Project

February 5, 2022

1 Update 5

- Please put a bulleted list of things you have accomplished since the last update
  - Include things that didn't work but you tried
  - Things you are planning on doing
  - Questions that you might have on your project.
- Reference the sections and figures you are dicussing here

2 Update 4

- Please put a bulleted list of things you have accomplished since the last update
  - Include things that didn't work but you tried
  - Things you are planning on doing
  - Questions that you might have on your project.
- Reference the sections and figures you are dicussing here

3 Update 3

- Please put a bulleted list of things you have accomplished since the last update
  - Include things that didn't work but you tried
  - Things you are planning on doing
  - Questions that you might have on your project.
- Reference the sections and figures you are dicussing here

4 Update 2

- Please put a bulleted list of things you have accomplished since the last update
  - Include things that didn't work but you tried
  - Things you are planning on doing
  - Questions that you might have on your project.
- Reference the sections and figures you are dicussing here

5 Update 1

- What dataset will be the focus of my research?

  - I have choosen a dataset that contains argumentative essays written by U.S students in grades 6-12. The essays were annotated by expert raters for elements commonly found in argumentative writing.
- What makes this project significant?
  - As students lack writing skills during their prime age of education, the aim is to aid tutors, teachers and students by providing feedback on the written essays and classifying them into categories that will conclude the style of writing which will help the students improve their skills.
- What is the question that my analysis seeks to answer?
  - Identify elements in student writing. More specifically, automatically segment texts and classify argumentative and rhetorical elements in essays written by 6th-12th grade students.
- What kind of EDA and modeling will I be doing?
  - The basic EDA techniques such as distribution of variables of the dataset. Lexical correlation will play an important role in modelling. I would be implementing RNN and LSTM algorithms to build the predictors on the dataset.
- What is the outcome of my analysis?
  - I expect to understand in depth the applications of natural language processing in education technology and how it impacts the human minds with it small contributions.

6 Excuetive Summary

- Summarize the key (This could be a bulleted list)
  - information about your data set
  - major data cleaning
  - findings from EDA
  - Model output
  - Overall conclusions

7 Abstract

The project seeks to predict the argumentative elements present in students writings for grades 6th-12th from the data set which contains different documents. By predicting the argumentative elements the documents will be broken down into sections as per the prediction class and final feedback will be provided.

8 Introduction

Writing is a critical skill for success. However, less than a third of high school seniors are proficient writers, according to the National Assessment of Educational Progress. Unfortunately, low-income, Black, and Hispanic students fare even worse, with less than 15 percent demonstrating writing proficiency. One way to help students improve their writing is via automated feedback tools, which evaluate student writing and provide personalized feedback. The majority of the available tools are proprietary, with algorithms and feature claims that cannot be independently backed up. More importantly, many of these writing tools are inaccessible to educators because of their cost. This problem is compounded for under-serviced schools which serve a disproportionate number of students of color and from low-income backgrounds. In short, the field of automated writing

feedback is ripe for innovation that could help democratize education. The dataset has been collected by Georgia State university which contains over 15000 documents which are annotated by expert human raters into different categories with their position in the essay. * train.csv - a .csv file containing the annotated version of all essays in the training set * id - ID code for essay response * discourse_id - ID code for discourse element * discourse_start - character position where discourse element begins in the essay response * discourse_end - character position where discourse element ends in the essay response * discourse_text - text of discourse element * discourse_type - classification of discourse element * discourse_type_num - enumerated class label of discourse element * predictionstring - the word indices of the training sample, as required for predictions

9 Data Science Methods

- To be applied (such as image processing, time-series analysis, spectral analysis etc
- Define critical capabilities and identify packages you will draw upon

10 Exploratory Data Analysis

10.1 Explanation of your data set

10.1.1 How many variables?

[1]:
```
import os
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from pprint import pprint
import pandas_profiling as pp
```

[3]:
```
train_df.head()
```

[3]:

