Spatial Machine Learning for Lead Service Line Detection
Harvard IACS - BlueConduit

Capstone Project, Fall 2021

Javiera Astudillo, Max Cembalest, Kevin Hare, and Dashiell Young-Saver

Table of Contents

Table of Contents **1**

Problem Description [Milestone 1] **2**

Data **3**

Flint Data [Milestone 1] 3

Parameterizing distances with OpenStreetMaps [Milestone 2] 4

Exploratory Data Analysis (EDA) [Milestone 1] 5

Literature review **8**

Previous work by BlueConduit [Milestone 1] 8

Additional work in spatial machine learning 9

Evaluation Framework & Cross-Validation [Milestone 2] **9**

Spatial Cross-Validation [Milestone 2] 10

Evaluation via the Hit Rate Curve [Milestone 2] 12

Modeling **15**

Baseline Model [Milestone 1] 15

Overview of Meta-Model [Milestone 2] 16

Gaussian Process (GP) [Milestone 1] 16

Network Diffusion [Milestone 2] 18

Graph Neural Network (GNN) [Milestone 2] 21

Stacking [Milestone 2] 24

Future Work [Milestone 2] **25**

Appendix A **26**

I. Problem Description [Milestone 1]

Most cities around the United States have a lurking health hazard spread out across thousands of homes: lead pipes. As a malleable and leak-resistant material, lead was a popular choice for water pipes for thousands of years.[1] Throughout the 20th Century, however, the negative health effects of lead became more apparent (particularly for younger children).[2] To combat these effects, American regulators have sought to remove lead from prominent environmental locations, namely household paint, service lines, and gasoline.[3] In this project, we are concerned with the second of these environments: lead service lines.

If lead could be easily and inexpensively removed, though, this project would have begun and concluded around the time that the U.S. Environmental Protection Agency (EPA) banned lead from new service lines in 1986. Of course, that is not the case. The lurking danger element of lead service lines manifests due to the fact that low levels of lead in water sources may not be immediately obvious. Moreover, the largest challenges occur when lead *leaches* into the water supply from corroded service lines, highlighting that minor environmental changes such as a change in water supply or failing solder can suddenly introduce significantly higher concentrations of lead into a home's water supply.[4] Finally, there are substantial data challenges in this domain. Cities typically have poor records of lead service lines, and many of these service lines are privately owned and without a public record at all. The cost associated with false positives and false negatives are both high, compounding this problem. Because failure to locate lead pipes could lead to lead poisoning for thousands, there is an obvious desire to limit the number of dangerous homes which are never investigated. On the other hand, verifying whether a home has lead pipes requires extensive excavation and incurring a cost of roughly $2,500[5], regardless of the outcome.

In this project, we will be assisting BlueConduit, an Ann Arbor, MI-based startup that develops bespoke machine learning algorithms to assist municipalities and water utilities find lead service lines so that they can be replaced. In particular, this project will focus on ***investigating whether spatial information can be incorporated into an existing machine learning pipeline to improve the "hit rate", or the rate at which digs find lead.*** To date, BlueConduit's models focus on parcel-level features only, incorporating information such

[1] https://www.safeplumbing.org/advocacy/health-safety/lead-in-water.
[2] https://www.cdc.gov/nceh/lead/prevention/health-effects.htm.
[3] https://www.epa.gov/lead/protect-your-family-sources-lead
[4] Service lines are the pipes that connect a residence or commercial property to the common water main. From discussions with BlueConduit, we understand that lead was not a popular choice for larger commercial-use properties due to its relative weaknesses when carrying high-flow water.
[5] https://arxiv.org/pdf/1806.10692.pdf

as the year the home was built and other neighborhood-level features to predict whether a particular parcel is likely to have lead. Because these observations are essentially viewed as independent and identically distributed, there is substantial spatial information loss. Intuitively, homes which are near one another are likely to share characteristics which cannot be inferred directly from home-level features. In a literal sense, this is the same intuition behind ML algorithms such as k-Nearest-Neighbors, where observations (or parcels) that are near to one another are assumed to provide information about each other as well.

The outcomes and goals of this project are twofold. First, following discussions with our partner organization, we plan to evaluate multiple approaches to incorporating spatial information to see whether it can improve their classification model. Importantly, from our partner's point-of-view, this investigation can be successful even if spatial information does not improve the model. BlueConduit has rarely tested these spatial methods. Indeed, they have stayed away from direct utilization of spatial features such as longitude and latitude, which are prone to overfitting and failure mode due to the lack of causal structure. During this evaluation phase, though, we would consider a model successful if it improves the "hit rate". As described above, this can be thought of as the precision over the "dangerous" homes. A higher hit rate means more excavations that find lead and fewer which do not, improving the rate at which lead can be removed and reducing the total program costs.

Second, we hope to evaluate whether the spatial information can improve predictions for low data environments. As described below, we will work with data for Flint, MI. Flint has been plagued by a water crisis which began unfolding in 2014 and culminated in widespread distribution of bottled water to residents for years due to the extreme levels of lead within the city. As a result, there are many data points of locations with high levels of lead. As BlueConduit moves from Flint to other cities, they are particularly interested in whether this spatial information can provide lift when few predictors have been captured and many homes have not yet been verified.

II. Data

A. The Flint Dataset [Milestone 1]

The primary data for this project have been provided by BlueConduit and focus on Flint, MI. The raw dataset is roughly 55,000 rows and contains 74 features. This proprietary data records a single observation for each parcel (home or commercial property) in Flint. Notably, however, only roughly half of the data have a determination of lead or no lead. These reflect homes which have been verified either before the Flint Water Crisis began in 2014 or were previously known to the city.

There is one notable bias here, which is that there is a real-world selection mechanism going on when each individual parcel is selected and verified. The homes in this dataset may be more likely than the average parcel in Flint to have lead, as those would have likely been targeted for removal. We do not believe that this is necessarily an issue for the external validity of the project for two main reasons. First, our approach is model and feature agnostic. While we will use the Flint data to validate the approach and measure progress, our charge is to assess the viability of the spatial information for Flint and future cities. Second, the overrepresentation of homes with lead may contribute to more of a balanced dataset for our model's purposes. Class imbalance is a well-known and omnipresent issue in machine learning, and if the selection effect were reversed (i.e. the dataset were likely to have very few homes with lead service lines), that would be cause for greater statistical concern.

Over the previous years, BlueConduit and the city of Flint have developed many novel sources of data, including the digitization of over one hundred thousand index cards detailing citywide maintenance.[6] Due to these efforts and the scope of our project, we do not intend to develop additional features apart from the spatial information.

B. Parameterizing distances with OpenStreetMaps [Milestone 2]

One outside source of data we use is OpenStreetMaps (OSM). A natural challenge of this project is measuring distance. Due to the design of traditional American infrastructure, pipes generally follow streets. So, true Euclidean (or Haversine) distance over a map may not accurately represent the way that spatial information should flow through a city. For example, if two houses share a backyard but are on different streets, then they might have small Euclidean distance but be connected to two different water mains. Thus, we queried OpenStreetMaps to retrieve walking distances between homes. These walking distances better mimic the "street distance" (Manhattan distance) between homes, since the walking paths follow the streets. Thus, a given house will be more influenced by neighbours in the same street than by a backyard neighbour, which likely has a different water main.

[6] https://arxiv.org/pdf/1806.10692.pdf.

