Predicting Median Income from Yelp Review Language

Stephanie Chen schen751 | Michael Zhu mzhu25

CS 229
Stanford University
Autumn 2016

Introduction

We use supervised learning methods to predict the median income of a given zip code based on the text of Yelp reviews for local businesses. We were initially motivated by curiosity in metrics for gentrification in urban areas. We input Yelp reviews and tips for a given business and output one of four median income ranges: (< \$40k, \$40k–\$55k, \$55k–\$70k, >\$70k). We trained a stochastic gradient descent classifier for this task and achieved approximately 40% subset accuracy. Alongside this, we also attempted to predict the city where a business was located based on the same features, in order to see their usefulness to a related task.

We use text data from the Yelp Dataset, which includes 2.7 million reviews across 86,000 businesses in ten cities across North America and Europe; we only use the American cities (Pittsburgh, Charlotte, Urbana-Champaign, Phoenix, Las Vegas, and Madison), data for which comprise roughly 80% of the whole set. Each business has a set of textual reviews and tips and is labeled with an address, from which we obtain a zip code that we map to a median income bracket from the University of Michigan's 2006-2010 Median Household Income set.

Approach

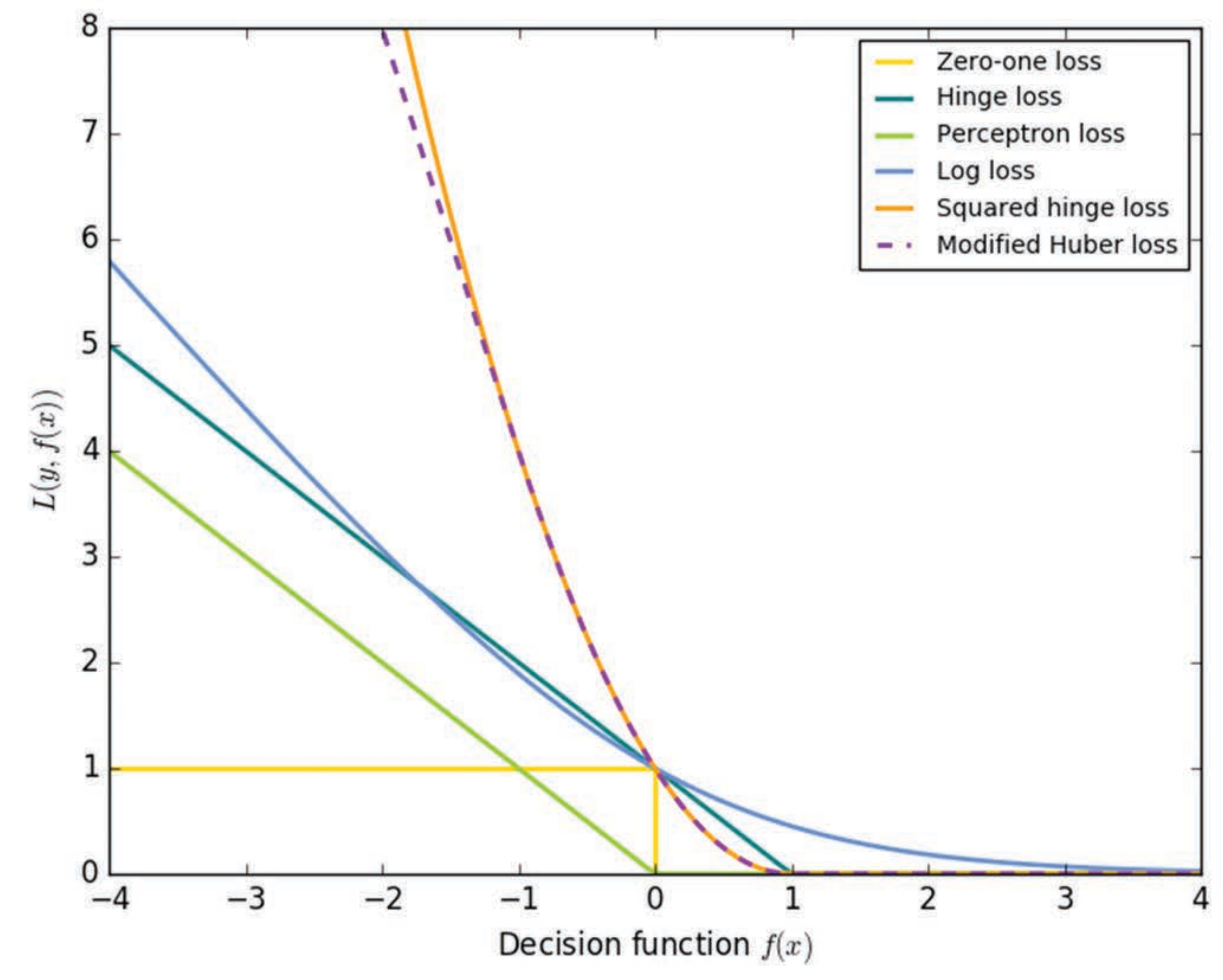
We remove identifying features like dollar costs and city and state names from the text and use placeholders for potentially useful punctuation like exclamation points and question marks, and we discretize incomes into four buckets that have relatively uniform support upon testing.