	ID_code	target	var_0	var_1	var_2	var_3	var_4	var_5	var_6	\
0	train_0	0	8.9255	-6.7863	11.9081	5.0930	11.4607	-9.2834	5.1187	
1	train_1	0	11.5006	-4.1473	13.8588	5.3890	12.3622	7.0433	5.6208	
2	train_2	0	8.6093	-2.7457	12.0805	7.8928	10.5825	-9.0837	6.9427	
3	train_3	0	11.0604	-2.1518	8.9522	7.1957	12.5846	-1.8361	5.8428	
4	train_4	0	9.8369	-1.4834	12.8746	6.6375	12.2772	2.4486	5.9405	

	var_7	…	var_190	var_191	var_192	var_193	var_194	var_195	\
0	18.6266	…	4.4354	3.9642	3.1364	1.6910	18.5227	-2.3978	
1	16.5338	…	7.6421	7.7214	2.5837	10.9516	15.4305	2.0339	
2	14.6155	…	2.9057	9.7905	1.6704	1.6858	21.6042	3.1417	
3	14.9250	…	4.4666	4.7433	0.7178	1.4214	23.0347	-1.2706	
4	19.2514	…	-1.4905	9.5214	-0.1508	9.1942	13.2876	-1.5121	

	var_196	var_197	var_198	var_199
0	7.8784	8.5635	12.7803	-1.0914

```
1   8.1267   8.7889  18.3560   1.9518
2  -6.5213   8.2675  14.7222   0.3965
3  -2.9275  10.2922  17.9697  -8.9996
4   3.9267   9.5031  17.9974  -8.8104

[5 rows x 202 columns]
```

[ ]:
```
train_df = pd.read_csv('./data/train.csv')
```

[5]:
```
train_df = pd.read_csv('./data/train.csv')
train_df['num_words'] = train_df.predictionstring.apply(lambda s: len(s.
 ↪split()))

train_df.head()
train_df.shape
```

[5]:
```
(144293, 9)
```

[7]:
```
print("Number of Essays:", train_df.id.nunique())
print("Number of Discourse examples:", len(train_df))
print("Number of Discourse Types:", train_df.discourse_type.nunique())
print("Number Of Discourse Type Numbers:", train_df.discourse_type_num.
 ↪nunique())
```

```
Number of Essays: 15594
Number of Discourse examples: 144293
Number of Discourse Types: 7
Number Of Discourse Type Numbers: 44
```

What are the data classes?

[8]:
```
profile = pp.ProfileReport(train_df, title = "Feedback-Prize")
profile.to_notebook_iframe()
```

```
HBox(children=(HTML(value='Summarize dataset'), FloatProgress(value=0.0, max=5.
 ↪0), HTML(value='')))


HBox(children=(HTML(value='Generate report structure'), FloatProgress(value=0.0,␣
 ↪max=1.0), HTML(value='')))


HBox(children=(HTML(value='Render HTML'), FloatProgress(value=0.0, max=1.0),␣
 ↪HTML(value='')))


<IPython.core.display.HTML object>
```

10.2 Data Cleaning

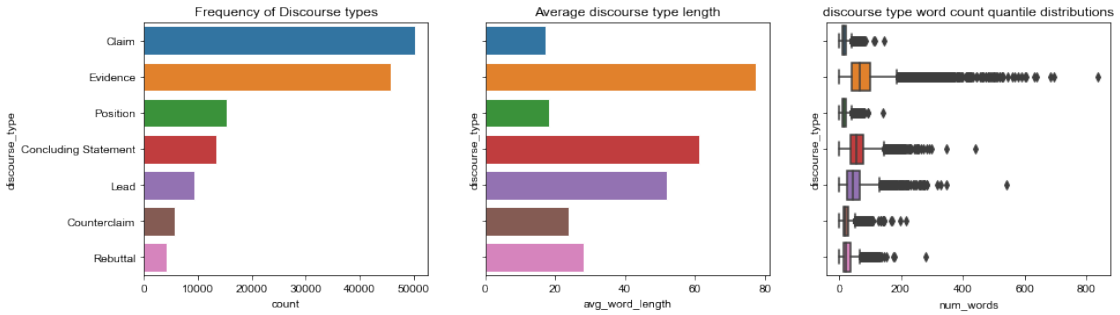
[9]:
```
train_df.isnull().any().any()
train_df.isna().sum()/(len(train_df))*100
```

[9]:
```
id                  0.0
discourse_id        0.0
discourse_start     0.0
discourse_end       0.0
discourse_text      0.0
discourse_type      0.0
discourse_type_num  0.0
predictionstring    0.0
num_words           0.0
dtype: float64
```

