Our team developed an algorithm that predicts product prices based on various attributes such as description, condition, category, and brand. This allows sellers to save time and resources when determining the value of their products.

**Inputs**
- Bag of Words (BoW): Binary vector of length \(|V|\) indicating presence or absence of each word in the vocabulary (Multi-hot vector).
- Word2Vec (W2V): 100-dimensional GloVe vector or vector trained on our corpus to capture meaning of word.
- Item condition: [scale of 1-5] One-Hot vector.
- Brand: (5K different brands) One-Hot vector.
- Category: Multi-hot vector.

**Outputs**
- SoftMax Output: Predicted price.
- LSTM: Output exact price using Root Mean Squared Logarithmic Error cost function.
- \(\text{SoftMax Output} (12 \text{ buckets})\)
- \(\text{Only item description input}\)
- \(\text{Item descriptions only as input}\)
- \(\text{Local categories only as input}\)
- \(\text{Multiple FC layers}\)
- \(\text{Linear Output}\)

**Models**
- **Neural Network** (2 hidden layers):
  - SoftMax Output (12 buckets)
  - Only item description input
  - Bag of Words
  - Word2Vec (using pre-trained GloVe vector & averaging vectors of words in description)
- **Recurrent Neural Network with LSTM**:
  - SoftMax Output (12 buckets)
  - Only item description input
  - Word2Vec (trained on our corpus)
  - Word2Vec (using pre-trained GloVe vector)
  - Word2Vec (trained on our corpus)
  - Item descriptions only as input
  - All categories as input
  - Only item description input
- **Best Models**
  - LSTM Network with multiple FC layers, 12 Buckets Output
  - LSTM Network with multiple FC layers, Descriptions Only, 12 Buckets Output

**Challenges**
- Had to train the word2vec vectors for our application.
- Bag of Words took too long to run so we couldn’t do very many epochs.
- Bi-Directional LSTM took too long to train so we just did normal LSTM.
- Had to pivot from storing all of our sentences encodings in CSVs to doing the encoding step at each mini-batch due to memory complexity.

**Discussion**
- **Bag of Words vs. W2V**
  - Bag of words at first seemed to outperform Word2Vec.
  - Standard NN that used BoW achieved same accuracy on dev set.
- **RNN (LSTM)** improved accuracy on train set by 10% on training set and 5% on dev set.
- Final architecture used an LSTM architecture followed by a fully connected multi-layer perceptron network.
- Though accuracy seems low, we believe that this is because this is an inherently hard problem and that the Bayes error is not much lower than our model’s.

**Future Work**
- Deeper architecture.
- Bi-directional LSTM instead of normal LSTM.
- Attention model instead of LSTM.
- More hyperparameter tuning.
- Characterize which combination of inputs add the most value in predicting prices.
- Find optimal bucketing strategy such that the buckets are not only representative of the market but also cover a more similar price range.
- Do more error analysis and penalize the most common errors more to speed up learning.
- Filter data used to train to account for outliers.

**Acknowledgements**
We want to thank all of the wonderful CS 230 TAs and specifically our mentor, Soroosh Hemmati! We’d also like to thank AWS for the free credits!