L2P: An Algorithm for Estimating Heavy-tailed Outcomes

Xindi Wang
Ctr. for Complex Network Research
Northeastern University
Boston, MA
wang.xind@husky.neu.edu

Onur Varol
Ctr. for Complex Network Research
Northeastern University
Boston, MA
ovarol@northeastern.edu

Tina Eliassi-Rad
Khoury College of Computer Sciences
Northeastern University
Boston, MA
eliassi@northeastern.edu

ABSTRACT

Many real-world prediction tasks have outcome (a.k.a. target or response) variables that have characteristic heavy-tail distributions. Examples include copies of books sold, auction prices of art pieces, etc. By learning heavy-tailed distributions, "big and rare" instances (e.g., the best-sellers) will have accurate predictions. Most existing approaches are not dedicated to learning heavy-tailed distribution; thus, they heavily under-predict such instances. To tackle this problem, we introduce *Learning to Place* (L2P), which exploits the pairwise relationships between instances to learn from a proportionally higher number of rare instances. L2P consists of two stages. In Stage 1, L2P learns a pairwise preference classifier: *is instance A > instance B?*. In Stage 2, L2P learns to place a new instance into an ordinal ranking of known instances. Based on its placement, the new instance is then assigned a value for its outcome variable. Experiments on real data show that L2P outperforms competing approaches in terms of accuracy and capability to reproduce heavy-tailed outcome distribution. In addition, L2P can provide an interpretable model with explainable outcomes by placing each predicted instance in context with its comparable neighbors.

KEYWORDS

Learning to place, heavy-tailed distributions, supervised learning

1 INTRODUCTION

We address the problem of predicting the value of a heavy-tailed outcome (a.k.a. target or response) variable in a supervised setting. By heavy-tailed, we mean a variable whose distribution has a heavier tail than the exponential distribution. Examples include predicting bestselling books, art auction price, detecting rare events [24, 26, 37] and viral content [30, 39, 41]. In most models, such instances are considered anomalies. For example, the start-up industry often refers to such big and rare instances as unicorns.

When predicting for heavy-tailed outcomes, traditional approaches produce large errors on the rare instances at the tail of the distribution (by under-predicting such instances). The limiting factor for prediction performance is the insufficient amount of training data on the rare instances. Collecting more data is not a solution to this problem because these instances are rare. Approaches such as over-sampling training instances [6], adjusting weights, and adding extra constraints [8, 13, 22, 28, 32, 40] do not properly address the aforementioned problem because they were introduced to address the class imbalance problem, which is different from predicting heavy-tailed distributed outcomes.

Existing classification and regression approaches for imbalanced data mostly assume groups of homogeneously distributed instances with proportionally different sizes. Separating instances from different groups is relatively easier since within group and between group distances can be significantly different. Higher-order moments and variance of heavy-tailed distributions are not well-defined. Thus, statistical methods with assumptions on the outcome distribution's variance lead to biased estimates on such data. In addition, defining distinct groups on a dataset with heavy-tailed outcomes is not trivial. Therefore, predicting the values of heavy-tailed variables is not merely a class imbalance problem; instead it is the problem of learning a heavy-tailed distribution.

Here, we propose a novel approach called *Learning to Place* (L2P) to estimate heavy-tailed outcomes and define performance measures for heavy-tailed target variables, while addressing the known limitations of previous methods such as under-prediction of the rare instances [22] and limitations of traditional regression performance measurements. Our approach learns to estimate a heavy-tailed distribution by first learning pairwise preferences between the instances and then placing new (i.e., never-before-seen) instances into the ordinal ranking of the known instances and generating a value for their outcome variables. Our contributions are as follows:

- We introduce *Learning to Place* (L2P) to estimate heavy-tailed outcomes by learning from the pairwise relationships between instances and placing the new instance into perspective with known samples from training data and predict outcomes. L2P produces interpretable models by providing additional context on relative relations with training instances and their features.
- We propose appropriate statistical metrics to measure the performance of heavy-tail distribution learning task and support performance of models by employing visual analysis.
- In an exhaustive empirical study, we demonstrate that L2P is robust, and consistently outperforms various competing approaches across diverse real-world datasets.

The outline of the paper is as follows. We describe L2P next. In Section 3, we present the experiments, followed by related work and discussion in Section 4. We conclude the paper in Section 5.

2 PROPOSED METHOD: LEARNING TO PLACE (L2P)

Our approach, L2P, takes as input a data matrix where the rows are data instances (e.g., books) and the columns are features that describe each instance (e.g., author, publisher, *etc*). Each data instance also has a value for the predefined target variable (e.g., copies of book sold). L2P learns to map each instance's feature vector to the value for its target variable. This is the standard supervised learning setup. However, the challenges that L2P addresses are as follows. First, it

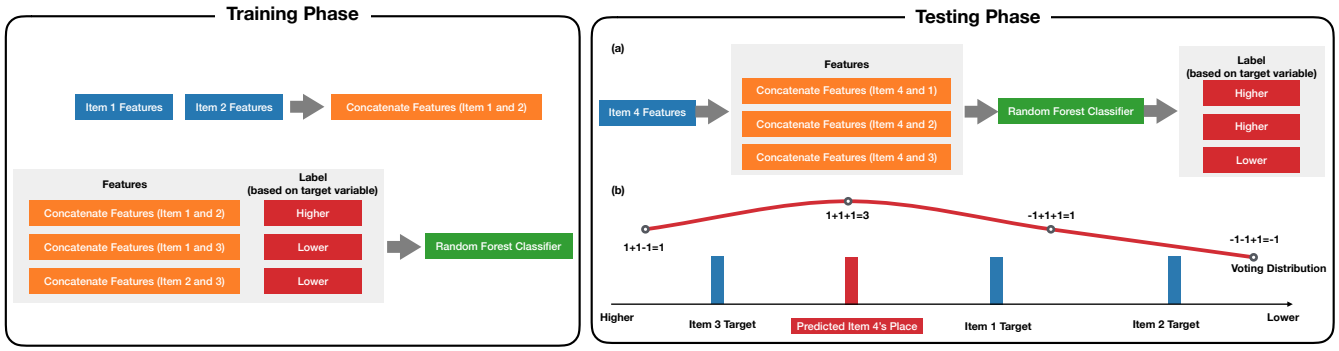

Figure 1: Learning to place (L2P) schema. Other than directly predicting the target variable, here we compare instances pairwise and predict the value based on the pairwise relationship. In the training phase, we train a classifier C on the pairwise relationship between each pair of train instances. In the testing phase, when a new (test) instance q coming in, we apply the classifier C on the test instance against all train instances and obtain the pairwise relationship between q and all train instances. Then, each instance in the training dataset contributes to the placement of testing instance q by voting on bins to its left or to its right depending on the estimated relation between instances. The middle value of the bin having the maximum vote is the prediction for the test instance.

learns the heavy-tailed distribution of the target variable; and thus it does not under-predict the "big and rare" instances. Second, it generates an interpretable model for the human end-user (e.g., a publisher) to employ; that is, the human end-user can interpret the reasoning behind the model.

