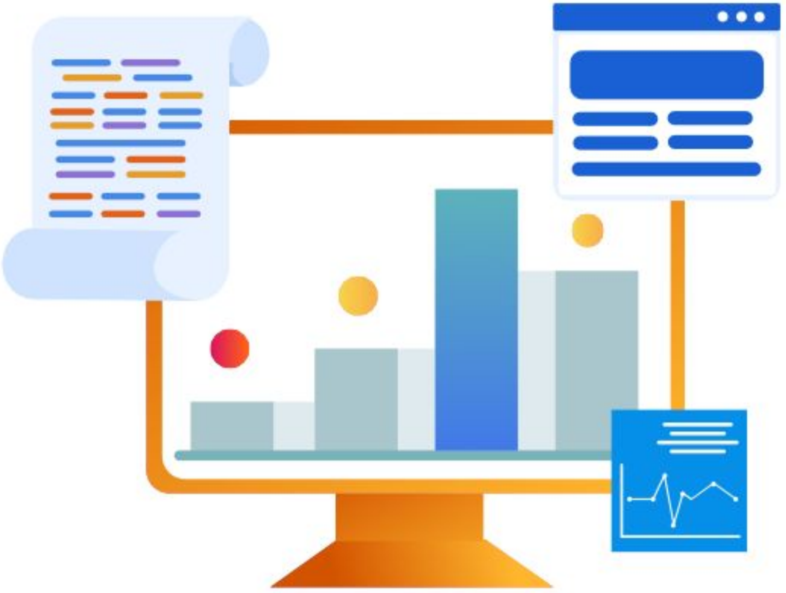
DEPRESSION DIAGNOSIS BASED ON NATURAL LANGUAGE



DSI-Project 3: Web APIs & Subreddit Classification with NLP

THANAPONG LIKHITPARINYA

(Physician Data Scientist)

KANITIN SUKDIT

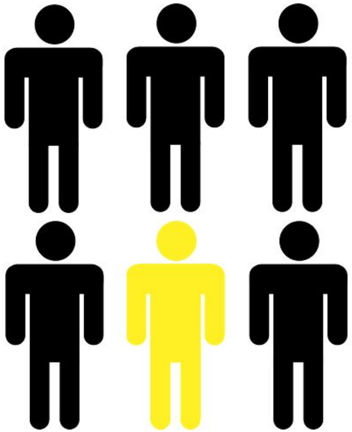
(Social Work Data Scientist)

PROBLEM STATEMENT

AFFECTED

Anxiety —> Self Harm —> Suicide — > **Violence in society**

ONE IN SIX



people experience depression at some time in their life.

Source : [Psychiatry.org - What Is Depression?]()

PROBLEM STATEMENT (CONT.)

DEPRESSION

Support & **Find a Way Out**
People struggling with depression

VENT

Listen & **Open Channel**
People who feel they can’t speak freely

Decrease risk of **Violence in Society**

OUTLINE

01 DATA ACQUISITION

02 DATA CLEANING AND EDA

03 FEATURE ENGINEERING
+ TUNING MODEL

04 EVALUATION MODEL

05 CONCLUSION
AND RECOMMENDATIONS

OUTLINE

01 DATA ACQUISITION

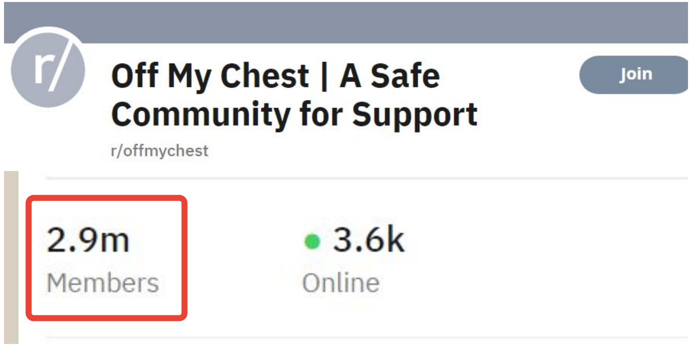
02 DATA CLEANING AND EDA

03 FEATURE ENGINEERING
+ TUNING MODEL

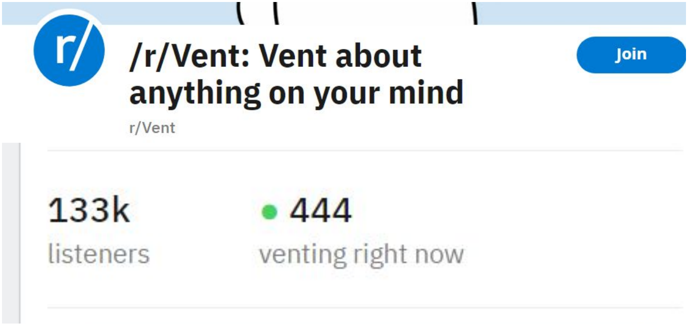
04 EVALUATION MODEL

05 CONCLUSION
AND RECOMMENDATIONS

01 DATA ACQUISITION



- **Reddit's API (requests library) → .JSON**
    - depression subreddit : **2001** posts
    - offmychest subreddit : **1973** posts



OUTLINE

01 DATA ACQUISITION

02 DATA CLEANING AND EDA

03 FEATURE ENGINEERING
+ TUNING MODEL

04 EVALUATION MODEL

05 CONCLUSION
AND RECOMMENDATIONS

02 DATA CLEANING AND EDA

- ## DATA CLEANING

Many missing values

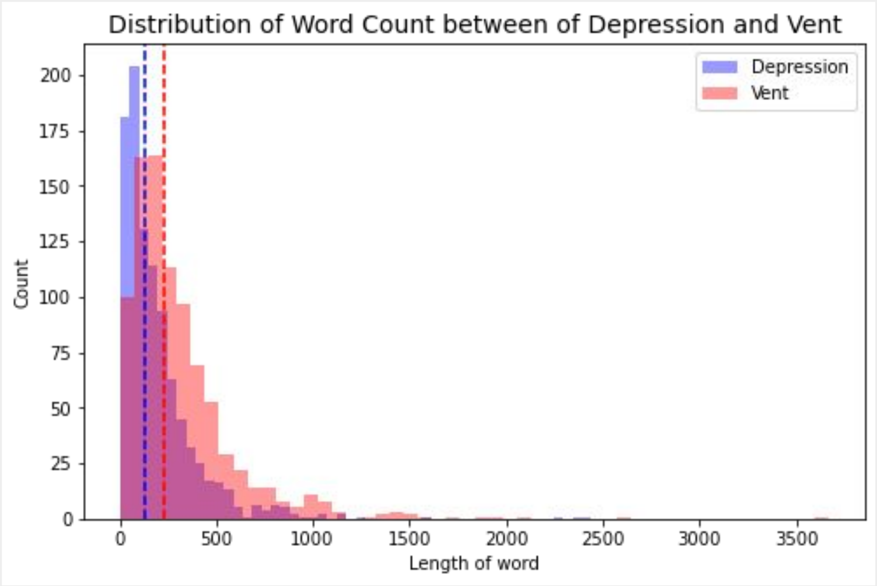
title	selftext	text	subreddit
Title of post	Body text of post		Name of subreddit