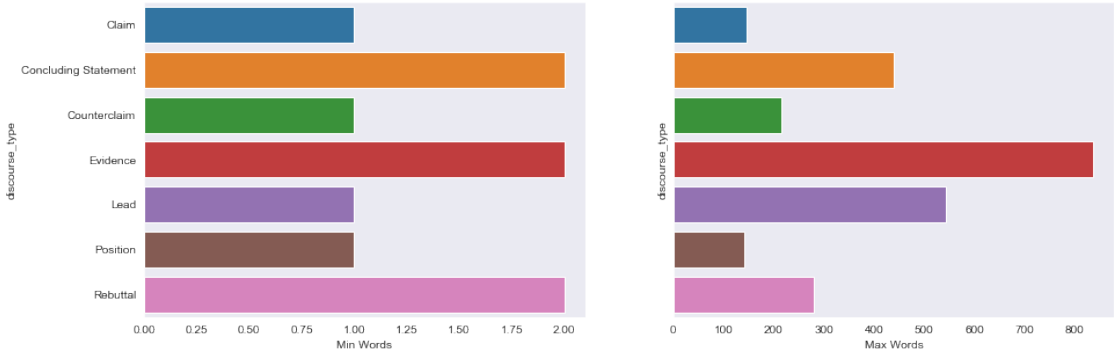
As move along the data processing the first of data cleaning is to find null values. Looking at the above output looks we have a clean dataset with no missing values.

10.3 Data Visualizations

[10]:
```
%matplotlib inline

data=train_df.groupby('discourse_type')[['num_words']].mean().reset_index().
 ↪rename(columns={'num_words': 'avg_word_length'})

fig, ax=plt.subplots(1, 3, figsize=(15, 4), sharey=True)
ax[0].set_title("Frequency of Discourse types")
ax[0].set_label('')

ax[1].set_title("Average discourse type length")
ax[2].set_title("discourse type word count quantile distributions")

sns.set_style('dark')
sns.countplot(data=train_df, y='discourse_type',
              order=train_df.discourse_type.value_counts().index, ax=ax[0])

sns.barplot(data=data.sort_values('avg_word_length', ascending=False),
            x='avg_word_length',
            y='discourse_type',
            order=train_df.discourse_type.value_counts().index,
            ax=ax[1])

sns.boxplot(data=train_df, y='discourse_type', x='num_words',
            order=train_df.discourse_type.value_counts().index,
            ax=ax[2])
plt.show()
```



[11]:
```
%matplotlib inline

data = train_df.groupby('discourse_type')[['num_words']].agg([min, max]).
 ↪reset_index()
data.columns=['discourse_type', 'Min Words', 'Max Words']

_, ax = plt.subplots(1, 2, sharey=True, figsize=(15, 5))
sns.barplot(data=data, y='discourse_type', x='Min Words', ax=ax[0])
sns.barplot(data=data, y='discourse_type', x='Max Words', ax=ax[1])
plt.show()
```



[12]:
```
train_df[train_df.num_words<=2].discourse_type.value_counts()
```

[12]:
```
Claim                   940
Position                 28
Rebuttal                  3
Lead                      3
Counterclaim              2
Evidence                  1
Concluding Statement      1
Name: discourse_type, dtype: int64
```

Looking at the claims we see that the words with smaller context have less number of words.

Lets explore some popular words from the discourse "Claim"

[13]:
```
print(train_df[(train_df.num_words<=2) &
         (train_df.discourse_type == 'Claim')
        ].discourse_text.sample(10).values)
```

```
['helping teachers,' 'be creative,' 'reduced polution '
 'different experiences. ' 'time consuming,' 'less pollution ' 'promoting'
 'more opinions ' ' confused,' 'less traffic, ']
```

Exploring the essays and using the popular discourse words to find their location in the essays

[14]:
```
essay_folder='./data/train'
essay_df = []
for filename in os.listdir(essay_folder):
    filepath = os.path.join(essay_folder, filename)
    with open(filepath) as file:
        essay_df.append({
            'id': filename.replace('.txt', ''),
            'content': file.read()
        })
essay_df = pd.DataFrame.from_dict(essay_df)
essay_df['total_num_chars'] = essay_df.content.apply(lambda x: len(x))
essay_df['total_num_words'] = essay_df.content.apply(lambda x: len(x.split()))

essay_df.head()
```

[14]:

	id	content	\
0	0000D23A521A	Some people belive that the so called "face" o…	
1	00066EA9880D	Driverless cars are exaclty what you would exp…	
2	000E6DE9E817	Dear: Principal\n\nI am arguing against the po…	
3	001552828BD0	Would you be able to give your car up? Having …	
4	0016926B079C	I think that students would benefit from learn…	

	total_num_chars	total_num_words
0	1343	251
1	3590	646
2	1527	274
3	2707	512
4	1395	266

