Learning from the mundane: enhanced, data driven real estate management

Mariya Markhvida markhvid@stanford.edu, Emma Lejeune elejeune@stanford.edu

CS229: Machine Learning Project Fall 2016

1 Introduction and background

Everyday, hundreds of tasks are performed to maintain and manage commercial real estate. High level metrics such as property performance and profitability are influenced by these ground-level actions, however a quantitative understanding of their relationship has historically been difficult because of the lack of robust data. In commercial real estate, data sets – to the extent that they exist – are typically highly focused on a particular aspect of the business, and there is no normalization of key data elements among disparate data sets. Boxer Properties, an integrated commercial real estate owner and management company, has utilized a unique data management software (Stemmons Enterprise: http://stemmons.com) to record and manage their day-to-day operations in a consistent manner. With this software, each ground-level actions is referred to as a "case", cases are grouped within "case types", and the structure of each case is highly normalized. Applying machine learning techniques to this data set has the potential to readily quantify relationships, connect seemingly unrelated concepts and measures, shift tasks towards automation, and take a more data-driven approach to management.

In industry, machine learning approaches have been useful for tasks such as understanding and clustering customer interactions [8], identifying unexpected deviations in customer purchases [1], and marketing [12]. However, the success of these techniques is highly dependent on effective strategy and robust databases [10]. Presently, property valuation used for decision making is mostly done in a subjective manner, where experience, intuition and basic regression techniques are leveraged [6]. In some cases machine learning techniques have been used to aid and automate mass appraisal of properties, where input features typically include cadastral data, property transaction history and and general building information such as year of building construction, area, number of storeys and others [9, 2]. However, day-to-day operations are not commonly incorporated into property appraisal models.

The overall goal of this project was to explore how machine learning techniques, widely applied to other types of data, could be used to understand, aggregate and make predictions with the significant amount of building operations data provided by Boxer Property, spanning multiple facets of property management. We started off by exploring the data within single case types. In this paper, we describe our procedure as it applies to the most prevalent case type, property work orders (maintenance tasks assigned to building staff). In Section 2.3 we describe pre-processing of this data, in Section 3.1 we show that we can make simple predictions via supervised learning within the case type, and in Section 3.2 we perform unsupervised learning to visualize and aggregate the information in a meaningful way. With our methods, we were able to identify clear clusters in the data, treat these clusters as *emergent features*, and use them for aggregating data within a single case type such that it was suitable for inclusion in a predictive model involving multiple case types. We chose to predict monthly property performance, specifically monthly Net Operating Income (NOI) per square foot. To do this, we used emergent features combined with building specific descriptors (average rent, gross area, year of acquisition. etc.) to build a model described in Section 4. Given these features, we used panel analysis, decision trees and boosting as potential models to predict if monthly NOI would be above or below a certain threshold value. The goal was to see whether the emergent time-variant features and property descriptors could be used in tandem to predict financial property performance, and investigate different models for exploring this relationship.

2 Data summary and pre-processing

The data was provided to us in table format (SQL) by Boxer Property (http://www.boxerproperty.com). Building specific information, 2 years worth of monthly property financial performance data, and data on property day-to-day functioning and maintenance spanned numerous data sets (each data set contains a distinct "case type"), which had to be pre-processed and aggregated. For the property specific data, we could create features without significant pre-processing.

Table 1: Case list summary for 8 properties.

Case Type	No. Cases
Property work order	12,345
Tenant request	9,426
Customer survey follow-up	1,797
Tenant ledger change	1,677
Check-ins	1,303
Property project	1,034
Other, ex: Help desk, Move out prep.	5,459

2.1 Property specific data

Each property had a set of quantitative and categorical features. Categorical features consisted of: Market, Owner, Management Status, Bank associated with the property, Unit condition and Paint/carpet condition. Quantitative features consisted of: Unit layout rating, Property gross area, Available leasing area, Property take over year, Months since takeover, Year built, Year last renovated, Number of floors, Starting rate, Occupancy rate, Electric expenses, Parking space per sq. ft, and Monthly Rent. In Section 4 we refer to these quantities as "time-invariant" features because they do not fluctuate on a month-to-month basis within the time scale we considered. In addition, for each of the properties "time-variant" features were created by methodology described in Section 3.2. Together both

time variant features from previous month and time invariant features were used to predict the next month's NOI.