Figure 0: *We use street distances (left) rather than geospatial distances (right) to construct our graphs*

These street distances inform our graph-based models. In the graph-based models, one node is created for each parcel. Then, the nodes are connected with edges that are weighted according to the magnitude of the walking distance between them. Due to the long time it takes to query the Open Street Maps API, we only calculated walking distances between homes that were within 0.5 km (Euclidean distance) of one another. This sparsity ensures that homes that are on opposite ends of the city are not connected by edges and, therefore, not spatially influencing one another in our models. In addition, this type of sparsity can help in regularizing large graph-based models.[7]

C. Exploratory Data Analysis (EDA) [Milestone 1]

The dataset has 26,863 rows, each representing a parcel (home or commercial property) in Flint, MI. Each parcel has 74 described features, such as the market value, size, location (latitude/longitude), year built, voting precinct, etc. Because our dataset size is relatively small, we are currently not too concerned about efficiency and computational issues.

pid int64	Property Zip Code float64	Owner Type object	Owner State object	Homestead object	Homestead Percent float64	HomeSEV
4012482018	48503	Private	MI	Yes	100	18400
4013226009	48503	Private	MI	Yes	100	11800
4012476011	48503	Private	FL	No	0	0
4012481022	48503	Private	MI	Yes	50	4550
4013226025	48503	Private	MI	Yes	100	12800

Table 1: *The head of our dataset, which shows the one-row-per-parcel structure of the data*

[7] https://arxiv.org/pdf/1706.02216.pdf

In addition to our features, we have a column for our target: a binary indicator of whether the home has dangerous pipes (1) or safe pipes (0). In the full dataset, which includes the homes that have been investigated in Flint, we found that 10,266 parcels had dangerous pipes, or about 38% of all homes. Because this is a relatively large proportion of all homes, we don't have to treat the target as a "rare" outcome in our models. Additionally, we understand that a full inventory of parcels in Flint would capture significantly more parcels which are either (a) abandoned or (b) are very unlikely to have lead due their date of construction and have therefore not been investigated.

Commercial Condition 2013	26760
B_aggregate_income	4666
B_imputed_value	1500
B_imputed_rent	1500
B_hispanic_household	1500
Housing Condition 2012	400
Residential Building Style	114
Housing Condition 2014	109
USPS Vacancy	59
Owner State	26
Longitude	6
Latitude	6
Use Type	2
Zoning	2

Table 2: *Missing value counts*

Table 2 shows the missing value counts for different features in our data. The highest count is for "Commercial Condition." However, the vast majority of these parcels in the dataset are residential, and this feature is coded as "missing" for residences. So, it's not truly missing but, rather, not applicable to residences. The other features with large missingness rates are the "B_" features. These are all features that were imported from the American Community Survey (part of the US Census). The missingness here is likely from homes that could not be linked to ID's from the American Community Survey. If these features are found to be important, we may go back to the census to see if we can successfully match these homes to American Community Survey data and fill in the missing values with the true values.

Papers published by BlueConduit show that "year built" is consistently the most important predictor of lead in their models. So, we explore this variable with particular interest. A histogram of "year built" is shown in Figure 1.

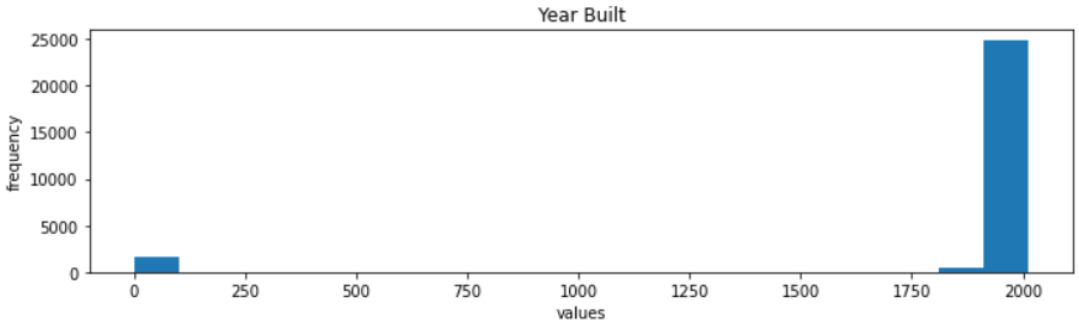


Figure 1: *Histogram of "year built," including the nonsensical values present in the data*

We saw that there were a fairly high number of nonsensically small values - homes that were built before the year 100 A.D. In total, there are 1,629 homes with nonsensically small year built values. We asked the partner why these values are present, and they reported that some city records are simply unreliable and that they treat these values as "missing." The Baseline XGBoost model is able to take this missingness in stride by splitting into different cases whether the year built feature is present or absent.

When looking just at the sensical year built values, we find an interesting pattern, visualized in Figure 2.

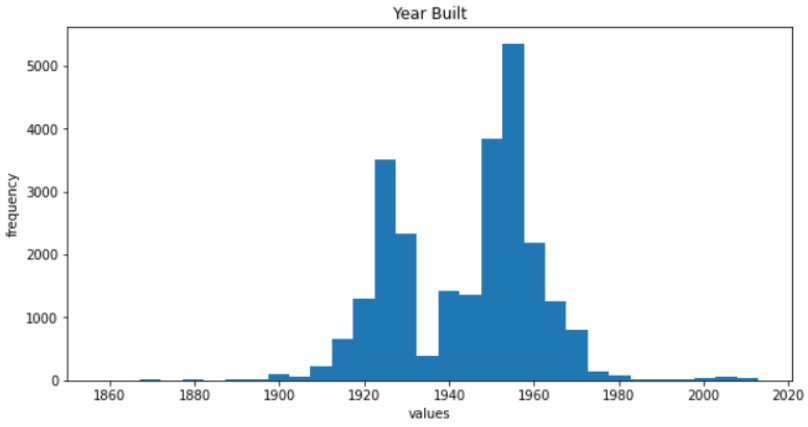


Figure 2: *Histogram of "year built," only including the sensical values*

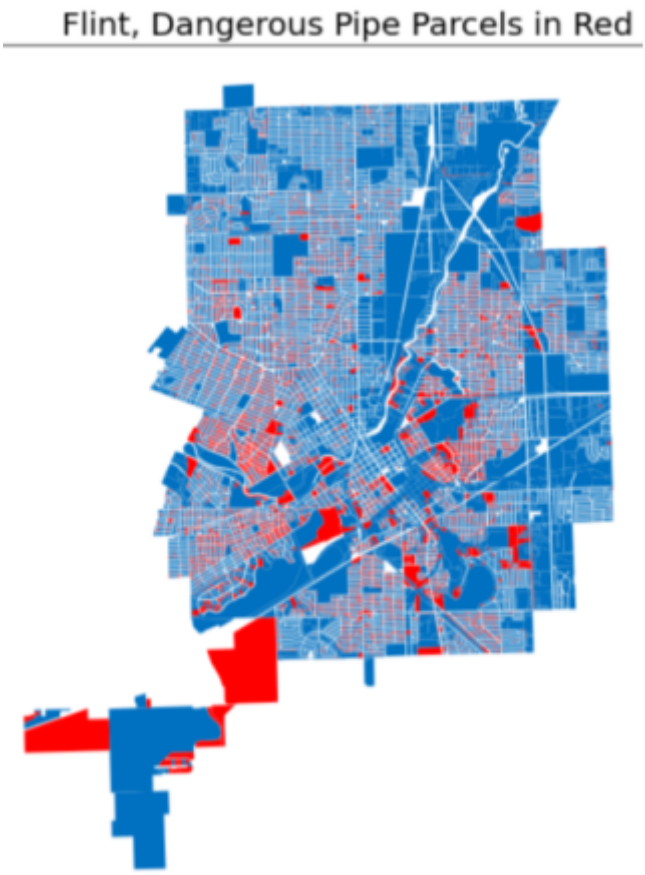


Figure 3: *Flint Michigan. Homes with known lead pipes in red.*

Figure 2 shows that year built appears to be somewhat bimodal, with large buildup of homes prior to 1935 and immediately following 1945. This is likely due to historical circumstances, such as a lower number of homes built during the Great Depression (1930s) and with a boom after the Second World War (1950s). It may end up being the case that these two eras (pre Depression, post WW2) are fairly predictive of dangerous pipes.

To test this theory, we found the rate of dangerous pipes within both eras. Among pre-Depression homes, 88.5% had dangerous pipes. Among post-World War 2 homes, only 7.2% had dangerous pipes. So, it's clear that year built is highly

predictive of lead presence. This explains why it's such an important feature in BlueConduit's current models.

