APPROACH

Feature Engineering

1. We are given two datasets – *train-data-prepared.json* and *val-data-prepared.json*, with *id*, *text* and *label* data for each data point. These are loaded into Pandas dataframes.

2. As a part of the pre-processing step, the *text* is cleaned up by removing any embedded URLs, punctuations, leading and trailing spaces and lower casing the text to make the data points consistent and to avoid any unnecessary influence on the learning process.

3. The cleaned text then undergoes the feature extraction process. Since the task involves classifying a text as a *Claim* or a *Non-Claim*, we extracted features keeping in mind the properties that distinguish the two based on the literature that we went through. We make use of the SpaCy library in Python to extract the features. Following are the features that we extracted:

    a. **Modals**: Boolean feature set to 1 if the text contains modal verbs like "should" etc, that indicate the possible presence of an argumentative claim, and 0 otherwise.

    b. **Verbs**: Boolean feature set to 1 if the text contains verbs and 0 otherwise. Verbs such as "feel", "believe" etc indicate the presence of claim.

    c. **Personal Pronouns**: Boolean feature set to 1 if the text contains personal pronouns like "I", "You" etc which occur more frequently in claims, especially in dialogical argumentation and debates. It is set to 0 if the text does not contain any personal pronouns.

    d. **Third Person Verbs in Present Tense**: Boolean feature indicating the presence of third person verbs such as "bring", "are" "do" indicate presence of claims sometimes. Feature variable is set to 1 if such verbs are present and 0 otherwise.

    e. **Adverbs**: Boolean feature set to 1 if the text contains adverbs like "personally" etc, that indicate the possible presence of a claim. Feature variable is set to 0 otherwise.

    f. **Adjective**: Boolean feature set to 1 if the text contains adjectives and 0 otherwise. Adjectives such as "awesome", "horrible" etc that describe a noun, often indicate claims with stance in it.

    g. **Discourse Markers and Prepositions**: Boolean features indicating the presence of words such as "because", "although" etc which are often used in

claims. The feature variable is set to 1 if the data point contains discourse markers and prepositions.

h. **Unigrams and Bigrams:** We use the *CountVectorizer()* method in Scikit-Learn library to create a matrix of all possible unigrams and bigrams present in the entire corpus as features. The feature matrix contains the frequency of occurrence of these unigrams and bigrams for each datapoint (text input). This is a powerful feature that helps in understanding the combination in which tokens might occur, thereby getting a sense of context in which, a token is used.

4. The different features that are extracted are combined to form a single final feature matrix for training and validation data separately. **This final feature matrix of the training data is then sent to a classifier for training.**

Classification

1. The final feature matrix after going through pre-processing and feature extraction processes is fed to a classifier for training.

2. We used **GridSearchCV** to find the best hyper-parameters of whatever classifier we wanted to use.

   a. We first tested with Logistic Regression with different C values and penalties and got the best score with one *C-penalty* combination of hyper-parameters.

   b. In order to improve the score even further, we used **Support Vector Classification (SVC)** with different set of 'C' and 'gamma' values.

   c. From the GridSearchCV, we got **C=10.0 and gamma=0.01** as the best parameters.

3. After training our model using **SVC** with the best hyper-parameters with the *training data*, we then tested the classifier by generating predictions on the *validation data* and evaluated the performance of the model using **f1-metrics**.

Results

1. We finally created an output JSON file called *"prediction_out.json"* with predictions by extracting data point IDs from the dataset and generating a dictionary of **ID-prediction** pair.

2. With respect to the runtime of our approach, it has been recorded as 1-2 minutes for the feature extraction process and 4-5 minutes for training the classifier. Overall, it would take around 7 minutes to run and produce the predictions.

EXECUTION INSTRUCTIONS

Before proceeding with the execution of the code, it must be ensured that the necessary packages and libraries are present in the system. From the scope of this assignment, the following packages and libraries needs to be installed for successful execution of the assignment:
numpy, pandas, spacy, json, re, sklearn, sys

In case any of the above-mentioned packages are missing, it can be installed using the command **pip install <package-name>**.

In order to download the Spacy "small English language model" used in our code, execute the command **python -m spacy download en_core_web_sm**

Steps for execution:

1. Unzip the file Assignment2-HardlyHuman.zip.

2. The unzipped folder will have the following files:

    a. **ClaimClassification.py**: Python code for classifying data as claim or non-claim.

    b. **Documentation.pdf**: PDF file describing our approach and execution instructions

3. Navigate to the folder containing the ClaimClassifcation.py file. Run the python code using the following command on the command line/terminal:

    python ClaimClassification.py <train data file path> <val/test data file path>

    Ensure that data files are present in the same folder as the python code. If the data files are in a different location, make sure to mention the file path accordingly.

4. Executing the code, will generate an output file called **prediction_out.json** with test/validation dataset predictions.

5. To check the performance of the classifier, run the ***eval.py*** file from command line/terminal using the following command:

    python eval.py --true <path_to_val_data> --predictions <path_to_pred.out file>

    Running this command will output the **F1 score** of the classification on test/validation dataset.

Group: Hardly humans
Siddhanth Janadri – 6882502
Sneha Hiremath – 6881479
Sanjay Gupta – 6882964
Sai Nikhil Menon - 6882524