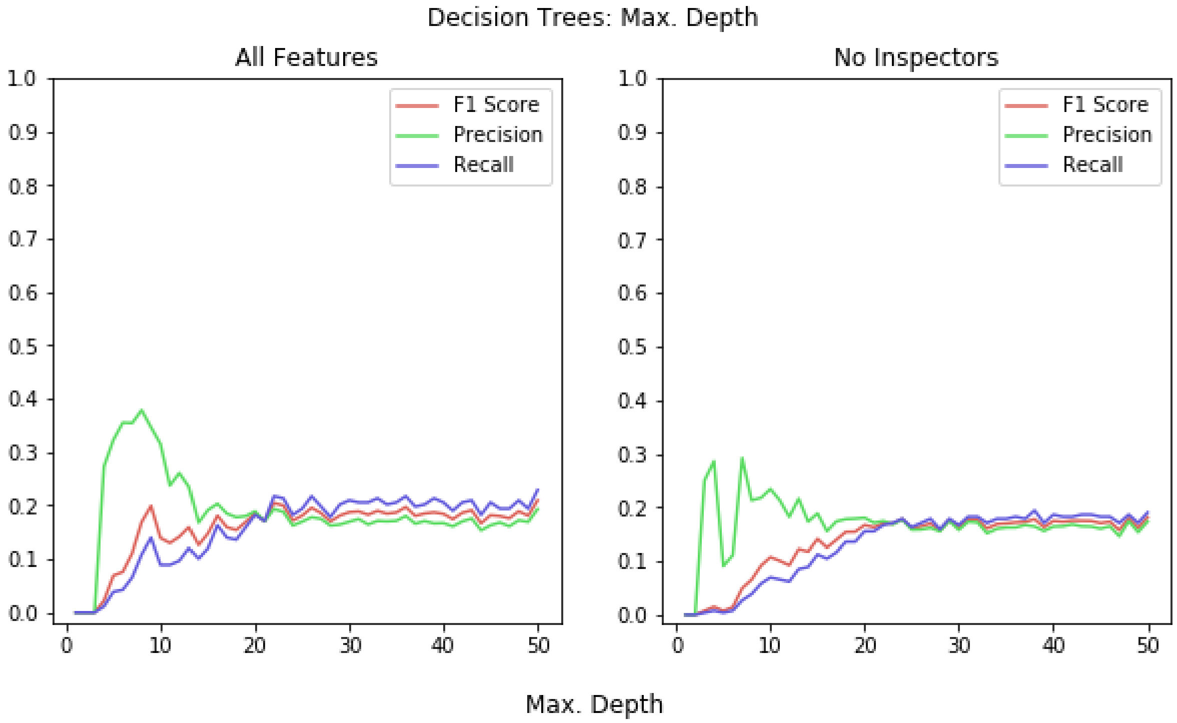


Figure 9. Evaluation metrics for decision trees with varying number of minimum samples to be a leaf node. Inclusive range: min=10, max=200, step=10.