Finally, in Figure 3, we visualized parcels where there is known lead in Flint (lead parcels shown in red). We see that lead pipes are distributed throughout the city; however, there is some variation in lead pipe density by neighborhood. This suggests that using geospatial information could help improve the BlueConduit models. In future map visualizations, we will vary color intensity by predicted probability of lead, and we will refrain from coloring the large commercial parcels (to preserve the granularity of seeing the smaller residential parcels).

III. Literature review

A. Previous work by BlueConduit [Milestone 1]

The very first work of BlueConduit with the Flint Council dates back to 2016 and is reviewed in Chojnacki at el. (2017)[8]. In this work, they frame Flint's water lead contamination, examine the water sampling process in detail and propose an initial model for house prediction lead presence. Their work also includes a discussion on how selection bias is a concern. Their models directly impact public health and, thereby, must address interpretability and accountability.

Their initial proposed model predicts a lead presence probability based on parcel dataset, service line dataset, and census dataset, for a total of 71 features. Their target variable uses water testing datasets based on residential and sentinel water testing programs, gathering information for around 15,000 parcels. Their findings show that the most predictive features for predicting lead levels include home value, demographic data from the census bureau and property age.

Their next work (Abernethy et al. 2018[9]) describes their model results in the context where the City of Flint took action in the water crisis. In 2016, Flint's Mayor contracted the Flint Fast Action and Sustainability (FAST Start) team. Their task was to remove as many hazardous service lines as possible up to the funding level. This publication analyses how their model considerably surpasses the FAST Start strategy and empirically shows the resources saved through their strategy.

[8] https://arxiv.org/pdf/1707.01591.pdf
[9] https://arxiv.org/pdf/1806.10692.pdf

They update their classification framework to handle unobserved variables with a Bayesian spatial model. Further, their targets are built based on excavations in conjunction with water samples lead level (ppb) in this new setting. As of September of 2017, the FAST Start had carried out 6,506 home excavations. Consequently, they pose their strategy in an active learning framework, simulating the actual scenario process. Every time they make a new batch of discovery excavations, they update their dataset and their predictions accordingly.

They quantify the cost savings between their strategy and Flint FAST Start's actual home selection during 2016-17. In simulating their results over Flint, MI, BlueConduit found that their strategy reduced the rate of costly unnecessary replacement visits from 18.8% (actual) to 2.0% (proposed). Suppose 18,000 total planned service line replacements; this would translate into savings amounting to as much as $11M from current spending. This is approximately equivalent to 2,100 additional homes in the city that would receive safe water lines.

B. Additional work in spatial machine learning

Bayesian Spatio-Temporal Gaussian Process
https://dl.acm.org/doi/fullHtml/10.1145/3300185

We found this paper when researching spatial models that take time into account as a feature. The temporal dimension is a very relevant consideration for our project since our EDA and the trained XGBoost model both show the importance of YEAR BUILT as a feature.

The SPDE approach for Gaussian and non-Gaussian fields:
https://arxiv.org/pdf/2111.01084.pdf

The paper will help us improve the runtime and kernel selection for our gaussian process. In addition, we are interested in implementing the spatio-temporal formulation given in section 4.3

A Comprehensive Survey on Graph Neural Networks
https://arxiv.org/pdf/1901.00596.pdf

This resource has helped us compare different graph neural network architectures and problem-solving paradigms at both a high-level and on the level of technical details.

GraphSAGE: Inductive Representation learning on Graphs
https://arxiv.org/pdf/1706.02216.pdf

This has been the most successful Graph Neural Network architecture used so far. We hypothesize the strength of GraphSAGE in our setting relies on its sub-sampling of

neighbors to improve runtime, and its ability to train via batches while the other graph methods train on the whole graph each train step.

IV. Evaluation Framework & Cross-Validation [Milestone 2]

Because we have considered a variety of models and frameworks, a consistent evaluation framework is of the utmost importance for this project. The evaluation framework for this project relies on two key concepts. First, because of the spatially-connected nature of the data, we aggregate data for cross-validation and evaluation by geographic partitions. This both prevents target leakage and simulates a more realistic environment for investigating parcels, where cities are more likely to select entire blocks or neighborhoods for investigation rather than working parcel-by-parcel in a scattershot manner. Additionally, we employ a custom loss metric, the hit rate curve, which is connected to the precision over the class of homes with lead pipes, but allows for a more dynamic understanding of the potential thresholds, once again simulating a more realistic choice for a policymaker. Both of these design decisions and implementations are described below in greater detail.

A. Spatial Cross-Validation [Milestone 2]

When designing our cross-validation strategy, it's essential to mind two key components that will bias our results if not adequately incorporated in our evaluation process: the spatial nature of the problem and the particulars of pipe excavation.

The spatial component regards the dependence structure in the data[10]. It presents itself as the high correlation of lead pipe presence between neighbour houses in the current set. Observation-level training-validation splits rely on the assumption of independence between observations in our dataset. However, we know that our observations (each home) have dependency by location (nearby homes are similar). Therefore, if we naively split train and test samples with simple random sampling, we could yield overly optimistic prediction error estimates. This is what we call the "target leakage" effect. Models appear more accurate than they are, enticing us to have more faith in their predictions than is actually warranted.

The second component concerns designing train-test splits that are consistent with the pipe excavation scenario. In reality, pipe digging will be carried at "blocks" or "neighbourhood" levels. This means that the city council targets locations with many positively predicted lead pipes houses and digs all those candidates at once. They do so to

[10] https://onlinelibrary.wiley.com/doi/10.1111/ecog.02881

make efficient use of excavation process resources. Consequently, labels are uncovered by "blocks", imposing a spatial structure in the train and validation sets.

A common solution addressing both these concerns of hold-out independence and accurate representation of splits is spatial cross-validation. It achieves unbiased error and better parameter estimates[11] by splitting the data into "blocks" by location. In the current scenario, we apply this methodology by dividing the city of Flint into spatial hexagons and assign each of them either to train or validation split. That way, we reduce the correlation between train and test samples, as most neighbours will be located either entirely within the train or entirely within the test set. Further, the digging will be executed at the hexagon level, emulating the actual scenario. We repeat this procedure multiple times (folds) to perform inference over various samples of the hold-out error metric. Furthermore, we try different train and test proportions to represent the different phases of lead pipe excavation in a city, starting with very few labels. Then, we progressively uncover labels to simulate pipe excavation.

The resulting cross-validation strategy is depicted in the following plots:

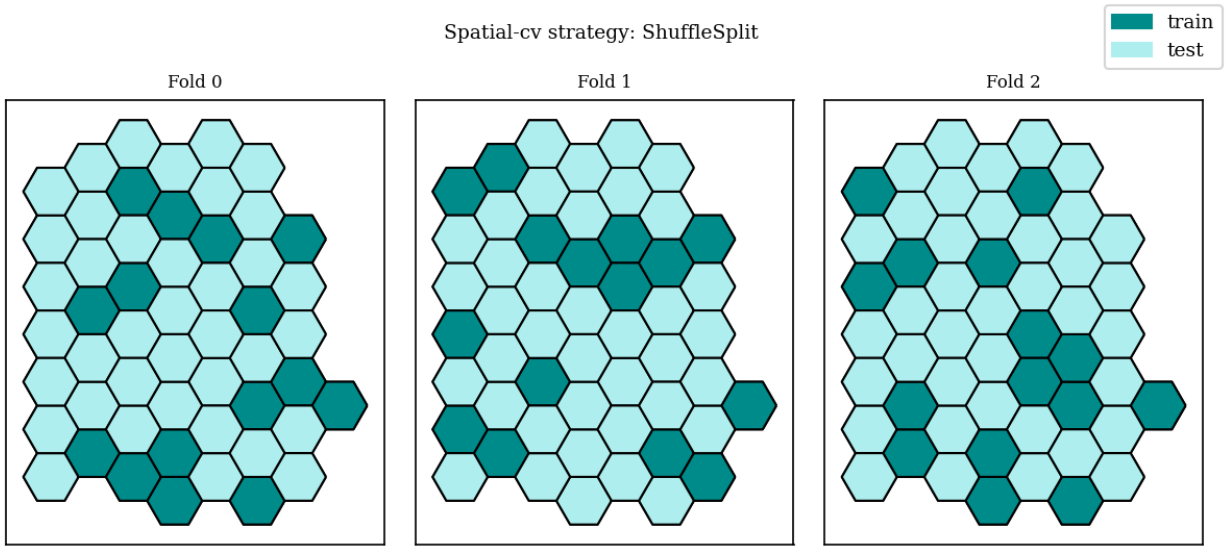


Figure 4. Spatial cross-validation train-test split [folds=3, hex_resolution=22, train_size=0.3]

