Human or AI

By: Zubeir Said

Problem Statement

Tthe scope of this project focuses on questions and responses gathered from Reddit, which provides a diverse collection of conversational text which will then be asked of OpenAI. Once we ask the question to OpenAI we will pull down the response the AI produces and compare it to top ranked human generated response on reddit and compare to see if our NLP model can identify which response came from a human and which one came from chatgpt. The goal is to get a model be as accurate as possible in making this disinction amongst the two option, essentailly making an optimal binary classification model to achieve this. Success will be evaluated based on the model's accuracy in correctly classifying the responses, as well as its ability to generalize to unseen data. In an era where AI-generated content is becoming ubiquitous, distinguishing between human and machine-generated text is crucial. Our stakeholders come from a variety of sources such as reddit platform/users, journalist, media, and end users of user generated content and AI content.

In short:

- We want a model that guesses BOTH well
- Guessing one label better than the other is no good to us
- The highest possible accuracy is desired

The Data

Collected Reddit Data from the following Forums

- Tech Support
- Ask Philosophy
- Ask Culinary
- Ask Academia
- Ask Statistics

Looking at the Data: Vocab Count w/ Stopwords

```
word_frequency(data['reddit_question'],10)
```

	Word	Frequency
0	the	14094
1	I	12867
2	to	11258
3	and	9572
4	a	9498
5	of	7220
6	in	5589
7	is	5131
8	that	4514
9	it	4449

```
word_frequency(data['human_response'],10)
```

	Word	Frequency
0	the	10147
1	to	8887
2	a	6721
3	of	6149
4	and	5910
5	you	4960
6	is	4075
7	that	3668
8	in	3363
9	it	2423

```
word_frequency(data['chatgpt_response'],10)
```

	Word	Frequency
0	the	4728
1	to	4008
2	a	2942
3	is	2840
4	and	2647
5	of	2550
6	in	1591
7	that	1590
8	you	1516
9	it	1321

Looking at the Data: Vocab Count wo/ Stopwords

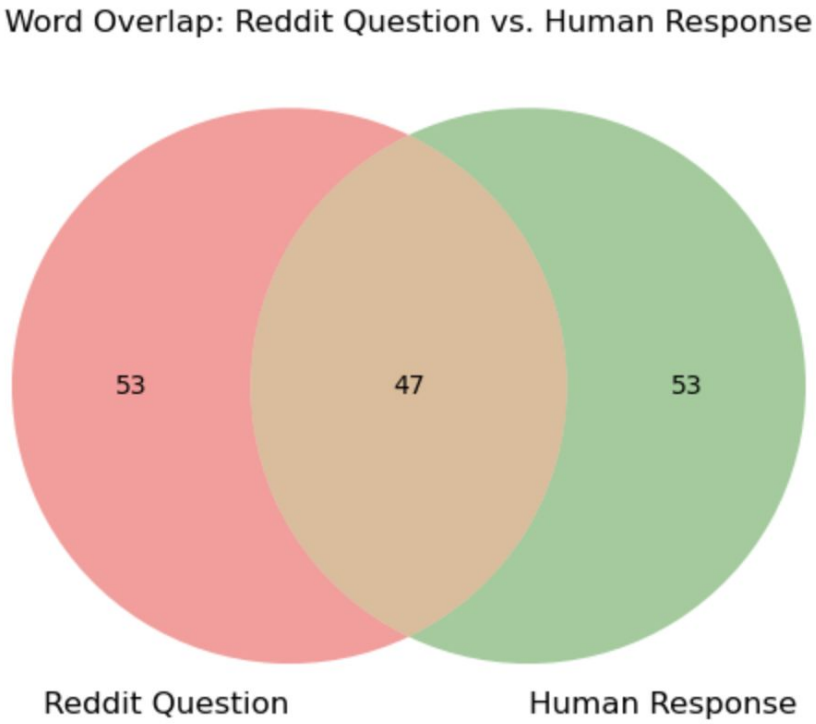
```
word_frequency(data['human_response'],10,True)
```

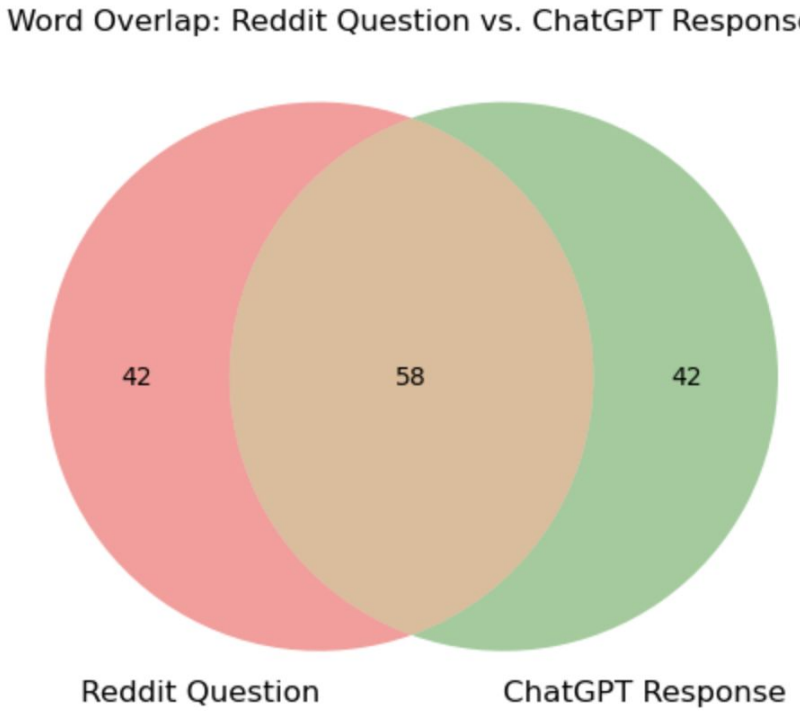
	Word	Frequency
0	would	852
1	like	795
2	questions	758
3	Please	715
4	make	709
5	see	641
6	get	588
7	one	577
8	think	504
9	read	461

```
word_frequency(data['chatgpt_response'],10,True)
```

