Predicting Water Main Breaks

Mitchell Sailsbery and Connor Robertson *

April 2018

Abstract

Each year, thousands of city water pipes crack, buckle, or tear due to age, corrosion, and external loads. These breaks incur enormous repair expenses, cause property damage, and waste treated water. In this project, we constructed a machine learning model that can be used to forecast these main breaks before they happen. To do so, we cleaned, engineered, and added to the data given to us by the public works departments of Springville and Orem, Utah. Our model demonstrated a greater forecasting accuracy than common "age-based" prediction methods. Specifically we achieved an accuracy of 15% on what were predicted to be high-risk water mains. Using the predictions from this model, cities can optimize their pipe replacement schedules to avoid the most breaks.

Introduction

Water is a key component of personal and commercial activity for everyone in the world. However, constant access to clean water is frequently taken for granted in the developed world. As a result, very little funding has been directed to the improvement and replacement of the water delivery infrastructure in the United States. Now, a majority of cities across the country utilize old and decrepit pipe delivery systems and deal with an increasing number of pipe breaks. On average, each break costs $3000 in pipe repairs, $3000 in lost water, and sometimes far more in property damages. Additionally, increased breaks cause an increase in water service outages for consumers, reducing the city's reputation.

The increase in water breaks over the past few decades has not gone unnoticed. Civil engineering and city planning researchers have worked hard to physically model the decay of water mains and have examined the statistical trends in pipe breaks to better anticipate them. The results of their studies have identified many factors which contribute to decay and main failure in water systems. These include material, age, soil acidity, temperature shifts, traffic load, soil moisture content, water treatment, and many more. However, until recently, data collection in city public works organizations has been minimal and there has been little large scale implementation of models created by researchers. In the last two decades, the growth and development of GIS systems has enabled cities to digitally store their water system's information and carefully log previous breaks. As a result, the data required to implement the researched models can now be gathered, generated, or approximated and an accurate predictive model can be constructed.

In the most recent years, a few attempts have been made by researchers to use machine learning for water main break prediction.[1] These projects have attempted to predict either the total number

*Project enabled by the City of Springville and the City of Orem

[1]https://www.sciencedirect.com/science/article/pii/S0895717710000051

of breaks that will occur in the next year or identify specific breaks in the system.[2] Though well considered, the majority of the machine learning models used in this research were naively implemented without hardly any tuning of parameters or careful consideration of architecture. As a result, few of the models found much success beyond the earlier statistical methods, which are used here as a baseline. We present a model that selects the most high-risk water mains in the city. We use more sophisticated methods and parameter tuning to capture the subtleties of the data and to incorporate more realistic engineered variables. Our model has demonstrated a .15 increase in accuracy over the baseline models of regression based on age or previous breaks and promises further accuracy with more development. Note here that accuracy in this context refers to the proportion of predicted high-risk water mains that actually break.

Data Collection, Cleaning, and Feature Engineering

Water system data was collected from both Springville and Orem in Utah County. These cities provided the location, length, and diameter of their city water mains. Springville provided additional information on the approximate installation year and material of some of their mains. In addition to that data, approximations for pipe age and soil acidity were incorporated by combining the collected data with a dataset from the state of Utah that documents residential construction on taxable land parcels[3] and a dataset of soil surveys in Utah from the USDA[4].

Once the data was collected and compiled, additional processing allowed the pipe materials of a subset of pipes in Orem to be collected from installation notes. The dataset of documented breaks then needed to be converted from addresses to latitude and longitude and then to a stateplane coordinate system in order to be assigned to specific pipes. The code for the connecting of breaks is included in `assign_breaks.py`. The variables were then normalized in order to be better weighted in the machine learning models.

Cleaning

In both city's datasets, various extreme values needed to be removed and almost everything needed to be converted from a string of some sort. Also, to use missing data categorically, missing values for installation year, material, and diameter were replaced with values such as "UNK". Since each row in the dataset represents a physical pipe in the system, replacing missing data with categories allowed all rows to be kept without disturbing the training process of the machine learning methods.

Additional Dataset Information

In addition to the city datasets, a dataset on Utah tax parcels and a soil dataset for the Central Utah region were collected. Both of these datasets required transformation into stateplane coordinates in order to be able to be matched up with the pipes. The tax parcel dataset contained the dates in which current buildings were constructed which offered an approximation of the age of nearby pipes. Thus, an approximated age feature could be constructed by finding the closest edge point to each pipe and assigning that pipe the building construction date. The soil data contained pH readings which allowed for an approximation of the soil acidity around each pipe. The Shapely Python package allowed each pipe to be identified as contained in a shape and then assigned an approximate pH value.