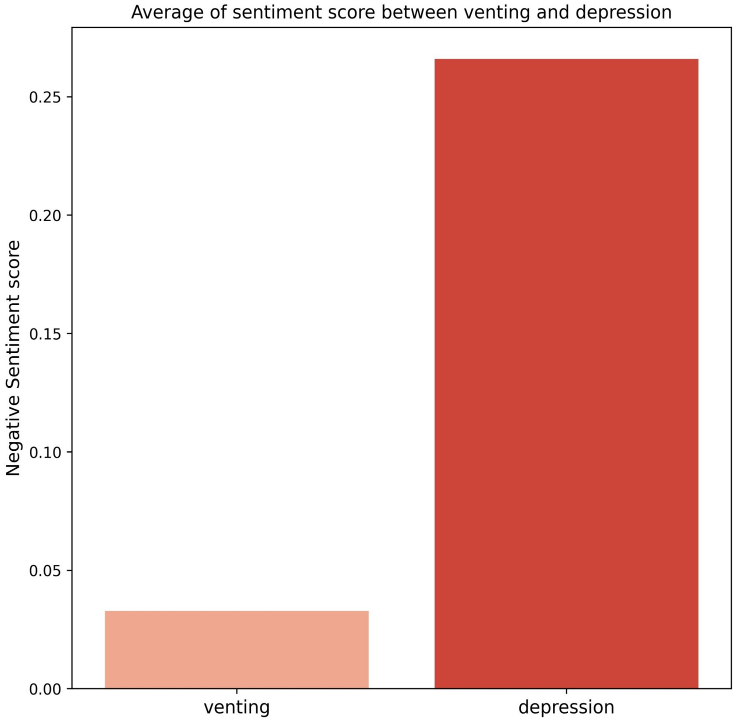
- **Combined :** Title and Selftext
- **Drop duplicate rows**
- **Drop missing value**
- **Drop character code :** amp;#x200B

- Final Dataset have **1863** rows
  - Depression : **975** posts
  - Offmychest(Vent) : **888** posts

02 DATA CLEANING AND EDA (CONT.)

- **EXPLORATORY DATA ANALYSIS :**



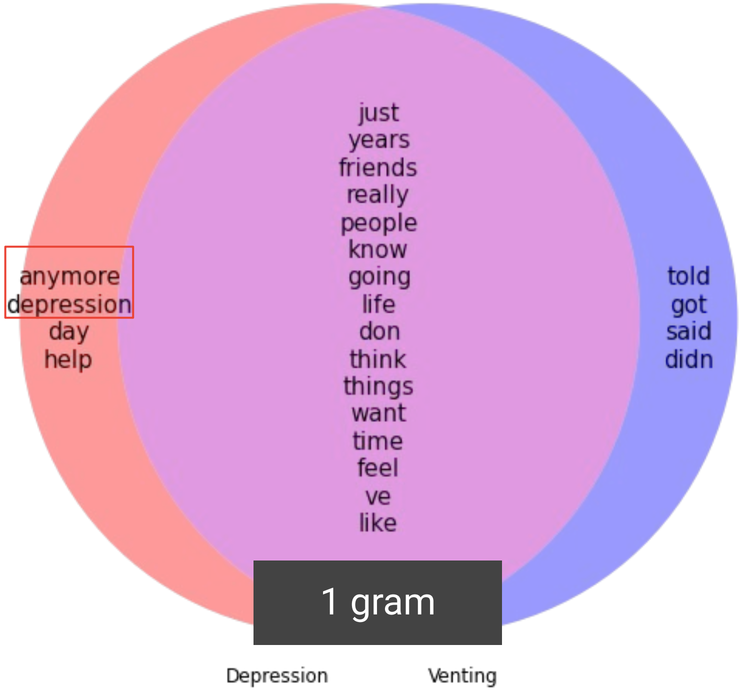


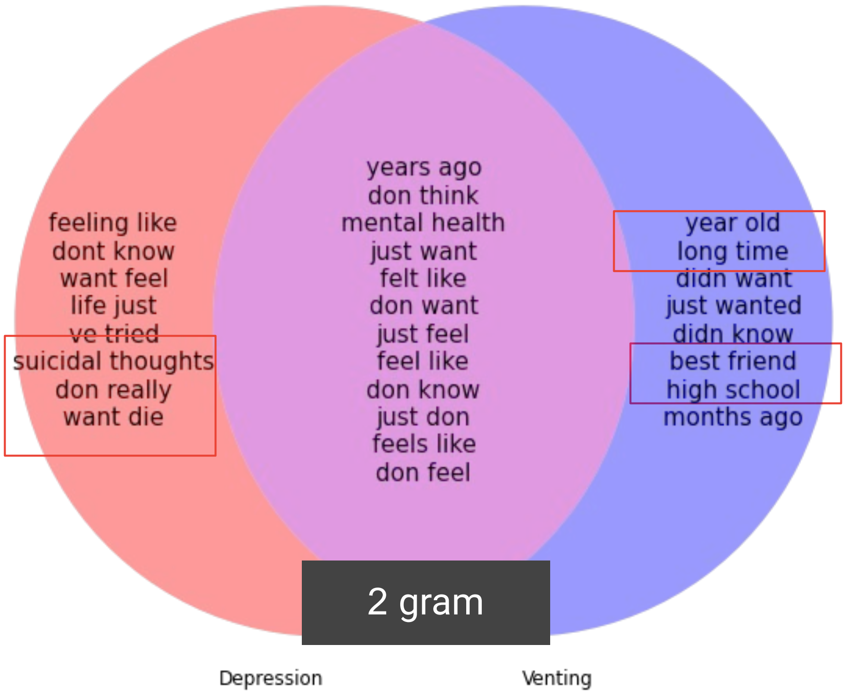
- Tokenizer : /w+
- Word count : **Vent > Depression**