L2P algorithm like any other supervised model learns from data and uses models learned in this phase to predict outcomes for a given test instances. In the training phase, L2P learns a pairwise relationship classifier, which predicts whether the target variable for an instance A is greater (or less) than another instance B. To predict outcomes in the testing phase, the new instance is compared with each training instances using the model learned in the training phase to predict pairwise relations. Those pairwise relations later use as "votes" to predict target outcomes. The detailed training and testing phase is described as follows and graphically in Figure 1:

Training Phase (Algorithm 1). For each pair of instances i and j with feature vector f_i and f_j, L2P concatenates the two feature vectors $X_{ij} = [f_i, f_j]$. If i's target variable is greater than j's, then $y_{ij} = 1$; otherwise, $y_{ij} = -1$ (ties are ignored in the training phase). Formally, denoting with t_i the target variable for instance i and with S the set of instances in the training set, L2P generates the following training data:

$$X_{ij} = [f_i, f_j], \text{ for each} (i, j) \in S \times S, i \neq j\ , t_i \neq t_j, \tag{1}$$

$$y_{ij} = \begin{cases} 1, & t_i > t_j \\ -1, & t_i < t_j \end{cases}. \tag{2}$$

Then a classifier C is trained on the training data X_{ij} and labels y_{ij}.[1] It is important to note that the trained classifier may produce conflicting results; for example, $A < B$ and $B < C$ but $C < A$. In the Experiments section, we discuss the robustness of L2P to such conflicts caused by the misclassification of the binary classifier learned in this phase.

Testing Phase (Algorithm 2). The testing phase consists of two stages. In Stage I, for each test instance q L2P obtains $X_{iq} = [f_i, f_q]$, for each $i \in S$ (recall S is the training set). Then, L2P applies the classifier C on X_{iq} to get the predicted pairwise relationship between the test instance q and all training instances ($\hat{y}_{iq} = C(X_{iq})$). In Stage II (the *estimate placement for test instance* stage), L2P treats each training instance as a "voter". Training instances (voters) are sorted by their target variables in descending order, dividing the target variable axis into bins. If $\hat{y}_{iq} = 1$, bins on the right of t_i will obtain an *upvote* (+1) and bins on the left of t_i will obtain a *downvote* (-1). If $\hat{y}_{iq} = -1$, i will upvote for bins on the left of t_i and downvote for bins on the right of the t_i. After the voting process, L2P obtains a voting distribution over the bins. It then takes the bin with the most "votes" as the predicted bin for test instance q, and obtains the prediction $\hat{t}_q$ as the midpoint of this bin.

We prove that L2P's voting process is the maximum likelihood estimation (MLE) of the optimal placement of an instance based on the pairwise relationships. Given the test instance q, our goal is to find its optimal bin m. For any bin b, we have: $P(b|q) \propto P(q|b) \times P(b)$. Since each train instance i contributes to $P(q|b)$, we have

$$P(q|b) = \frac{1}{Z} \sum_{i \in S_{train}} P_i(q|b), \tag{3}$$

where $P_i(q|b)$ is the conditional probability of test instance q placing in the given bin b based on its pairwise relationship with training instance i; and Z is the normalization factor, $Z = \sum_b \sum_i P_i(q|b)$.

L2P assigns two probabilities to each pair of training instance i and test instance q: $p_i^l(q)$ and $p_i^r(q)$, which respectively denote the probability that the test instance q is smaller than (i.e., to the left of) or larger than (i.e., to the right of) training instance i. Obviously, $p_i^l(q) + p_i^r(q) = 1$. Let $R_b^i \in \{l, r\}$ be the region defined by training instance i for bin b, and $|R_b^i|$ as the number of bins in this region.

[1] Having a single (i, j) pair or training on symmetric pairs (including both (i, j) and (j, i) do not lead difference on model performance.

Algorithm 1: Training phase of L2P

Input: Training data S consisting of feature matrix F and target variable vector t
Output: Pairwise relationship classifier C

```
X = [ ];          // Concatenated feature matrix
y = [ ];                              // Label vector
for i ← 1 to |S| do
    for j ← i + 1 to |S| do
        X.append([f_i, f_j]);                 // Concatenate
        if t_i > t_j then
            y.append(1);
        else if t_i < t_j then
            y.append(-1);
    end
end
C.train(X, y);                              // Train a model
return C
```

We know that

$$P_i(q|b) = \frac{p_i^{R_b^i}(q)}{|R_b^i|} \tag{4}$$

assuming the test instance is equally probable to fall in each bin in region R_m^i. Therefore, the optimal bin

$$m = \arg\max_b \frac{1}{Z} \sum_{i \in S} \frac{p_i^{R_b^i}(q)}{|R_b^i|}. \tag{5}$$

We observe that $\frac{p_i^{R_b^i}(q)}{|R_b^i|}$ is actually the "votes" the training instance i gives to bin b for test instance q, therefore the optimal bin m is the one with the most "votes". Notice that by using upvotes (+1) and downvotes (-1) in our approach, we are basically standardizing $p_i^l(q)$ and $p_i^r(q)$.

L2P can incorporate any method that takes pairwise preferences and learns to place a test instance among the training instances. Besides voting, we experimented with other approaches to estimate placement of a test instance. Specifically, we examined SpringRank [11], FAS-PIVOT [2] and tournament graph related heuristics [10]. We found that the performances of these approaches are quite similar to voting. However, voting – with its linear runtime complexity – is the most efficient method among them.

Complexity analysis. The training phase of L2P requires learning pairwise relationship of all the pairs in the training set, leading to a $O(n^2)$ complexity. Since this is computationally expensive for large dataset [33], we later discuss techniques to reduce the number of pairs needed for the classifier. The testing phase has $O(n)$ complexity.