2.2 Case list raw data

The bulk of Boxer Property's data set is a detailed record of "cases", ground-level data where each case corresponds to someone at the company taking some action to remedy an issue, either preemptively or by request. Due to computational power limitations, the scope of our class project was limited to 8 properties in Dallas, Texas, under 5% of the total data set. However, our techniques are designed to be extensible to a larger data set. The case data used for this project is summarized in the Table 1, where each category of information is referred to as a "case type". In order to get access this data, both of us have signed a NDA (https://sites.stanford.edu/ico/ndas).

2.3 Pre-processing case data

As a baseline method for aggregation, for each case type in Table 1, we can construct model features without significant processing based on the number of case types opened, active, and closed at a given property for a given month. For the two largest case types, property work orders and tenant requests, we performed additional analysis, in part to gain additional insights to the data set. In this section, we will explain this process for work orders. A nearly identical process was performed for tenant requests, but is not discussed in depth here due to limited space. Figure 1 shows a sample

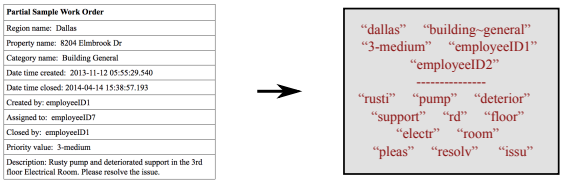

Figure 1: Selected fields of a sample work order are shown on the left. Each work order is converted to a "bag-of-words" prior to analysis. Here we show 5 out of 456 fields.

property work order, a collection of largely categorical data. Some of the data fields are selected from a drop down menu where a set number of words, phrases or identification numbers are selection options, we call these "fixed" fields. The work orders also contain an option to freely type a description. We extract the text from this field, remove numbers, symbols and punctuation, separate each word, remove stop words, and apply the Porter stemming algorithm [11]. We call the set of words created from this process "free" fields. Then, we concatenate the free fields and the fixed fields to turn each work order in to a "bag-of-words" model where many of the tokens from the fixed fields are not real words, instead they are phrases such as `2-major` or `employeeID1`. A simple alternative to the "bag-of-words" approach would be to use "one-hot" encoding to represent the categorical data in the fixed fields. However, since many of the fixed field words could only appear in one or two places, switching to one-hot encoding did not enhance the results seen in either Section 3.1 or Section 3.2 and required a much larger data matrix because many fixed fields have $10s - 100s$ of input options. In addition, a representation of the free fields that acknowledges that different words are highly correlated was not implemented, though it could lead to improvements in the future. The result of creating a "bag-of-words" model was that each work order could be represented as a feature vector, and all work orders could be represented as a feature matrix X with dimensions (no. work orders) x (no. tokens) where there are $3,358$ tokens.

3 Applying machine learning techniques within a single case type

3.1 Prediction of Work Order duration

Given this representation of data described in Section 2.3, we could not resist using it to try and make some simple predictions. Specifically, we decided to try and predict work order duration (date closed - date created) using matrix X. The median work order duration is 10 days, the mode is 0 days and the mean is 28 days. We attempted a binary classification, treating a positive result as a work order duration that exceeds 10 days. We implemented a naive Bayes multinomial event model with Laplace smoothing, a support vector machine (SVM) with a linear kernel and k-nearest neighbor (KNN) based prediction. Each algorithm was evaluated by creating a learning curve with a variable number of training samples and 500 test samples, shown in Fig. 2, and summarized in Table 2. The errors reported here correspond to the highest number of training samples tested for each method. For naive Bayes, the tokens most strongly associated with positive results were `employeeID2`, `present`, `employeeID5`, `employeeID6`, and `employeeID7`. The tokens most strongly associated with negative results (short-duration) were `employeeID8`, `preventative maintenance`, `employeeID9`, `employeeID10`, and `telesforom`. In addition, we performed naive Bayes with fixed tokens only, free tokens only, and all tokens, and saw that the best performance was achieved when all tokens were used, learning curve shown in Fig. 2. The performance of SVM was comparatively poor, both in computation time ($\approx$ 3 orders of