[15]:
```
position_df = train_df[['id', 'discourse_type', 'discourse_start',␣
 ↪'discourse_end']].copy()
position_df = position_df.merge( essay_df )
position_df['discourse_start_percentile'] = 100 * position_df.discourse_start.
 ↪div(position_df.total_num_chars)
```

```
position_df['discourse_end_percentile'] = 100 * position_df.discourse_end.
 ↪div(position_df.total_num_chars)

position_df.head()
```

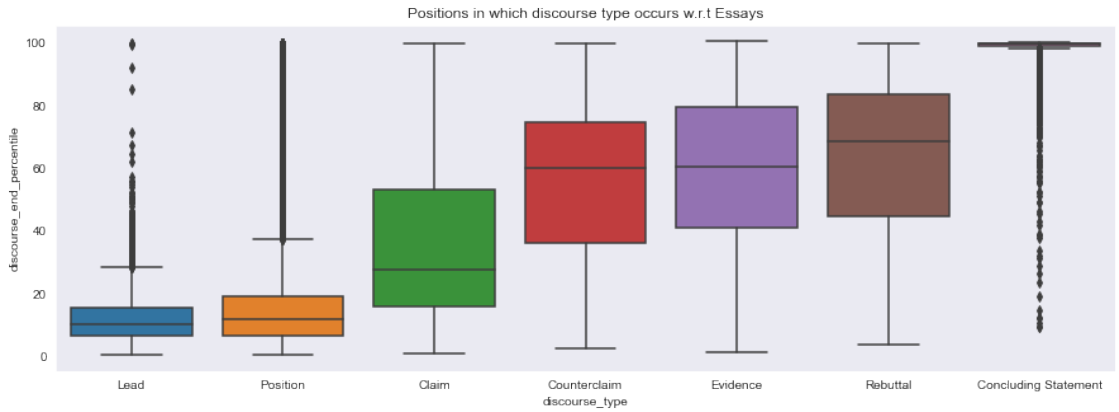
[15]:

	id	discourse_type	discourse_start	discourse_end	content	total_num_chars	total_num_words	discourse_start_percentile	discourse_end_percentile
0	423A1CA112E2	Lead	8.0	229.0	Phones\n\nModern humans today are always on th…	2030	379	0.394089	11.280788
1	423A1CA112E2	Position	230.0	312.0	Phones\n\nModern humans today are always on th…	2030	379	11.330049	15.369458
2	423A1CA112E2	Evidence	313.0	401.0	Phones\n\nModern humans today are always on th…	2030	379	15.418719	19.753695
3	423A1CA112E2	Evidence	402.0	758.0	Phones\n\nModern humans today are always on th…	2030	379	19.802956	37.339901
4	423A1CA112E2	Claim	759.0	886.0	Phones\n\nModern humans today are always on th…	2030	379	37.389163	43.645320

[16]:
```
train_df.discourse_type.unique()
```

[16]:
```
array(['Lead', 'Position', 'Evidence', 'Claim', 'Concluding Statement',
       'Counterclaim', 'Rebuttal'], dtype=object)
```

[17]:
```
%matplotlib inline
plt.figure(figsize=(15, 5))
plt.title("Positions in which discourse type occurs w.r.t Essays")
sns.boxplot(data=position_df,
            x = 'discourse_type',
            y='discourse_end_percentile',
            order=['Lead',  'Position',  'Claim',
                   'Counterclaim', 'Evidence', 'Rebuttal','Concluding␣
 ↪Statement',]
           )
plt.show()
```