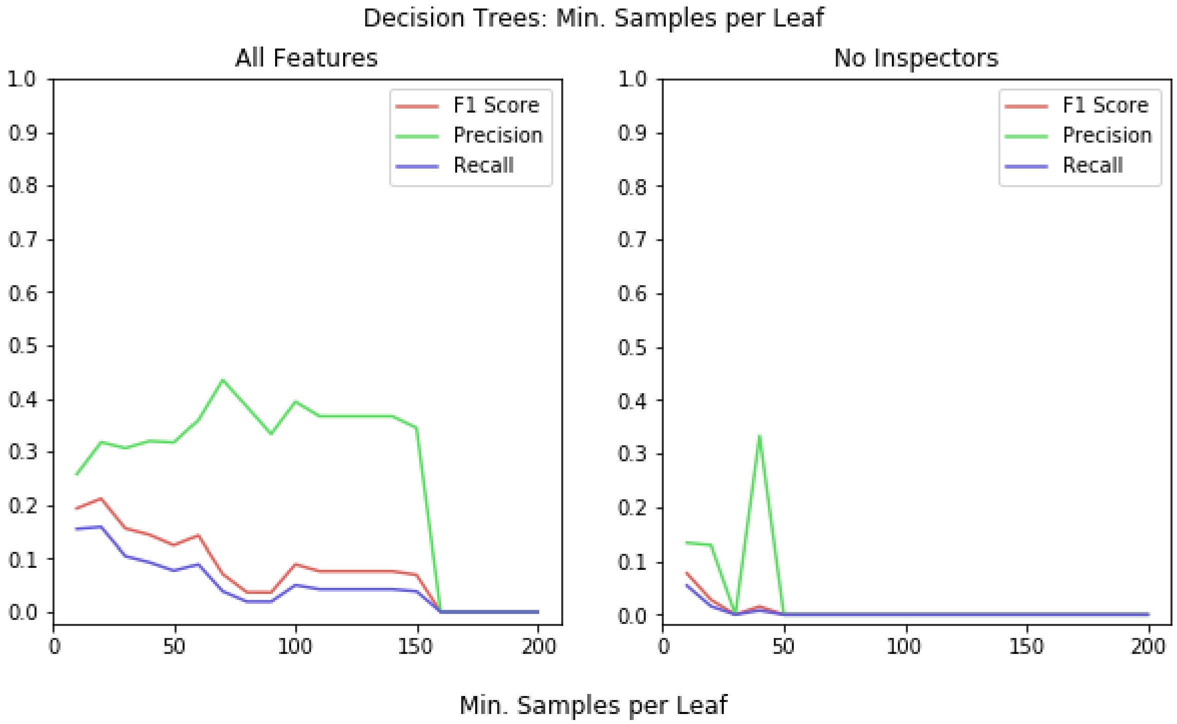


Figure 10. Evaluation metrics for decision trees with varying number of maximum leaf nodes. Leaf nodes chosen by order of relative information gain. Inclusive range: min=10, max=500, step=10.

