Q1: What data cleaning did you end up doing in Task 2 to create your features in task 3 and why?

In Task 2 I removed the following with reasoning:

1. **HTML URLs**: First I removed all html urls, both (https and http). This is because a URL cannot help determine authorship and was only making the data noisy.
2. **Hashtags:** Most tweets have hashtags that often do not give any information that could be used to identify an author.
3. **Mentions:** I removed mentions because they are unnecessary and do not add any value in identifying the author. Since my user was an Australian Prime Minister, there were alot of mentions that are not repeated and hence a pattern could not be determined.
4. **Punctuations:** I removed all types of punctuations because again they do not convey any significant information
5. **Numbers:** I also removed numbers because they do not help us identify who the tweet could've belonged to. They don't add any value to our collected vocabulary.
6. **Emojis:** Since this is not a sentiment analysis system we don't need emojis.

Even though Authorship attribution is mainly dependent on the stylistic attributes, content words can add value in determining the frequency of mentioning a specific word or type of content. For example, given that my profile was a prime minister, most of the tweets are similar in nature such that they either include meetups with other world leaders or are political in nature. The author can also have favorite words that we can note using the frequency and BOW approach.

Q2: In relation to the problem of authorship attribution, which classifiers do you feel would be most appropriate and why? Give a thorough comparison across all the classifiers you have studied in class.

For authorship attribution, SVM would be most effective. The comparison to the classifiers we have studied in class are:

1. **KNN Classifier:** KNN Classifier can work in Authorship attribution however, it would only be effective for a small dataset as stated by Fatma Howedi et al [1] and work well on limited training data. This however, can be a drawback for large datasets and the accuracy is often low. It also requires the data to be normalized or scaled so that one feature does not influence the decision significantly as studied in class. Also since KNN computes distances for every point at runtime, it is costly and not effective in real-life. Furthermore, the chance of overfitting is high.
2. **Logistic Regression:** Logistic regression model is a binary model that results in 2 class classification which is not effective in this problem. However, multinomial regression can be used for the softmax activation function which would give the probabilities of which

author the tweet could belong to. Logistic regression also works better on large datasets however, compared to SVM it is more vulnerable to overfitting. It also assumes linearity.

3. **SVM:** SVM has the ability to work even if data isn't linearly separable using kernels which is effective and is mostly used for text classifications since it works well with unstructured and semi-structured data as compared to the other classifiers. It is also able to process huge amounts of data.
4. **Perceptron**: Perceptron is a good option if the data is linearly separated which might not be the case in this project and hence would not be the most effective approach.
5. **Neural Networks/ MLP Classifier:** Neural Networks can automatically identify implicit features and do not require manual feature generation. These implicit features help in identifying the author. It also gives a good accuracy.

Q3: What is the ambient dimensionality of your solution and how would you determine the intrinsic dimensionality? Report both in your answer.

A dataset has high dimensionality if the number of features are more than the number of observations, which in this case is true as:

Ambient Dimensionality:

Features = Length of vocabulary (approx 3200)

Number of observations = Number of Tweets (approx 1000)

This can result in the curse of dimensionality since the size of the dimension is the size of the vocabulary. On top of that, the BoW representation will result in a lot of null values and sparse vectors. In this case, the **intrinsic dimensionality** can be determined by redundant dimensions in the vectorised BoW and performing reductions to find the minimum number of features or in this case words required such that the model works efficiently and computes faster. Since I suggested that the SVM Model should be used, it works well with high dimensionality data and automatically regularizes it to prevent overfitting.