- Used sent.polarity_scores
- Negative Sentiment: **Depression > Vent**

02 DATA CLEANING AND EDA (CONT.)

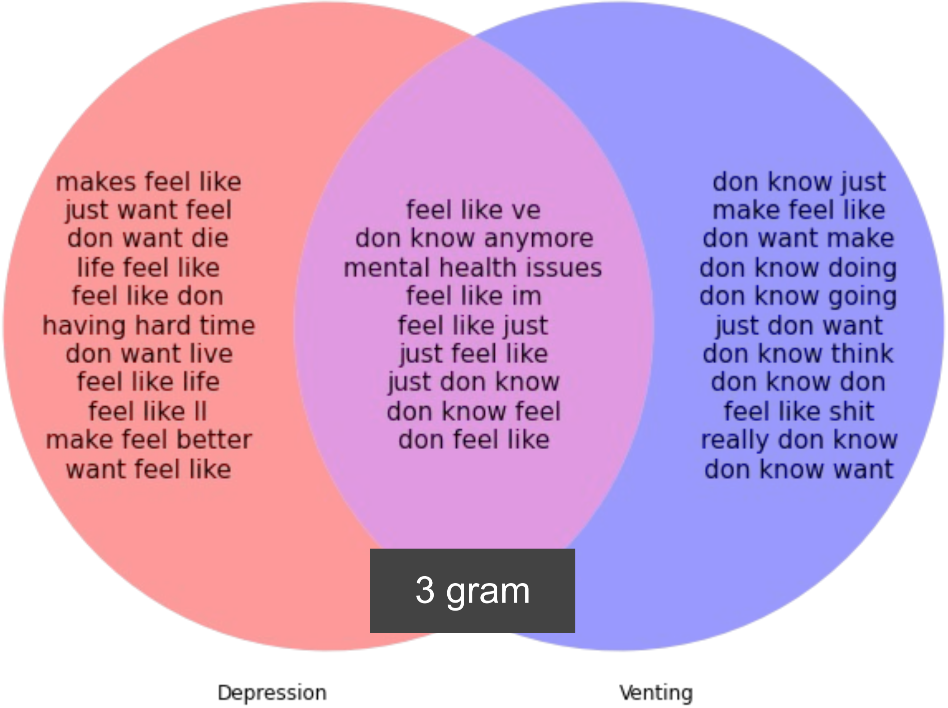
- **EXPLORATORY DATA ANALYSIS : FREQUENT WORDS (TOP 20 OF EACH SUBREDDIT)**
  - Countvectorizer() – > tokenzier
  - Stopword = “english”





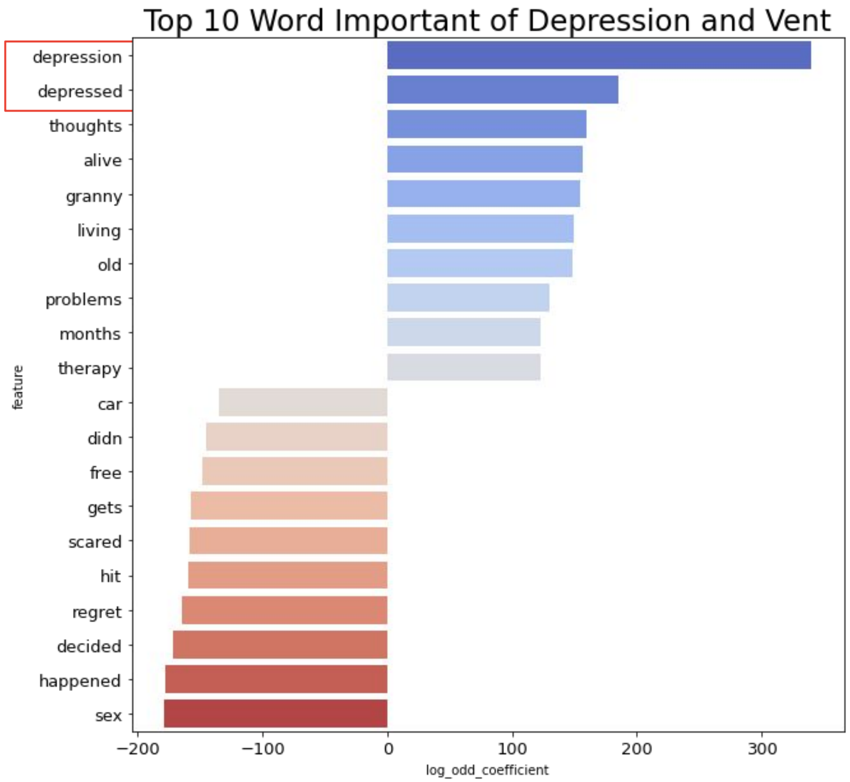
02 DATA CLEANING AND EDA (CONT.)

- **EXPLORATORY DATA ANALYSIS : FREQUENT WORDS (TOP 20 OF EACH SUBREDDIT)**
  - Countvectorizer() – > tokenzier
  - Stopword = “english”



02 DATA CLEANING AND EDA (CONT.)

- ## EDA - PREPROCESSING NOISE REDUCTION



Benchmark model

CountVectorize : Tokenize words (stopwords=”english”)

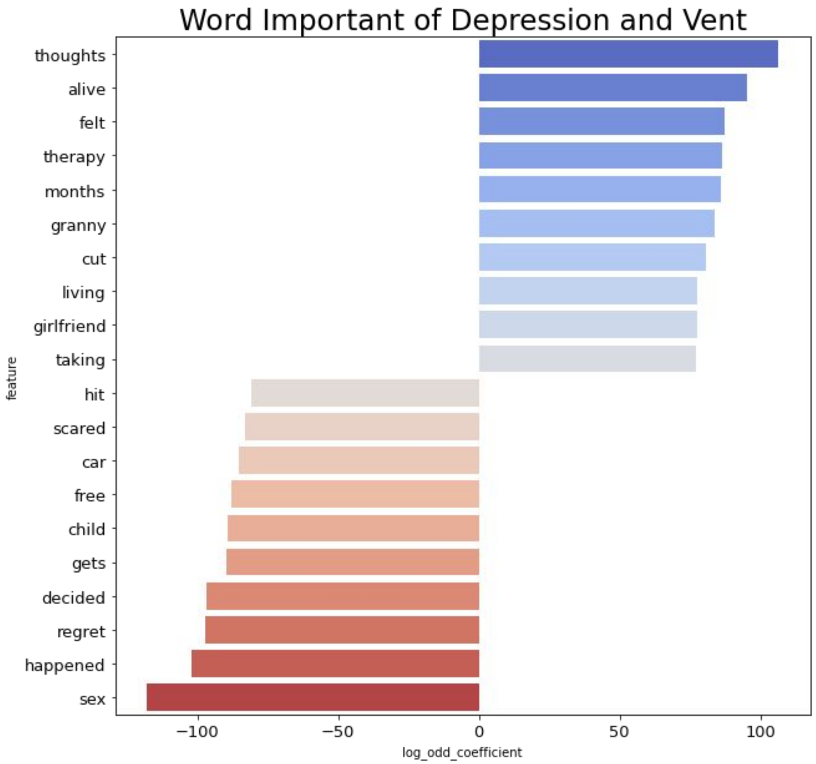
Logistic Regression : Classification

Show Word Importance : Coefficient

Remove Impact words : “depression” and “depressed”

