CUNY MSDS – DATA622

Homework 1

Introduction

For this assignment, I've chosen two datasets:

1. News Headlines by Category ("Large") – This dataset contains over 200,000 headlines for news articles, along with short descriptions of each article. The articles are all with HuffPost, published between 2012 and 2022. The target label is the news category (e.g. Business, Sports, Politics). Sourced from Kaggle.com [1].
2. NYC Property Sales by Price ("Small") – This data contains details ~4,800 housing sales in New York City between September 2016 and 2017. The target variable is sale price. Sourced from Kaggle.com [2].

The two datasets are primarily differentiated by two factors: size and type. The news headlines dataset is much larger, and the target variable is categorical, requiring a classification model. The NYC property dataset is smaller, and the target variable is continuous, requiring regression.

Code for all the following analysis is saved here: https://github.com/kac624/cuny/tree/main/D622.

Exploratory Data Analysis and Feature Selection

1. Large

The large dataset contains two text columns: the article headline and a short description of the article. The category target has 41 classes. There are two additional features: date and author. For the sake of this exercise, however, I wished to focus on textual analysis alone, so I've excluded those additional features.

I wished to use both the headline and description fields, so I examined the distribution of token lengths of each. In modeling, I would pass both as a sequence of tokens, so I also examined the number of tokens in the combined string. Most observations have less than 50 tokens, but there was a long tail to the right, with some observations having over 200 tokens. See Figure Large-1 for details. This indicated I would need to try different values for the max_sequence_length hyperparameter.

The data also exhibited significant imbalance across classes. In particular, the Politics category predominated. Moreover, many classes appeared poorly defined, with a high potential for overlap. For example, the data contained three nearly identical labels related to world news: WORLD NEWS, WORLDPOST, and THE WORLDPOST. Similarly, we had three nearly identical arts and culture labels: ARTS, ARTS & CULTURE, and CULTURE & ARTS'. See Figure Large-2 for details. To address these poor distinctions, I consolidated categories to 15 distinct, meaningfully different classes.

For this data, correlation across columns is not a major concern, since we really only have a single column: the combined headline and description.

2. Small

The small dataset contains a combination of numerical and categorical features. See Figure Small-1 for details on the distribution of each.

Unfortunately, the data contain a number of location-related features that appear to be of poor quality. Specifically, these features have inconsistent granularity. For example, the LOCALITY feature contains very high-level values such as “New York” or even just “United States”, along with very granular values associated with specific neighborhoods, such as “Flatbush” and “Ditmas Park”. Moreover, many of these location features were largely redundant. Ultimately, I kept just two, both of which I engineered from other features. Specifically, I kept a ZIPCODE feature, and I calculated a DISTANCE feature, indicating the distance from midtown Manhattan (specifically the Empire State Building) based on each building’s latitude and longitude.

The data also contains property specific features. One of them is categorical, assigning each property to one of 13 home types (e.g. Condo, Multi-family, etc.). The others are numerical, related to the number of bedrooms, the number of bathrooms, and the square footage of the property. I examined pair-wise correlation among the numerical features and found that the numbers of beds and baths were highly correlated (~0.78), as were both with the property square footage (~0.45). See Figure Small-2 for details. This indicated I would need to examine regularization techniques, as well as modeling approaches that are robust to multicollinearity.

Finally, careful examination of a sample of observations indicated some potential data quality issues in the target variable. In particular, the dataset contained a number of observations with surprising low prices (i.e. less than $100,000). Moreover, the target variable exhibited severe skew.

Modeling

The primary difference in modeling frameworks across the two datasets was related to the target variable. The Large dataset’s categorical target required classification models, whereas the Small dataset’s continues numerical variable required regression models. Additionally, the size of and nature of data drove many differences in the approach. Because the Large dataset had so many observations, I was able to employ relatively more complex models. Additionally, the fact that I was dealing with text sequences allowed me to take advantage of pre-trained, foundational language models and employ fine-tuning. The more bespoke nature of the Small dataset precluded the use of foundational models. Moreover, the limited number of observations placed a greater emphasis on regularization to maintain generalizability to whatever degree I could. Finally, the difference in size across datasets necessitated different proportion of training / testing splits. For the Large dataset, I used 70% of the data for training, 15% for validation and 15% for final test evaluation. For the Small dataset, I wanted to ensure I had a reasonable number of observations for evaluation, so I used 60%, 20% and 20%, respectively.

1. Large

I examined two commonly used classification models to establish a baseline: Logistic Regression and XGBoost. For both, I vectorized the text sequences using the Term Frequency – Inverse Document Frequency (TF-IDF) metric. They were able to achieve Accuracy in the validation subset of 64.7% and 59.63%, respectively.

The primary model I wished to focus on was the Bidirectional Encoder Representations from Transformers (BERT) pre-trained large language model. Specifically, I used the BertForSequenceClassification setup from the Huggin Face library available via the transformers

package. Without any adjustments and some fine-tuning on the news dataset, BERT provided decent results, with Accuracy and F-1 scores in the validation set of 69.4% and 60.7%, respectively. However, I employed a number of techniques to try and improve performance.

- Data augmentation: For a subset of samples from minority classes on which the model performed poorly, I created synthetic observations using backtranslation, random deletion and random substitution of tokens.
- Weighted random sampling: To address the heavy imbalance in the data, I employed weighted random sampling to create training datasets that better represented minority classes.
- Hyperparameter tuning: I trained the model on various permutations of hyperparameters to identify those combinations that performed best on the validation subset. Specifically, I aimed to optimize the max token length used by the BertTokenizer, the mini-batch size, the learning rate, the dropout factor (for regularization), and the balance factor used in the weighted random sampling (mentioned above).

With these enhancements, I increased Accuracy and F-1 scores in the validation set to 77.6% and 79.1%, respectively. See Figure Large-3 for a summary of performance across the baseline and final models.

2. Small

I looked at a number of regression techniques, with a focus on ensemble methods. Given the relatively limited data, I wanted to maintain as much generalizability as possible, which ensemble methods can support. In particular, I considered the following approaches:

- Ridge Regression
- Random Forest
- eXtreme Gradient Boosted trees (XGBoost)
- Adaptive Boosting (ADA)
- A Stacked Ensemble regression using all of the above models

I scaled all numerical features between 0 and 1 and one-hot encoded all categorical features. I also decided to scale the target variable, given its heavy skew. Ultimately, I found that this scaling affected different algorithms differently. Specifically, it improved performance for tree-based models, but impacted ridge regression and stacking negatively.

Then, I used the default hyperparameters in sklearn for each model to first establish a baseline. Performance was pretty weak from the start. The best models learned the data well in-sample (~20% MAPE for random forest), but in the validation data performance was much worse (~69% for random again). Some models performed really poorly, namely ridge, ADA and stacking, which all had MAPE >100%.

Next, I ran through a hyperparameter tuning protocol to find the optimal values for most hyperparameters for each model (using validation MAPE as the target). Ultimately, however, performance left much to be desired. The best models were random forest and XGBoost, which both

had ~50% MAPE in the validation data. In my ultimate evaluation against the test holdout, performance was actually better at 44% and 41%, respectively. See Figure Small-3 for full results.

I suspect the weak performance relates to (1) the quality of the data, and (2) some key missing features. A time component may have helped identify seasonal factors. More property-specific details (e.g. amenities) may have also helped. Finally, the best models still exhibit some overfitting (performance on training data is much better than performance on the holdouts). So, I might explore more severe regularization, or decreasing the proportion of data used for training to enhance the models' generalizability.

Takeaways

Given the poor performance on evaluation subsets for the Small regression model, I would *not* be comfortable using that model in a production setting. Further refining the models (and likely getting some better data) would be required. The news classifier, however, has pretty solid performance that I do believe could provide some value in a real-world setting. At ~80% accuracy, however, I suspect that it may require human oversight for final decisioning.

Figures

Figure Large-1: Distribution of Tokens Length

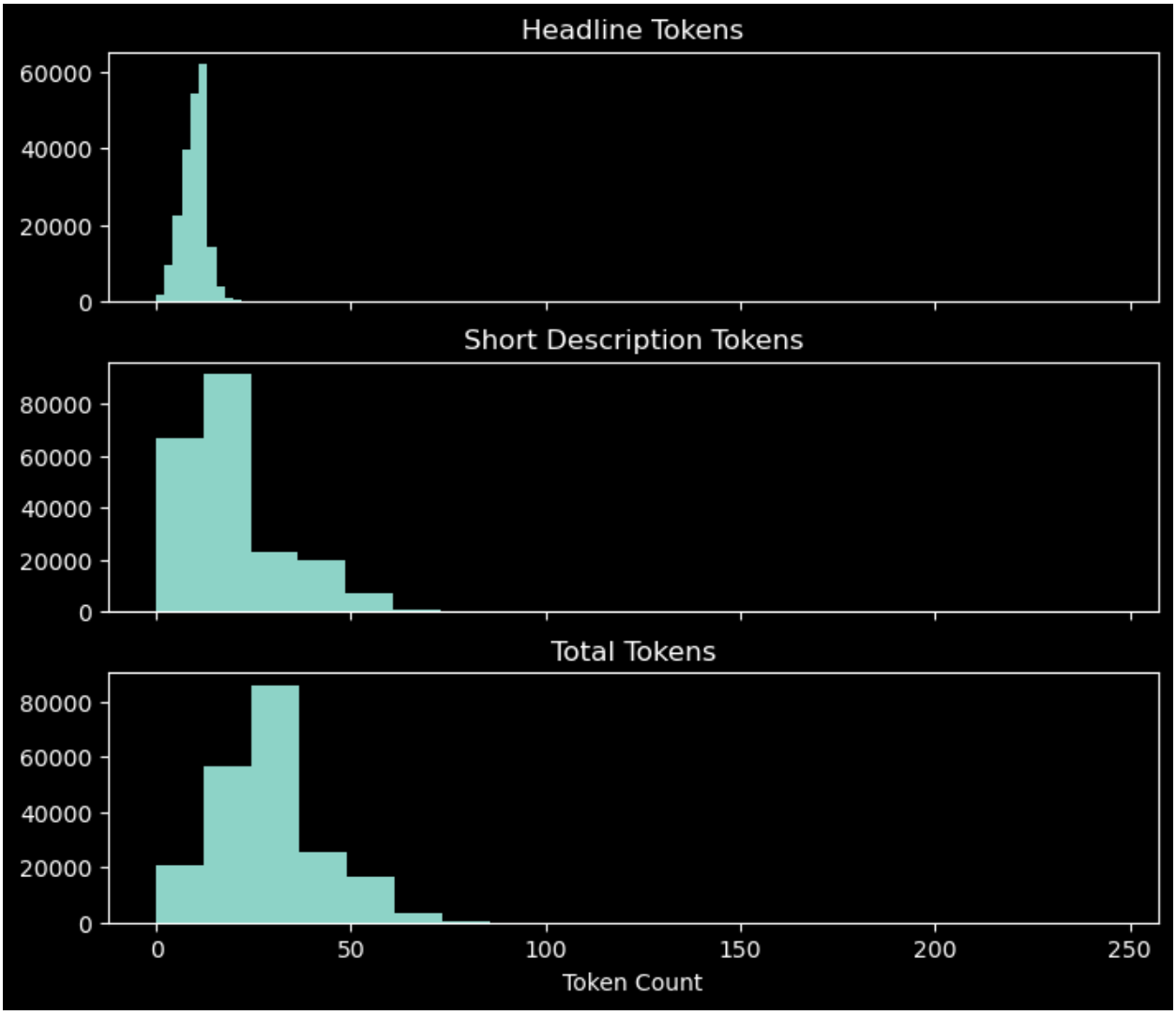

Figure Large-2: Distribution of Classes

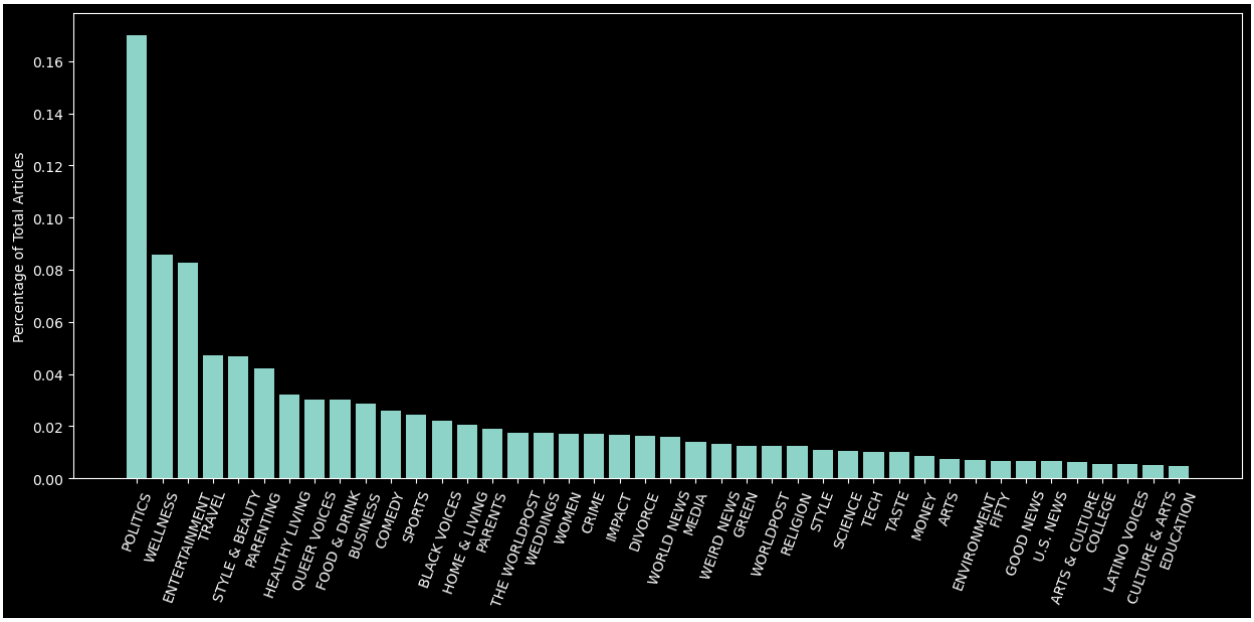

Figure Large-3: Evaluation Results

		BASELINE - Logistic Regression	BASELINE - XGBoost	BASELINE - BERT	FINAL - BERT Optimized
Training	Accuracy	71.1%	68.8%	80.8%	87.5%
	Precision	--	--	75.4%	89.4%
	Recall	--	--	77.7%	89.2%
	F-1 Score	70.8%	43.0%	76.4%	89.3%
Validation	Accuracy	64.7%	59.6%	69.4%	77.6%
	Precision	--	--	60.9%	81.4%
	Recall	--	--	61.3%	77.4%
	F-1 Score	65.0%	60.4%	60.7%	79.1%
Testing	Accuracy	--	--	--	77.0%
	Precision	--	--	--	81.0%
	Recall	--	--	--	77.1%
	F-1 Score	--	--	--	78.8%

Figure Small-1: Distribution of Features

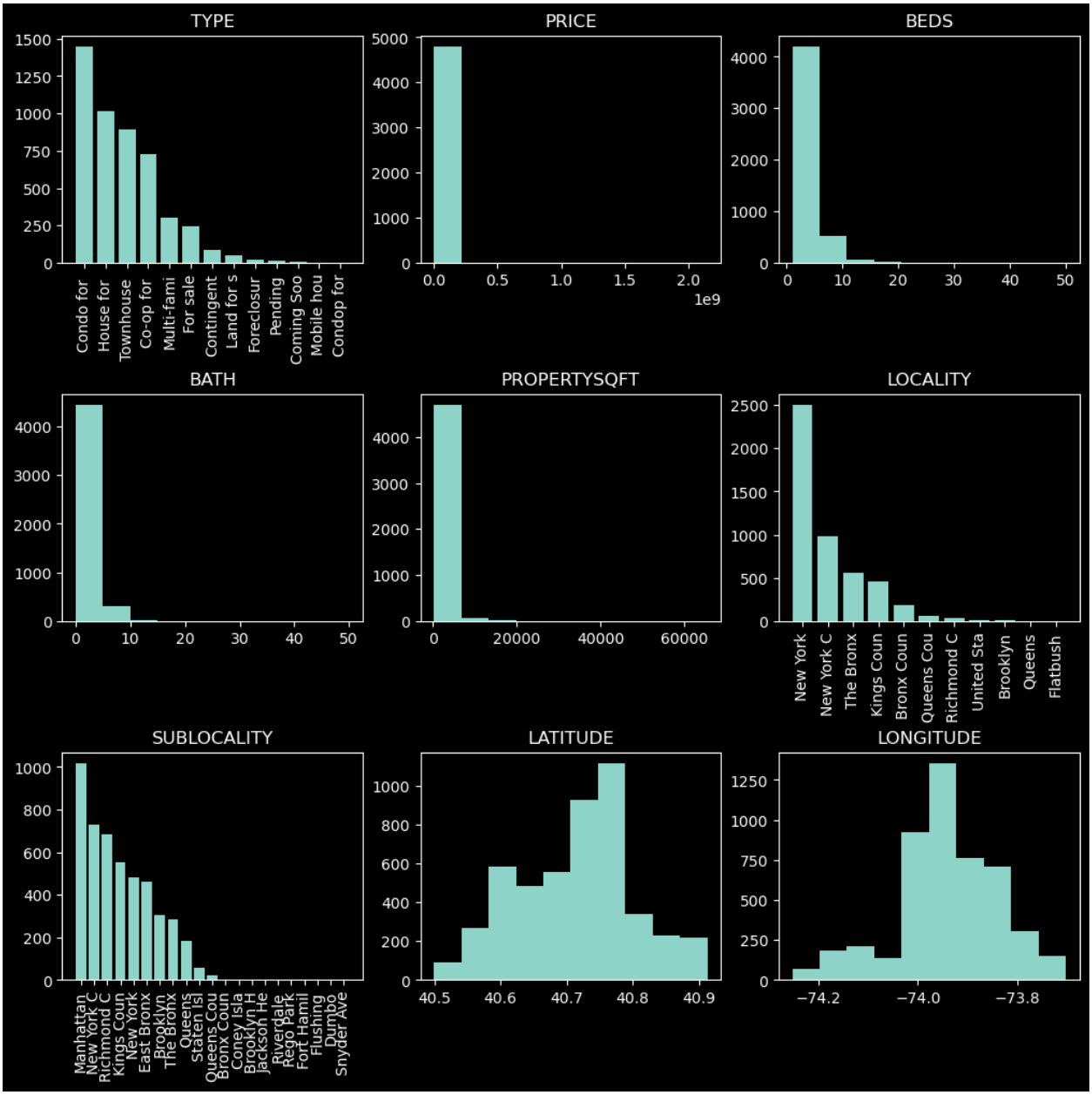

Figure Small-2: Assessment of Multicollinearity

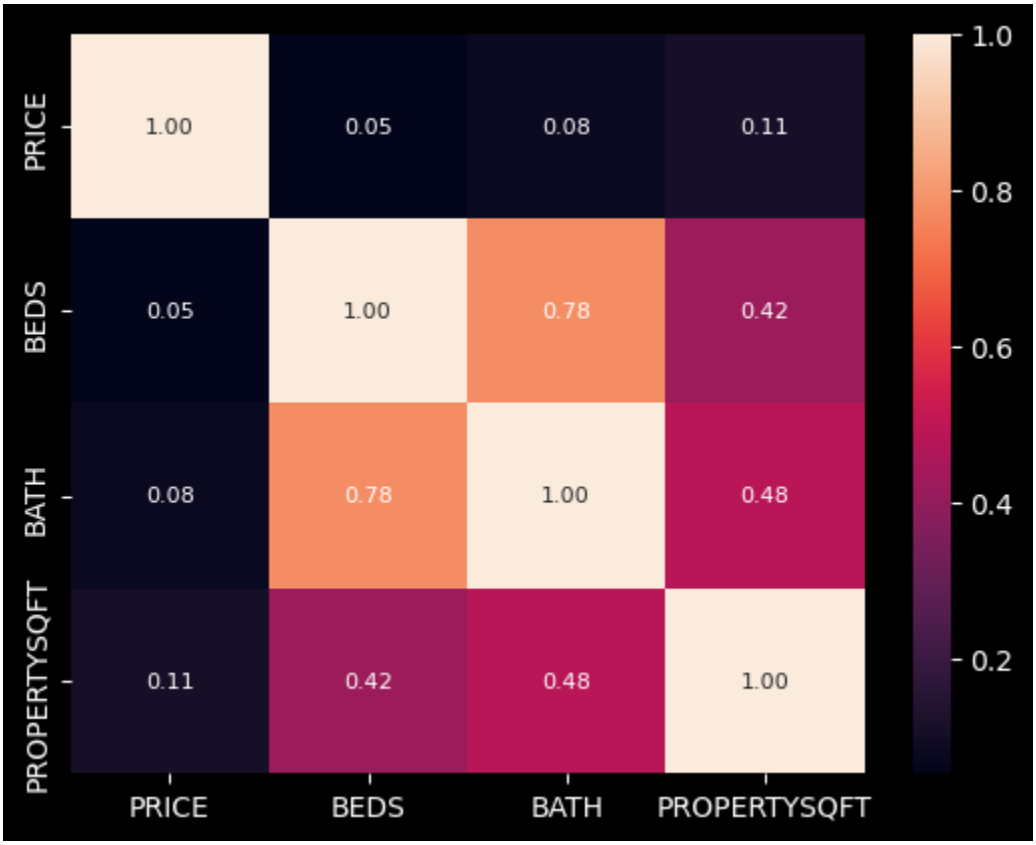

Figure Small-3: Evaluation Results

		MAPE		
		Training	Validation	Testing
Baseline	Ridge	360.7%	340.5%	--
	Random Forest	20.1%	68.6%	--
	XGBoost	30.8%	51.1%	--
	ADA	413.7%	376.3%	--
	Stacking	321.4%	291.5%	--
Optimized	Ridge	286.0%	232.8%	264.2%
	Random Forest	18.7%	51.2%	43.6%
	XGBoost	5.9%	49.4%	41.4%
	ADA	125.9%	121.0%	107.7%
	Stacking	225.9%	219.0%	198.6%

Sources

[1] Misra, Rishabh. "News Category Dataset." arXiv preprint arXiv:2209.11429 (2022). Sourced from https://www.kaggle.com/datasets/rmisra/news-category-dataset.

[2] Bilogur, Aleksey. “NYC Property Sales”. Sourced from https://www.kaggle.com/datasets/new-york-city/nyc-property-sales.