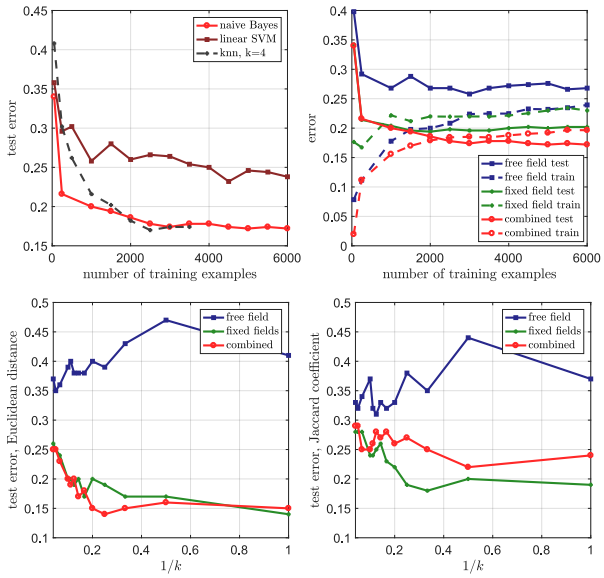

Figure 2: The upper left learning curve compares NB, SVM and KNN. The upper right learning curve compares the performance of naive Bayes with free fields only, fixed fields only, and both fields combined for both test and training error. For these curves, we used 500 test samples. The lower left curve is used to choose k with a Euclidian distance metric, and the lower right curve is used to choose k with the Jaccard coefficient. With 100 test samples and $1,000$ training samples, the best performance is attained with $k = 4$, combined fields and the Euclidian distance.

magnitude larger than naive Bayes for this sample size) and in error rates.

Table 2: Classification of work order duration

	Precision	Recall	Test Error
NB	0.82	0.84	0.17
SVM	0.77	0.45	0.24
KNN	0.90	0.74	0.174

Finally, to make a prediction with KNN, we had to first select k by choosing the value with the lowest test error, and choose a distance metric for computing the distance between two points. For our distance metric, we tried both searching for the points with the smallest Euclidian distance $d = ||x^{(1)} - x^{(2)}||$ and the smallest Jaccard coefficient, $J = (t^{(1)} \cdot t^{(2)})/(||t^{(1)}||^2 + ||t^{(2)}||^2 + t^{(1)} \cdot t^{(2)})$ where $t_i = 1\{x_i > 0\}$. Among many potential metrics of text similarity, the Jaccard coefficient has been shown to perform well [7]. The results of this small parameter study are shown in Fig. 2. For KNN with $k = 4$ and a Euclidian distance metric, corresponding to the best performance in the selection step, performance on the test data in the learning curve is summarized in Fig. 2 and Table 2. Because of the large computational time associated with KNN for high dimensional data (≥ 4 orders of magnitude than naive Bayes) we stopped the analysis as soon as the learning curve flattened. If Fig. 2, we also plot both the test and training error for the naive Bayes method. The fact that they converge indicates that the imperfect performance of the model is most likely due to bias. It is possible that relying on the naive Bayes assumption, that each feature element is conditionally independent given the duration category, is leading to this bias. KNN performed was nearly identical to naive Bayes, but was by far the slowest algorithm. If the main objective of this paper was to predict work order duration we would conduct this analysis in more detail by performing cross validation, investigating more potential methods to overcome the bias present in our current best performing method, and attempt to predict work order duration as a continuous variable.

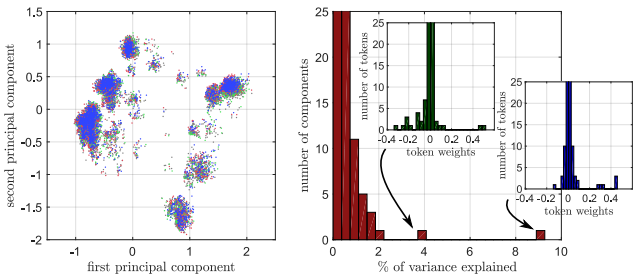

Figure 3: Left: the gray dots plot work orders in principal component space when PCA is applied to all work orders at once. Red, green, and blue dots show the results of applying PCA to three randomly selected sub-samples of 3,000 points each. Right: Histograms of the variance explained by each principal component, and histograms of the weights of different tokens within a principal component.