We use a simple unigram bag-of-words approach for our language features; experimentation with bigram and trigram features produced no improvement in results. After preprocessing, we vectorize the text using scikit's HashingVectorizer with 2^{18} features, which converts a text corpus into a sparse matrix representation of counts.

We use scikit-learn's stochastic gradient descent (SGDClassifier) model and perform a grid search over different loss (hinge, log, modified Huber, squared hinge, perceptron) and penalty (l1, l2, elasticnet with mixing parameter 0.15) functions. We find that modified Huber loss with l1 loss performed the best for both classification tasks. The scikit-learn SGDClassifier uses a one-vs-all (OVA) strategy for multiclass classification, which yields a probabilistic classifier when using the modified Huber loss.

$$L(y, f(x)) = \begin{cases} \max(0, 1 - y\,f(x))^2 & \text{for } y\,f(x) \geq -1, \\ -4y\,f(x) & \text{otherwise.} \end{cases}$$

Modified Huber Loss Function (for binary classification)[2]



Comparison of Loss Functions (from sklearn docs)

Results & Analysis

Median Income Prediction			
Loss	Penalty	Train accuracy	Test accuracy
Hinge	L1	0.3237	0.3286
	L2	0.3968	0.3931
Modified Huber	L1	0.3980	0.3879
	L2	0.3826	0.3855
City Prediction			
Loss	Penalty	Train accuracy	Test accuracy
Hinge	L1	0.5712	0.5661
	L2	0.5643	0.5712
Modified Huber	L1	0.6331	0.6378
	L2	0.6210	0.6186

Subset Accuracies by Model

Our results for income prediction were not as good as expected, even after optimizing our model parameters. This could be due to uninvestigated underlying patterns in our data, like Yelp usage frequency or business density across zip codes, or because of our feature selection process – including an entire review as simple unigrams might be too noisy with regards to the specific task of income prediction. Our results for city prediction were much better, though our accuracy for Illinois was consistently low, potentially due to sampling issues or underlying business patterns in the college town of Urbana-Champaign. However, even after removing identifying features like raw city names, we still predict cities with >60% accuracy.

scikit's HashingVectorizer doesn't preserve feature names, so we use TfidfVectorizer on a subset (we don't use it on the whole set due to memory usage) to get an idea of the most predictive word features. Though our performance in predicting the cities where businesses are located was significantly better than our performance in predicting the income brackets, it's interesting to note that the most significant predictive words in the income task seem more semantically related to income than those in the city task seem related to location.

Our dataset covered cities with similar median incomes and intra-city income distributions, making our model potentially susceptible to patterns as discussed above; with an expanded range of median incomes (ex. San Francisco) and income distributions (ex. New York), our model might learn more predictive textual features. We're also interested in unsupervised approaches to similar problems, for example using PCA with regards to the cities task to see if businesses could be clustered geographically by certain features.

Predictive Terms: Income	Predictive Terms: Cities
uptown	town
valley	casino
wine	smog
airport	school
hospital	here
members	valley
downtown	west
busy	patio
campus	downtown
mall	sun

Unsorted List of Predictive Terms

[1] B. Zadrozny and C. Elkan, "Transforming classifier scores into accurate multiclass probability estimates," Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 694–699, Jul. 2002. [2] T. Zhang, F. Damerau, and D. Johnson, "Text Chunking based on a Generalization of Winnow," Journal of Machine Learning Research, vol. 2, pp. 615–637, Mar. 2002.
[3] T. Zhang, "Solving large scale linear prediction problems using stochastic gradient descent algorithms," Proceedings of the twenty-first international conference on Machine learning, p. 116, Jul. 2004.