A Novel Method for Predicting the End-Price of eBay Auctions

Stanford CS229, Fall 2013

David Nicholson
Department of Applied Physics
Stanford University
Stanford, CA 94305
Email: davidjn@stanford.edu

Rohan Paranjpe
Department of Electrical Engineering
Stanford University
Stanford, CA 94305
Email: rap2363@stanford.edu

***Abstract*—The eBay marketplace offers millions of items for sale each day. Understanding and being able to better predict the outcome of these auctions is important to buyers and sellers alike for individual profit maximization. In this paper, we explore the use of multinomial logistic regression (softmax), Naive Bayes (NB), and uniform prior Naive Bayes (UPNB) algorithms to predict both whether or not an item will sell and how much it will sell for. In predicting whether an item will sell or not, we find that the NB classifier performs with greater than 75% accuracy over a test data set. For final price prediction, we find that for multi-class binary prediction decision tree models, UPNB is able to outperform both the general NB and softmax algorithms. Unlike algorithms from previous works, our UPNB classifier uses the item title as the sole contributor to the feature vector. We provide a discussion on the results, as well as some insight about our particular data set and avenues for future exploration.**

I. Introduction and Motivation

eBay has been the leading online auction marketplace for nearly two decades. While auction final sale prices generally reflect the intrinsic item value [1], there exists variability in sale prices that may be attributed to factors such as item start price, seller ratings, or item visibility. In this paper, we aim to build a model which identifies and utilizes important item features to predict both whether or not an item will sell and how much it will sell for. Armed with this knowledge, sellers can optimize their listing strategies so as to maximize estimated profits, while buyers can more easily identify good and bad purchases.

Previous studies of final sale price predictions have used a large selection of features with classification and regression trees (CART) [2] or nearest neighbor clustering [3]. Another approach focused primarily on the text description, but also included additional weighting features in a regressive analysis [4].

The work presented in this paper extends upon these previous works as follows. Each of the three papers trained and tested on only items that sold; we aim to make this as a prediction. In our NB and UPNB schemes, we use only the item title text, while others used a mix of item features. Finally, to our knowledge, we are the first to use a classifier based solely on the item title text to make our predictions.

General:	
Data collection period	10/24/13-12/10/13
Category studied	Music - Records
Total completed auctions, pre/post filtering	12373 / 8694
Total sold auctions, pre/post filtering	3043 / 2066
Percent sold	24.6%
Average / total bids	3.82 / 11643
Average start / end price	\$14.94 / \$24.33
Total sales volume	\$74,050
Words in title dictionary, all items / sold items	3957 / 1149
Approximate classification errors, ϵ_S:	
Naive Bayes, sold/unsold	25%
Softmax regression, final bin	66%
Multi-class Naive Bayes, final bin	44%
Multi-class Uniform prior NB, final bin	30%
Common words, classical music	
Indicative of selling	concerto, tchaikovsky, brahms, organ, schumann
Indicative of not selling	digital, tulip, ensemble, great, ed1

TABLE I: Data at a glance.

II. Data Collection

Auction data is accessed via eBay's application programming interface (API) [5] using the Python software development kit (SDK) [6]. After signing up for a developer account, API calls can be made in an automated fashion, allowing autonomous data scraping. Keyword searches are made using the `findItemsAdvanced` function, and specific item information is obtained using the `GetSingleItem` function. Data is returned in XML format, which is then parsed for insertion and storage in an SQL database.

We gathered as much information per item as possible. Features stored included: item number, item title, start time, end time, text description, detailed item specifications[1], seller feedback percentage and total score, shipping costs, return policy, and image presence. Furthermore, each item was queried regularly through its complete auction duration so that page views, current price, and bid counts could be tracked as a function of time.

[1]These vary by auction category, and are optional for sellers to include. For Music-Records, this includes genre, condition, and many other specifications.