[2] https://ascelibrary.org/doi/abs/10.1061/(ASCE)WR.1943-5452.0000354
[3] https://gis.utah.gov/data/cadastre/parcels/
[4] https://websoilsurvey.nrcs.usda.gov/app/

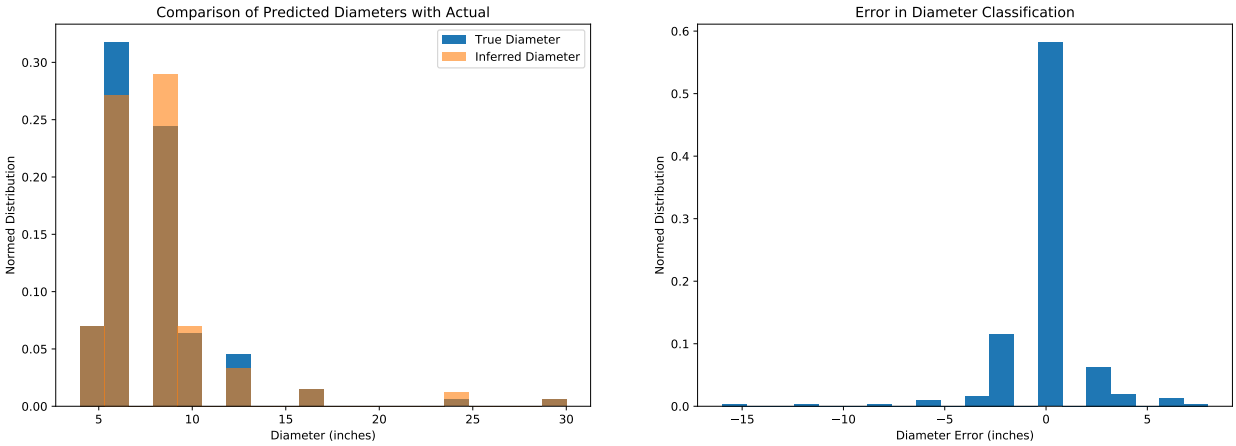


Figure 1: Comparisons of the inferred and true diameter distributions for a test set as well as the associated error. distributions of the inferred diameter with

Feature Engineering

In order to more realistically represent the contributing factors to pipe corrosion and failure, we computed several new variables for the dataset. We assigned variables to capture if the pipe considered was metal, how it could respond to surface load, and how far it was from another break. In order to account for age as an ordinal feature and material as a categorical feature we converted these columns into several columns of dummy variables.

Data Imputation

Pipe diameter was determined as an important attribute via linear regression, and its missing values were imputed. The initial imputation model used a specially designed metric that put in-line pipes closer to the pipe in question than pipes at a perpendicular angle to on another. This embodied the assumption that in-line pipes are usually installed together and are therefore more similar to each other than their perpendicular neighbors. Testing of this first method showed around 70% exact accuracy.

To attempt improvement on this method, various machine learning methods were used to determine the diameter. These methods included random forests, linear regression, and support vector machines. However, their testing demonstrated only 50% accuracy.

Due to the lack of success of the machine learning methods to impute diameter, we concluded that diameter was mostly independent of the other variables and that the "in-line" metric would be more successful in imputing the missing diameters. To add further information to the metric, extra distance was added between pipes which did not share the same material and age attributes. Thus, the nearest pipe to the pipe in consideration would be a pipe that is in-line with it and which shares its material and age. After modifying the metric, the diameter imputation yielded around 84% exact inference and 95% of inference were within 2 inches of the correct value. Figure 1 demonstrates a normed histogram of the error for the misclassified diameters.

The code used for the imputation is included in the file `inference_methods.py` and its dependencies.

Methods

Various machine learning algorithms were considered to give accurate results. However, in addition to standard algorithms, particular methodology was considered to overcome the difficulties posed by the data.

Rare Event Methods

Only about 1% of the pipes in cities have broken in the past ten years of data collection. Due to this rarity, predicting the breaking of the correct pipes is a difficult problem to overcome. For example, most out of the box methods will predict that no water mains in a city will break and will get a score of 98% accuracy (and the model has vanishing coefficients, so it qualifies as simple).

To overcome this challenge, new metrics and methods were considered to accurately calculate and portray the success of the model.