[11] https://onlinelibrary.wiley.com/doi/10.1111/ecog.02881

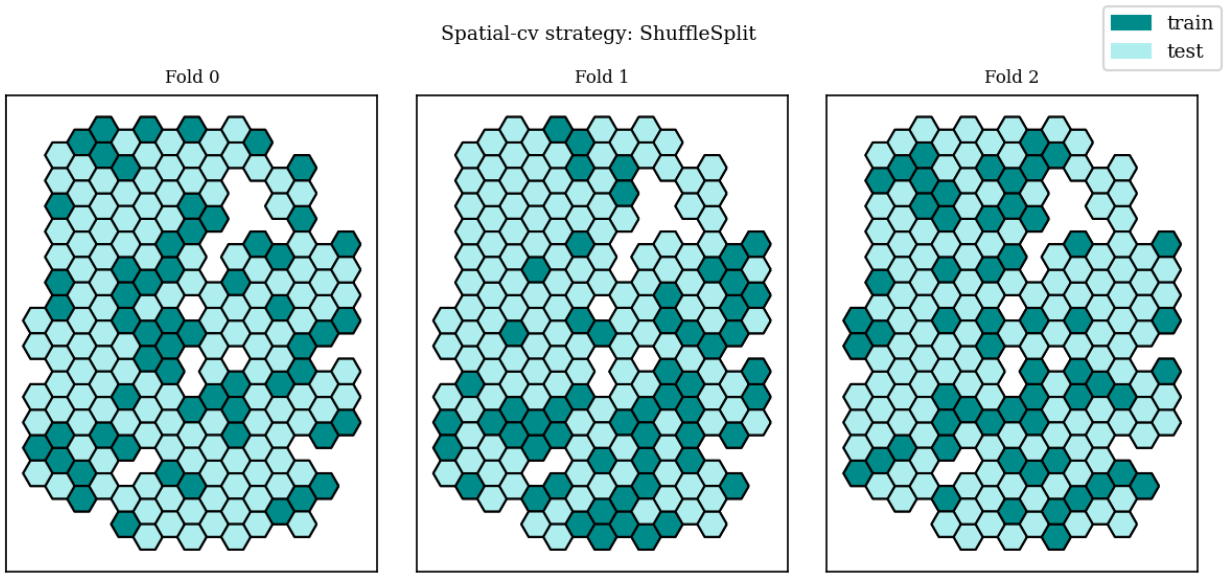


Figure 5. Spatial cross-validation train-test split [folds=3, hex_resolution=47, train_size=0.3]

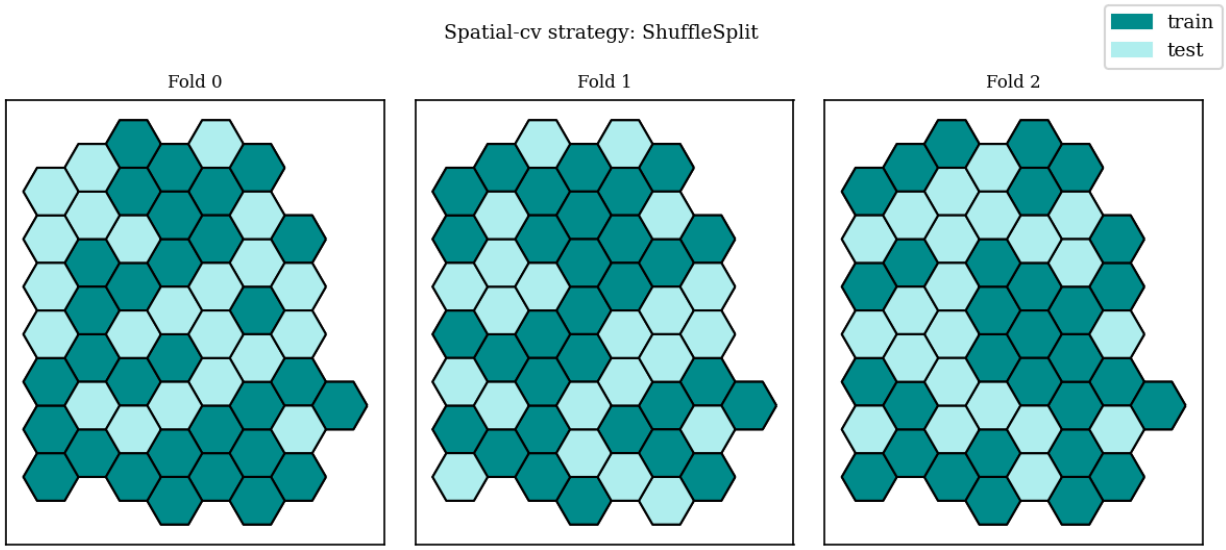


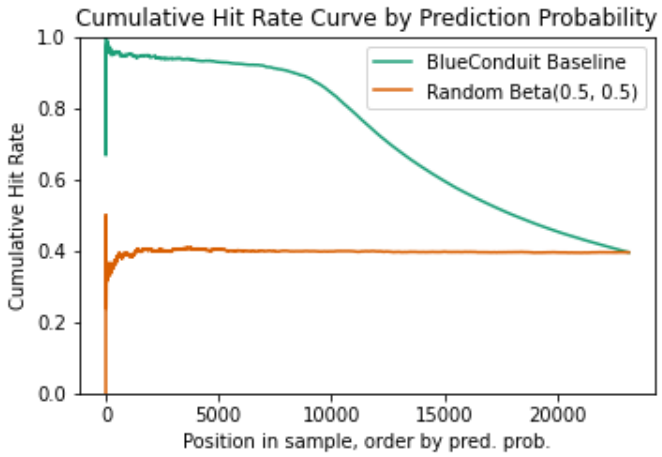
Figure 6. Spatial cross-validation train-test split [folds=3, hex_resolution=22, train_size=0.6]

B. Evaluation via the Hit Rate Curve [Milestone 2]

To properly evaluate our models, we will use a BlueConduit customized loss metric referred to as the “hit rate curve”. First, it is important to define the hit rate. As described in Abernathy et al. (2018), the hit rate is the, “percentage of homes visited for replacement

that required replacement".[12] In technical terms, this is the precision over the class of homes having lead. In traditional machine learning, the hit rate would be calculated across the entire test set, or across multiple test sets and then averaged. However, doing so would naturally require setting a threshold for each prediction.

Rather than evaluate models via a single number, BlueConduit makes use of a richer representation of the outputs: the prediction probabilities. In this framework, the "hit rate curve" essentially demonstrates the hit rate for various prediction probability thresholds. Naturally, most trained models yield a high hit rate when the threshold is close to one and converge to the test set average proportion of the positive label as the threshold approaches zero.[13] One implementation detail is that rather than plot the hit rate vs. the prediction threshold, we have elected to plot the hit rate vs. the parcel visited, following conversations with our partner. Thus, if 20% of the test set has a predicted probability of lead of greater than 0.9, this will still be represented at roughly the 20th percentile of the x-axis.



Here, we demonstrate a visualization of the Hit Rate Curve for the Blue Conduit Baseline model (described below) as well as a non-model which guesses according to a Beta(0.5, 0.5) distribution to both demonstrate the visual as well as the performance of the baseline relative to random guessing.

Appendix A demonstrates a side-by-side comparison of the Hit Rate Curve organized by the prediction probability and parcel investigated for the BlueConduit baseline model.

One weakness of the standard Hit Rate Curve described above is that it does not take into account the reality of investigating lead throughout a city. In general, cities would prefer not to receive a list of 100 parcels to dig on 100 separate city blocks, even if the model outputs those parcels as the highest probability of lead. There are economies of scale to

[12] Abernathy, Jacob, Alex Chojnacki, Arya Farahi, Eric Schwartz, and Jared Webb, "ActiveRemediation: The Search for Lead Pipes in Flint, Michigan," *KDD '18*, London, United Kingdom, 2018. Available at https://arxiv.org/pdf/1806.10692.pdf.

[13] In

removal crews, city shutdowns of traffic, and pooling of municipal resources. Thus, we alter the algorithm to investigate by cross-validation partition, which works as follows:

By-Partition Hit Rate Curve Algorithm

- Initialize threshold (e.g. 0.9)
- Repeat until all parcels investigated:
  - Calculate count of parcels in partition with prediction probability exceeding threshold.
  - Order by partition.
  - For each partition:
    - Investigate all parcels above threshold; remove from partition.
  - Increment threshold (i.e. threshold = threshold - 0.1)

This methodology is less efficient than the original due to two underlying mechanisms. First, some parcels which have a very high probability of lead may not be in large or high-lead areas. In the initial algorithm, those homes will still be equally prioritized. In our more realistic evaluation scenario, though, these homes will be somewhat de-prioritized by being put into a partition that is investigated in a later time frame.

Second, our framework relies on the total number of expected hits, and thus partitions -- which are of equal geographic area but not of equal population size -- with more parcels will tend to be selected for excavation earlier. We also believe that this is a reasonable and realistic assumption because a dense, high lead area is likely to be identified and prioritized before a single parcel in a non-lead dense area. Below, we see that the new evaluation algorithm causes the Hit Rate to decrease in the more realistic scenario, highlighting the tradeoff between following the model entirely and balancing the costs of shutting down infrastructure such as water or roads for some amount of time.

V. Modeling

A. Baseline Model [Milestone 1]

Our chosen baseline model consists of BlueConduit's current model, mentioned in the previous literature review. We agreed on this with our partner since this project aims to improve based on what they have built so far by incorporating spatial modelling.

More specifically, the baseline model consists of an XGBoost, with 102 input features, spanning parcel, service line and census data. Their target is a binary variable indicating whether there's a lead pipe presence or not. We are mainly concerned about this model's hit rate, representing the lead rate discovery behaviour of the samples ordered by their predicted probability of having a lead pipe. We will consistently compare our developed models with this same metric. In Figure 4, we depict the hit rate of BlueConduit's baseline model alongside some basic comparison models included as a reference.

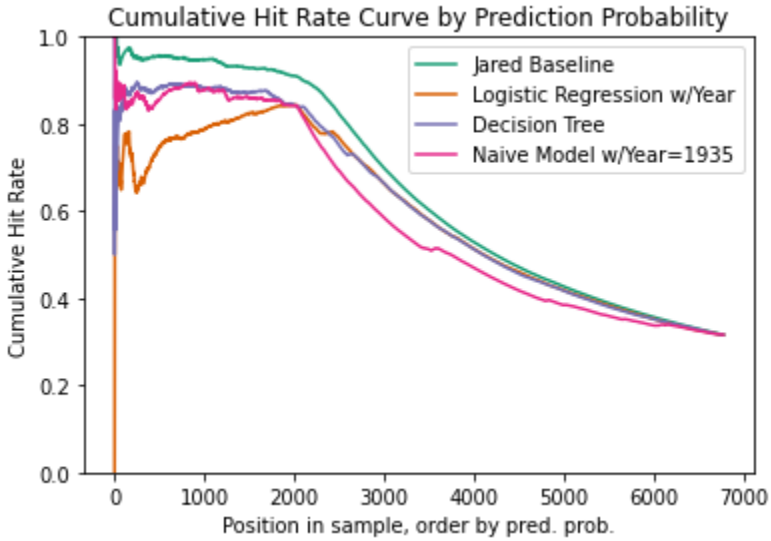
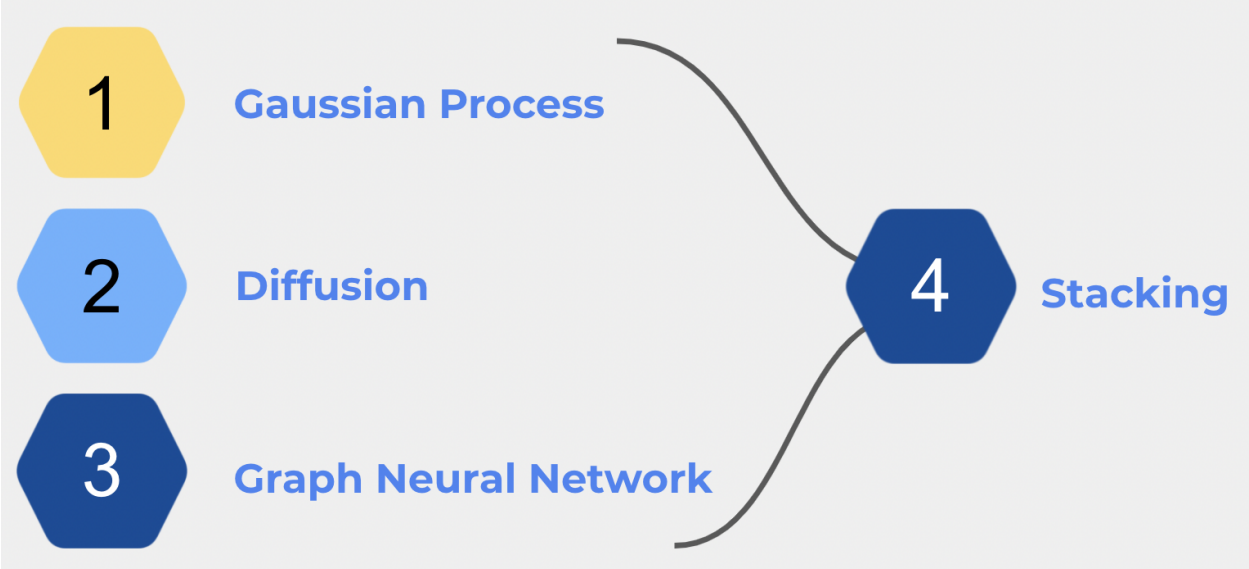


Figure 4. Baseline model and other additional model references.

In Figure 4, the BlueConduit XGBoost baseline model is represented by 'Jared Baseline'. Based on our EDA findings, we included reference models trained only on the year predictor ("Logistic Regression w/Year" and "Naive Model w/Year") to get a sense of how much information is conveyed by the year. Also, we included a simple decision tree as a comparative reference of the XGBoost baseline model.

It's clear that each of our naive models have lower hit rate curves than the baseline BlueConduit XGBoost model. Our goal will be to use spatial information models that can raise the hit rate curve beyond BlueConduit's current XGBoost model.

B. Overview of Meta-Model



Our group has working implementations of three different classes of spatial models so far: 1) a Gaussian Process (section C), 2) a Diffusion algorithm (section D), and 3) a few different Graph Neural Network architectures (section E).

Each uses a different feature set, and each has useful properties. For example, the Gaussian Process uses *only* latitude and longitude, and is therefore useful early in the data-collection process as a baseline spatial model. The Diffusion algorithm consists of only slightly tweaking the XGBoost baseline estimates with a weighted average of neighboring probabilities, resulting in a smoothing effect that seems to yield our strongest results so far. The Graph Neural Network leverages *all* features collected so far, and learns* how to estimate a probability of lead for a home based on all available information about the nearby homes.

We intend for our final model to be an implementation of Stacking (section F) which will make a final prediction of lead for each home based on the probabilities of lead from XGBoost as well as the probabilities of lead from our 3 spatial models. In Section F, we describe (only hypothetically, we haven't implemented this so far) what a decision tree might look like as a stacking model for our data.

*to what extent this information is robustly and faithfully "learned" depends on the Graph Neural Network architecture implemented. We have reason to believe GraphSAGE is the most straightforward, reliable choice. The other deep architectures are somewhat promising but likely unreliably overfitting to the data.