02 DATA CLEANING AND EDA (CONT.)

- ## EDA - PREPROCESSING NOISE REDUCTION



Important words in Depression :
Thoughts, alive, felt, therapy, mouths, granny, living, girlfriend taking

Important words in Vent:
Sex, happened, decided, gets, child, free car, scared

OUTLINE

01 DATA ACQUISITION

04 EVALUATION MODEL

02 DATA CLEANING AND EDA

05 CONCLUSION AND RECOMMENDATIONS

03 MODELLING

03 MODELLING - BASELINE

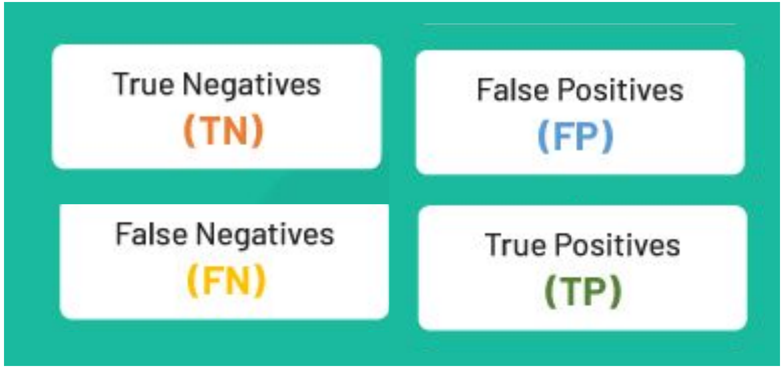
BASELINE SCORE

Depression class: 52.33 %
Vent class: : 47.67 %

BASELINE MODEL

Training accuracy : 100 %
Testing accuracy : 71 %