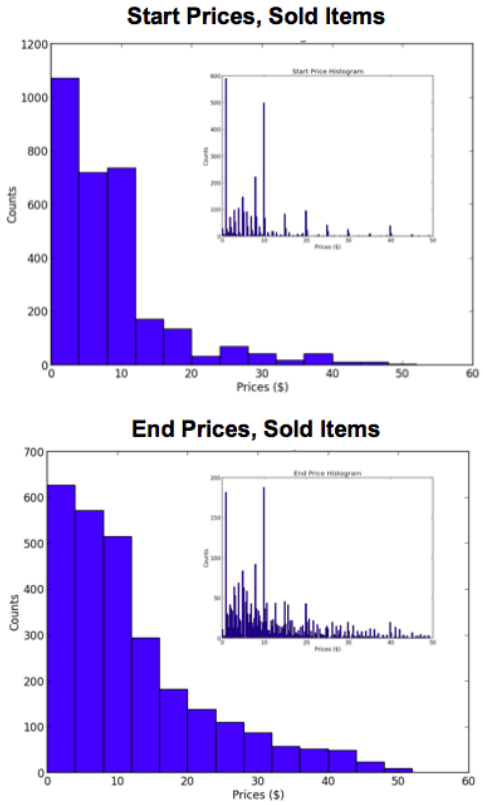

Fig. 1: Main: Histograms for the start and end price distributions of the items in our data set. Inset: The same histograms shown with smaller bin sizes. The peaks at common start and end prices show a non-smooth distribution, which may play a role in our classification errors.

In an effort to reduce the size and scope of our data, we restricted ourselves to a single category, Music - Records. Calls to the `findItemsAdvanced` function were made daily, with parameters set to a blank keyword search and to return the most recently listed items with each call. Likewise, we called the `GetSingleItem` function daily for each live auction in order to trace the temporal data.

Before analysis, the auction data is processed to remove potentially complicating outliers. For the work herein, the filter removes items where the complete text description (not just the title) contains any of the following words: skip, skips, damage, damaged, broken, warped, scuffs, and scratches. Previous work performed similar filtering to reduce their set by 25% [3]. This filtering reduces our data set by 30% and increases our UPNB prediction accuracy by 6-8% percent.

III. Modeling the Problem

Like other works [1-3], we discretize the end prices as a set, B, of k equally sized bins, each of size $\$b$:

$$B = \{0, b, 2b, \ldots, (k-1)b\}$$

where the jth element of B represents end prices in the interval $\$[(j-1)b, jb-.01]$. We model the prediction problem using k discretized end prices labeled by y, the n-dimensional feature vector x, and a training set S as follows:

$$\begin{aligned} S &= \{x^{(i)}, y^{(i)}\} \qquad \text{for } i = 1 \ldots m \\ x^{(i)} &\in \mathbb{R}^n, \qquad n = \text{number of features} \\ y^{(i)} &\in \{1 \ldots k\} \end{aligned}$$

For a given feature vector x, we will want to maximize a likelihood function over the bins:

$$\hat{y} = \arg\max_y p(y) \prod_{i=1}^{n} p(x_i|y) \tag{1}$$

The algorithm needs to identify the appropriate features for x as well as to model $p(x_i|y)$. With this framework, the sell/not sell prediction becomes a binary decision problem and the end price prediction becomes a multi-class classification problem.

A. Naive Bayes

In order to predict whether or not a given item would sell, we implemented a multinomial Naive Bayes classifier with the item title as the sole contributor to the feature vector. A complete word list $V = \{1, ..., |V|\}$ was created from all item titles in the complete set. For each new item on which to make a prediction, our feature vector $x^{(i)}$ is of length $|V|$, where the v^{th} entry represents the number of times that the v^{th} word appears in the title.

The maximum likelihood estimates, $\phi_{j|y=0,1}$, for each word are calculated from the training set data:

$$\begin{aligned} \phi_{j|y=0} &= p(x_j = 1|y = 0) \\ &= \frac{\sum_{i=1}^{m} \sum_{j=1}^{n_i} \mathcal{I}\left\{x_j^{(i)} = k \wedge y^{(i)} = 0\right\}}{\sum_{i=1}^{m} \mathcal{I}\left\{y^{(i)} = 0\right\} n_i} \\ \phi_{j|y=1} &= p(x_j = 1|y = 1) \\ &= \frac{\sum_{i=1}^{m} \sum_{j=1}^{n_i} \mathcal{I}\left\{x_j^{(i)} = k \wedge y^{(i)} = 1\right\}}{\sum_{i=1}^{m} \mathcal{I}\left\{y^{(i)} = 1\right\} n_i} \end{aligned} \tag{2}$$

with n_i as the number of words in the title of the i^{th} training example, $y^{(i)} = 0$ representing unsold items, and $y^{(i)} = 1$ representing sold items. With $p(y = 0)$ as the percent of items unsold in the training set and $p(y = 1)$ as the percent of items sold in the training set, a prediction is made on a new test item using equation (1).