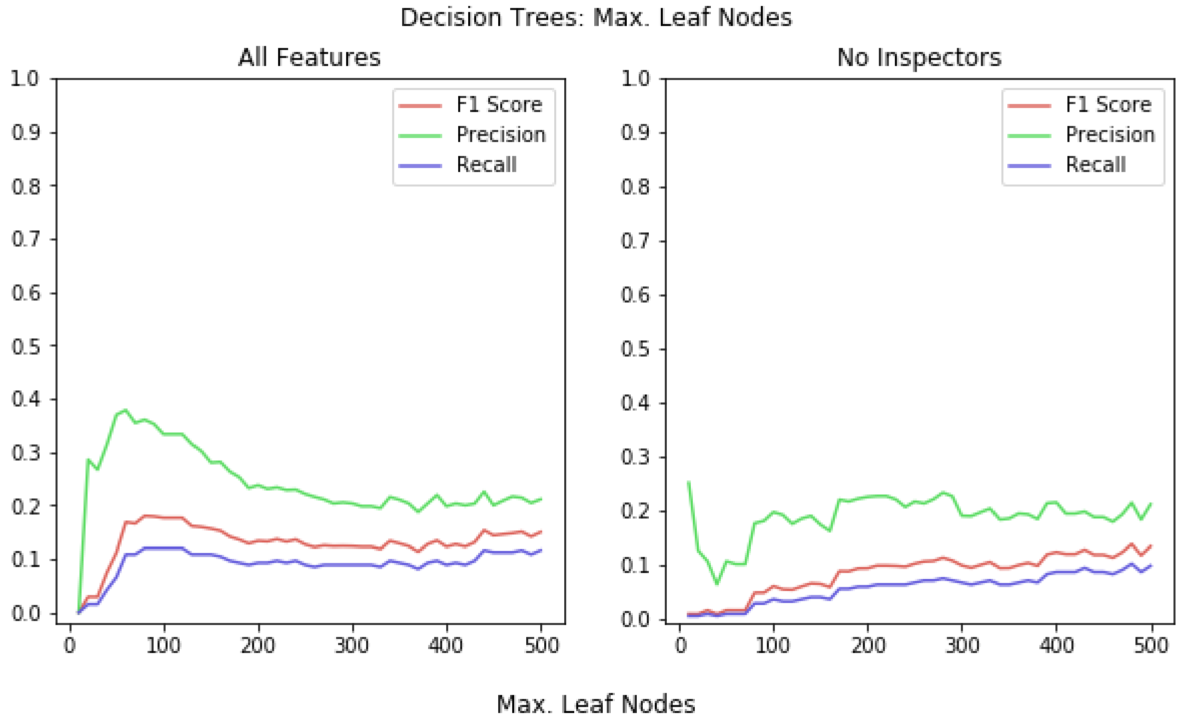
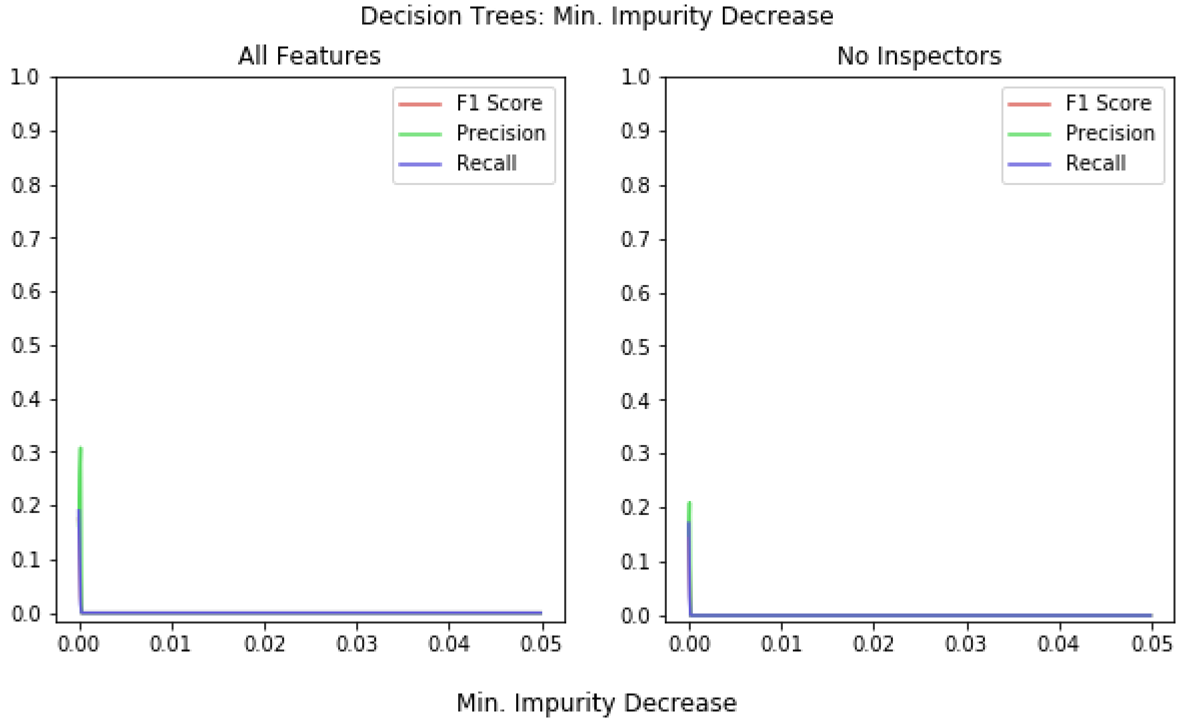


Figure 11. Evaluation metrics for decision trees with thresholds of minimum impurity decrease needed to make a split. Inclusive range: min=0, max=0.05, step=0.0001. Scores appear to stay level for values above 0.05, up to `min_impurity_decrease=0.5`.



Help: Why do higher thresholds for minimum impurity decrease appear not to improve performance?

Some features had Gini importance near `0.18`, are those influenced by splits further down in the tree?

To visualize a sample decision tree, I chose `max_depth=7` somewhat arbitarily. The tree is quite big, so at greater depths, it becomes harder to read the diagram. Producing the diagram is time-intensive, so a copy can be found in `figures/tree_height_7.png`.

Table 6. Confusion matrix for decision tree with maximum depth of 7.

```
Decision Tree (depth = 7)
------------------
F1 Score = 0.11111
Precision = 0.35417
Recall = 0.06589
```

Out[30]:

	Predicted +	**Predicted -**
Actual +	17	241
Actual -	31	1348

Decision tree visualization produced with the following code:

```
CLASSES = ["No Critical", "Critical Found"]
COLORS = ["#57D2DB", "#DB5E56"]
im = visualize_tree(tree_lim, PREDICTORS, CLASSES, COLORS)
imfile = "figures/tree_height_{}.png".format(tree_depth)
with open(imfile, "wb") as file:
    file.write(im.data)
im
```

To visualize decision tree in the notebook, run with:

```
jupyter notebook --NotebookApp.iopub_data_rate_limit=10000000000
```