Siddarth Srinivasan

Professor Wenpeng Yin

CIS 4526

27 October 2022

Machine Learning Midterm

1. For my paraphrase identification machine learning model, I had created a total of five features: . Here is the breakdown of what features I used: ['wordcount difference', 'fuzz ratio', 'char difference', 'Overlapping ratio', 'synonyms']
    - ‘wordcount difference’

Wordcount difference is the absolute value of the difference of the number of words between sentence 1 and sentence 2.

- 'fuzz ratio'

    Fuzz ratio is a ratio that I had created by using the fuzzywuzzy package and using fuzz.ratio to find the similarity ratio of the two sentences via the levenshtein distance.

- ‘char difference’

    Char difference is the difference of the total number of characters from sentence 1 and sentence 2.

- ‘Overlapping ratio’

    To get the overlapping ratio, I got the total number of overlapping words between sentence 1 and sentence 2, and then divided this number by the number of words in the shorter sentence.

- 'synonyms'

  To get the number of synonyms between the two sentences, I used the nltk package and imported wordnet to do this. With this, I created a function called getSynonyms and found the number of synonyms between sentence 1 and sentence 2. After getting this number, I divided it by the sentence with the lesser number of words. This gave me the 'synonym ratio'. After testing this feature, It did not perform as well as the regular 'synonym' feature so I ended up removing it.

2. Data preprocessing and feature preprocessing
   - For data preprocessing, I did not do much to the data. Perhaps this could have been where my accuracy could have improved in retrospect. What I did was separate the data as specified with a '\t', and then dropped any null values in the datasets. I also set the 'gold label' column in both the training set and the development set to an integer. Here is a screenshot on what I had done:

```
columns = ['id', 'sentence 1', 'sentence 2', 'gold label']
training_data = '../input/mlmidterm/MLMidTerm-main/train_with_label.txt'
development_data = '../input/mlmidterm/MLMidTerm-main/dev_with_label.txt'
test_data = '../input/mlmidterm/MLMidTerm-main/test_without_label.txt'

df_test = pd.read_csv(test_data, sep = '\t', names = ['id','sentence 1', 'sentence 2'])
df_test = df_test.dropna().reset_index(drop = True)

df_dev = pd.read_csv(development_data, sep = '\t', names = columns)#.apply(lambda x: x.

df_dev['gold label'] = pd.to_numeric(df_dev['gold label'], errors='coerce')
df_dev = df_dev.dropna().reset_index(drop = True)
df_dev['gold label'] = df_dev['gold label'].astype(int)

df = pd.read_csv(training_data, sep = '\t', names = columns)#.apply(lambda x: x.astype(

df['gold label'] = pd.to_numeric(df['gold label'], errors='coerce')
df = df.dropna()
df['gold label'] = df['gold label'].astype(int)
```

3. Algorithms and Libraries used

- Numpy, Pandas, String, fuzzywuzzy (fuzz.ratio), difflib (SequenceMatcher), nltk (nltk.corpus, wordnet), matplotlib (pyplot), sklearn (svm, make_pipeline, StandardScaler, LogisticRegression)

4. Experiences and Lessons learned
    - Doing this machine learning project has taught me a lot about machine learning and artificial intelligence. The research part on what features I can use for this project allowed me to learn many things such as the numerous packages that were available to me, how to create a machine learning model using svm or logistic regression, and also has allowed me to understand what deep learning is. After completing this project, I am now comfortable building a machine learning model for any data given to me and hope to delve into more of this topic in the future.
5. Results
    - Here is the result of the Logistic Regression model with the above features mentioned on the development data and the accuracy:

```
from sklearn.linear_model import LogisticRegression
#features = ['wordcount difference', 'fuzz ratio', 'char difference', 'Overlapping ratio', 'synonyms', 'synonym ratio',]
features = ['wordcount difference', 'fuzz ratio', 'char difference', 'Overlapping ratio', 'synonyms']
X_train = df[features]
X_dev = df_dev[features]
y_train = df['gold label']
y_dev = df_dev['gold label']

lr = LogisticRegression(solver='lbfgs', max_iter=1000)
lr.fit(X_train, y_train)

y_pred = lr.predict(X_dev)

print(lr.score(X_dev,y_dev))

X_test = df_test[features]
y_test_pred = lr.predict(X_test)
```

```
0.6337625178826896
```

    - The accuracy as shown above for the Logistic Regression model is at: 0.6337625