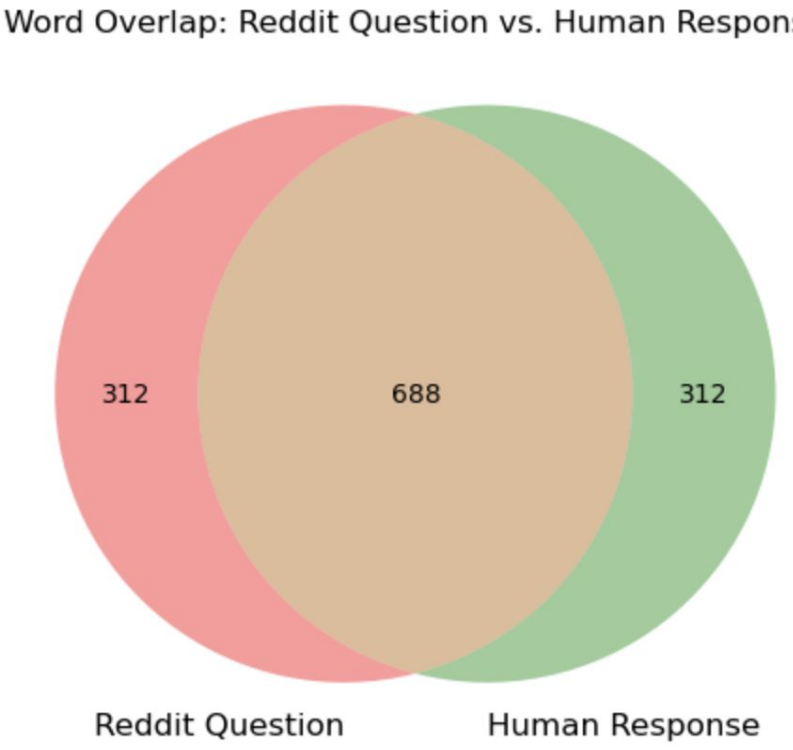
	Word	Frequency
0	may	441
1	would	419
2	could	338
3	make	258
4	like	242
5	use	235
6	Yes,	222
7	answer	216
8	way	215
9	However,	213