F1 Score

Instead of standard accuracy measures, the algorithms' performance was evaluated using the F1 score (also referred to as F score). The F1 score has many equivalent formulations, but can be represented as,

$$\frac{\text{True Positives}}{2 * \text{True Positives} + \text{False Positives} + \text{False Negatives}}$$

This score doesn't reward True Negatives, which was the exact problem plaguing the normal accuracy metric. (accurately predicting that a water main does not break) However, it values the True Positives which are the most actionable results that can be used by a city in avoiding water main breaks. Unfortunately, due to syntax and code structure, the F score was not able to be used as an objective function in standard algorithms. Its implementation is included in the file `Metrics_Success.py`.

Balancing Data Weights

Most machine learning methods treat all sorts of errors in classification with the same weight by default. Luckily, there are some of the out-of-the-box machine learning algorithms that have an easy way to account for rare events.

In the case of scikit-learn's Random Forest algorithm, the *class-weight* parameter allows for the identification of classes for which a correct prediction should weigh more. If this parameter is set to *balanced*, each class weighs an amount inversely proportional to its incidence. Thus, rare classes (such as a water main break) weigh a good deal more than common ones (such as a non-break). This allows rare classes to be strongly represented in the training of an algorithm.

Choosing a Classification Probability Cutoff

Even with the balancing of classes, many classification methods do not predict enough main breaks. To avoid this, the algorithms were used to output the probability of classification rather than the classification itself. A classification cutoff point, τ, was then determined to classify all pipes with a greater probability as a predicted break.

To determine the classification cutoff, a variant of the ROC curve was used to demonstrate the F score with different values of τ. After running each method on various test and train splits, the F scores at each τ value were averaged and the maximizer of the curve was determined. This

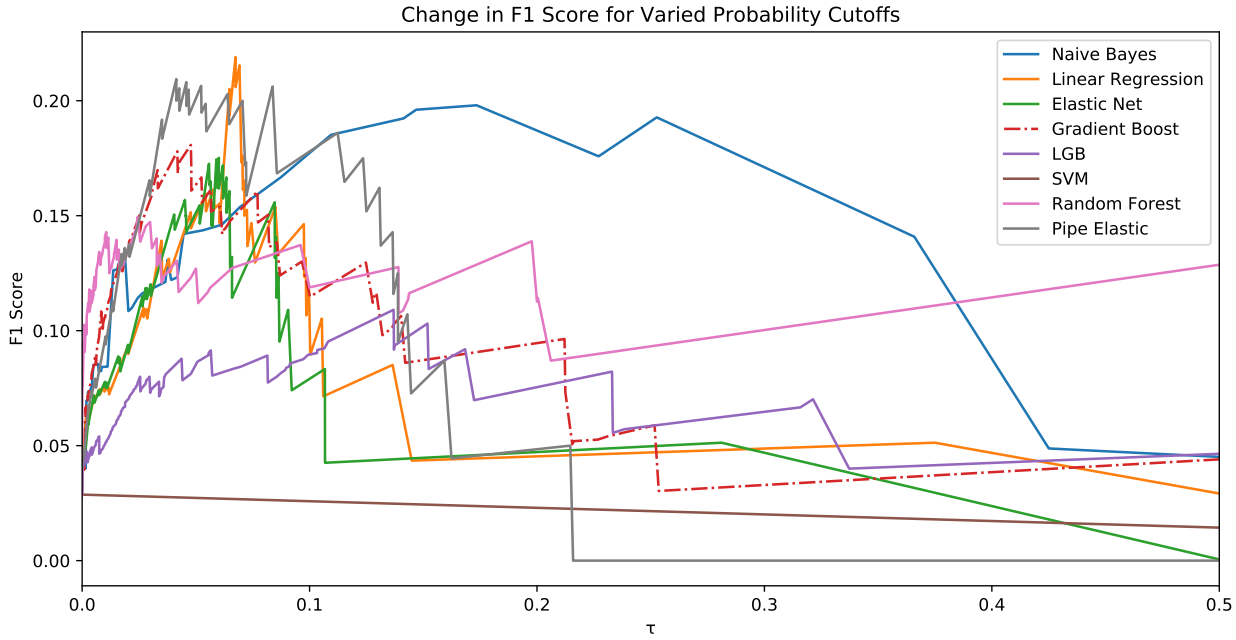


Figure 2: An example of the modified ROC curve using F Score instead of Accuracy.

maximizer proved to be an effective τ value to maximize the F1 Score on later predictions. An example of a modified ROC curve which incorporates the F1 Score can be seen in Figure 2. Code for the modified ROC curve is included in `ROC.py`.