CONFUSION MATRIX

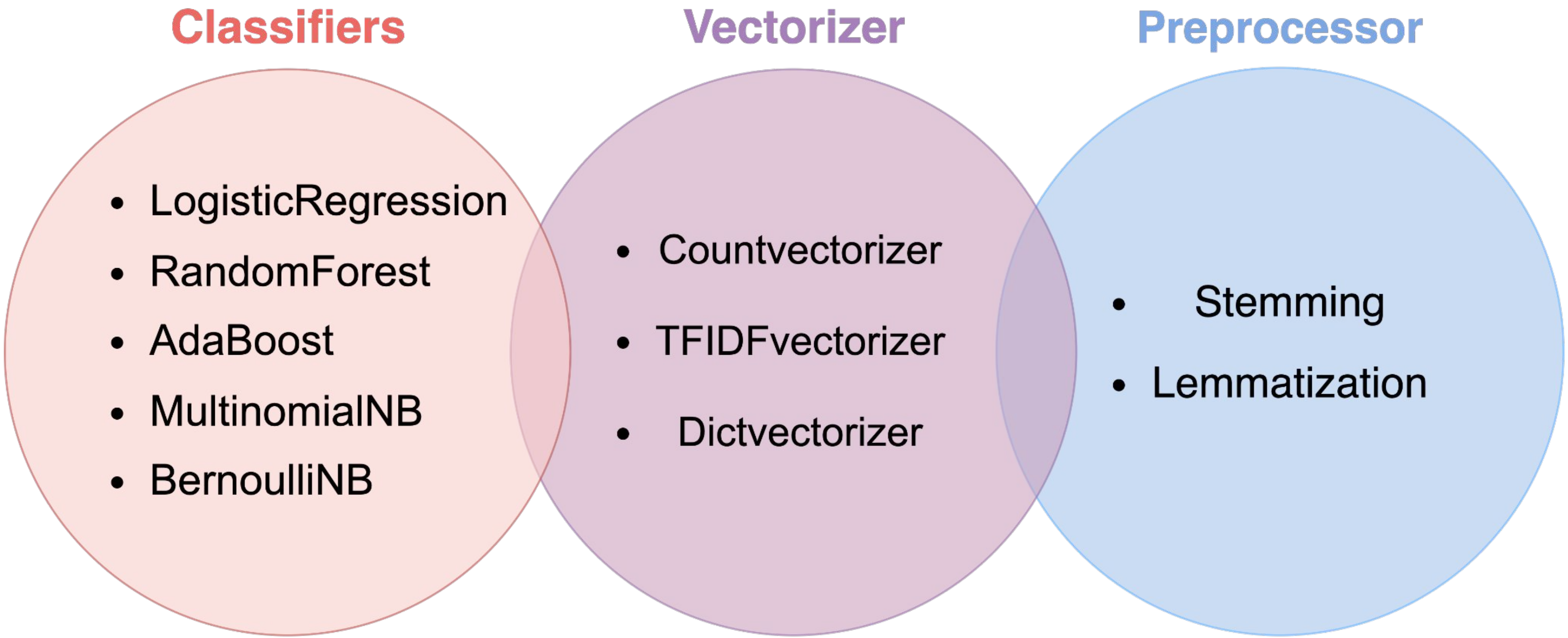


TP and TN : Accuracy score

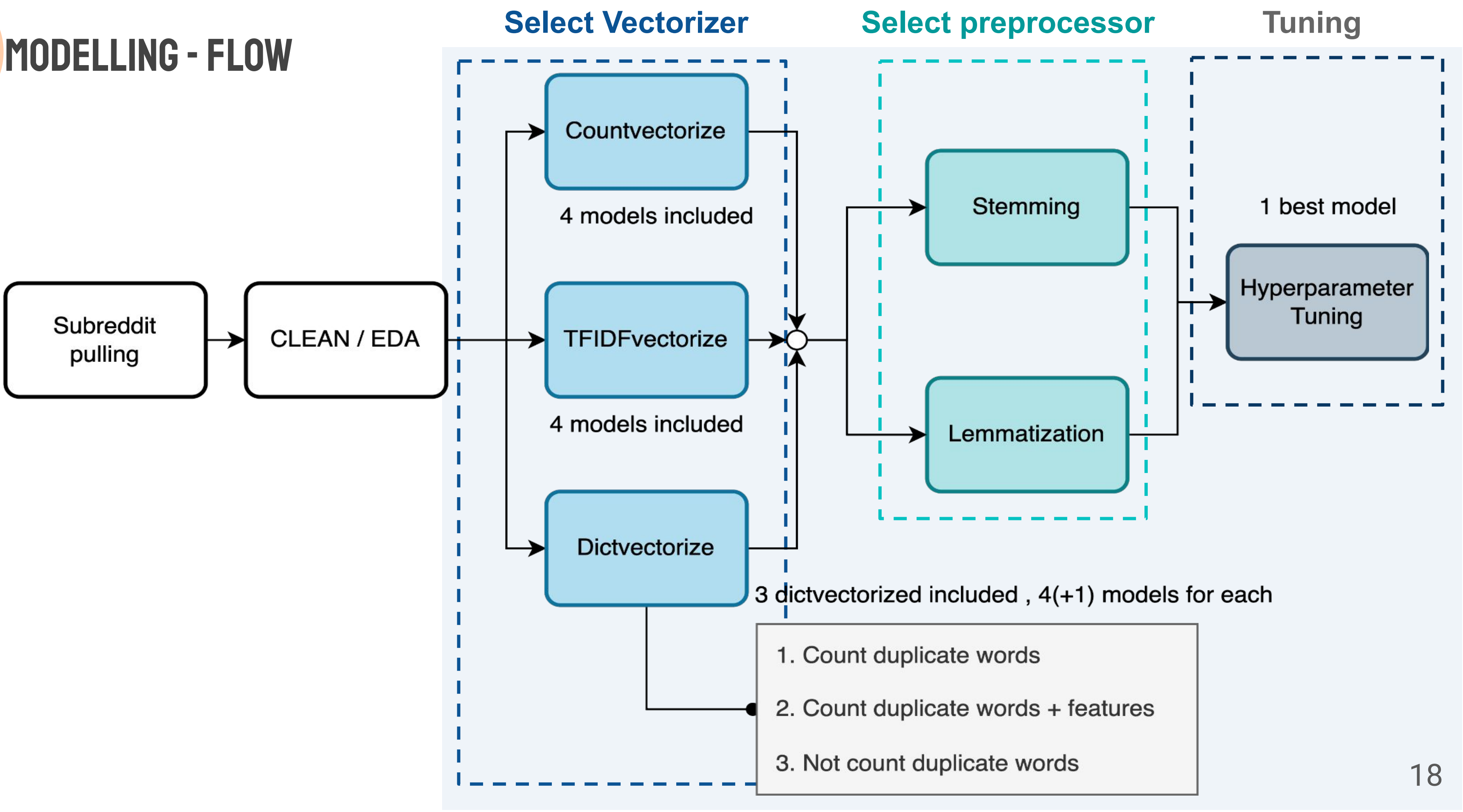
Recall score

FN values : People with depression but the model fail to capture

03 MODELLING - TOOLS

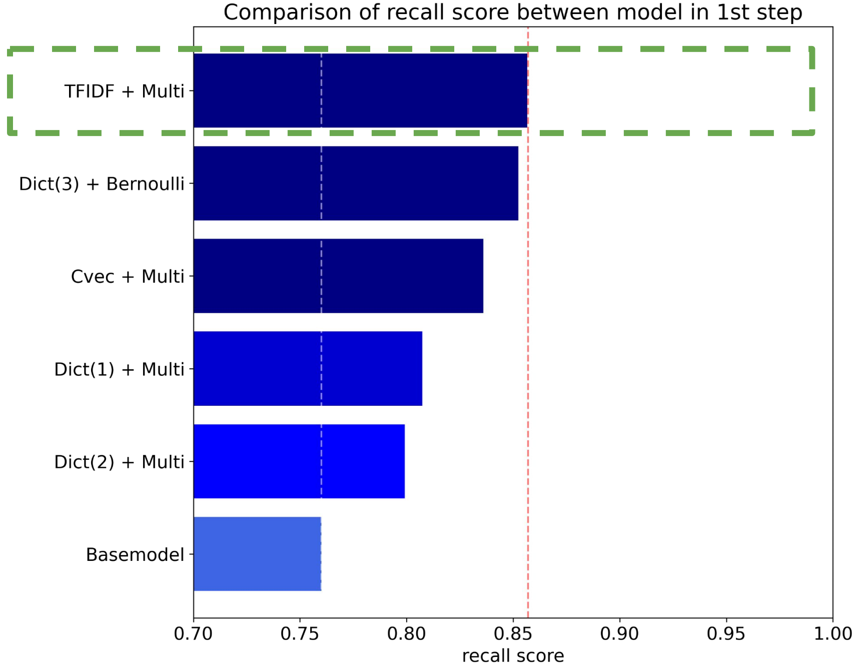


03 MODELLING - FLOW



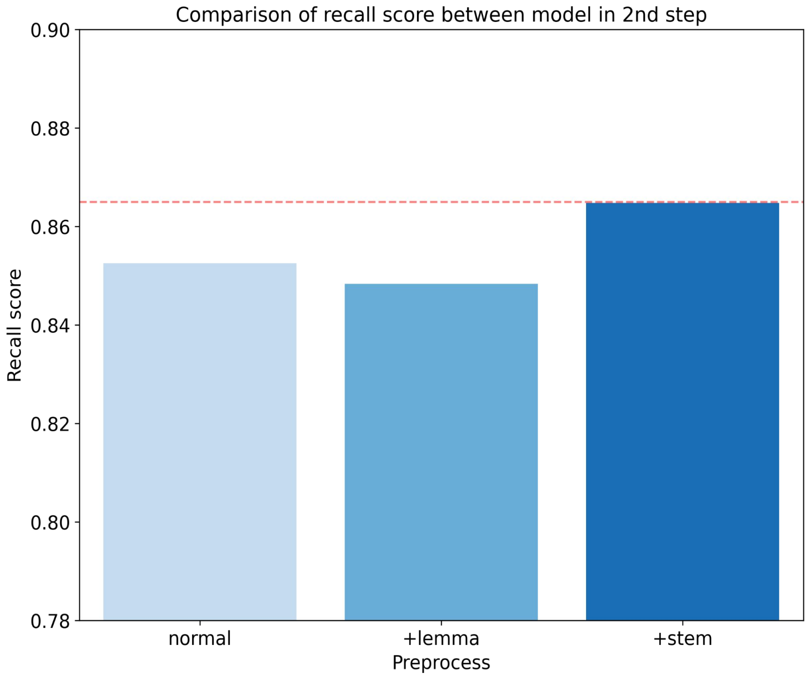
03 MODELLING - PERFORMANCE I

1st step - Vectorizer Selection



- **The top model can increase recall score by ~13 %**

2nd step - Preprocessor Selection



Preprocessing can improve the recall score but more overfitting occurs.