C. Gaussian Process (GP) [Milestone 1]

For a first pass at a spatial model, we chose to use a Gaussian Process to predict the probability a parcel contains dangerous materials, given only latitude/longitude as features.

From our EDA we could see the homes with lead are generally clustered in the

middle of the city. Our GP would need to be able to identify the spread of these regions and construct a set of highly nonlinear decision boundaries to account for the twists and turns of the structure of the underlying streets and neighborhoods.

The biggest drawback of a GP we have encountered is its runtime, which is $O(n^3)$; therefore we have not trained a GP on all n=26k homes. Instead, we trained 100 different GPs on a set of bootstrapped data samples, with n=2,000 homes in each sample. Figure 5a shows to what extent the fit of the GPs varies when changing the particular sample of homes they were trained on, and Figure 5b shows what the average of the 100 GP predictions looks like.

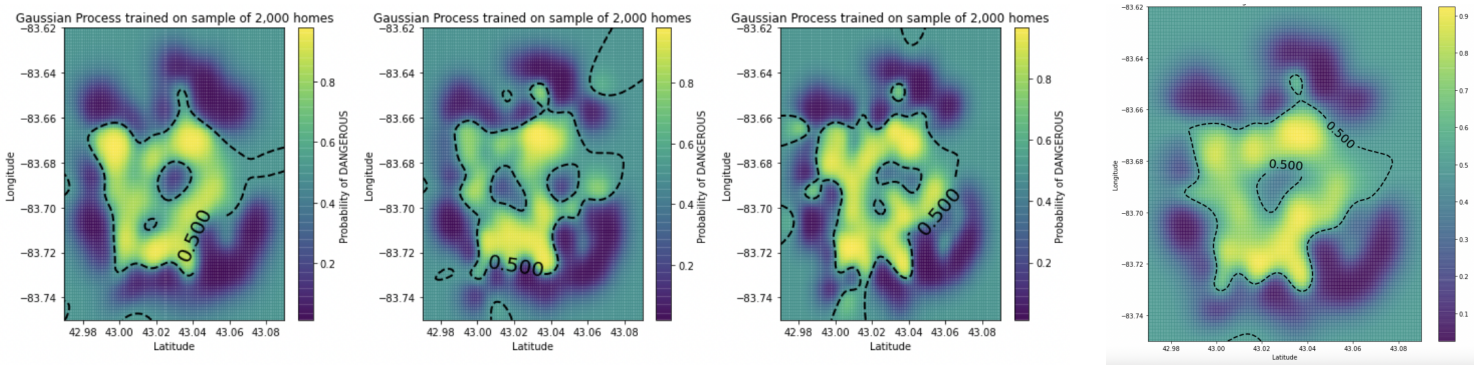


Figure 5a (left three): Three of the 100 trained GPs, plotting the predicted probabilities of lead across the latitude/longitude values spanning Flint, MI. Each of the GPs identifies that lead in Flint is mostly concentrated towards the center of the city.

Figure 5b (rightmost image): Predicted probabilities of the AVERAGE of the 100 trained GPs. This average of GPs had an AUC of 0.8288, higher than the vast majority of individual GPs in the 100.

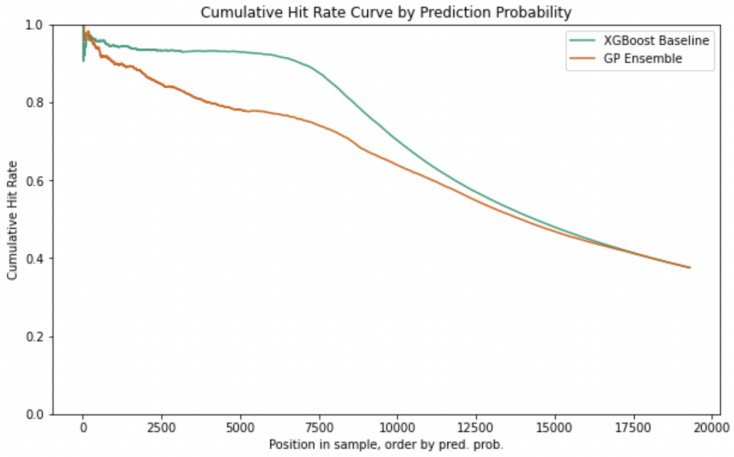


Figure: Hit Rate Curve for the Gaussian Process Ensemble (with samples ordered the old way, by individual predicted probability of lead. Our newer hit rate curves on our graph models are calculated via the Partition-Ordered hit rate curve, described in section V.B)

D. Network Diffusion [Milestone 2]

A second attempt to incorporate the spatial information into the modeling process has featured a "diffusion" process. Inspired by anisotropic diffusion to blur images and opinion dynamics over networks, the intuition for this approach is that neighboring homes should share information about one another's lead status.[14] Cities do not grow as random entities, with homes inserted at random times and places. Instead, blocks and neighborhoods tend to be built in similar eras, and thus with similar building materials. Thus, if the baseline parcel-level features-only model outputs a low predicted probability of lead for a single parcel on a block with many high-lead predictions, either the parcel-level features provide some indication of a different construction material or time frame or the model may be incorrectly focusing on a non-relevant feature in the data. In the ideal setting of the diffusion model, the process will "blur" the prediction probability to make it more like its neighbors, hopefully identifying additional high-risk parcels.

We have begun to investigate two possible formulations of this diffusion process, both of which are graph-based. As described above, nodes are individual parcels, and edges are formed on homes which are determined to be neighbors. Moreover, edges are weighted by the street distance between parcels. In this generalized framework, however, the number of neighbors, as well as the particular kernel used to implement diffusion, are hyperparameters which need to be tuned (see Future Work).

[14] https://en.wikipedia.org/wiki/Anisotropic_diffusion; https://academic.oup.com/poq/article-abstract/26/4/578/1868671.

The first formulation of our graph-based diffusion model starts each home at its Baseline Model predicted probability. Then at each step of diffusion, the node value is updated to be a weighted average of its own probability and that of its neighbors.[15] The results from this process can be visualized for pure performance as well as for an input in the stacking model below.

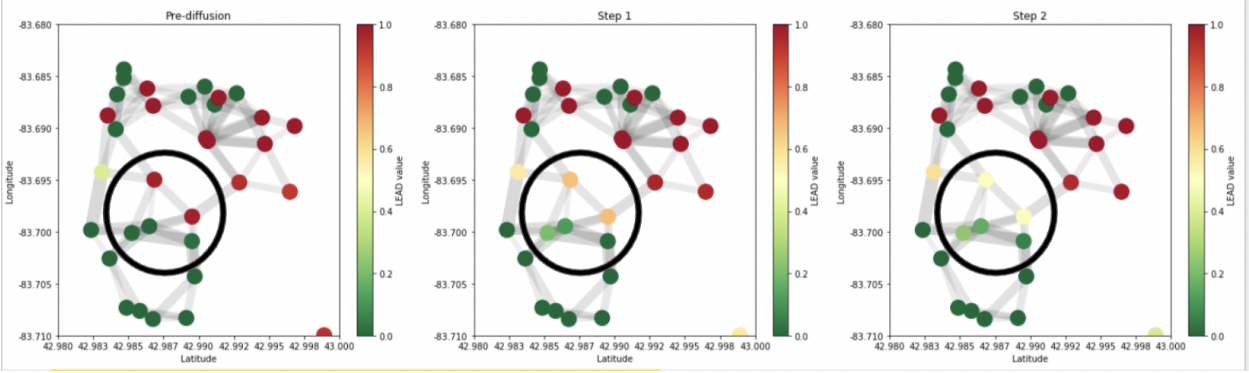
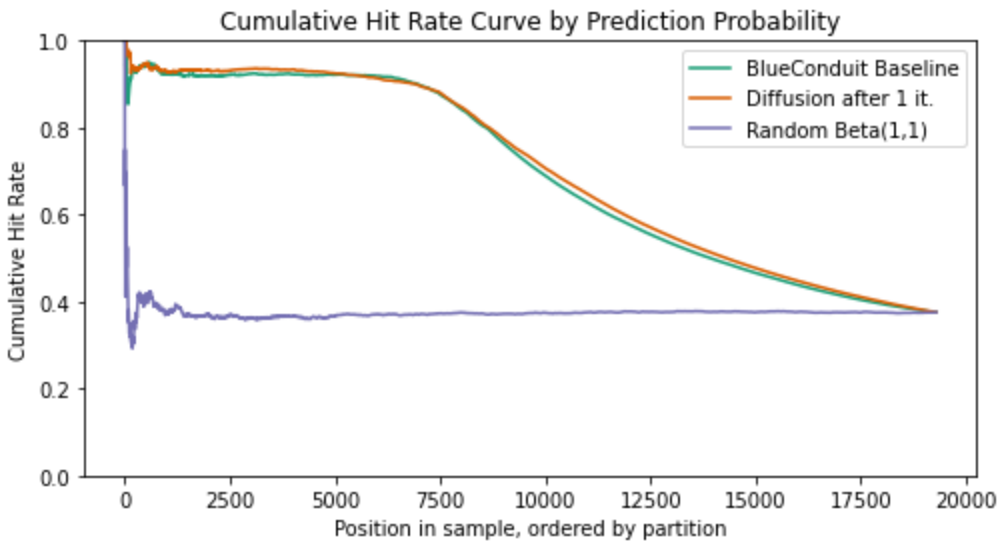


Figure: We show what diffusion looks like step by step in a small neighborhood. The Pre-Diffusion plot shows the XGBoost Baseline Model's predicted probabilities. Then, we update each node's probability via a weighted average using the values of their neighbors. Notably, for the parcels circled in the geographic center of this plot, their neighbors on one side have mostly a low probability (green), and their neighbors on the other side have mostly a high probability (red). After two diffusion steps our probability estimates for this group of houses gets closer to 50%.

Below, we show the results for a single iteration of diffusion (i.e. one update step).[16] From these results, it is clear that under the partition-based evaluation metric, the diffusion alone suffices to improve the model. This is particularly important for the second half of the hit rate curve. These preliminary results are encouraging, and suggest that while the diffusion model may not improve the high-end predictions (i.e. those that the model is most confident over), it can help in areas of lower confidence, as represented by the prediction probability.

[15] As with the number of neighbors and the distance metric, the exact weighting of the self-probability and its neighbors will be tuned. Results will be presented in upcoming milestones.
[16] Figures for additional splits can be found in our GitHub repository. Please contact authors for access.



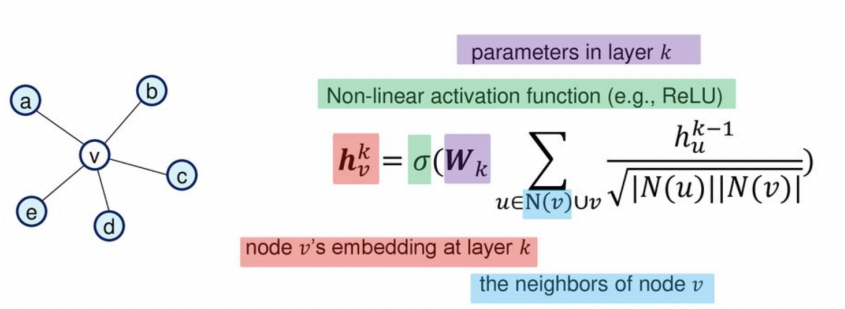
The second methodology for diffusion begins with only a set of labeled homes. From there, the network is evolved over many diffusion steps such that the probabilities of the test set(s) are changed over time to reflect the presence of lead in homes in the train set(s). In principle, this is similar to the Gaussian process. Rather than parameterize the effect of each lead home via a 2-dimensional Gaussian distribution, however, we would parameterize a graph and diffuse the information via that mechanism. One particular challenge for this model is that it is not clear how we would obtain prediction probabilities for the train set to be fed into a stacking model (described in Section E). As such, this is an active area of development and may be considered in addition to the to-be-tuned framework for diffusion described above.

E. Graph Neural Network (GNN) [Milestone 2]

Our investigation into graph neural networks is motivated by the need to preserve BlueConduit's XGBoost *when it is working*. We have at our disposal a) the OpenStreetsMap road distance graph, B) our dataset of 301 features for each home, and C) the baseline XGboost probabilities. Our intuition is that the graph neural network paradigm has the following benefit: it can *learn* how to use nearby home features to inform how much we should spatially diffuse probabilities.

Below, we describe the three different graph neural network architectures we have examined so far, and show the results for each so far by plotting their hit rate by partition.

Graph Convolutional Network (GCN or Conv-GNN)



http://keg.cs.tsinghua.edu.cn/jietang/

Implemented directly using the official Keras tutorial in order to research best practices (e.g. using batch normalization in all fully-connected MLP components).
Features: XGBoost probabilities, Year Built, Residential Building Value, Land Value, Parcel Acres
Model settings:

- Layers: preprocess, conv1, conv2, conv3, postprocess, dense
- We use a standard setup for a fully connected MLP: two layers of [BatchNormalization, Dropout with 50%, and Dense with 8 hidden units]
- Each convolutional layer uses a *prepare*, *aggregate*, and *update* function. Our prepare function is our standard MLP.
- We use a skip connection between the call of each convolutional layer:

  Output = *Dense* (*Postprocess* (X + *Conv3* (X + *Conv2* (X + *Conv1* (*Preprocess*(X))))))

- Our preprocess & postprocess layers are each our standard MLP
- The final dense layer is linear, outputs logits directly

Graph Attention Network (GAT)

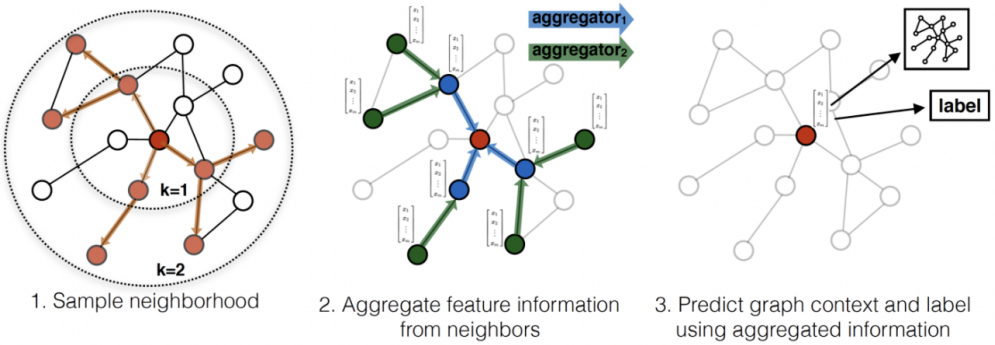
$$h_v^k = \sigma\Big(\sum_{u \in N(v) \cup v} \alpha_{v,u} W^k h_u^{k-1}\Big)$$

Learned attention weights

http://keg.cs.tsinghua.edu.cn/jietang/

Implemented using the StellarGraph library: see here for the GAT demo we followed
Features: XGBoost probabilities & ALL features collected by Blue-Conduit
We use multi-headed attention using 8 heads. Each attention layer has 8 hidden units and uses Dropout with 50%.