3.2 Clustering within a case type and emergent feature construction

In order to search for patterns within a given case type, we began by attempting k-means clustering of the work order data. However, the initial attempt went poorly because of high dimensionality and lack of significant evidence of distinct clusters even existing. Therefore, we turn to principal component analysis (PCA) to first visualize the data and look for patterns. Given matrix X, PCA computes the

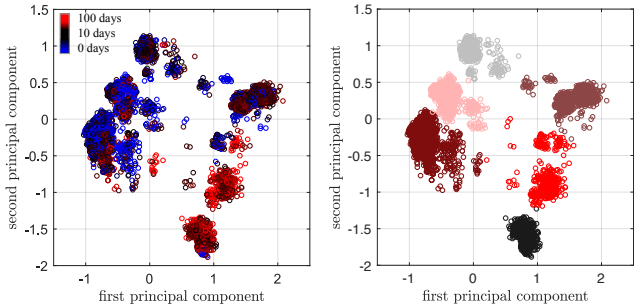

Figure 4: When the first two principal components of the work order data are plotted (12,345 work orders in total), several clusters emerge. In the left plot, work order duration is superimposed on the clusters. In the right plot, the six clusters divided by k-means clustering are shown.

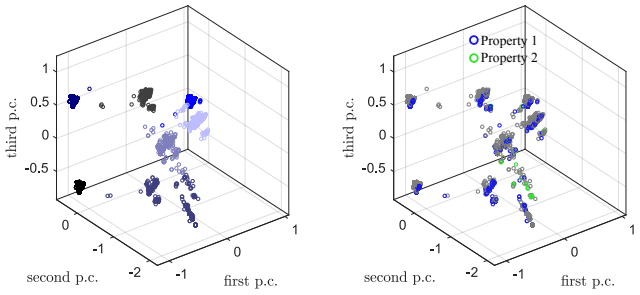

Figure 5: When the first three principal components of the tenant request data (9,426 tenant requests in total) are plotted, several very tightly packed clusters emerge. The left plot shows the seven clusters divided by k-means clustering and the right plot shows the variability between locations of tenant requests in principal component space from two different buildings.

eigenvectors associated with the largest eigenvalues of covariance matrix $\Sigma = \frac{1}{m} X X^T$. We did this using the MATLAB built in function `pca`. Figure 3 shows a histogram of how much variance is explained by each principal component, and histograms of the weights of each token in the first two principal component. We chose to cluster and visualize work order data with only the first two components because they explain a significantly higher amount of the variance in the data than the other components (outliers in the red histogram in Fig. 3, and because visualizing the data with three principal components did not significantly change the cluster behavior. When we attempted PCA using either "fixed" or "free" fields only the clusters were not nearly as well defined. Therefore, we focus only on analysis using both. For tenant requests, shown in Fig. 5, we found that the data clustered best in the first three principal components. For the first principal component of property work orders, the highest magnitude weights (outliers in the blue histogram in Fig. 3) were `resolv`, `issu`, `pleas`, `2-major`, `building general`, `employeeID1`, and `3-medium`. For the second principal component, the highest magnitude weights (outliers in the green histogram in Fig. 3) were `building general`, `2-major`, `employeeID2`, `new`, `pleas`, `issu`, `resolv`, `other`, `employeeID9`. However, the weights of unlisted tokens were not trivial. The orientations of principal component space is also consistent even when different

random samples of work orders, illustrated in Fig. 3, are chosen for analysis.

In the left plot in Fig. 4, color indicating work order duration is superimposed on the data in principal component space. From this visualization, we can easily identify clusters that take much longer to complete. We then grouped the data into six clusters using a K-means clustering algorithm with a Euclidian distance metric. Given these defined clusters, we note that the shortest duration cluster has a mean time to completion of 4 days while the longest duration cluster has a mean duration of 85 days. For each cluster identified using the procedure in Sec. 3.2, we defined the number of clusters of each type per property per month that were "opened" "active" and "closed". The remainder of the cases were simply grouped by case type. In the subsequent section, we refer to these case driven features as "time-variant" features.

4 Prediction of property performance using multiple case types

The general motivation behind building a model for predicting monthly property performance is that the property performance and building operations data from a given month could be used to anticipate the following month's NOI. For this analysis, we had access to monthly NOI data from January 2015 to November 2016. In this section, when we report our final test error, it is for the split where the training set was taken as January 2015 to July 2016, and the test set as August 2016 to November 2016 i.e. the most recent available data. While no time series analysis was performed, the idea of this model was to predict NOI in the future.

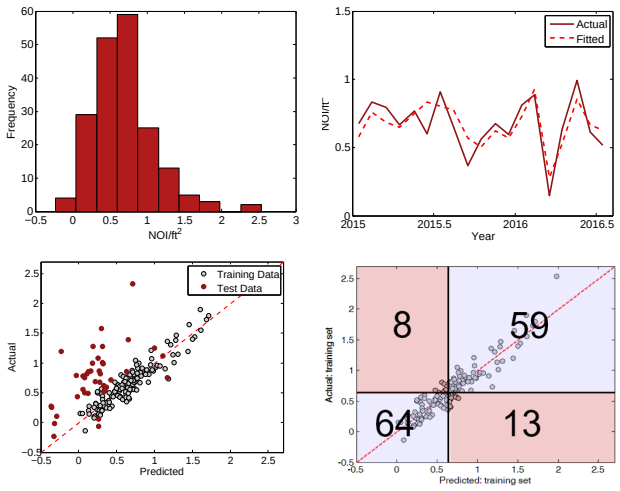

Figure 6: Upper Left: distribution of monthly NOI per square foot for 8 properties. Upper Right: actual NOI data and fitted model from panel analysis for one of the properties. Lower Left: predicated and actual NOI values for August-December 2016 based of the previous month's operations and performance data. Lower Right: predicted value assignment to 2 class confusion matrix, with "profitable" and "non-profitable" classification labels

.

4.1 Methods

Panel Analysis: Linear panel analysis is a regression commonly used in econometrics and social sciences. Panel or longitudinal data refers to data that is collected over time for multiple units. In our case this was data for 8 properties aggregated on a monthly basis, with input features that included both time variant and invariant inputs. To handle the time invariant features, "random effects" model was used as in Equation 1, where α and β are model coefficients to be estimated, x_{it} is a vector of features for property i at time t, μ_i is the individual error term (assumed to be uncorrelated with regressors), and ϵ_{it} is the idiosynchratic error independent of both regressors and individual error.

$$y_{it} = \alpha + \beta^T x_{it} + \mu_i + \epsilon_{it} \quad (1)$$

The model coefficients are obtained by using general least squares (GLS) regression [3]. Panel analysis was implemented using the *plm* R package. The model revealed several significant coefficients (p-value < 0.05) from the emergent features: number of active "type 1" tenant requests (negative coeff.), number of closed "type 5" tenant requests (positive coeff.), number of tenant check-ins (positive coeff.), number of other types of cases active (positive coeff.) and closed (negative coeff.), and number of active tenant ledger changes (negative coeff.). The root-mean-square-error (RMSE) of the model on the training data was 0.145, while on the test data it was 0.692, which means the model has limited predictive accuracy or utility. The results of the fitted model are visualized in Figure 6. Since the rest of the learning models used categorical labeling "profitable" (+) and "not profitable" (-), the panel analysis results were converted to classification results as shown in Figure 6. The threshold for classification was chosen as the median of all NOI data (0.67 dollars per ft^2).

Decision Trees: Classification decision trees were considered next, as they are relatively interpretable and are able to capture interactions between different features. Using this learning method allowed us to see which time-variant features constructed using PCA and K-means clustering would show up in our prediction of monthly NOI. In decision trees, each of the splits is made to maximize impurity reduction (as per Equation 2), where $I(A)$ is the impurity of node A, $p(A)$ is the probability of a future observation of node A, and A_L and A_R are two sons of the split of node A [13].

$$\Delta I = p(A)I(A) - p(A_L)I(A_L) - p(A_R)I(A_R) \quad (2)$$

The analysis was done using the R package *rpart*, where Gini index was used as the impurity function. In order to prevent overfitting, the size of the tree was chosen based the complexity parameter (cp) that yielded the lowest cross validation error with 10-fold cross validation (cp=0.041), as shown in Figure 7. The final decision tree with optimal pruning is also shown in Fig. 7. Figure 7 shows that while the first two splits are made on time invariant features (previous month's NOI and year build) further splits are made using the emergent time variant features, in particular, splits are made on work order clusters and tenant request cluster, which is similar to the panel analysis results. For example, work order type 2, which corresponds to the shortest duration work orders, shows up as a split. The error results are presented in Table 3.

Boosting: The boosting algorithm, which uses decision stumps as weak learners, was applied to the training data via the *ada* R package. The algorithm uses Stochastic Gradient Boosting on exponential loss function shown in Equa-

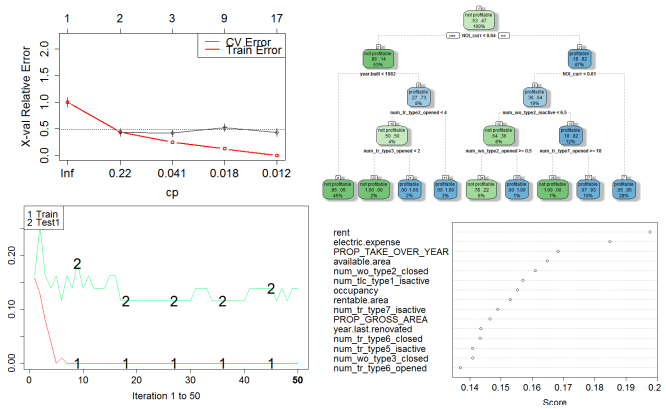

Figure 7: Upper Left: relative cross validation error for different number of splits, where the optimal cp=0.041. Upper Right: final pruned decision tree for prediction of monthly NOI. Lower Left: training and test (cross-validation) errors as a function of the number of iterations. Lower Right: variable importance scores.

tion 3, with regularization parameter ν [4].

$$L(\theta) = e^{-y\theta^T \phi(x)} \tag{3}$$

The model was tuned using hold-out cross validation (70/30). The regularization parameter was chosen as $\nu = 0.5$, a value that corresponded to the lowest cross-validation error. The number of iterations was chosen to be 18, since it was the number of iterations required to achieve the lowest error on the cross-validation test set (Figure 7). The variable importance, as defined by Hastie et al. [5], in shown in Fig. 7. Consistent with the results from the other modeling methods, boosting showed that while the first four most important features are the time invariant property descriptors, the subsequent features include emergent time variant features.

4.2 Result summary

Table 3 summarizes the errors from 144 training samples and 40 tests samples, as well as precision and recall. We can see that on the test data, boosting performed marginally better than decision trees, while panel analysis did not perform well on test set predictions. Boosting is also the preferred method when considering precision and recall. Although these results are highly preliminary, it is possible that boosting is performing well because the presence of each different case type is a form of NOI predicting "weak learner" which suits the theory behind boosting. However, since we have so many features and so few training examples, and our training error is significantly lower than our test error, we suspect that our results are hurt by and largely indicate high variance. In the future, implementing these models with $1-2$ orders of magnitude more data is how we would choose to remedy this issue.

Table 3: Summary of the results for three methods used - precision and recall are calculated on the test data.

	Training Error	**Test Error**	**Precision**	**Recall**
Panel Analysis	14.6%	50%	100%	20%
Decision Trees	5.6%	22.5%	78.5%	88%
Boosting	0%	20%	79.3%	92%

5 Conclusions and Future Work

We began this project with a limited sense of what the building operations data set contained and a desire to demonstrate that a machine learning approach had the potential to extract useful information from this data. During the initial investigations, our analysis focused on understanding data within a specific case type. To handle the large volume of categorical data present within a case, we used a "bag-of-words" model. Then, we used this model to classify work order duration. The results of this analysis are summarized in Table 2. The naive Bayes approach performed the best with a test error of 0.17. We then utilized PCA and showed that dimensionality reduction helps visualize patterns in this data set. Because clear clusters emerged in principal component space, we could construct features out of building operations data using k-means clustering, to be further utilized in prediction of monthly property performance. The emergent time variant features along with property description features showed potential for predicting future monthly net operating income - a novel approach to financial performance estimation with greater insight from aggregation of day-to-day operations.

The next steps to this project follow three major directions. **First**, we would revise our work flow, and upgrade hardware such that analysis of more data is possible. With a revised approach, processing significantly more building operations data would be possible, which would allow for a larger number of examples and a more robust model for predicting NOI with proper cross validation techniques. In addition, more features such as internal data on tenants and external data on the economy should be introduced to the model once we have enough data such that high variance is less of an issue. We also note that nearly all of our results thus far indicate correlation, rather than causation. With further analysis, particular conducted by someone with high familiarity with the company, inferences about causal relationships may be more feasible. **Second**, we would more rigorously study the patterns that emerge from clustering the data. As a starting point, the initial choice of six clusters for the work order duration data was somewhat arbitrary. With further analysis, we would be able to more rigorously identify clusters and sub-clusters and potentially some unintuitive yet meaningful groupings. The real power of this analysis will come when we are able to identify patterns that exceed what is possible with human intuition alone. **Third**, perhaps the least novel from a machine-learning perspective, but the most pragmatic from an industry perspective, would be to use simple supervised learning techniques, such as the ones we implemented to predict work order duration, to make other predictions within case types. For example, a similar supervised learning approach may be a successful way to deal with categorical tenant data and predict the probability of a certain tenant paying their rent late. Or, supervised learning techniques could be used to automate simple decision making tasks within a given case type. With enough years of recorded data, learning techniques could also be used to predict a need for building repairs by anticipating work orders before they are issued. We believe that by capitalizing on the potential of their data, Boxer Property can become an industry leader in applying machine learning techniques to operations data for enhanced property management.

References

[1] Emin Aleskerov, Bernd Freisleben, and Bharat Rao. Cardwatch: A neural network based database mining system for credit card fraud detection. In *Computational Intelligence for Financial Engineering (CIFEr), 1997., Proceedings of the IEEE/IAFE 1997*, pages 220–226. IEEE, 1997.

[2] Marco Aurélio Stumpf González and Carlos Torres Formoso. Mass appraisal with genetic fuzzy rule-based systems. *Property Management*, 24(1):20–30, 2006.

[3] Yves Croissant and Giovanni Millo. Panel data econometrics in R: The plm package. *Journal of Statistical Software*, 27(2), 2008.

[4] Mark Culp, Kjell Johnson, and George Michailidis. ada: An r package for stochastic boosting. *Journal of Statistical Software*, 17(2):9, 2006.

[5] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*, volume 1. Springer series in statistics Springer, Berlin, 2001.

[6] Magdalena Graczyk, Tadeusz Lasota, and Bogdan Trawiński. Comparative analysis of premises valuation models using keel, rapidminer, and weka. In *International Conference on Computational Collective Intelligence*, pages 800–812. Springer, 2009.

[7] A. Huang. Similarity measures for text document clustering. In *Proceedings of the sixth new zealand computer science research student conference (NZCSRSC2008)*, pages 49–56, Christchurch, New Zealand, 2008.

[8] Siu Cheung Hui and G Jha. Data mining for customer service support. *Information & Management*, 38(1):1–13, 2000.

[9] Olgierd Kempa, Tadeusz Lasota, Zbigniew Telec, and Bogdan Trawiński. Investigation of bagging ensembles of genetic neural networks and fuzzy systems for real estate appraisal. In *Asian Conference on Intelligent Information and Database Systems*, pages 323–332. Springer, 2011.

[10] Vincent P Magnini, Earl D Honeycutt Jr, and Sharon K Hodge. Data mining for hotel firms: Use and limitations. *Cornell Hospitality Quarterly*, 44(2):94, 2003.

[11] M. F. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

[12] Michael J Shaw, Chandrasekar Subramaniam, Gek Woo Tan, and Michael E Welge. Knowledge management and data mining for marketing. *Decision support systems*, 31(1):127–137, 2001.

[13] Terry M Therneau and Elizabeth J Atkinson. An introduction to recursive partitioning using the rpart routines, 2015.