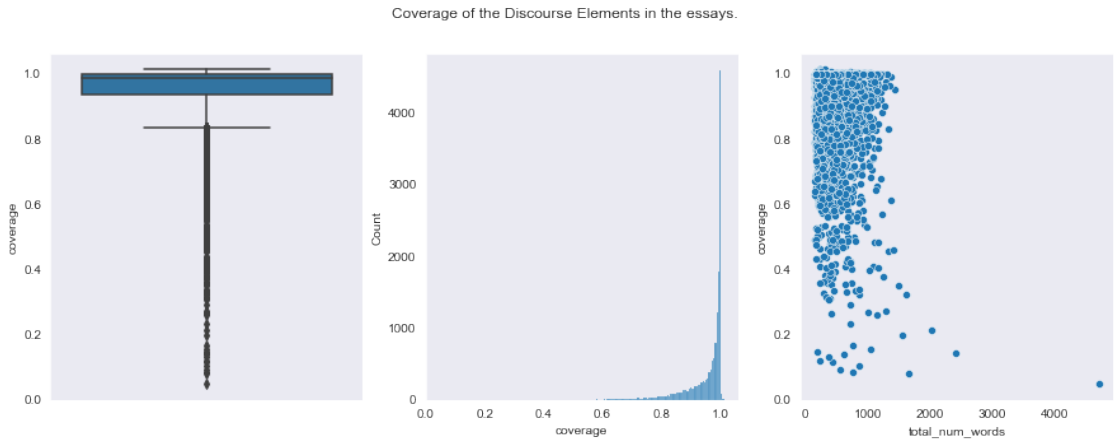
Looking at the plot we can conclude that Lead occurs at the start of the essays and concluding at the end (which is but obivious). But this proves that we can have a insights on the training dataset and the model we build will have some good predictions.

[18]:
```
data = train_df.groupby('id')[['num_words']].sum().reset_index()
data = data.merge(essay_df[['id', 'total_num_words']].copy())
data['coverage'] = data.num_words.div(data.total_num_words)

data.head()
```

[18]:

	id	num_words	total_num_words	coverage
0	0000D23A521A	251	251	1.000000
1	00066EA9880D	622	646	0.962848
2	000E6DE9E817	245	274	0.894161
3	001552828BD0	512	512	1.000000
4	0016926B079C	252	266	0.947368

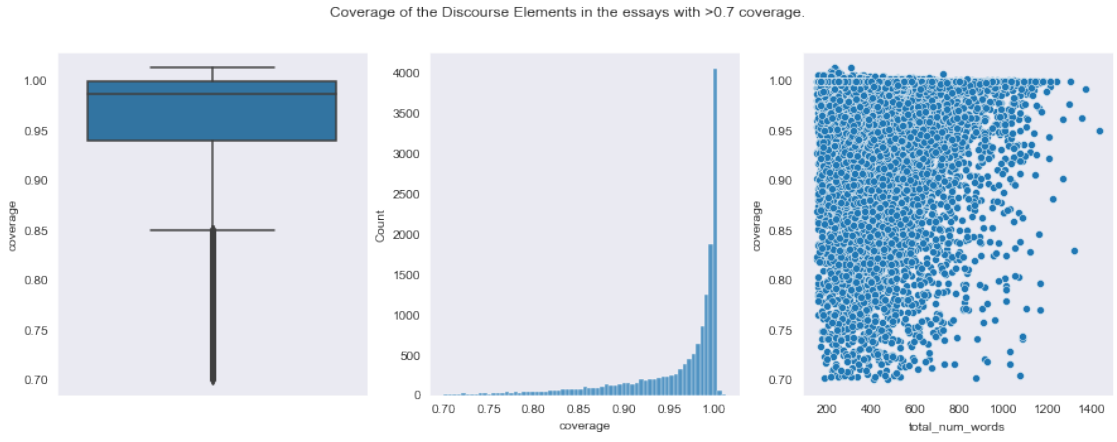
[20]:
```
_, ax=plt.subplots(1, 3, figsize=(15, 5))
plt.suptitle("Coverage of the Discourse Elements in the essays.")
sns.boxplot(data=data, y='coverage', ax=ax[0])
sns.histplot(data=data, x='coverage', ax=ax[1])
sns.scatterplot(data=data, x='total_num_words', y='coverage', ax=ax[2])
plt.show()
```



[21]:
```
print("Number of essays with <0.2 coverage:", len(data[data.coverage<0.7]) )
print("Percent of essays with <0.2 coverage: {:.4f}".format( 100 *␣
↪len(data[data.coverage<0.7])/len(data) ) )
```

```
Number of essays with <0.2 coverage: 328
Percent of essays with <0.2 coverage: 2.1034
```

[22]:
```
_, ax=plt.subplots(1, 3, figsize=(15, 5))

plt.suptitle("Coverage of the Discourse Elements in the essays with >0.7␣
↪coverage.")
sns.boxplot(data=data[data.coverage>0.7], y='coverage', ax=ax[0])
sns.histplot(data=data[data.coverage>0.7], x='coverage', ax=ax[1])
sns.scatterplot(data=data[data.coverage>0.7], x='total_num_words',␣
↪y='coverage', ax=ax[2])

plt.show()
```



[ ]: