Alenna Zweiback
Project #1

Background

In this project, I worked as a cybersecurity defender interested in intercepting communication between two adversaries. These adversaries were using two types of classical ciphers, Ceasar and Playfair. The Ceasar Cipher had keys of 3,4, and 5. The Playfair Cipher used keywords CRYPTOFUN and SHERLOCK. I was able to compromise the adversaries' channel and collect some of their plaintext and ciphertext. My goal for this project was to build a classifier that would correctly determine which cipher was used to generate each ciphertext given a plaintext message.

Approach

To create the training data, I used AI for assistance. I had AI generate 500, unique, human-readable plaintexts. Each word was 9 characters long. I carefully validated each word that was given to me to make sure there was no duplicates or longer than 9 characters long. I adjusted wherever was needed but luckily did not run into any issues.

After creating the data training set full of plaintexts, I used the *pycipher* library to use to Ceasar and Playfair Cipher functions and encrypt the plaintexts. Since there were 5 classifications total, I decided to divide Ceasar with 300 for its three classifications and left 200 for Playfair's two classifications. I started with Ceasar first, where I took the first 300 words from the plaintext list and used a for loop to loop through the first 100 words using Ceasar with key =3, the next 100 using Ceasar with key=4, and the last 100 to use Ceasar with key=5. These were all then labeled with class = 0 at the end of the loop.

I moved onto the Playfair encryption function, where I also utilized a for loop that took the last 200 words in the list that had not been labeled yet. The first 100 of them were encrypted using the key CRYPTOFUN and then last 100 were encrypted using the key SHERLOCK. I ran into some issues because the Playfair cipher requires an even number of characters and cannot process any repeated letters in the same digraph. I resolved this by making my keys "CRYPTOFUNABDEGHIKLMQSUVWXYZ"and "SHERLOCKABDFGIMNPQTUVWXYZ." This was very helpful because it made sure that all characters used in the plaintexts were present in the cipher grid and compatible with the Playfair encryption rules. This avoided a lot of errors that I was receiving about missing certain values. At the end of the loop, they were given the class label of 1.

Once I completed these two encryption methods of the plaintext, I combined them to output a single training data CSV file which contained the plaintext, the newly generated ciphertext from my previous functions and the corresponding class label.

For Feature Engineering, I merged each plaintext and ciphertext into a single 18-character string. For example, plaintext of "drivegoal" and ciphertext of "GULYHJRDO" became "drivegoalGULYHJRDO." From there, I created a function that converted each character into an integer between 0 to 25. This resulted in 18 numerical features per row, and these became my predictor variables. Unexpectedly, I encountered that there were now 19 features with NaN values, as some of the Playfair had spilled over. Initially, I considered dropping that column.

However, I decided not to remove the column because when I did, it reduced my accuracy to 0.87. I kept the column, and it ended up improving my overall accuracy. To reduce any chance for error, I filled up the NaN values with a function that replaced any missing values with 0.

For training the model, I used Logistic Regression. I split the data to 80/20 into training (80) and testing (20) because it had the best accuracy results. My model achieved 100% accuracy on the test set which showed that it was able to perfectly distinguish and predict between Ceasar and Playfair ciphered messages based on the numerical features.

To ensure that my model can handle new test data from my professor, I included a dedicated section in the code that reads the CSV file with the plaintext and ciphertext columns. The data will then be processed using the same feature engineering steps as the training data and predicted against the trained model. I also added some additional code in case of any errors such as filling in any missing values with 0. Overall, this project was really helpful and interesting because it showed the entire process of creating the testing data, encrypting using classical models, feature engineering for machine learning, and then model training with prediction. It took us through the whole process which was exciting to see. I was able to create a classified that distinguished between Ceasar and Playfair ciphers with an accuracy of 100%.

Note: For this project, I uploaded my **full_plaintexts.csv** which is the list of the 500 plaintexts I used at the beginning. The output I used to train the model is called **full_train_data.csv**.