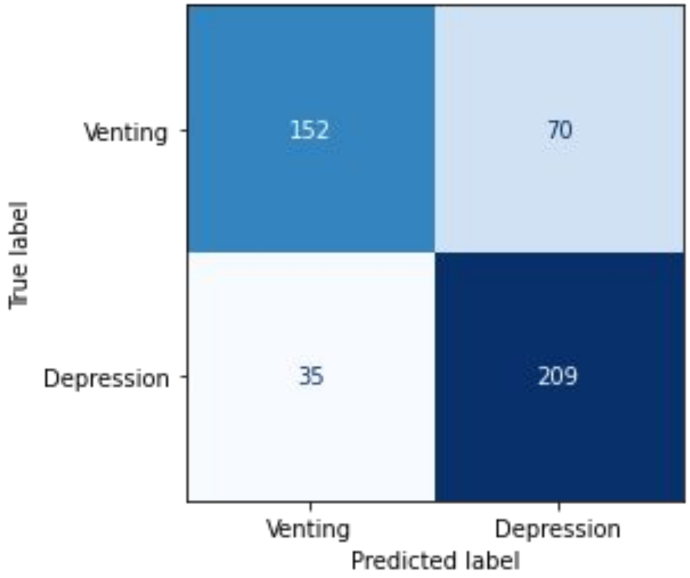
03 MODELLING - PERFORMANCE II

Last step - Hyperparameter Tuning Selection

	Before Tuning	After Tuning
Training Accuracy	**0.867**	**0.8260**
Testing Accuracy	**0.770**	**0.775**

Best hyperparameters :
'max_df : 1 , 'fit_prior' : True , ' alpha ' : 1, 'Max_features : 1500' , 'min_df' :2 , ngram_range : (1,2), 'stopword' : english

Confusion Matrix



- **The confusion matrix of the final model**

OUTLINE

01 DATA ACQUISITION

02 DATA CLEANING AND EDA

03 FEATURE ENGINEERING
+ TUNING MODEL

04 EVALUATION MODEL

05 CONCLUSION
AND RECOMMENDATIONS

04 EVALUATION MODEL

Number of data : 466

Testing Accuracy : ~ 77.5%

Recall score : ~ 86%

Number of data : 790,184

Testing Accuracy : ~ 49%

Out of word!

OUTLINE

01 DATA ACQUISITION

02 DATA CLEANING AND EDA

03 FEATURE ENGINEERING
+ TUNING MODEL

04 EVALUATION MODEL

05 CONCLUSIONS
AND RECOMMENDATIONS

05 CONCLUSIONS

- ## CONCLUSIONS

  - The last model that we have selected performed well **(Accuracy > 75%)**
  - All models exceed the baseline accuracy **(52.33%)**
  - Definite Winner : **Multinomial Naive Bayes with TFIDfVectorizer Model and Stemmong process**
  - Top 4 Important words Depression: **Thoughts, alive, felt, therapy**

05 RECOMMENDATIONS & FUTURE WORK

- ## RECOMMENDATIONS

  - This model could be used for effectively detecting depressed individuals on social media.
  - Social media posts which contain the words: **thoughts, alive, felt, therapy** should be flagged as cause for concern.
  - Notify the family so they can offer care and support.

- ## FUTURE WORK

  - Rework modeling flow and explore more possible model options.
  - Collect more data such as from **comment** as well as from other platforms.
  - Use superlative NLP model such as **BERT.**

Depression

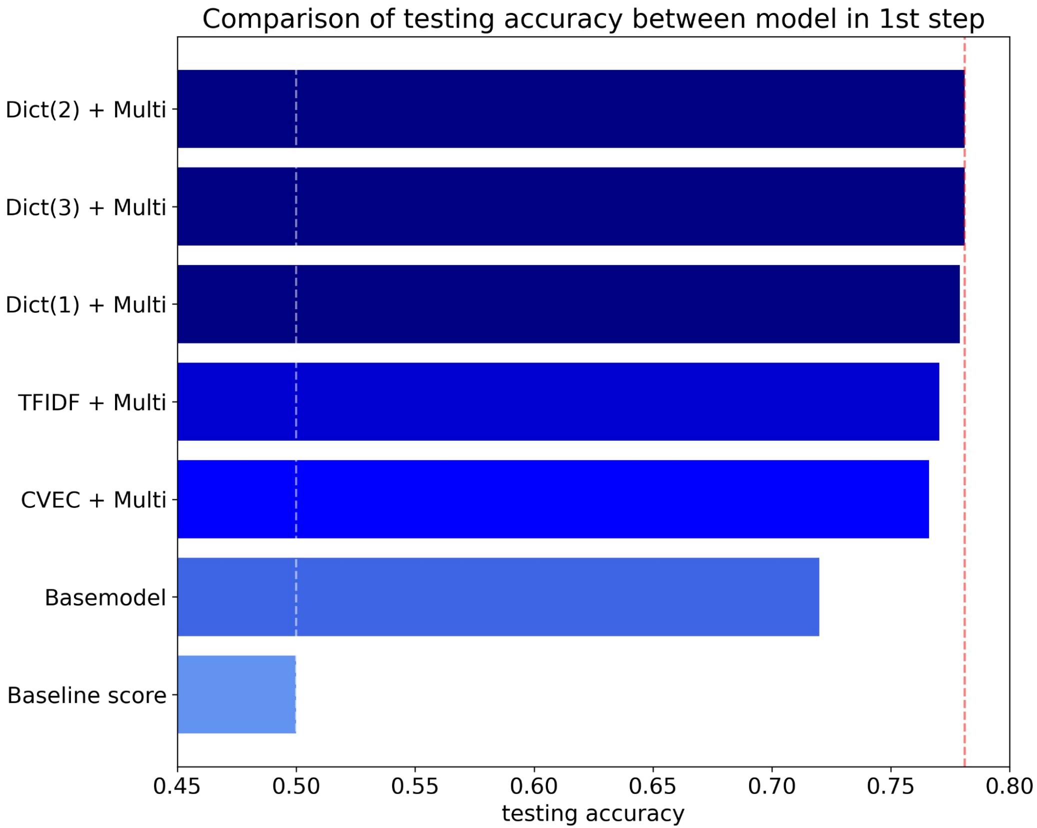
Because nobody should be alone in a dark place

“Life, there is always tomorrow.”

Thank you for your attention.