3 EXPERIMENTS

We study the performance of L2P in various datasets. In this section, we describe the data used in our experiments, the baseline and competing approaches, our experimental methodology and evaluation metrics to introduce concepts required to interpret our experimental results in the following section.

Algorithm 2: Testing phase of L2P

Input: Classifier C, Training data S=(F, t), Test instance q represented by its features f_q
Output: t_q = predicted value for test instance q

```
B = [ ];                                  // Vote counter
bins = sort(unique(t));     // Unique target values,
 highest to lowest
for i ← 1 to |bins| do
    B[i] = 0;
end
for i ← 1 to |F| do
    ŷ_iq = C.predict([f_i, f_q]);
    for j ← 1 to BinEdgeIndex(t_i) - 1 do
        B[j] -= ŷ_iq;              // Vote preceding bins
    end
    for j ← BinEdgeIndex(t_i) to |B| do
        B[j] += ŷ_iq;              // Vote subsequent bins
    end
end
b = GetHighestBin(B);                      // MLE over bins
t_q = Mean(bins[b-1], bins[b]);          // Get prediction
return t_q
```

3.1 Datasets

We present results on four real-world applications – prediction of book sales, artwork auction prices, COMPAS recidivism risks, and a synthetic network dataset. Table 1 provides the summary statistics of these datasets. Specifically, we calculate the kurtosis for each target variable. Kurtosis measures the "tailedness" of the probability distribution of a real-valued random variable. The kurtosis of any univariate normal distribution is 3, and the higher the kurtosis is, the heavier the tails. Distribution of real outcomes are also presented as complementary cumulative function (CCDF) in Fig. 2. Real-world datasets exhibit at least two order of magnitude difference between extreme values of the distribution.

Book sales: This dataset consists of information about all print nonfiction books published in the United States in 2015, including details about authors publication history, a summary of the book, as well as Wikipedia pageviews as a proxy to authors popularity [41]. The goal is to predict the book sales using the given features prior to the book's publication [12].

Art auctions: This dataset was collected by a company operating in the art world that combines information on artists exhibits, auction sales, and primary market quotes. It was previously used to quantify success of artists based on their trajectories of exhibitions [14]. We only select the paintings in the dataset and sampled 7,764 from them using vertical logarithmic binning [17]. Here we try to predict auction sale of an art piece based on artists previous sale and exhibition history.

COMPAS recidivism: We use the COMPAS (Correctional Offender Management Profiling for Alternative Sanctions) dataset released by ProPublica[2] for our analysis. This dataset records criminal

[2]https://github.com/propublica/compas-analysis

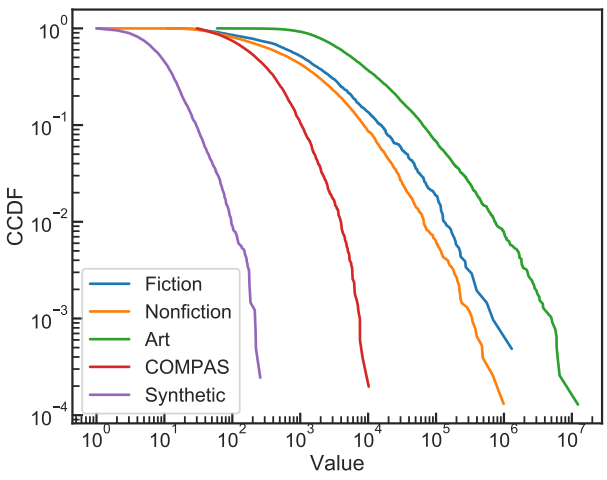

Figure 2: Distribution of target variables presented for each dataset. Heavy-tailed nature of those distribution and saturation on smaller values can be seen from the distributions.

Table 1: Dataset summary statistics and measure of "tailedness" by kurtosis of the target distributions.

Name	Instances	Features	Target Variable	Kurtosis
Fiction	2,061	55	Book sales	443.08
Nonfiction	7,641	55	Book sales	619.73
Art	7,764	21	Sale price	597.82
COMPAS	5,061	16	Inter-crime time	40.58
Synthetic	4,096	9	Node degree	36.40

defendants in Broward County, Florida, including information such as demographics and the individual's past criminal record [12]. The goal is to predict the time between adjacent crimes (inter-crime time) based on features regarding the defendant's crime history.

Synthetic dataset: In addition to the real-world datasets, we generate a network with heavy-tail degree distribution using the Multiplicative Graph Model [21], where nodes are assigned to attribute vectors. The goal is to predict the degree of a node based on its attribute vector.

3.2 Baseline and Competing Approaches

To compare the predictive capabilities of `L2P`, we experiment with other competing approaches from the literature.

K-nearest neighbors regression (kNN): We employ regression based on k-nearest neighbors. In this model, target variable is predicted by local interpolation of the targets associated with the nearest neighbors in the training set. We consider 5 neighbors ($k = 5$) and Euclidean distance between instances.

Kernel regression (KR): Since the dataset might have non-linear relations, we employ Ridge regression with a RBF kernel to estimate non-linear relation between coveriates and target variable.

Heavy-tail linear regression (HLR): Hsu *et al.* [19] proposed heavy-tail regression model, in which a median-of-means technique is utilized. They proved that a random sample of size $\tilde{O}(dlog(1/\delta))$ is sufficient to obtain a constant factor approximation to the optimal loss with probability $1 - \delta$. However, this approach is not able to capture non-linear relationships, which exist in our setting.

Neural networks (NN): We train a multi-layer perceptron regressor with one hidden layer of 100 neurons and with a regularization term $\alpha = 0.0001$ added to the loss function that shrinks model parameters to prevent overfitting. Adam solver [23] is used to minimize the squared-loss and optimize model parameters. However, neural networks are in general not interpretable.

XGBoost (XGB): Efficient and scalable implementation of gradient boosting proposed by Friedman *et al.* [15] and it has been optimized to perform tree boosting approach to various tasks such as regression, classification and ranking [9].

RankSVM: Joachims [20] introduced RankSVM, which is a two-stage approach. It translates optimization of learning weights for ranking functions into an SVM classification problem. RankSVM is designed to predict the ranking among a list of items; here we adjust it by directly computing predicted value as the midpoint between the actual values of adjacent ranked items.[3] However, RankSVM is a computationally expensive approach in both training and testing, since the testing phase is ranking all the train instances plus the test instance into an ordinal ranking, which requires $O(n^2)$ complexity.