Before performing this analysis, words that appeared only once amongst all titles were removed from the word list. Furthermore, words shorter than three characters, and the following common words, were removed from the list: and, for, from, that, there, these, this, and those. This analysis provided an error $\epsilon_S = 0.25$ when predicting on all genres.

B. Multinomial Logistic Regression

The first multi-class algorithm we attempted was a softmax regression. Our main assumption is that there exist k multivariate Gaussian distributions for each $x|y$, with equivalent covariance matrices after zeroing out the mean.

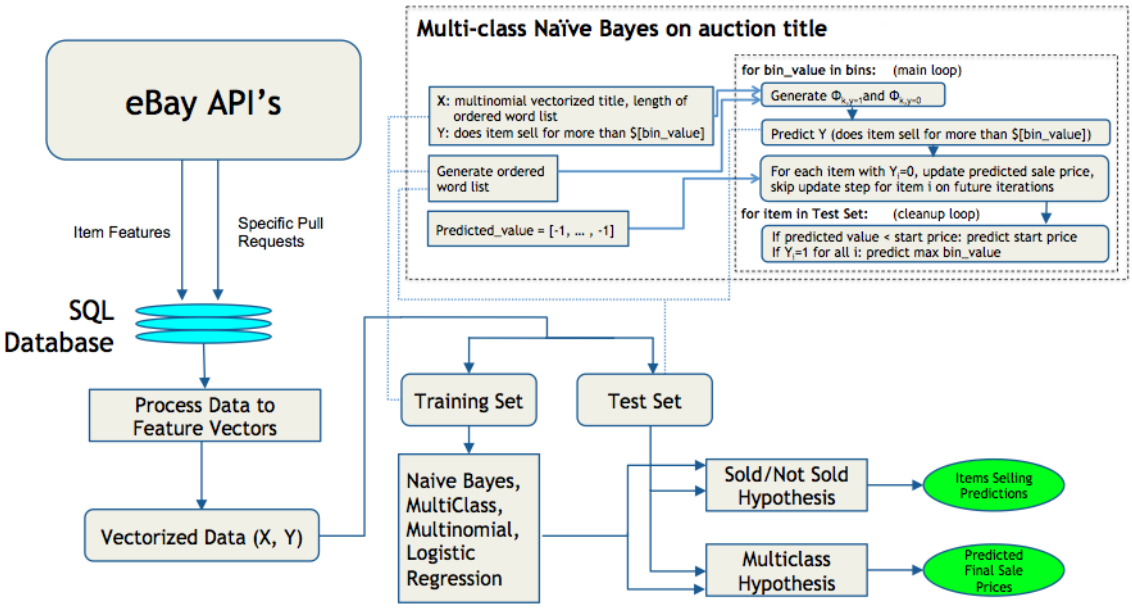

Fig. 2: Flowchart description of the algorithm

The feature vector for each item included the starting price, starting time, condition, seller feedback score, whether returns were accepted, shipping price, duration of the auction, genre (e.g. rock, country, etc.), and a constructed "page views" parameter. The page views parameter roughly corresponds to the average page views per hour during the auction listing. Our reasoning was that this gives some insight to the popularity of the item and may help in estimating its end price.

The softmax model and equations are as follows:

$$p(y=j|x;\Sigma,\phi_1,\ldots,\phi_k,\mu_1,\ldots,\mu_k) = \frac{e^{-\theta_j^T x}}{\sum_{i=1}^{k} e^{-\theta_i^T x}}$$

$$\theta_j = \begin{pmatrix} \frac{1}{2}\mu_j^T \Sigma^{-1}\mu_j + \log(\phi_j) \\ \Sigma^{-1}\mu_j \end{pmatrix}$$

where Σ is the zero-mean adjusted covariance matrix of the data, ϕ_k are the class priors for y, and μ_j is the averaged feature vector for items that end in bin j. We then argmax j over each $\theta_j^T x$ for each item.