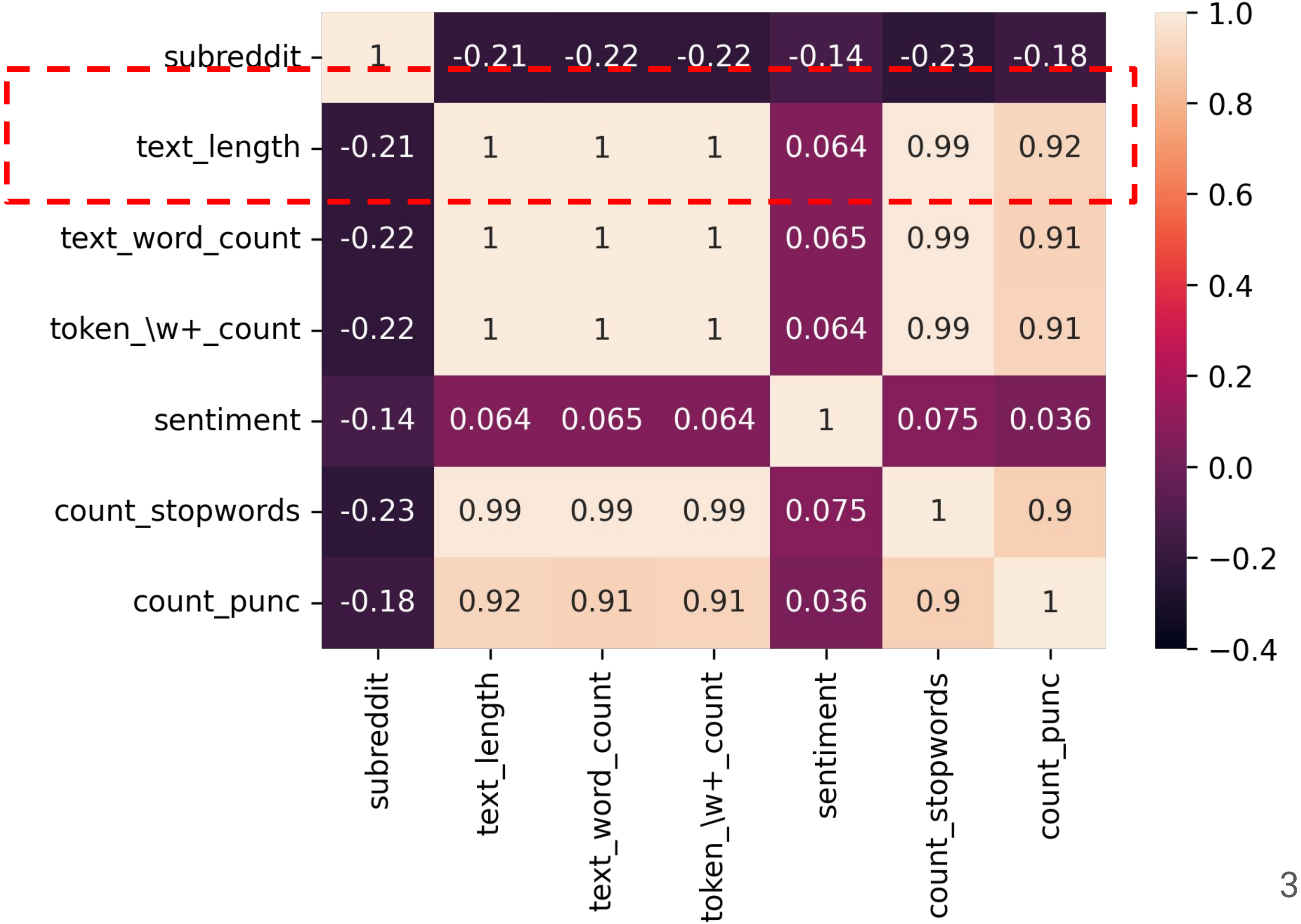
Q&A

APPENDIX



- Top 5 accuracy order is different from recall score, since there are an overfitting occur, even the predicted correct of the venting person tend to increase but we're focusing the to reduce the number of false negative as much as we can . So, we give up accuracy at this point.

APPENDIX



subreddit
text_length
text_word_count
token_\w+_count
sentiment
count_stopwords
count_punc
1 -0.21 -0.22 -0.22 -0.14 -0.23 -0.18
-0.21 1 1 1 0.064 0.99 0.92
-0.22 1 1 1 0.065 0.99 0.91
-0.22 1 1 1 0.064 0.99 0.91
-0.14 0.064 0.065 0.064 1 0.075 0.036
-0.23 0.99 0.99 0.99 0.075 1 0.9
-0.18 0.92 0.91 0.91 0.036 0.9 1
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4

	index	accuracy	precision	recall	f1_score	tn	fp	fn	tp	name_model
0	0	0.688841	0.692607	0.729508	0.710579	143	79	66	178	LogisticRegression(penalty='none')
1	1	0.680258	0.713004	0.651639	0.680942	158	64	85	159	RandomForestClassifier()
2	2	0.671674	0.673004	0.72541	0.698225	136	86	67	177	AdaBoostClassifier()
3	3	0.766094	0.747253	0.836066	0.789168	153	69	40	204	MultinomialNB()
4	0	0.682403	0.706897	0.672131	0.689076	154	68	80	164	LogisticRegression(penalty='none')
5	1	0.718884	0.7251	0.745902	0.735354	153	69	62	182	RandomForestClassifier()
6	2	0.665236	0.677419	0.688525	0.682927	142	80	76	168	AdaBoostClassifier()
7	3	0.781116	0.757246	0.856557	0.803846	155	67	35	209	MultinomialNB()
8	0	0.716738	0.733333	0.721311	0.727273	158	64	68	176	LogisticRegression(penalty='none')
9	1	0.67382	0.678295	0.717213	0.697211	139	83	69	175	(DecisionTreeClassifier(max_features='sqrt', r...
10	2	0.67382	0.67037	0.741803	0.70428	133	89	63	181	(DecisionTreeClassifier(max_depth=1, random_st...
11	3	0.77897	0.778656	0.807377	0.792757	166	56	47	197	MultinomialNB()
12	4	0.716738	0.733333	0.721311	0.727273	158	64	68	176	LogisticRegression(penalty='none')
13	5	0.678112	0.682171	0.721311	0.701195	140	82	68	176	(DecisionTreeClassifier(max_features='sqrt', r...
14	6	0.660944	0.666667	0.704918	0.685259	136	86	72	172	(DecisionTreeClassifier(max_depth=1, random_st...
15	7	0.781116	0.78629	0.79918	0.792683	169	53	49	195	MultinomialNB()
16	8	0.699571	0.718487	0.70082	0.709544	155	67	73	171	LogisticRegression(penalty='none')
17	9	0.678112	0.683594	0.717213	0.7	141	81	69	175	(DecisionTreeClassifier(max_features='sqrt', r...
18	10	0.695279	0.693182	0.75	0.720472	141	81	61	183	(DecisionTreeClassifier(max_depth=1, random_st...
19	11	0.770386	0.784232	0.77459	0.779381	170	52	55	189	MultinomialNB()
20	12	0.654506	0.624625	0.852459	0.720971	97	125	36	208	BernoulliNB()

```
Model : GridSearchCV(cv=3,
             estimator=Pipeline(steps=[('tf',
                                        TfidfVectorizer(tokenizer=<__main__.StemTokenize object at 0x7fb1fd492880>)),
                                       ('nb', MultinomialNB())]),
             param_grid={'nb__alpha': [0.001, 0.1, 1, 10, 100],
                         'nb__fit_prior': [True, False], 'tf__max_df': [1.0],
                         'tf__max_features': [1200, 1500], 'tf__min_df': [1, 2],
                         'tf__ngram_range': [(1, 2), (2, 2)],
                         'tf__stop_words': ['english']},
             verbose=1)


Train Score:0.8260558339298497

Test  Score:0.7746781115879828


Model classification report:
              precision    recall  f1-score   support

           0       0.81      0.68      0.74       222
           1       0.75      0.86      0.80       244

    accuracy                           0.77       466
   macro avg       0.78      0.77      0.77       466
weighted avg       0.78      0.77      0.77       466
```