Random (RDM): We define a random baseline as the actual outcomes are shuffled at random and assigned as predictions.

3.3 Experimental Setup and Evaluation Metrics

For all results presented in this work, we follow the same experimental setting and evaluation metrics. We employ 5-fold stratified cross-validation to estimate confidence of model performance. For `L2P`, we choose random forest classifier with 100 trees and Gini impurity as split criteria to learn pairwise preferences. Like others, we find that random forest has good performance (does not overfit) and provides interpretablity of features and results.

In this work, we emphasize importance on learning heavy-tailed outcome distributions. Traditional regression metrics are not of good fit in this problem setting. For example R^2 is calculated under the assumption that the error is normally distributed, which is not the case for heavy tailed distributions; root mean square error (RMSE) will be dominated by the errors on the high end since the values on high-end are extreme. Other statistical metrics such as Mann-Whitney statistics and rank correlations also biased due to ties between instances at the lower-end.

Under our problem definition, the model with the best performance will (i) reproduce heavy-tailed outcome distribution, (ii) predict all instances accurately especially at the tail of distribution. Therefore, we have the following metrics that fit better under our circumstances for evaluation:

Quantile-quantile (Q-Q) plot: Q-Q plot can visually present the deviations between true and predicted target variable distributions. In addition, we can also investigate parts of distributions contributing to deviations.

Kolmogorov–Smirnov statistic (KS): Kolmogorov–Smirnov statistic is a commonly used statistic to measure the distance between two underlying one-dimensional probability distributions. Since our

[3] RankSVM code is obtained from https://gist.github.com/fabianp/2020955.

goal is to predict attributes that are heavy-tailed distributed, methods should be able to recover the true distribution of the actual values. Therefore, we calculate the two-sample Kolmogorov–Smirnov statistic between the predicted values and the true values. A better method will have smaller Kolmogorov–Smirnov statistic.

Earth mover distance (EMD): Similar to Kolmogorov–Smirnov statistic, EMD (also called Wasserstein metric) is another commonly used statistic to measure distance between two distributions. Small earth mover distance indicates higher similarity between distributions and in our analysis better reproducing underlying distribution.

Receiver operating characteristic (ROC): We calculate true positive rates and false positive rates introduced below to compute ROC curve and the corresponding area under the curve (AUC) score. Traditional way of calculating ROC and corresponding AUC lies in the classification space. Here, we adapt it to regression setting by calculating true positive rate and false positive rate at different threshold (all possible actual values in our case), and obtain the curve and corresponding AUC.

True positive rate at threshold. The calculation of true positive rate (TPR) at threshold follows the fashion of the metric recall@k frequently used in information retrieval literature. Here, we investigate whether an instance with true value (y) higher than threshold t actually be predicted ($\hat{y}$) to be higher than t.

$$TPR@t = \frac{|\{\hat{y}_i \geq t, y_i \geq t\}|}{|\{y_i \geq t\}|}.$$

False positive rate at threshold. Similar to TPR at threshold, one can calculate false positive rate at threshold as:

$$FPR@t = \frac{|\{\hat{y}_i \geq t, y_i < t\}|}{|\{y_i < t\}|}.$$

For various thresholds, we can compute corresponding TPR and FPR scores to create ROC curve. Similar to traditional ROC curve, a better performing method would have a curve that is simultaneously improving both true positive rate and false positive rate, leading to a perfect score of AUC = 1. A random model leads to a performance of AUC = 0.5 and corresponding ROC curve aligns with 45-degree line indicating that TPR and FPR are equal for various thresholds.

We also want to note that each individual measure is not sufficient enough to judge the goodness of a model. KS, EMD and Q-Q plot are measuring the reproducibility of the heavy-tailed distribution, but are not able to measure the prediction accuracy for each instance. AUC is measuring the accuracy of the prediction, but didn't take into account model's ability to reproduce the distribution.

3.4 Results

Here, we present the experimental results of L2P in comparison with different baseline algorithms. We present our experimental results on various datasets. In particular, we show performance comparisons, robustness analysis, and case studies to illustrate interpretability of L2P outcomes.

3.4.1 Performance comparison study. Here we present performance of L2P and other competing methods using metrics that we proposed above.

Q-Q plot. We investigate distribution of predicted outcomes presented in Fig. 3. Best performing models in this analysis should produce curve closer to $y = x$ line and we can study when predicted quantiles deviate from this line. In all datasets except synthetic, we see deviation at the high-end. In Fiction, L2P is among the top 3 methods that produce the smallest deviation from the high end; the other two are RankSVM and Neural Network, but they produce larger deviation at low end than L2P. In nonfiction, L2P produces the least deviations on both low and high ends. In art dataset, RankSVM and L2P produce least deviation on the high end but L2P has lower deviation at the low end than RankSVM. Performance on COMPAS dataset is challenging for all methods. They exhibit deviations on lower and higher values, while in the mid-range RankSVM and L2P achieves best performance. Finally on the simple synthetic dataset, all methods produce similar results.

KS and EMD. To quantify differences between distributions of predicted outcomes and target values, we compute KS statistics and earth-mover distance and results are presented in Table 2. CCDF of the outcome distributions against actual distributions for selected methods are shown in Figure 4. Outcomes of L2P leads the smallest KS statistics for almost all datasets (except COMPAS). RankSVM shows an advantage on minimizing EMD as well as visually comparing with the actual distribution. However, we want to note that reproducing distribution of true outcome is not sufficient to evaluate models by itself, since error between predicted and true values are not directly measured by these metrics.

AUC. To quantify performance of models using the prediction errors, we calculate the AUC measure introduced earlier. Performance of models on different datasets summarized in Fig. 5. In this comparison, L2P achieves the best performance 0.895 ± 0.002 and 0.872 ± 0.005 AUC score on fiction and nonfiction datasets respectively. Experiment on art dataset points similar performance of AUC 0.821 ± 0.007 but differences between top four methods are statistically insignificant. We notice that RankSVM, which has good performance under KS and EMD, has very low AUC score, indicating its inability to have accurate prediction on each instance. Synthetic dataset is a simple dataset compared to the real datasets (less features and smaller kurtosis) and all methods achieves comparable performance on it.