GraphSAGE: http://snap.stanford.edu/graphsage/



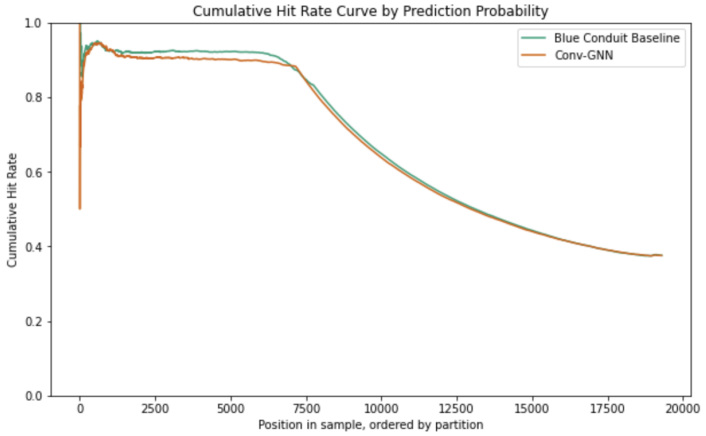
1. Sample neighborhood
2. Aggregate feature information from neighbors
3. Predict graph context and label using aggregated information

Implemented using the StellarGraph library: see here for the GraphSAGE demo we followed.
Features: XGBoost probabilities & ALL features collected by Blue-Conduit
Optimization notes:

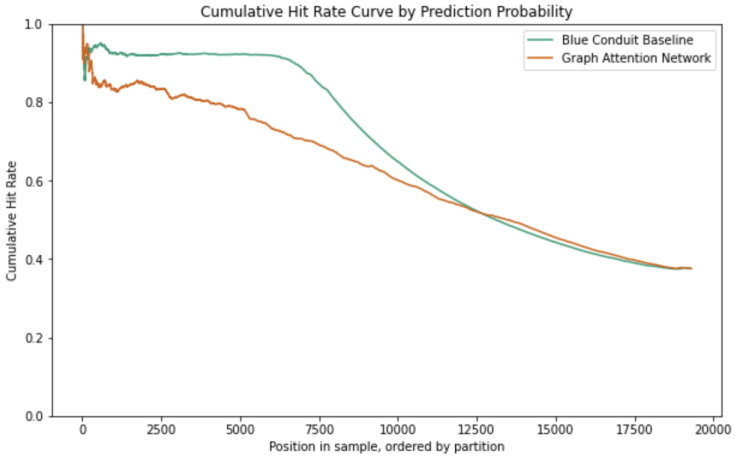
- Batch size: 256
- Neighborhood sampling: K=2 layers of sampling, # neighbors per sample [15, 10]
- Hidden layer embedding sizes [64, 64]

Hit Rate Curves

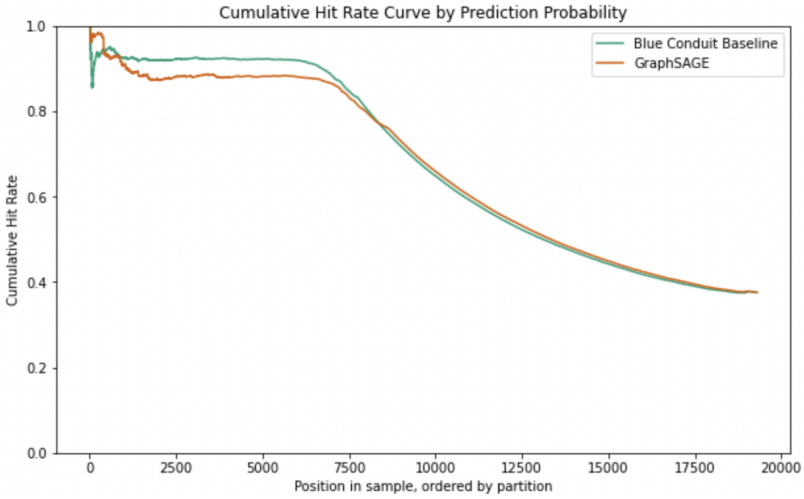
Conv-GNN:



GAT:



GraphSAGE:

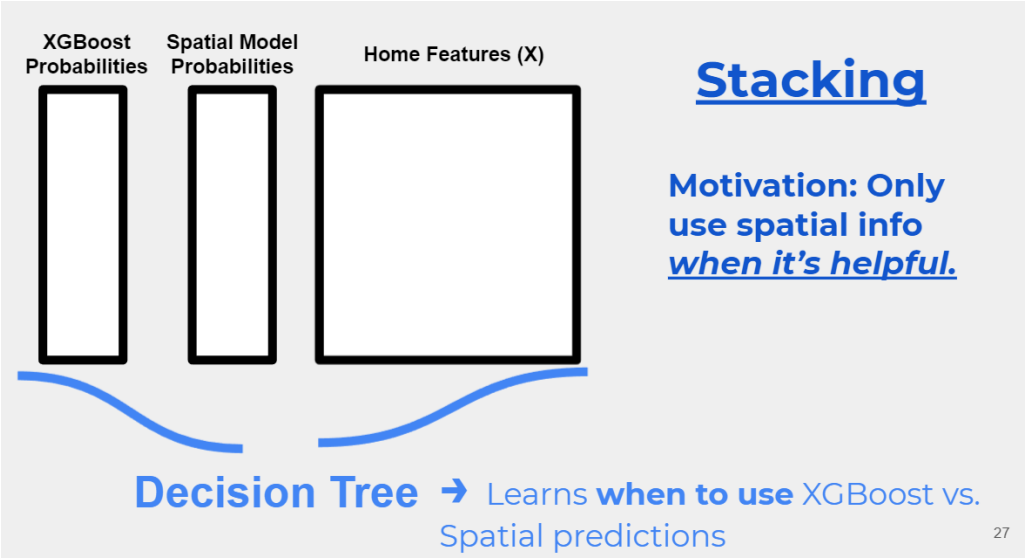


F. Stacking [Milestone 2]

During model building, we noticed that the baseline XGBoost model had very confident predicted probabilities for lead (or not-lead) in many parcels. Our spatial models would often take these probabilities and "smooth them out" over nearby homes. In other words, confidence was distributed from "confident homes" (for which the XGBoost model had predicted lead probabilities close to 0 or 1) to nearby "unconfident" homes (for which the XGBoost predicted probabilities were far from 0 and 1).

The upside of the above process is that the "unconfident" homes get to borrow information from nearby "confident" homes, ideally leading to more confident and accurate predictions. The downside is that the "confident" homes have their confidence diffused away from them, leading to less confident and potentially less accurate probabilities among them.

The above tradeoff motivated our use of **stacking**. By stacking models, we can create a model ensemble that learns to distinguish situations in which spatial information is helpful from situations in which spatial information is unhelpful. Specifically, the stacked model learns which particular sets of home features and XGBoost predicted probabilities indicate that a home's prediction would benefit from spatial information. Then, it only uses spatial information when predicting among those homes. Otherwise, it defaults to simply using the baseline XGBoost predicted probabilities.



As the above figure demonstrates, stacking works by learning a "meta model" over the XGBoost probabilities, spatial model probabilities, and home features. When we implement stacking, we expect to use a tree-based meta model, in order to precisely capture the best decision splits of when to use XGBoost probabilities vs. spatial probabilities over particular sets of home features. As we build out the stacking model, we

expect to incorporate multiple sets of spatial model probabilities from the various spatial models we discussed above. In addition, we will replace a simple decision tree with an ensemble tree method, such as random forest or XGBoost.

Preliminary stacking results: As of the second milestone, we have only implemented a simple stacked model, incorporating a naive k-NN as the only spatial model. The naive k-NN predicted each home's probability by taking the simple average of the XGBoost probabilities of the 5 nearest homes (in terms of street distances). A simple decision tree was fit as the meta model. At the moment, the decision splits are only based on the XGBoost probabilities (>50% XGBoost probability means predict lead; predict no lead otherwise). So, according to the decision tree, there are no situations in which our k-NN model provides better predictions. We assume that when we add our more advanced and well-tuned spatial models, the stacking meta model will find more useful splits based on the higher-quality spatial information.

VI. Future Work [Milestone 2]

Our future work will mainly focus on improving our spatial models and robustly incorporating them into the stacking framework. In particular, we will:

- Find the best kernel and number of iterations for Diffusion.
- Investigate the best graph representation and hyperparameters for our Graph Neural Network.
- Incorporate our Gaussian Process, Diffusion, and Graph Neural Network predicted probabilities into our stacking model.
- Replace the simple decision tree meta model in our stacking framework with an ensemble tree method.
- Create a hit rate curve comparison between our final stacking model and the baseline XGBoost model.
- Use other metrics (e.g. effective cost) to evaluate our final stacking model.
- Explore different resolutions for hex-tiling the city (in terms of train/test splits) and different train/test split iterations to ensure our results are robust.

VII. Appendix A:

Figure A1: Comparison of the Hit Rate Curve by Parcel Investigated and Prediction Probability