This method worked decently, predicting with $\epsilon_s = 0.56$ for the items in our test set with $b = 5$ and a maximum price range of \$50. After implementing a feature selection process, a surprising result we found was that the starting price of the item dominated the regression more than we initially predicted. Excluding the starting price did not improve our results. Seeing this, we decided to look for new methods to make our predictions.

C. Multi-class Naive Bayes

In an effort to extend the title-based Naive Bayes algorithm, we developed a multi-class Naive Bayes scheme. The final prices are discretized into bins of size b dollars, and a series of title-based Naive Bayes classifications are made.

The first classifier asks, *does the item sell for more or less than \$0?* The second asks, *does the item sell for more or less than \$(b)?* The third, *does the item sell for more or less than \$(2b)?*, and so on. Each decision is made using a standard Naive Bayes classifier and requires a retraining of the data set. The first classification is identically the sold/unsold problem and the remaining classifications determine which final price bin, $\hat{y}$, that the algorithm should predict.

The practical implementation is illustrated in figure (2). For the first price at which an item is deemed not to sell, the algorithm predicts the preceding bucket as the final sale price. In the case where $\hat{y}_i < \text{binOf}(\texttt{START\_PRICE}_i)$, $\hat{y}$ is reassigned to $\hat{y} \equiv \text{binOf}(\texttt{START\_PRICE}_i)$. Those that were deemed to sell at greater than all bin values are assigned the maximum bin value.

When training on all genres and with the same bin values as the softmax regression, this algorithm performs with $\epsilon_s \approx 0.37$.

D. Multi-class, Uniform Prior Naive Bayes

Our final algorithm is derived from an analysis of the end price distributions. We were finding that weighting the probabilities, $p(x|y)$, by their class priors, $p(y=j)$, tended to skew the multi-class classifier predictions. We propose something new: given a multi-class binary decision tree process, we assume a uniform prior over all classes. This effectively states that the probability of a given item for selling above a price is equal to the probability of it selling below that price.

This algorithm predicts better than softmax and NB in general, reducing our classification error to $\epsilon_S = 0.28$ with \$5 bins and $\epsilon_S = 0.3$ with \$2 bins. This result is presented in

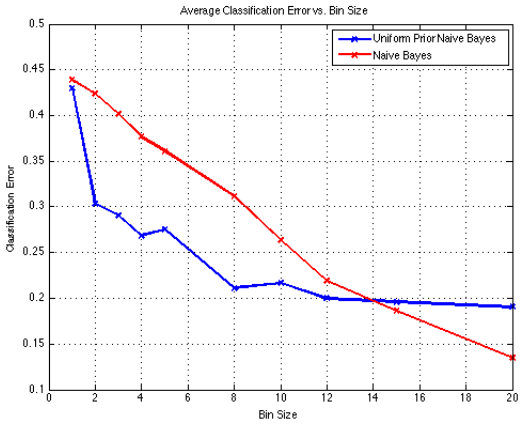

Fig. 3: Classification Error vs. Bin Sizes for the NB (red) and UPNB (blue) classification schemes. The spikes at $5 and $10 in classification error with UNBP are likely off-by-one misclassifications due to huge amounts of end prices distributed at places like $9.99, $14.99, etc. (see figure 1).

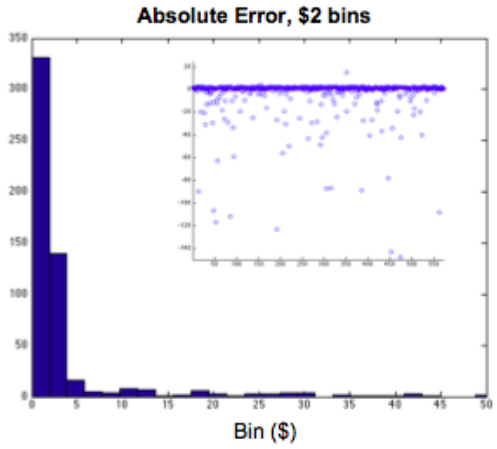

Fig. 4: Main: The Binned Error histogram for the UPNB scheme. Note the majority of misclassifications are off-by-one errors. Inset: Relative error of the raw data. Most of the misclassifcations made by UPNB overpredict the end price because the uniform class prior assumption does not skew the likelihood.

comparison with the pure Naive Bayes algorithm performance in figure (3). We hypothesize that this improvement could be attributed to our decision making process, and we are still exploring the full repercussions of assuming a uniform prior.

