For this project, the majority of the features that I utilized were found in the literature related to this task. The primary article in the literature that I utilized was the paper *Re-Examining Machine Translation Metrics For Paraphrase Identification*. In this paper, they utilized a variety of metrics, such as BLEU1-BLEU4, NIST1-NIST5, TER, and METEOR. These metrics help determine the accuracy of the translation of a sentence, making them useful for paraphrase identification purposes. On top of this, I also utilized a metric known as Levenshtein, or edit distance. While I initially explored the use of sentence length as its base metric, I found it to be not that effective. Instead, the number of edits it takes to transform one sentence into another is a better indicator of sentence meaning, as two sentences with similar lengths but different words would have an equal distance but a dissimilar edit distance. Levenshtein distance was discussed as a potential metric for paraphrase identification in the paper *Low-Level Features For Paraphrase Identification.*

In terms of data preparation, I preprocessed the data in various ways. For example, I noticed some syntactical errors in the training data, which I corrected through preprocessing. Also, when feeding the data into the machine translation metrics, I noticed that eliminating the periods and commas from the tokenization of the sentences before they were fed into the machine translation algorithms improved the accuracy. While I briefly experimented with removing the stop words in a sentence, this does not notably improve the model's accuracy. However, the biggest change to my model would be duplicating the entries in the model that had a ground truth value of zero three times before the data was processed. This made the training data have a distribution of approximately 50/50, which is the distribution of the development and test set, as these sets should ideally all come from the distribution. This preprocessing step improved my model from scoring in the high .60s to the mid .70s. In terms of feature preprocessing, I used a MinMaxScaler to assist in scaling my features before they were fed into the machine learning model. I also experimented with using a StandardScaler, but ultimately the MinMaxScaler led to slightly higher accuracy on dev.

In terms of the libraries I used, the two main ones I used were the sklearn library and the nltk library. The nltk library had many machine translation metrics, such as the BLEU, NIST, and METEOR scores. However, I did have to import an additional library, the pyter library for TER scores. I also had to import the Levenshtein library to use the distance function to calculate the edit distance. I also utilized the CSV library to assist in creating CSV files that contained the data that was used as input for my model. I used the pandas library to convert these CSV libraries into data frames and alter the training data to make it balanced. After this, I utilized the sklearn library to scale my data and train my data on various models and report the accuracy on dev for those models. I used a logistic regression model with l1 loss and C = 0.8, an SVM with a linear kernel function, and an 'rfbgf' kernel function based on the results on the dev set. The result was a majority voting classifier that utilized these 3 models, resulting in the highest development set score of 0.74.

In terms of my experiences in this project and the lessons that I learned from this project, I gained experience with crafting features for the first time. While I have had prior experience using machine learning algorithms to work with data, the datasets were always prefilled with features; I never had to generate the original features used in the data. This was an interesting challenge, and it led me to look at the literature for the types of features I may use. Utilizing the current literature on a topic to assist in building a model was another key lesson that I learned from this project. I have never looked towards academic papers or research to assist in my model construction, as I mainly relied on what I had previously learned in class. Learning how to gather the key information I needed in the literature and learning how to implement these findings worked to improve my model. Also, in this project, I gained experience utilizing majority-voting-based models. While I was aware of the concept before this project, I never actually implemented one until this project and I learned how to implement this type of ensemble model using popular libraries. In total, this project gave me experience with classifying features, reviewing the literature, and utilizing different types of models and I am excited to use this knowledge in future projects.

Works Cited

Bird, Steven, Edward Loper and Ewan Klein (2009), *Natural Language Processing with Python*. O'Reilly Media Inc. (NLTK library)

Madnani, Nitin, Joel Tetreault, and Martin Chodorow. "Re-examining machine translation metrics for paraphrase identification." *Proceedings of the 2012 conference of the north american chapter of the association for computational linguistics: Human language technologies*. 2012.

Scikit-learn: Machine Learning in Python, Pedregosa *et al.*, JMLR 12, pp. 2825-2830, 2011.

Pronoza, Ekaterina, and Elena Yagunova. "Low-level features for paraphrase identification." *Mexican International Conference on Artificial Intelligence*. Springer, Cham, 2015.