Kaggle Dataset

Source : [Sentiment140 dataset with 1.6 million tweets | Kaggle]()

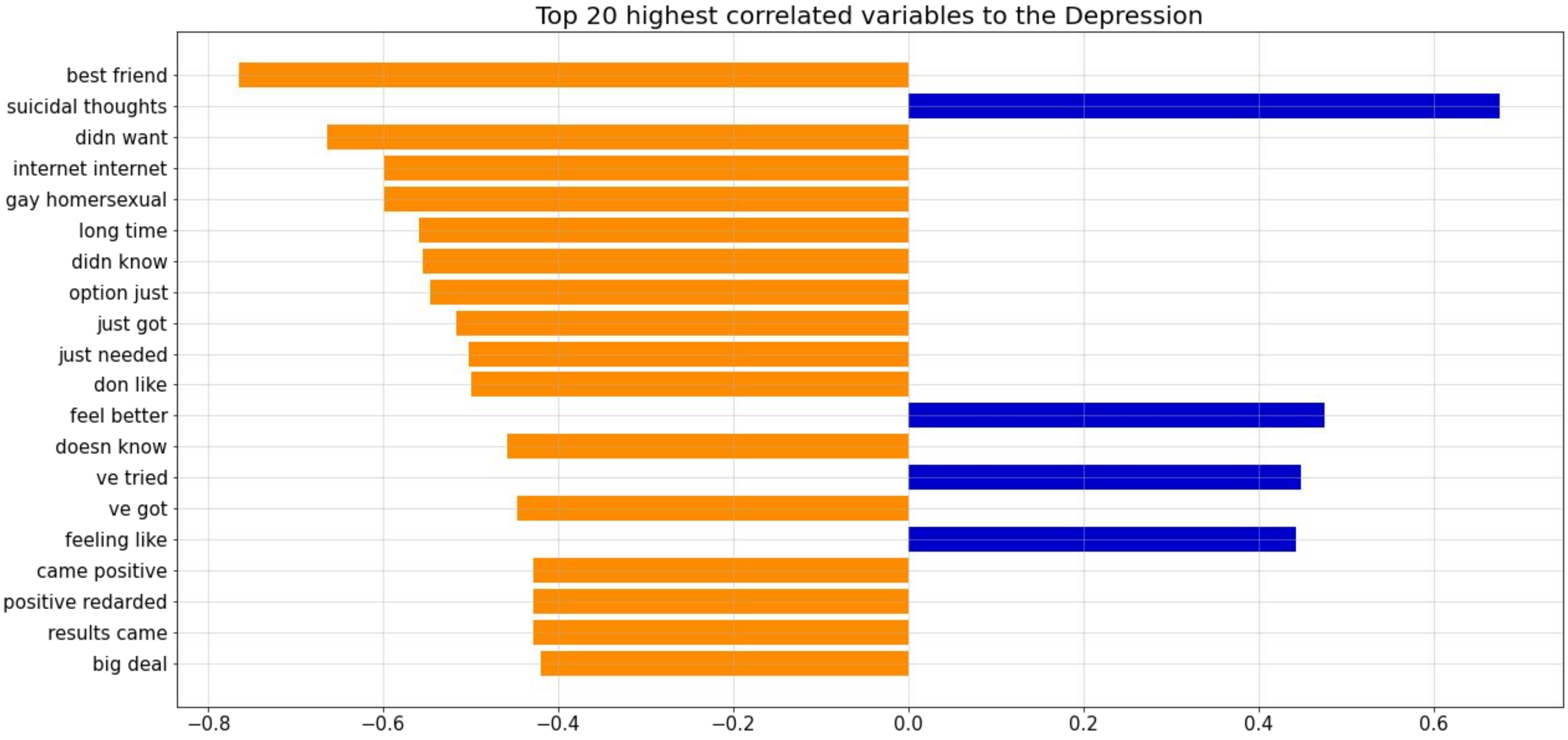
Context

This is the sentiment140 dataset. It contains 1,600,000 tweets extracted using the twitter api . The tweets have been annotated (0 = negative, 4 = positive) and they can be used to detect sentiment .

Content

It contains the following 6 fields:

1. **target**: the polarity of the tweet ($\boxed{\textit{0} = \text{negative,}}$ *2* = neutral, *4* = positive)
2. **ids**: The id of the tweet (*2087*)
3. **date**: the date of the tweet (*Sat May 16 23:58:44 UTC 2009*)
4. **flag**: The query (*lyx*). If there is no query, then this value is NO_QUERY.
5. **user**: the user that tweeted (*robotickilldozr*)
6. **text**: the text of the tweet (*Lyx is cool*)


Top 20 highest correlated variables to the Depression
best friend
suicidal thoughts
didn want
internet internet
gay homersexual
long time
didn know
option just
just got
just needed
don like
feel better
doesn know
ve tried
ve got
feeling like
came positive
positive redarded
results came
big deal
−0.8
−0.6
−0.4
−0.2
0.0
0.2
0.4
0.6