IV. Analysis

A. Errors and Results

The absolute error for each item, $\epsilon_{i,abs}$, and the classification error on the entire set, ϵ_S, are measured as:

$$\epsilon_{i,abs} = \left|\hat{y}^{(i)} - y^{(i)}\right|$$
$$\epsilon_S = \frac{1}{m}\sum_{i=1}^{m} \mathcal{I}\left\{\hat{y}^{(i)} \neq y^{(i)}\right\} \quad (3)$$

As expected, figure (3) shows that the classification error improves with increasing bin size. The absolute error, shown in figure (4), helps display where our prediction inaccuracies lie. The histogram's largest bin represents all items where $\hat{y}_i = y_i$. The second largest bin represents where our predictions were off by one bin. The inset shows the raw data; the solid line represents points where $\hat{y}_i = y_i$, points below the line represent where $\hat{y}_i > y_i$, and points above the line represent where $\hat{y}_i < y_i$.

We can see the large majority of our errors are *overpredicting* the final sale price, and that most of the inaccurate $\hat{y}_i$ predictions fall in the closest bin to what would have been the correct prediction. We hypothesize that this is a direct result of the uniform prior assumption. Still, there exists a substantial amount of outliers where $\hat{y}_i \gg y_i$. Dealing with these outliers will help improve the robustness of our predictions, as well as make this tool a safer bet for practical use.

Finally, some representative successes and mistakes in our predictions are shown in figure (5).

V. Conclusions

A. Remarks

One of the more interesting insights we were able to draw from this project was the importance of particular features over others. One of our biggest assumptions prior to the start of this project was the importance of the start price. In reality we found that the start price is not always a good indicator of the end price. This can be attributed to the item's intrinsic value or the flexible valuation of an item from user to user. These factors may be predominant in categories featuring collectibles.

We found that eliminating the start price from the classifier and focusing on the words in the item titles improved the accuracy. This came as a surprise initially, but after scrutinizing the data, we began to see trends associated with particular words that affect the final sale price. This became clearer as we narrowed the data set down to test on particular genres, and artist names or keywords in the title tended to boost or lower the probabillity of selling for some price. As an example, the words most indicative of selling and not selling for the classical music genre are shown in Table 1.

B. Further work

There are many directions to proceed in improving the performance of our algorithm. Keeping with the title-based Naive Bayes theme, we hope to extend the classifier to include bigrams and trigrams to capture particular phrases (e.g. "very rare") and complete artist names, which may help deal with the outlier problem discussed in figure (4).

Another direction we'd like to try is to use a high-dimensional kernel mapping and subsequent set of SVM's on our model. This approach can also help capture the bigram and trigram features without explicit representation, and may perform better than the pure NB algorithms.

The novelty of our work is that our NB classifiers use feature vectors based entirely on the titles of the items. As

Fig. 5: Left: Some remarkable end-price predictions made using UPNB. Right: Misclassifications due to the over-predicting nature of UPNB.

previous works have shown, there is wealth of information available in the rest of the features extracted for each auction. Future work will seek to identify the most relevant of these features and incorporate this information with our title-based predictions.

REFERENCES

[1] W. Vickrey, "Counterspeculation, auctions, and competitive sealed tenders", *The Journal of Finance, 16(1), 8-37*, 1961.

[2] R. Ghani and H. Simmons, "Predicting the end-price of online auctions", *Decision Support Systems 44 (2008) 970-982*, 2004.

[3] I. Raykhel and D. Ventura, "Real-time automatic price prediction for eBay online trading", *Association for the Advancement of Artificial Intelligence*, 2009.

[4] D. van Heijst, R. Potharst, & M. van Wezel, "A support system for predicting eBay end prices", *Decision Support Systems 44*, 2008, 970-982.

[5] `http://developer.ebay.com/common/api/`

[6] `https://github.com/timotheus/ebaysdk-python/`