Bonus of Assignment 1

To Run

Please `cd A1_bonus` to go to the bonus directory and run `python3 a1_bonus.py -i1 preproc_bonus.json -i2 feats_bonus.npz -o bonus_output`

Modifications to each procedure

Preprocessing (`a1_preproc_bonus.py`)

- Added a dictionary of words that are abbreviations (e.g., "*i.e.*", "*Dr.*", "*No.*") and kept them together with the punctuations.
- Added a more thorough weblink removal procedure.

Feature Extraction (`a1_bonus.py` - `get_new_feats(feats_old, data)`)

- Added the controversiality and score, which were also categories in the original .json files. Also added the count of "Obama" and "Clinton" (not Trump yet).
- I evaluated the newly added features adn added them to `a1_bonus_features.txt` under `bonus_output`.
    - **p-value**: Controversiality got a really low (e-116) p-value, indicating that it's actually a very useful feature, while the p-values of the other 3 new features were just mediocre.
    - **Accuracy**: With the new features, we can see that the accuracy (with full data and all featrues) increased by 4% (from 39.6% to 43.6%). It also increased slightly when we use only the top 20 features, indicating that at least some of the added features were considered the top-20 most important, and that they had a positive impact on the classification.

Classification (`a_bonus.py` - `classifers(X_train, X_test, y_train, y_test, output_dir)`)

- Experimented with multiple other classifiers and different parameters on these as well as the original ones we used in the non-bonus part.

```
MODELS = {1: "SGDClassifier",
          2: "GaussianNB",
          3: "RandomForestClassifier",
          4: "MLPClassifier",
          5: "AdaBoostClassifier",
          6: "Linear SVC",
          7: "Gaussian SVC",
          8: "DecisionTreeClassifier"}

TESTS = {1: [0],
         2: [0],
         3: [1, 5, 10, 15],  # max depths in RandomForestClassifier
         4: [0.01, 0.05, 0.1, 0.5, 1],  # alphas in MLPClassifier
         # learning rate in AdaBoostClassifier
```

```
            5: [0.05, 0.1, 1.0, 1.5, 2.0],
            6: [0],
            7: [0],
            8: [None, 'sqrt', 'log2']}  # max_features in
DecisionTreeClassifier
```

- The results are recorded in `a1_bonus_classifiers.txt`, and I also formulated a table at the end of that file. In terms of top performers:
  - Top-feature based ones generally performs better:
    - AdaBoostClassifier with learning rate = 1 -- Accuracy = 0.43875
    - AdaBoostClassifier with learning rate = 1.5 -- Accuracy = 0.431875
    - MLPClassifier with alpha = 0.05 -- Accuracy = 0.426125
  - The highest in ones that use all features:
    - MLPClassifier with alpha = 0.01 -- Accuracy = 0.40225
  - Note that the top perfomers are actually all relying on only the top 20 features, perhaps because it reduces chances of overfitting (especially as I was using full data).