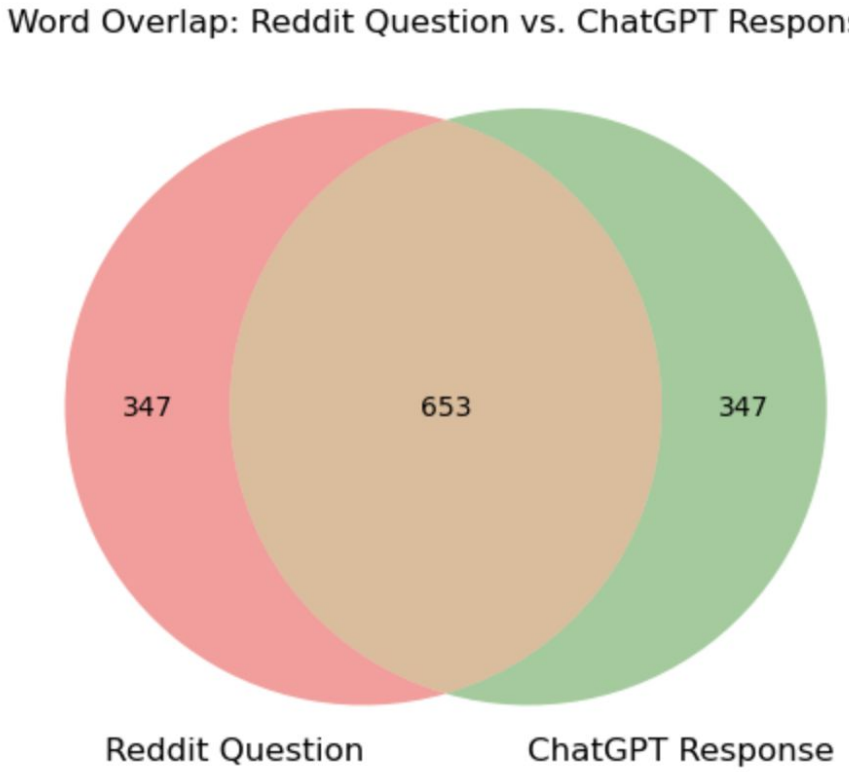
Word Overlap



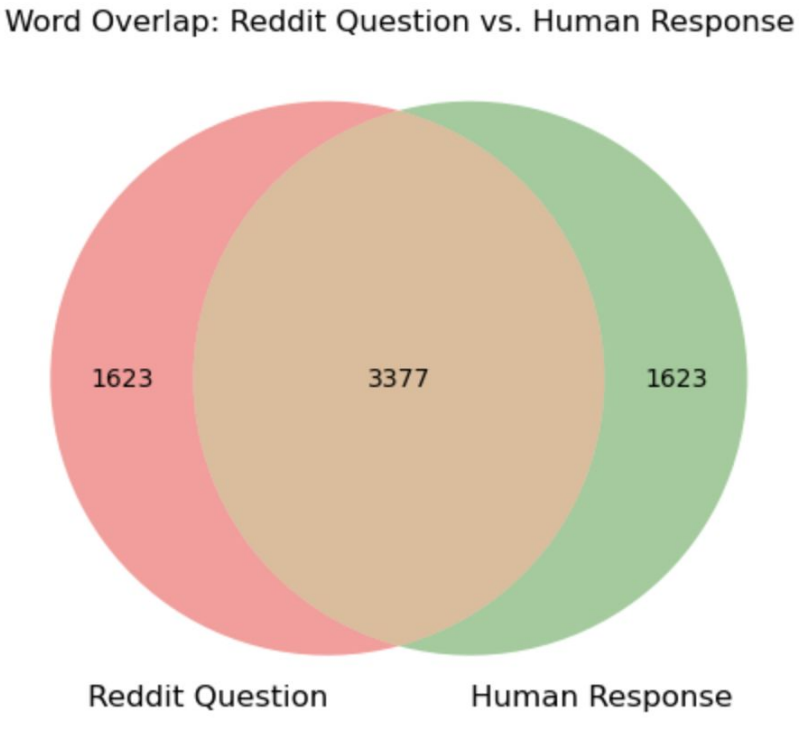


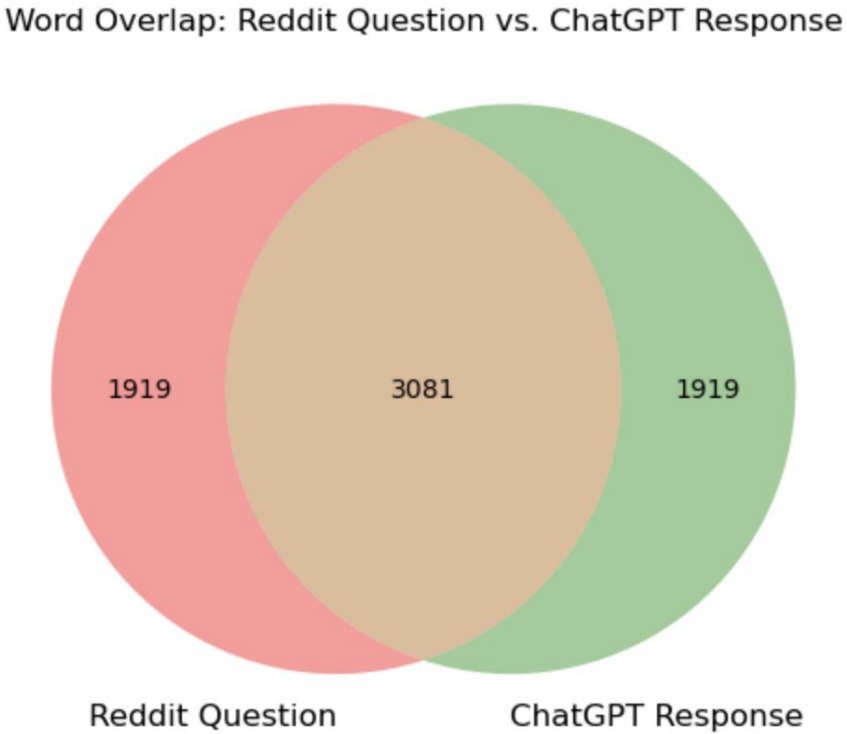
Word Overlap



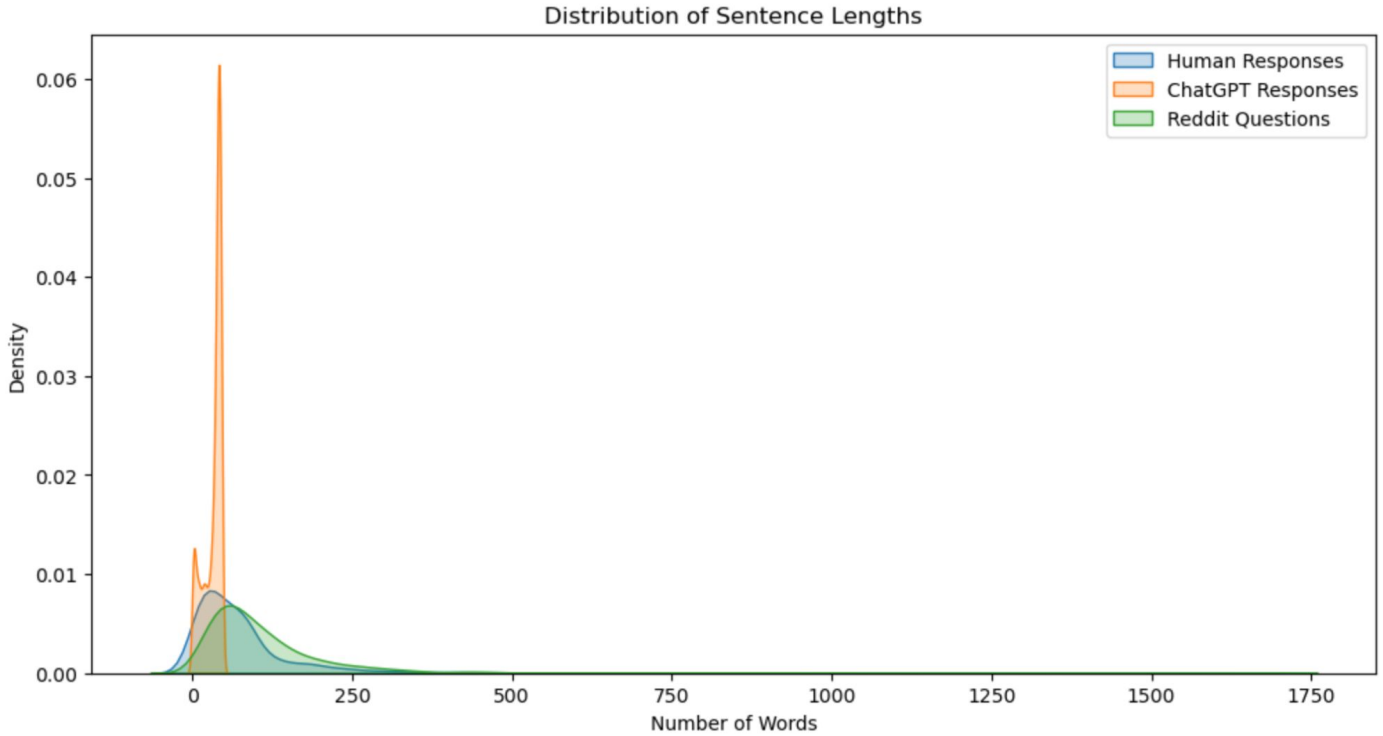


Word Overlap





Sentence Variety



Unique Word and Clause Usage

```
Average unique word ratio in reddit questions: 0.7699014300006813
Average unique word ratio in human responses: 0.8369412439804145
Average unique word ratio in chatGPT responses: 0.8692428037275195
```

```
Average number of clauses in reddit questions: 11.091174751607248
Average number of clauses in human responses: 8.26709526592636
Average number of clauses in chatGPT responses: 3.8664523670368207
```

Slang and Punctuation Usage

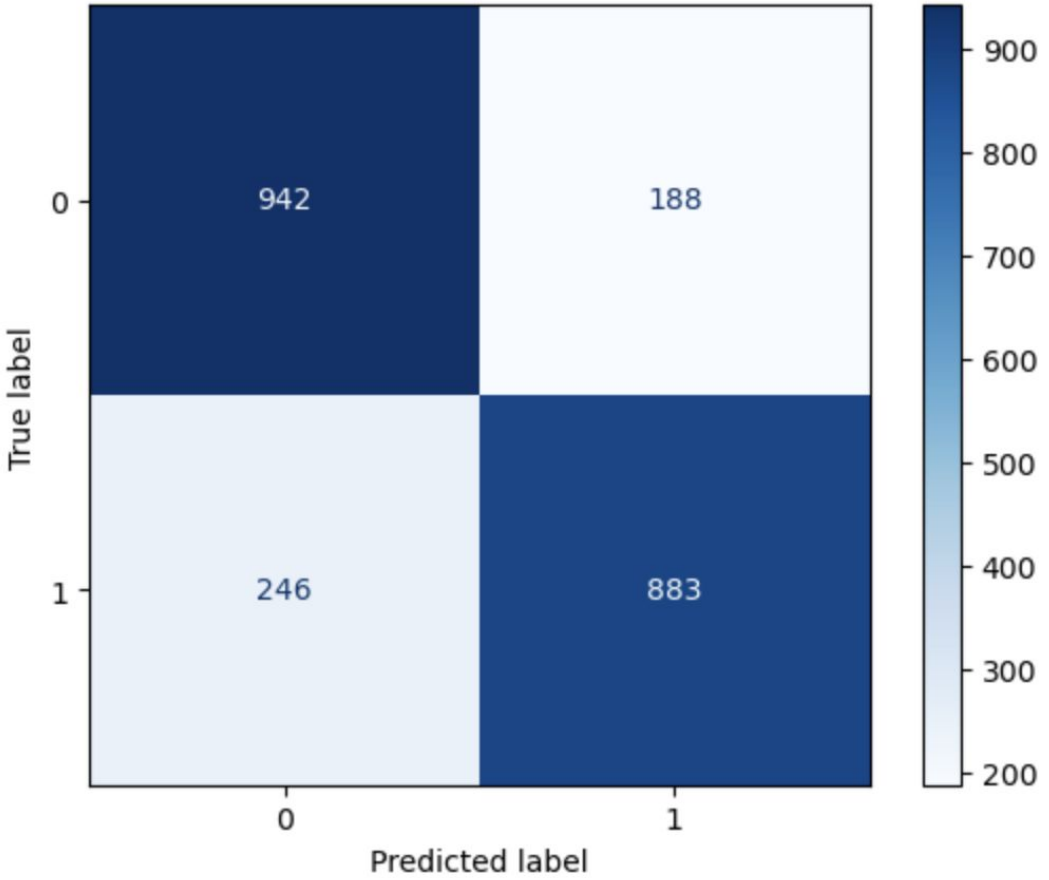
```
Average slang count in reddit questions: 0.0078901227352425 49
Average slang count in human responses: 0.010227936879018119
Average slang count in ChatGPT responses: 0.0008766803039158387
------------------------------------------------------------
Average punctuation count in reddit questions: 20.8801870251315
Average punctuation count in human responses: 23.048509643483342
Average punctuation count in ChatGPT responses: 4.333430742255991
------------------------------------------------------------
```

Modeling

- Used very low parameters to start off with
- Explored both CVEC and TFIDF
- Models looked at:
  - Logistic Regression
  - Bernoulli
  - Multinomial
  - Bagging
  - DTC
  - SVM
  - Random Forest
  - ADA Boosting
  - Gradient Boosting

Logistic Regression TVEC

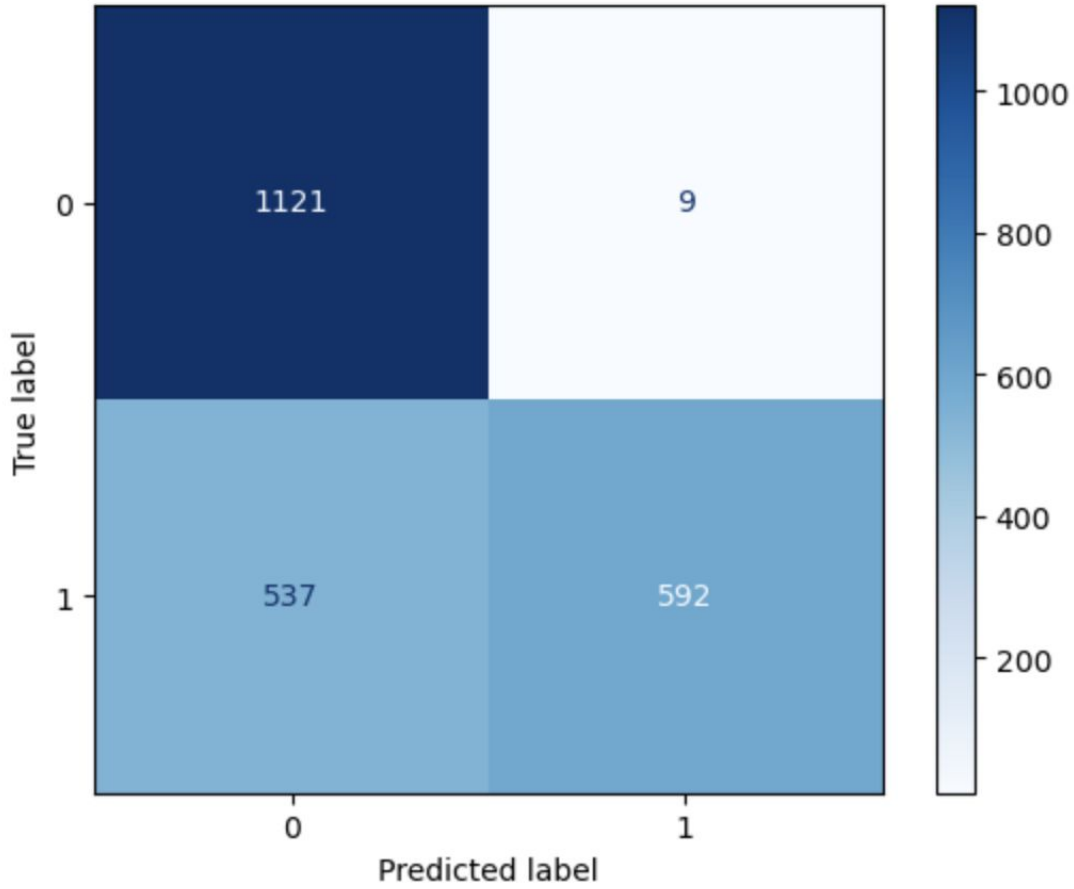
```
'cvec__max_features': [1000, 5000],
'cvec__min_df': [3, 5],
'cvec__max_df': [0.9, 0.95],
'cvec__ngram_range': [(1, 1), (1, 2)],
'cvec__stop_words': [stopwords]
```




```
TVEC Logistic Regression had a sensitivity of: 0.7821080602302923
TVEC Logistic Regression had a specificty of: 0.8336283185840708
TVEC Logistic Regression had a f1 score of: 0.8027272727272727
```

Bernoulli CVEC

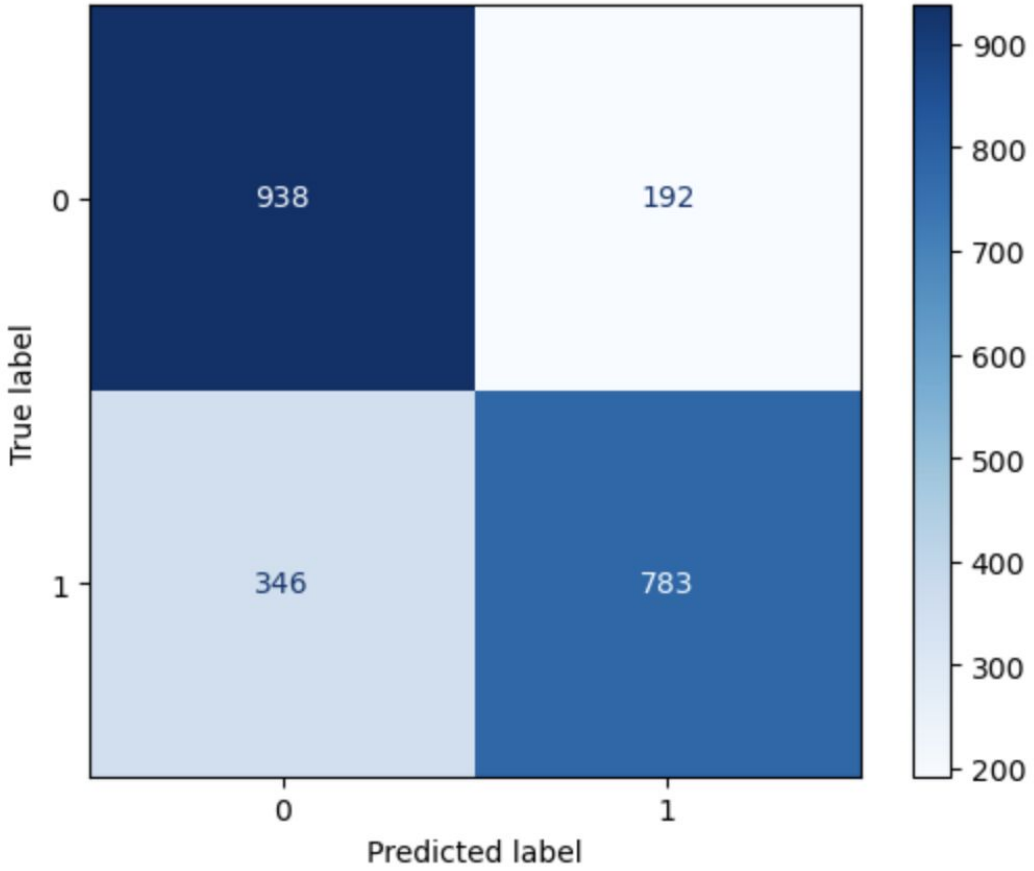
```
'cvec__max_features': [1000, 5000],
'cvec__min_df': [3, 5],
'cvec__max_df': [0.9, 0.95],
'cvec__ngram_range': [(1, 1), (1, 2)],
'cvec__stop_words': [stopwords]
```



TVEC Bernoulli had a sensitivity of: 0.5243578387953941
TVEC Bernoulli Regression had a specificty of: 0.9920353982300885
TVEC Bernoulli Regression had a f1 score of: 0.6843930635838149

Multinomial CVEC

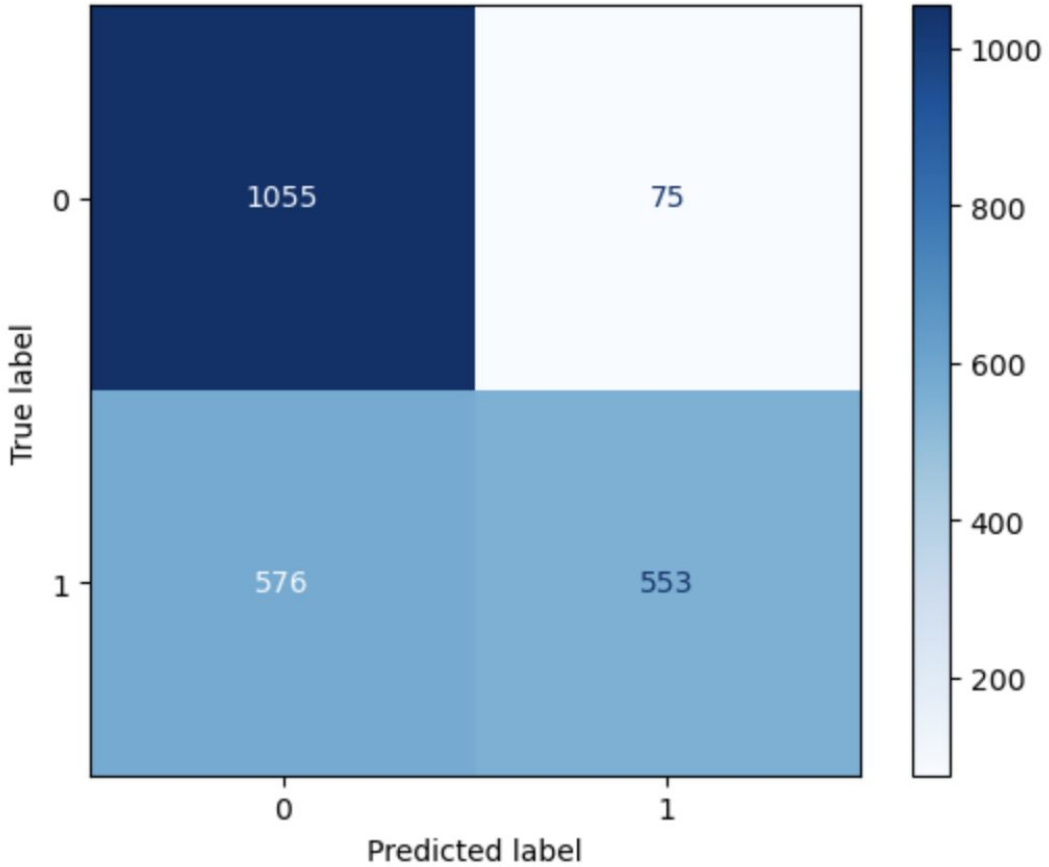
```
'cvec__max_features': [1000, 5000],
'cvec__min_df': [3, 5],
'cvec__max_df': [0.9, 0.95],
'cvec__ngram_range': [(1, 1), (1, 2)],
'cvec__stop_words': [stopwords]
```




```
CVEC Multinomial Regression had a sensitivity of: 0.6935341009743136
CVEC Multinomial Regression had a specificty of: 0.8300884955752212
CVEC Multinomial Regression had a f1 score of: 0.744296577946768
```

Bagging with DTC

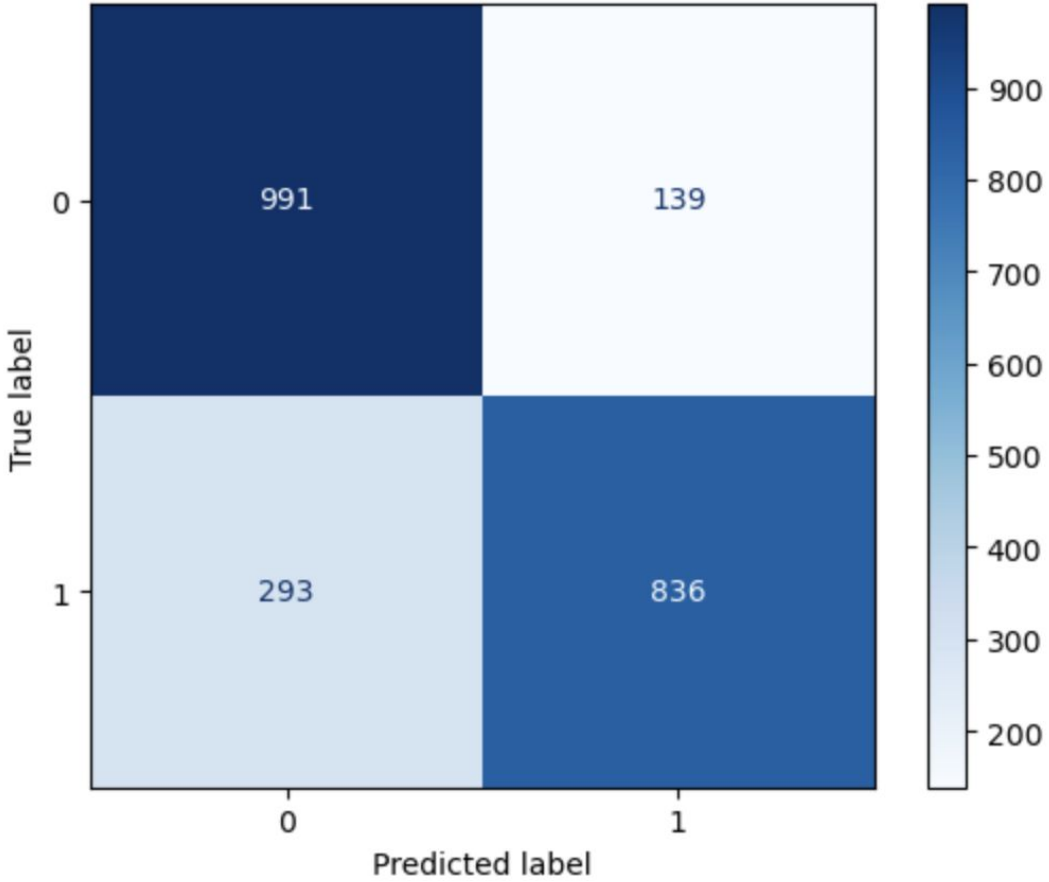
```
params_bag_dtc = {

'n_estimators' : [50, 100,150],

'base_estimator__max_depth' : [3, 5]  ¶

}
```




```
Bagging with a base DTC had a sensitivity of: 0.48981399468556247
Bagging with a base DTC had a specificty of: 0.9336283185840708
Bagging with a base DTC had a f1 score of: 0.6294820717131474
```

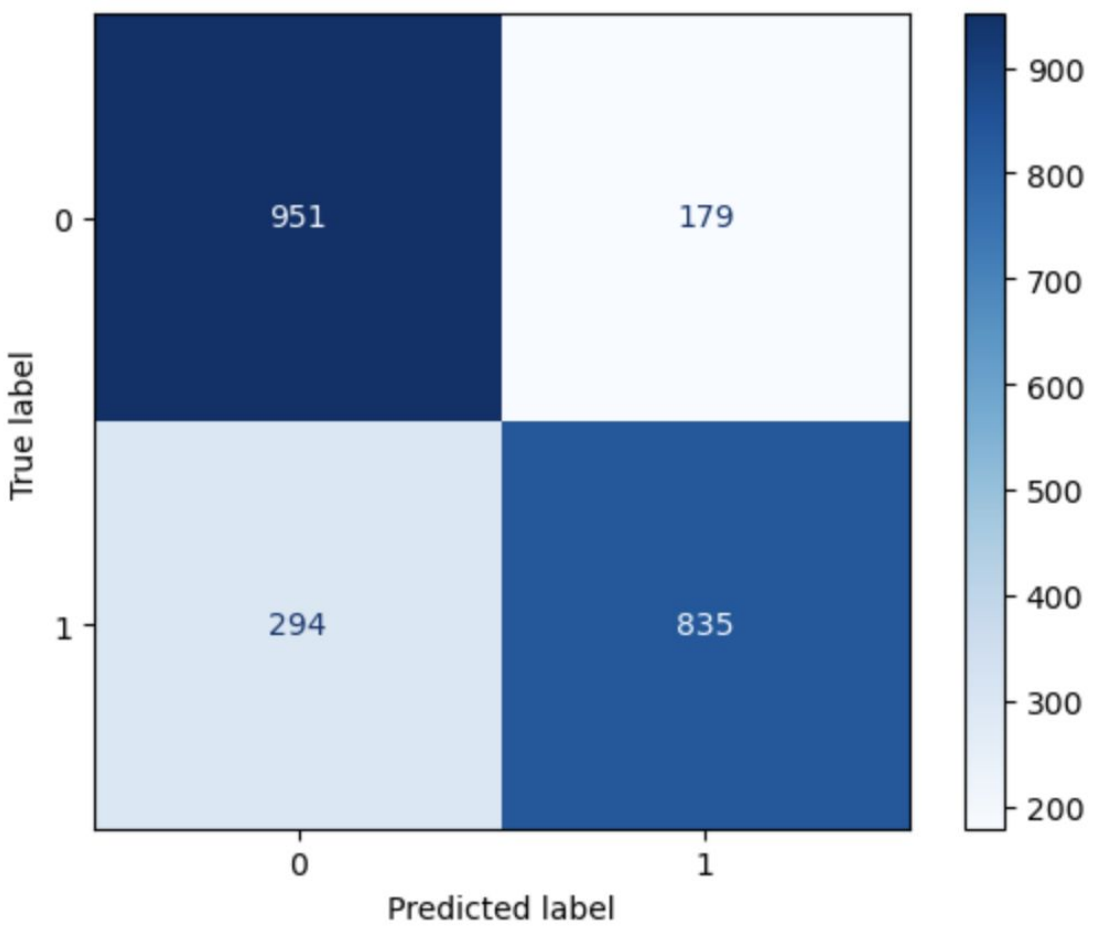
Random Forest Classifier

```
rf_params = {

'n_estimators': [150,250],

'max_depth': [None, 3,]

}
```




```
RF had a sensitivity of: 0.7404782993799823
RF had a specificty of: 0.8769911504424779
RF had a f1 score of: 0.7946768060836502
```

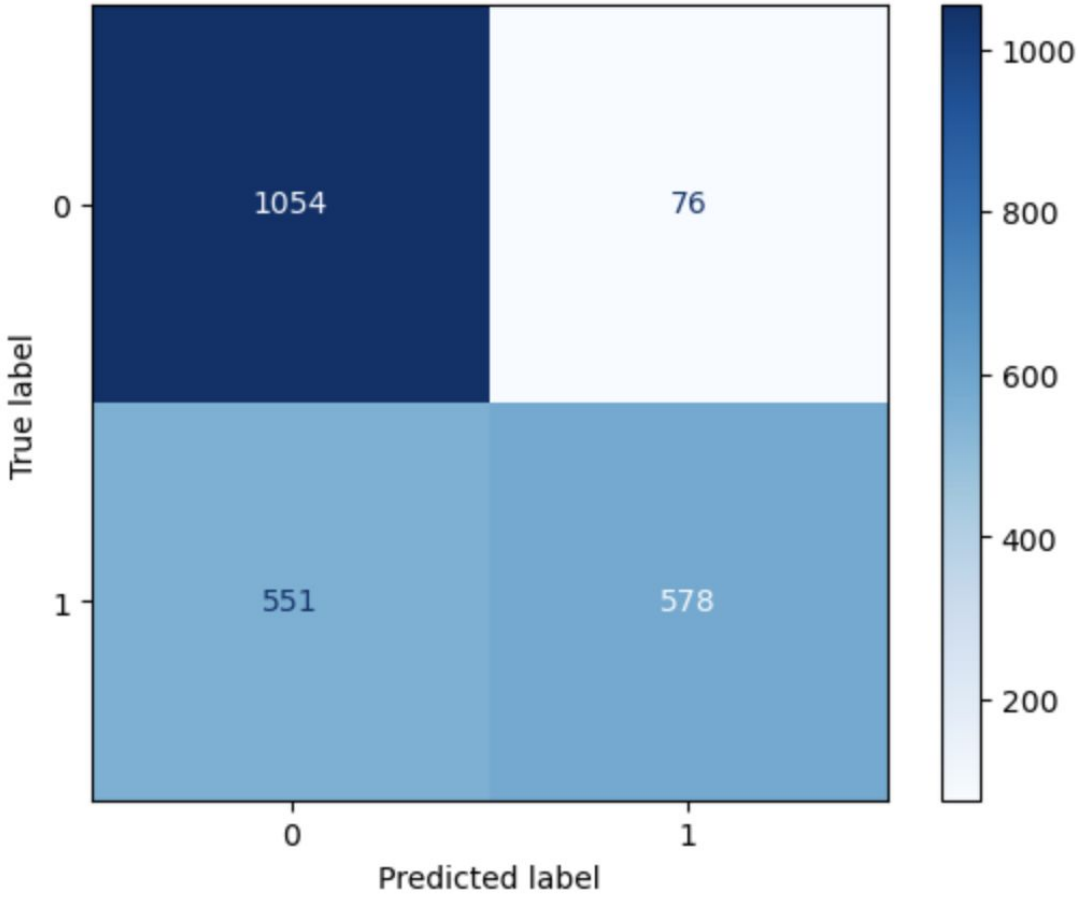
SVM



SVM had a sensitivity of: 0.8184233835252436
SVM with a base LR had a specificty of: 0.8619469026548673
SVM with a base LR had a f1 score of: 0.7792813812412505

Ada Boosting

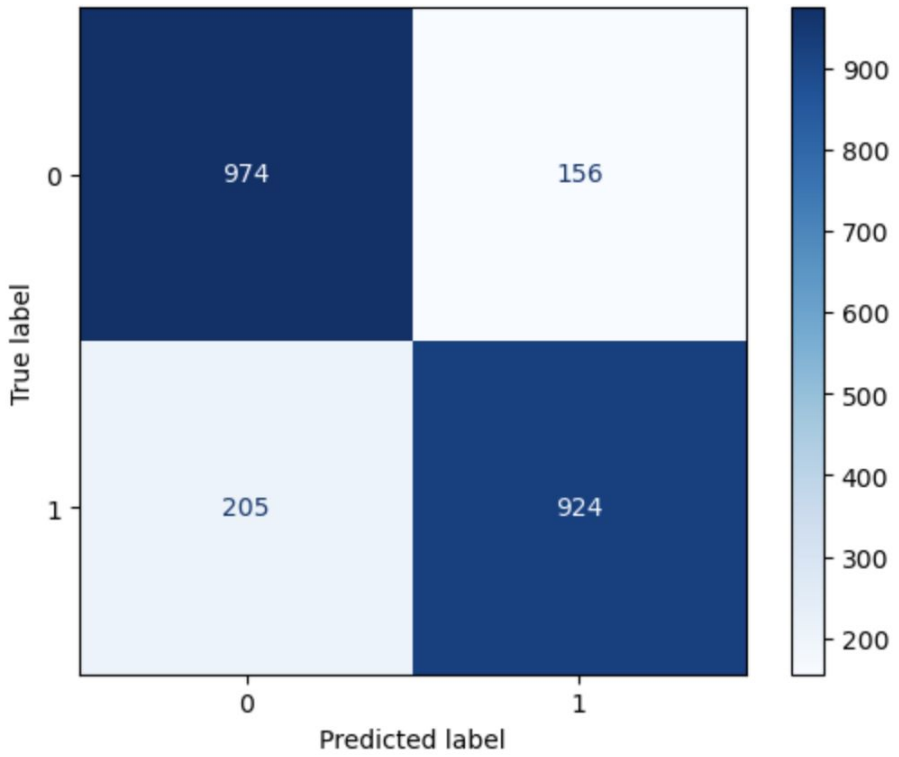
```
abc = AdaBoostClassifier(random_state=42,
base_estimator=LogisticRegression(),
n_estimators = 150)
```




```
ABC with a base LR had a sensitivity of: 0.5119574844995571
ABC with a base LR had a specificty of: 0.9327433628318584
ABC with a base LR had a f1 score of: 0.6483454851374089
```

Best Model: Logistic Regression TVEC

```
params = {
    'tvec__max_features': [15000,20000],
    'tvec__min_df': [3],
    'tvec__max_df': [0.9, 0.95],
    'tvec__ngram_range': [(1, 5), (1, 4)],
    'tvec__stop_words': [None, 'english', nltk_stop]
}
```



TVEC Logistic Regression had a sensitivity of: 0.8184233835252436
TVEC Logistic Regression had a specificty of: 0.8619469026548673
TVEC Logistic Regression had a f1 score of: 0.8365776369397918

Conclusions

- Logistic seems to be the strongest when it comes to classification
- Does the best overall at predicting BOTH human and AI
- Would love to see what other types of boosting exists with Logistic Regression
- Would love to run thousands of n-estimators on decision trees
- Would have loved to have a ton more problem free data