Summary. With comprehensive consideration of all three evaluations, we can see that L2P is the best method in both reproducing the underlying heavy-tail distribution and providing accurate predictions. We also observe that for the three most heavy-tailed datasets – fiction, nonfiction and art, L2P is the method having the highest performance, which shows the power of L2P on heavy-tail distributed outcomes; for the less heavy-tailed dataset COMPAS and synthetic, L2P is still one of the top methods.

3.4.2 Robustness analysis. L2P is a two-stage method consisting of a pairwise learning algorithm in the first stage and a voting algorithm in the second stage. Previously we showed that voting itself is a maximum likelihood estimation, therefore the performance of L2P is highly relying on the performance of pairwise relationship learning. Here, we investigate the robustness with respect to classification error in Stage 1 of L2P.

To quantify the error tolerance of "voting" and estimation for the new instance (Stage 2) of L2P, we conduct a set of experiments where we introduce errors on predicting pairwise relationships. Here

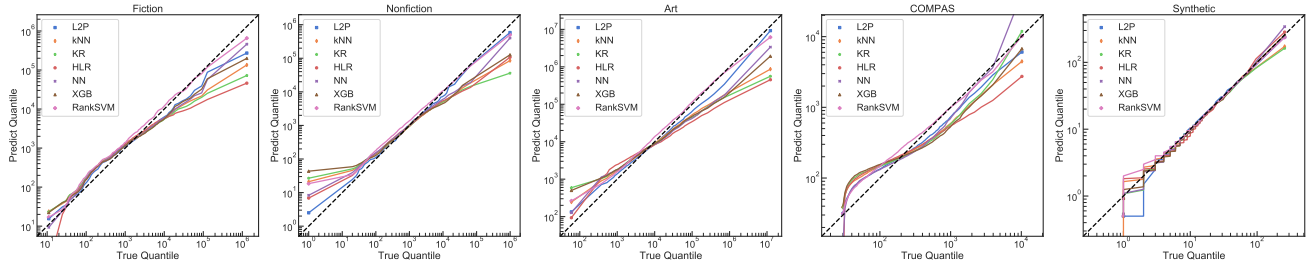

Figure 3: Predicted outcomes are compared with respect to underlying true distribution. Visualization of Q-Q plot points the part of the distribution leading the greatest deviation. Outcomes of `L2P` reproduce underlying distribution closely at the both lower and higher quantiles comparing to other methods. `L2P` demonstrates one of the best methods aligned with the true distribution and produce minimum deviation at the lower and higher quantiles. RankSVM produces small deviation on the high end as well but it generally has larger deviation in the low end comparing to `L2P`.

Table 2: Kolmogorov–Smirnov (KS) statistic and Earth mover distance (EMD). We measure KS statistic and EMD to compare prediction distribution and the actual distribution. We also highlight best two models for each measure. We can see that across various datasets, `L2P` is always among the top 2 of lowest KS statistic and EMD across different datasets. Other approaches are not consistent across various datasets.

Method	Fiction		Nonfiction		Art		COMPAS		Synthetic	
	KS	EMD	KS	EMD	KS	EMD	KS	EMD	KS	EMD
L2P	**0.058**	**4136**	**0.059**	**1334**	**0.068**	**20680**	**0.129**	**113**	**0.063**	0.78
kNN	0.084	5809	0.090	2880	**0.102**	30718	0.169	171	**0.067**	0.81
KR	0.082	6137	0.096	3003	0.117	33194	0.198	166	0.100	1.13
HLR	0.092	6437	**0.077**	2958	0.195	36268	0.182	210	0.115	1.22
NN	0.076	5105	0.089	2643	0.112	32400	0.135	118	0.098	**0.65**
XGBoost	**0.073**	4761	0.108	2749	0.118	31762	0.214	188	0.097	0.80
RankSVM	0.098	**1515**	0.093	**723**	0.105	**10127**	**0.126**	**64**	0.077	**0.49**

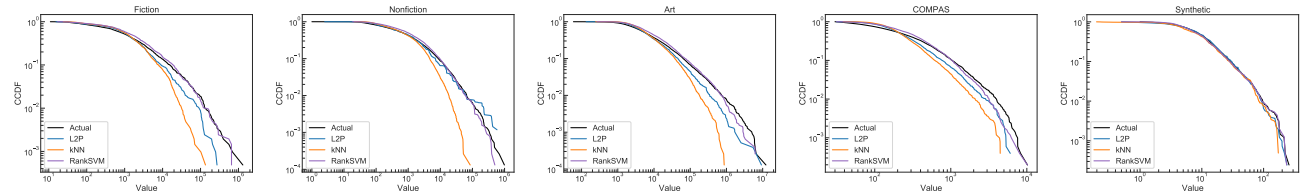

Figure 4: Predicted outcomes CCDF are compared with respect to underlying actual values CCDF. We select methods that are ranked top 2 at least twice using KS and EMD. Visually, for most of the datasets RankSVM has the closest distribution to the true distribution. `L2P` is among the top 2 methods that have the closest distribution. However one has to note that having the closet distribution does not necessarily mean the method has the best performance.

we simulate the pairwise relationship error with two mechanisms: (i) *random error*: constant probability $p = p_c$ flips the label for each pair, (ii) *distance-dependent error*: probability of error is proportional to the true ranking percentile difference between items; here we use the percentile of the ranking because the sizes of the datasets vary. We define the flipping probability as $p_{ij} = e^{-\alpha|r_i - r_j|}$, assuming it would be easier to learn the pairwise relationship for items that are further away. This is observed in our experiments as well. In nonfiction data, we notice that more than 48% of the pairwise relationship error occurs in item pairs that have a ranking percentile difference smaller than 10. We can control the rate of errors introduced by the two mechanisms by tuning p_c or α.

In Figure 6, we present the overall performance (AUC) of `L2P` when various degrees of errors are introduced to the system in Stage 1 for different datasets. First thing to notice is that if the pairwise relationship has no error (see left panel of Figure 6 when classifier accuracy is 1), `L2P` has an accurate prediction, showing that the performance of the voting stage is only influenced by the quality of the pairwise relationships learned by the model. Moreover, the voting stage can actually compensate errors in pairwise relationships. We observe that error tolerance is significantly high towards random

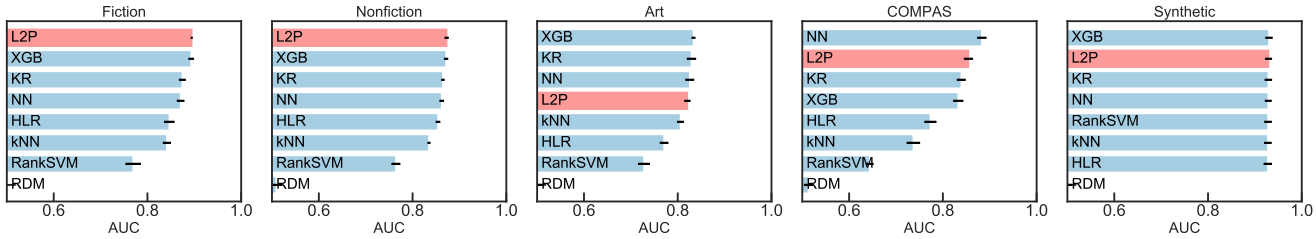

Figure 5: Performance comparison using AUC scores of different methods on different datasets. `L2P` achieves the highest performance rank on book datasets and obtains similar score on art dataset. It is the second highest method on COMPAS dataset. However, given kurtosis score in Table 1, COMPAS is less heavy-tailed than the previous three datasets. Synthetic dataset is a simple dataset where all methods achieves comparable performance. XGBoost (XGB) and neural networks (NN) provides acceptable accuracy as well.

error. That is, performance of `L2P` is stable until more than 45% of the pairwise relationships are mistaken. For distance-dependent mechanism to simulate errors, we observe robust performance for up to 30% error in Stage 1 predictions resulting just 20% reduction of the overall performance.

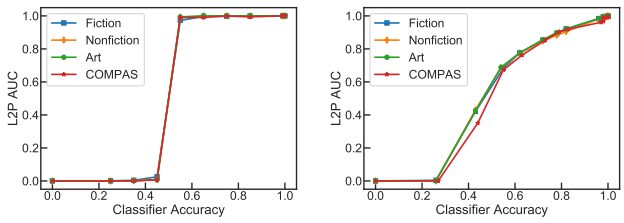

Figure 6: Robustness analysis for varying degree of errors introduced to the experiments by two mechanisms: random error mechanism (left) and distance-dependent error mechanism (right). The x-axis is the classifier accuracy and y-axis is the final AUC score. Across all the datasets we can see that `L2P` is highly robust to random error: performance is stable until more than 45% of the pairwise relationships are wrong. For distance-dependent error, we observe up to 30% error in the classifier leads to only 20% reduction of the overall performance.

3.4.3 Case studies. As mentioned earlier, one of the advantages of `L2P` methodology is its interpretability. Here, we demonstrate examples of model outcomes and how pairwise comparisons and certain features leads to more intuitive explanations than what other methodologies can provide.

First, let's look at a case where `L2P` performs better than others. We have the example of nonfiction book *Why not me?* by Mindy Kaling published by Crown Archetype. Our prediction is about 218,000 copies while the actual sales is about 230,000. The key features explaining the success of this particular book are the author's popularity (as measured by Wikipedia pageviews) and the previous sales of author – 6,228,182 pageviews and about 638,000 copies, respectively. However, performance of neural network leads to significant under-prediction as its prediction of 27,000 sales is an order of magnitude lower. While understanding the factors causing this significant under-prediction is not clear, `L2P` can provide context of its prediction. `L2P` places *Why not me?* between *Selp-Helf* by Miranda Sings and *Big Magic* by Elizabeth Gilbert. *Selp-Helf* has the author popularity as 1,390,000 and the author has no prior publishing history, while *Big Magic* has the author popularity as 1,596,000 and previous sales as 6,954,000. We see our query instance *Why not me?* has higher author popularity than *Big Magic* and *Selp-Helf*, but since it has a lower publishing history than *Big Magic*, `L2P` places it between these two books.

We also want to demonstrate an example case where `L2P` fails to achieve accurate prediction. The nonfiction book *The Best Loved Poems of Jacqueline Kennedy Onasis* by Caroline Kennedy under *Grand Central Publishing*, with claimed publication year 2015 in Bookscan, is predicted to sell 53,000 copies while the actual sales is 180 copies in the dataset. After an extensive analysis, it turns out that the book was initially published in 2001 and was a New York Times bestseller, which `L2P` captures its potential and predict high sales. Therefore this incorrect prediction is rooted in data error and our overprediction can be attributed to the initial editions performance as being a best seller. Neural network predicts 6,800 copies, though closer to the actual sale 180.

4 RELATED WORK AND DISCUSSION

Related research can be divided into three categories: (1) Learning to rank methodologies which rank list of instances into ordinal outcomes, (2) regression for rare events with heavy-tailed outcomes, and (3) methods addressing insufficient training data.

Learning to rank methodologies. Although `L2P` is designed for predicting heavy-tailed outcomes, methodological contributions show some parallels with the existing ranking algorithms. Cohen *et al.* [10] proposed a two-phase approach that learns from preference judgments and subsequently combines multiple judgments to learn a ranked list of instances. Similarly, RankSVM [20] is a two-phase approach that translates learning weights for ranking functions into SVM classification. Both of these approaches have complexity $O(n^2)$, which is computationally expensive.

In real-world applications like search engines and recommendation systems, systems provide ranked lists tailored to users and their queries [1, 5, 20]. In some cases, mapping those preferences into an ordinal variable leads to better user experience. Such tasks require the use of regression and multi-class classification methods [18].

Heavy-tail regression. Regression problems are known to suffer from under-predicting rare instances [22]. Approaches proposed to correct fitting models consider prior correction that introduces terms capturing a fraction of rare events in the observations and weighting the data to compensate for differences [28, 32]. Hsu and Sabato [19] proposed a methodology for linear regression with possibly heavy-tailed responses. They split data into multiple pieces, repeat the estimation process several times, and select the estimators based on their performance. They analytically prove that their method can perform reasonably well on heavy-tailed datasets. Quantile regression related approaches are proposed as well. Wang *et al.* [38] proposed estimating the intermediate conditional quantiles using conventional quantile regression and extrapolating these estimates to capture the behavior at the tail of the distribution. Robust Regression for Asymmetric Tails (RRAT) [35] was proposed to address the problem of asymmetric noise distribution by using conditional quantile estimators. Zhang and Zhou [42] considered linear regression with heavy-tail distributions and showed that using $l1$ loss with truncated minimization can have advantages over $l2$ loss. Like all truncated based approaches, their method requires prior knowledge of distributional properties. None of these regression techniques can capture non-linear decision boundaries.

Imbalance Learning. Data imbalance, as a common issue in machine learning, has been widely studied, especially in classification space. In [4], the problem of imbalance learning is defined as instances have different importance value based on user preference. There are in generally three categories of methods tackling this problem: data pre-processing[6, 16], special-purpose learning methods [29, 36] and prediction post-processing [3, 34]. However, one should notice that learning heavy-tailed distributed attributes is different from imbalance learning: in most imbalance learning, there is a majority group and a minority group, but within group items are mostly homogeneous; however in heavy-tailed distribution, there is no clear cut to define majority/minority group and even if forcing a threshold to form majority/minority group, within each group, the distribution is still heavy-tailed. Additionally, one need to choose a pre-defined relevance function for a lot of methods designed in this space.

Efficient algorithm for pairwise learning. In L2P, pairwise learning approach enhances model performance by constructing quadratically more training instances and presenting rare instance more frequently in comparison with all other instances. Such learning task leads to $O(n^2)$ complexity. However, in real practice, n^2 comparison is not necessary, and one can improve the scalability of the approach by reducing the number of pairs to be trained on to get comparable performance. We tested a naive approach based on the intuition that pairs which are further away are easier to be learned. For this efficient algorithm, we have two parameters: n_s denoting the number of samples to compare with for each instance and k denoting the number of instances that we consider as neighbors to each instance. The efficient algorithm will take all neighboring instances and sample $n_s - k$ non-neighboring instances for comparison for each instance. The intuition behind this efficient algorithm is that it is easier for a classifier to judge the pairwise relationship between instances that are far apart than instances that are closer to each other. Our experiments with efficient implementation of L2P leads similar outcomes and by sacrificing small accuracy we can reduce the number of comparisons in the training phase to $n_s n << n^2$ pairs.

In literature, efficient methodologies were proposed to learn pairwise relations more efficiently than comparing all n^2 pairs exhaustively. Qian *et al.* proposed using two-step hashing framework to retrieve relevant instance and nominate pairs whose ranking is uncertain [31]. Similar approaches to efficiently search similar pairs and approximately learning pairwise distance are proposed in the literature for information retrieval and image search [7, 25, 27].

5 CONCLUSIONS

We presented L2P, our Learning to Place algorithm. L2P accurately estimates heavy-tailed outcome variables; and it is robust and interpretable. Through learning pairwise relationships, L2P preserves the heavy-tailed nature of the outcome variables and avoids under-prediction of rare instances. We observe the following:

(1) L2P yields the best performance in majority of cases as measured by ROC, Kolmogorov-Smirnov statistic, and Earth mover distance. L2P consistently ranked among the top 2 models on learning outcome distributions that are the closest to the target distributions. L2P has the highest predictive performance on the book dataset, whose target variable (book sales) is very heavy tailed.
(2) We demonstrate experiments on various datasets having heavy-tailed outcome distribution. We select datasets exhibiting various range of values and degrees of kurtosis and prediction tasks from different application domains. We notice that L2P outperforms significantly on datasets that are more heavy-tailed (e.g., sales of fiction and nonfiction books).
(3) L2P has robust performance against pairwise relationship errors. Under random error setting, L2P can tolerate up to 45% error in pairwise relationship prediction; and under distance-dependent error setting, L2P only has a accuracy drop of 20% with 30% of pairwise relationship error.
(4) L2P is an interpretable approach where one can investigate the reason behind each prediction, and explore the placing of a test instance to obtain more context. This is highly important in markets such as book publishing and movie producing, where executives need reasons to make huge investments.

Future work. L2P's performance can be improved by slightly modifying its stage II. Currently, we are using the midpoint of the assigned bin as the predicted value for the test instance. This heuristic can introduce larger errors if the bin width is large. An alternative approach is to use a weighted average between neighboring bins for the test instance's predicted value.

Reproducibility

The Python implementation of `L2P` method is freely available at `https://github.com/xindi-dumbledore/L2P`.

REFERENCES

[1] Eugene Agichtein, Eric Brill, and Susan Dumais. Improving web search ranking by incorporating user behavior information. In *Proc. of the 29th Annual Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 19–26. ACM, 2006.

[2] Nir Ailon, Moses Charikar, and Alantha Newman. Aggregating inconsistent information: Ranking and clustering. *Journal of the ACM*, 55(5):23, 2008.

[3] Gaurav Bansal, Atish P Sinha, and Huimin Zhao. Tuning data mining methods for cost-sensitive regression: A study in loan charge-off forecasting. *Journal of Management Information Systems*, 25(3):315–336, 2008.

[4] Paula Branco, Luís Torgo, and Rita P Ribeiro. A survey of predictive modeling on imbalanced domains. *ACM Computing Surveys*, 49(2):31, 2016.

[5] David Carmel, Guy Halawi, Liane Lewin-Eytan, Yoelle Maarek, and Ariel Raviv. Rank by time or by relevance? revisiting email search. In *Proc. of the 24th ACM Int'l on Conf. on Information and Knowledge Management*, pages 283–292. ACM, 2015.

[6] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357, 2002.

[7] Gal Chechik, Uri Shalit, Varun Sharma, and Samy Bengio. An online algorithm for large scale image similarity learning. In *Advances in Neural Information Processing Systems*, pages 306–314, 2009.

[8] Chao Chen, Andy Liaw, and Leo Breiman. Using random forest to learn imbalanced data. Technical report, University of California, Berkeley, 2004.

[9] Tianqi Chen and Carlos Guestrin. XGBoost: A scalable tree boosting system. In *Proc. of the 22nd ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining*, pages 785–794. ACM, 2016.

[10] William W Cohen, Robert E Schapire, and Yoram Singer. Learning to order things. In *Advances in Neural Information Processing Systems*, pages 451–457, 1998.

[11] Caterina De Bacco, Daniel B Larremore, and Cristopher Moore. A physical model for efficient ranking in networks. *Science Advances*, 4(7):eaar8260, 2018.

[12] Julia Dressel and Hany Farid. The accuracy, fairness, and limits of predicting recidivism. *Science Advances*, 4(1):eaao5580, 2018.

[13] Seyda Ertekin, Jian Huang, Leon Bottou, and Lee Giles. Learning on the border: Active learning in imbalanced data classification. In *Proc. of the 6th ACM Conf. on Information and Knowledge Management*, pages 127–136. ACM, 2007.

[14] Samuel P Fraiberger, Roberta Sinatra, Magnus Resch, Christoph Riedl, and Albert-László Barabási. Quantifying reputation and success in art. *Science*, 362(6416):825–829, 2018.

[15] Jerome H Friedman. Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, pages 1189–1232, 2001.

[16] Haibo He, Yang Bai, Edwardo A Garcia, and Shutao Li. Adasyn: Adaptive synthetic sampling approach for imbalanced learning. In *Prof. of the IEEE Intl. Joint Conf. on Neural Networks*, pages 1322–1328. IEEE, 2008.

[17] Keith Henderson, Brian Gallagher, Lei Li, Leman Akoglu, Tina Eliassi-Rad, Hanghang Tong, and Christos Faloutsos. It's who you know: Graph mining using recursive structural features. In *Proc. of the 17th ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data mining*, pages 663–671. ACM, 2011.

[18] Ralf Herbrich, Thore Graepel, and Klaus Obermayer. Support vector learning for ordinal regression. In *Proc. of the 9th International Conference on Artificial Neural Networks*. IEEE, 1999.

[19] Daniel Hsu and Sivan Sabato. Heavy-tailed regression with a generalized median-of-means. In *Proc. of the Int'l Conf. on Machine Learning*, pages 37–45, 2014.

[20] Thorsten Joachims. Optimizing search engines using clickthrough data. In *Proc. of the 8th ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining*, pages 133–142. ACM, 2002.

[21] Myunghwan Kim and Jure Leskovec. Multiplicative attribute graph model of real-world networks. *Internet Mathematics*, 8(1-2):113–160, 2012.

[22] Gary King and Langche Zeng. Logistic regression in rare events data. *Political Analysis*, 9(2):137–163, 2001.

[23] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv:1412.6980v9*, 2014.

[24] Yufeng Kou, Chang-Tien Lu, Sirirat Sirwongwattana, and Yo-Ping Huang. Survey of fraud detection techniques. In *Proc. of the IEEE Int'l Conf. on Networking, Sensing and Control*, volume 2, pages 749–754. IEEE, 2004.

[25] Brian Kulis, Prateek Jain, and Kristen Grauman. Fast similarity search for learned metrics. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(12):2143–2157, 2009.

[26] Richard P Lippmann, David J Fried, Isaac Graf, Joshua W Haines, Kristopher R Kendall, David McClung, Dan Weber, Seth E Webster, Dan Wyschogrod, Robert K Cunningham, et al. Evaluating intrusion detection systems: The 1998 darpa off-line intrusion detection evaluation. In *Proc. of the DARPA Information Survivability Conference and Exposition*, volume 2, pages 12–26. IEEE, 2000.

[27] Haomiao Liu, Ruiping Wang, Shiguang Shan, and Xilin Chen. Deep supervised hashing for fast image retrieval. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, pages 2064–2072, 2016.

[28] Maher Maalouf, Dirar Homouz, and Theodore B Trafalis. Logistic regression in large rare events and imbalanced data: A performance comparison of prior correction and weighting methods. *Computational Intelligence*, 34(1):161–174, 2018.

[29] Marcus A Maloof. Learning when data sets are imbalanced and when costs are unequal and unknown. In *ICML Workshop on Learning from Imbalanced Data Sets II*, volume 2, pages 2–1, 2003.

[30] Henrique Pinto, Jussara M Almeida, and Marcos A Gonçalves. Using early view patterns to predict the popularity of youtube videos. In *Proc. of the 6th ACM Intl. Conf. on Web Search and Data Mining*, pages 365–374. ACM, 2013.

[31] Buyue Qian, Xiang Wang, Jun Wang, Hongfei Li, Nan Cao, Weifeng Zhi, and Ian Davidson. Fast pairwise query selection for large-scale active learning to rank. In *Proc. of the 13th Int'l Conf. on Data Mining*, pages 607–616. IEEE, 2013.

[32] Max Schubach, Matteo Re, Peter N Robinson, and Giorgio Valentini. Imbalance-aware machine learning for predicting rare and common disease-associated non-coding variants. *Scientific Reports*, 7(1):2959, 2017.

[33] D Sculley. Large scale learning to rank. In *NIPS Workshop on Advances in Ranking*, pages 58–63, 2009.

[34] Atish P Sinha and Jerrold H May. Evaluating and tuning predictive data mining models using receiver operating characteristic curves. *Journal of Management Information Systems*, 21(3):249–280, 2004.

[35] Ichiro Takeuchi, Yoshua Bengio, and Takafumi Kanamori. Robust regression with asymmetric heavy-tail noise distributions. *Neural Computation*, 14(10):2469–2496, 2002.

[36] Luis Torgo and Rita Ribeiro. Utility-based regression. In *European Conf. on Principles of Data Mining and Knowledge Discovery*, pages 597–604. Springer, 2007.

[37] Chih-Fong Tsai, Yu-Feng Hsu, Chia-Ying Lin, and Wei-Yang Lin. Intrusion detection by machine learning: A review. *Expert Systems with Applications*, 36(10):11994–12000, 2009.

[38] Huixia Judy Wang, Deyuan Li, and Xuming He. Estimation of high conditional quantiles for heavy-tailed distributions. *Journal of the American Statistical Association*, 107(500):1453–1464, 2012.

[39] Lilian Weng, Filippo Menczer, and Yong-Yeol Ahn. Virality prediction and community structure in social networks. *Scientific Reports*, 3:2522, 2013.

[40] Xiao Yu, Jin Liu, Zijiang Yang, Xiangyang Jia, Qi Ling, and Sizhe Ye. Learning from imbalanced data for predicting the number of software defects. In *Proc. of Intl. Conf. on Software Reliability Engineering*, pages 78–89. IEEE, 2017.

[41] Burcu Yucesoy, Xindi Wang, Junming Huang, and Albert-László Barabási. Success in books: A big data approach to bestsellers. *EPJ Data Science*, 7(1):7, 2018.

[42] Lijun Zhang and Zhi-Hua Zhou. l1 regression with heavy-tailed distributions. In *Advances in Neural Information Processing Systems*, pages 1076–1086, 2018.