Overall Examination of Methods

Once the tools of rare event prediction were working properly, various standard algorithms were tested for efficiency. The effectiveness of each method ultimately depended on which city it was being tested on. For example, though the features of the dataset are not truly independent, the assumption that they are independent in Naive Bayes was the most effective for prediction on the Springville dataset. However, on the Orem dataset, Naive Bayes was not very effective. Yet, Linear Discriminant Analysis, a closely related method, was very effective. Both of these methods are classifiers which assume a certain amount of linearity. The effectiveness of linear assumptions is further reinforced by the overall success of linear regression and its relative Elastic Net on both datasets. Elastic Net is a linear regressor like ridge regression that incorporates a combination of both L_1 and L_2 regularizing terms.

To complement the linear methods, random forests and gradient boosted forests were able to correctly classify pipes which were infrequently caught by the linear methods. Hence, the optimal solution appeared to be an ensemble of both linear classifiers and random forest classifiers. Ensembling is discussed further in a later section. Code for the implementation of the models is included in the files `modeling.py` and `run_ml_tests.py`.

LightGBM

One of the more effective methods we used was a form of gradient boosting trees developed by Microsoft called LightGBM. In many algorithms for boosted decision trees, the trees are grown level-by-level, where by level means splitting on each leaf at each iteration. However, LightGBM

grows its trees using a leaf-by-leaf approach, producing trees that have varying depths depending on which path is taken. This means that a deeper and more specifically tailored tree can be constructed in a shorter amount of time, making the algorithm very fast. Yet, because the algorithm builds off of each leaf individually, there is a risk of overfitting, which needs to be controlled by limiting the depth of the trees.[5]

Ensembling

After the initial successes of the individual models, their results were pooled together in an ensemble in an attempt to increase the F score. The ensembles are among the best of the models, but still are not superior in F score to the best individual models in Springville. However, after closer examination, it appeared that the true strength of ensembling was that it reduced the total number of predicted breaks. Other methods, such as XGBoost, would often report a lot of true positives along with an enormous number of false positives. Just as a model that predicts no breaks is poorly suited to the problem, a model that predicts 40% of the pipes to be high-risk is almost as unusable for a city. In order to submit a report of predictions to a city, the number of predicted positives needs to be reasonable enough to warrant budgeting. Too many predicted breaks overwhelms the city's capacity to act.

In order to properly ensemble and combine the results of multiple methods, a voting system was used. This voting system allows each method to individually classify each pipe and then combine their decisions into one decision by tallying up votes and comparing the total to a predetermined threshold. As previously mentioned, this ensembling method was able to reduce inaccurate classifications from various methods. A table of ensembling results follows in the next section.

Although the voting ensemble was an effective way of reducing false positives, it appeared that there was still some noise being introduced into the results. In an attempt to further improve, we introduced a weighted model of voting in which algorithms with better F1 Scores received more weight in the voting process. This method was decently effective, but its results were comparable to the regular voting method and it was not pursued further.

Results

The results of a model run for Springville are presented in the following table. Our goal was to attain an F1 score of .25. This would mean our product can save large cities hundreds of thousands of dollars.

Results for Springville

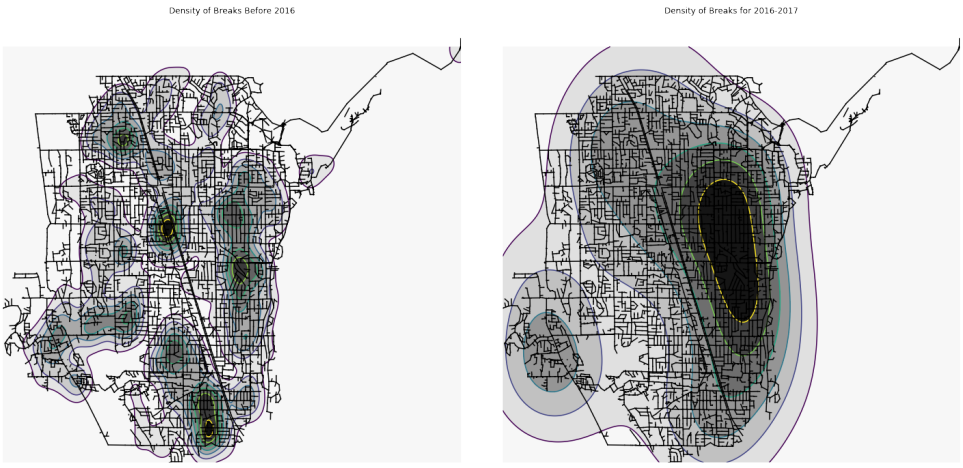
Individual Method	F1 Score	Ensemble	F1 Score
Linear Regression (LR)	.162	(QDA,LDA,SVC,NB,XGB,RFC,PEN) > 3	.184
Pipe ElasticNetCV (PEN)	.188	(LDA,NB,XGB,PEN)>2	.203
Random Forest Class (RFC)	.153	(LDA,NB,XGB,PEN)>1	.205
XGBClass (XGB)	.175	(LDA,SVC,NB,XGB,RFC,PEN)>3	.188
LightGBM (LGB)	.174	(LDA,SVC,NB,XGB,RFC,PEN)>2	.195
Naive Bayes (NB)	.209		
Support Vector Machine (SVC)	.144		
Linear Discriminant (LDA)	.172		
Quadratic Discriminant (QDA)	.108		

[5]Visit https://www.analyticsvidhya.com/blog/2017/06/which-algorithm-takes-the-crown-light-gbm-vs-xgboost/ for more information.

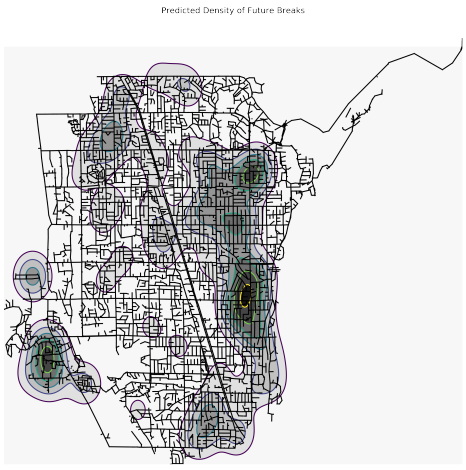
In Orem the lack of initial information about pipe ages may have contributed to lower success for the model. As a result, the ensembles seemed to have the ability to pick up on more patterns than the single model and so were comparably more successful.

Results for Orem

Individual Method	F1 Score	Ensemble	F1 Score
Linear Regression	.179	(LR,LGB,NB)>0	.188
Pipe ElasticNetCV	.177	(NB,PEN,RF,XGB)>1	.168
Random Forest	.153	(NB,LR,XGB,LGB,SVM,RF,PEN)>2	.180
XGBClass	.150		
LightGBM	.154		
Naive Bayes	.135		
LDA	.183		



(a) Density of training breaks from 2007-2015 (b) Density of testing breaks from 2016-2017



(c) Density of predicted breaks on test data

Figure 3: Comparison of predicted density to actual density of breaks. The algorithms were trained on data from 2007-2015 and tested on the data from 2016-2017.

Analysis

Though many of the methods used are simple linear classifiers, when combined with variations of decision tree forests, they provided several unique perspectives that were effective for prediction. After all testing, it appeared that the greatest struggle in the problem of predicting water mains is the scarcity and incompleteness of the data. As a result, more complete datasets and further feature engineering will lead to greater accuracy. Each new variable or added feature resulted in a significant increase in the F Score of the majority of the algorithms.

Once we selected a model, we demonstrated its performance by comparing a heat map of high-risk water mains across the city with actual breaks. This heat map represents a deliverable that could help cities to identify the most pressing areas for pipe replacement. Example heat maps are contained in Figure 3.

Cities sometimes use heat maps of previous break locations to guide their decision making. As can be seen in the figures, the algorithms used were able to identify the at-risk areas of the sample city much more accurately than the heat map of previous break locations.

Conclusion

In conclusion, there is a lot of potential to predict main breaks when the correct variables are collected or computed and the correct methods are used and tuned properly. Though individual methods are accurate alone, ensembling their results into one prediction reduces the number of high-risk water mains to a more manageable scale and gives a concentration of actionable predictions. These predictions can then be used to save thousands of dollars in repair and water costs. It is astounding how a simple application of machine learning ideas can add so much value to a field.

Going forward, a focus should be put on accurate data collection and feature engineering to guarantee good results. It is unlikely that these methods will ever lead highly accurate predictions of when and where a water main will rupture, but the information that we can currently get out of the model, when used with already existing infrastructure improvement budgets, will help cities avoid a valuable portion of water main breaks.