NLP project depression classifier

Duvivié Jasmin, Flor Nicolas, Pozdena Nicolas, Tiefengraber Marlis

github

https://github.com/smoothjass/this-is-depression

data

https://www.kaggle.com/datasets/infamouscoder/depression-reddit-cleaned

In [3]:

```
import pandas as pd
import numpy as np
import re
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import confusion_matrix, accuracy_score, f1_score, classification_report

import nltk, keras, string, html, math
from nltk.tokenize import word_tokenize, sent_tokenize
from nltk.corpus import stopwords, wordnet
from nltk.stem import WordNetLemmatizer
from sklearn.preprocessing import LabelEncoder
from collections import Counter, defaultdict
from keras.preprocessing.text import Tokenizer
from keras_preprocessing.sequence import pad_sequences
from gensim.parsing.preprocessing import remove_stopwords
from gensim.models import Word2Vec
from tensorflow.keras import layers
nltk.download('punkt')
nltk.download('wordnet')

import matplotlib.pyplot as plt
from wordcloud import WordCloud
nltk.download('stopwords')
```

```
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\nickb\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package wordnet to
[nltk_data]     C:\Users\nickb\AppData\Roaming\nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\nickb\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

Out[3]:

True

In [2]:

```
pd.set_option('display.max_colwidth', None)
```

In [3]:

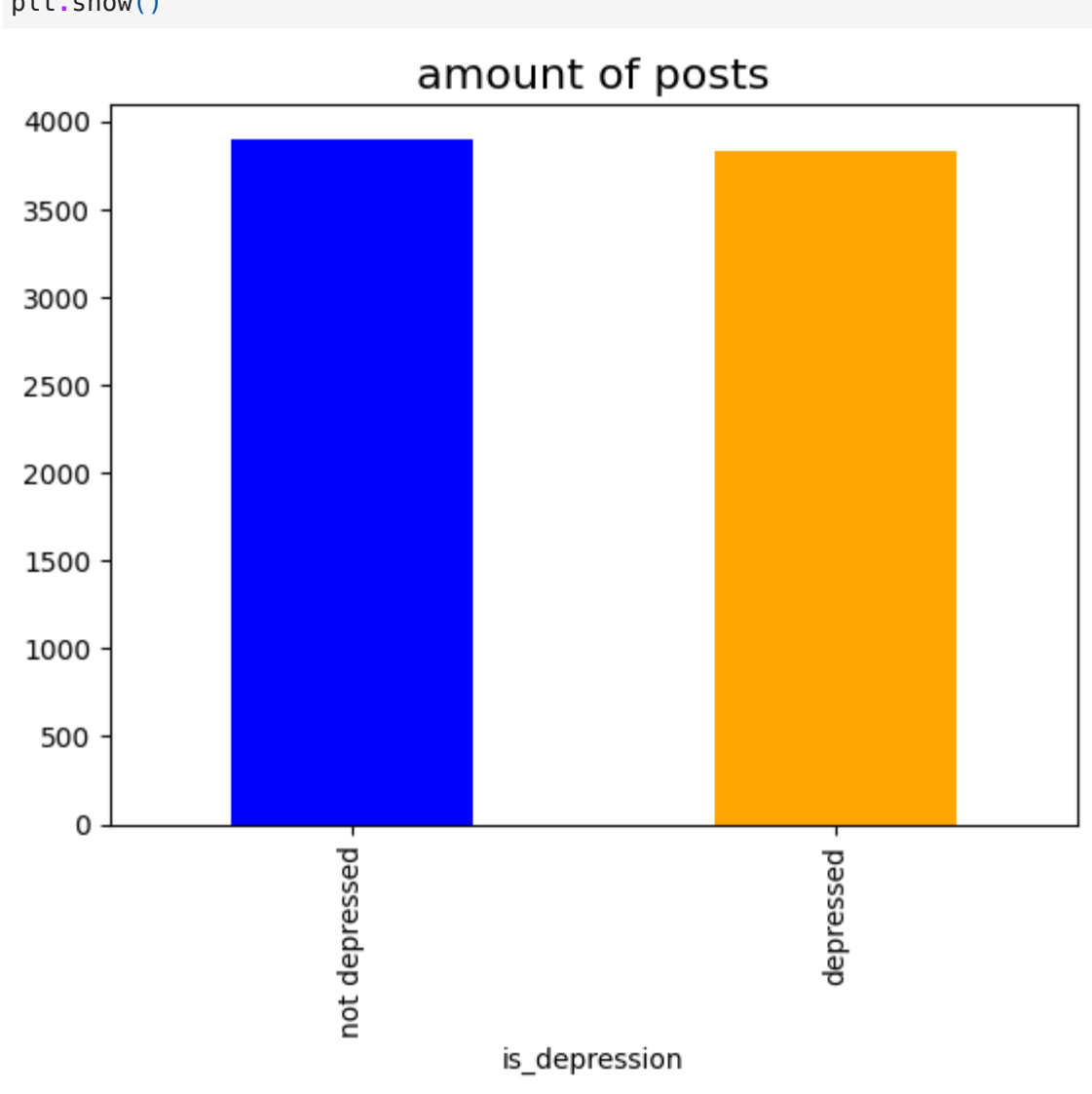
```
# read data, assign column names, skip header
data = pd.read_csv("depression_dataset_reddit_cleaned.csv", names=["text", "is_depression"], header=0)
```

Visualiziation of data

not depressed = blue | depressions = orange

In [4]:

```
counts = data["is_depression"].map({0: "not depressed", 1: "depressed"}).value_counts()

counts.plot(kind='bar', color=['blue', 'orange'])
plt.title("amount of posts", fontsize=16)

plt.show()
```



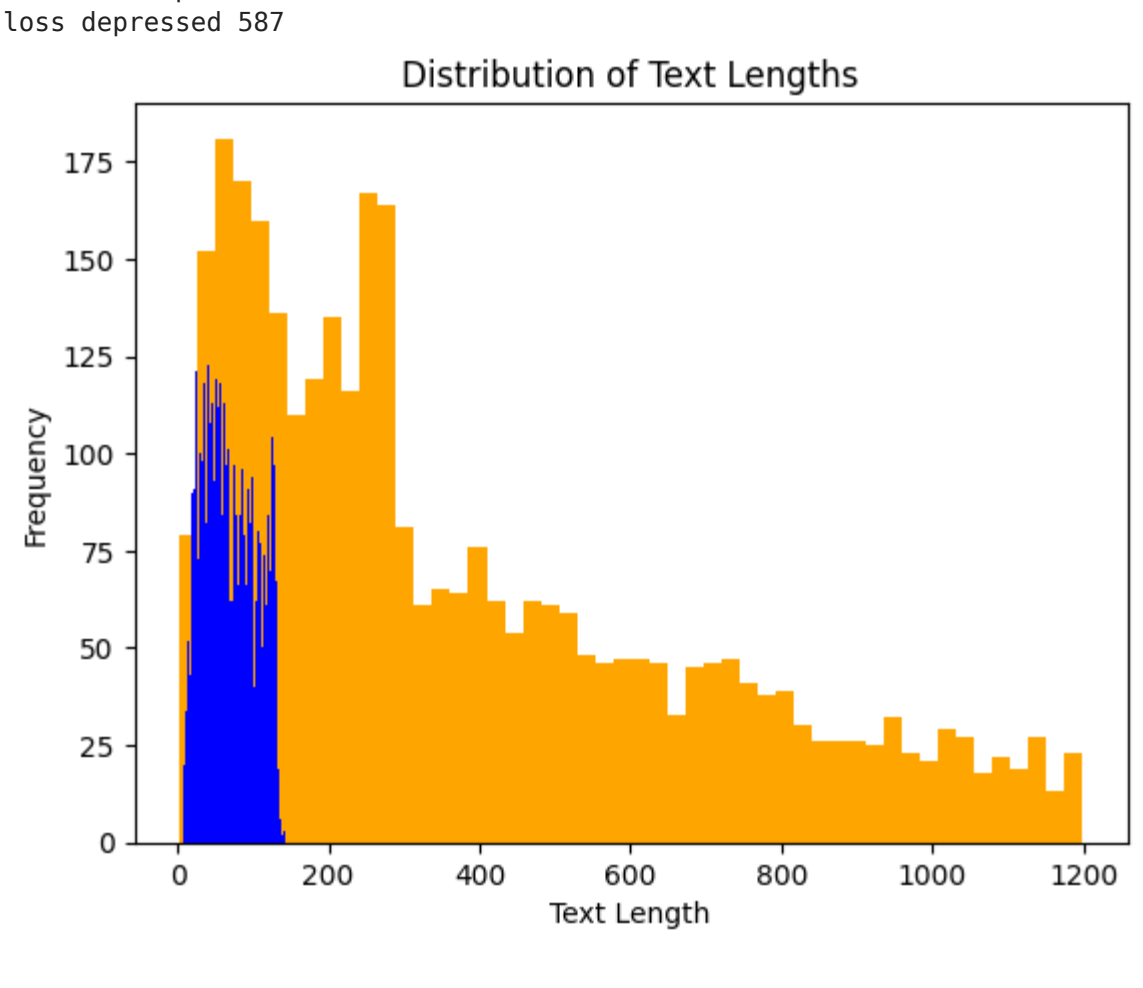
the data is evenly distributed

Comparing the lenght of the texts

In [5]:

```
data["length"] = data["text"].apply(len)

notDepressed = data[data["is_depression"] == 0]
depressed = data[data["is_depression"] == 1]

nD = len(notDepressed)
d = len(depressed)

# for better visualization
notDepressed = notDepressed[notDepressed["length"] < 1200]
depressed = depressed[depressed["length"] < 1200]

# how many posts did we lose
print(f"loss not depressed {nD - len(notDepressed)}")
print(f"loss depressed {d - len(depressed)}")

plt.hist(depressed['length'], color='orange', bins=50)
plt.hist(notDepressed['length'],color='blue', bins=50)



plt.title('Distribution of Text Lengths')
plt.xlabel('Text Length')
plt.ylabel('Frequency')
plt.show()
```

```
loss not depressed 0
loss depressed 587
```



Preprocessing

Removing Stop Words => keeping important words

In [6]:

```
def removingStopWords(text):
    words = text.split(' ')
    valuedWords = [word for word in words if word not in stopwords.words('english')]
    return valuedWords

data["valueWords"] = data["text"].apply(removingStopWords)
data["valueLength"] = data["valueWords"].apply(len)

data
```

Out[6]:

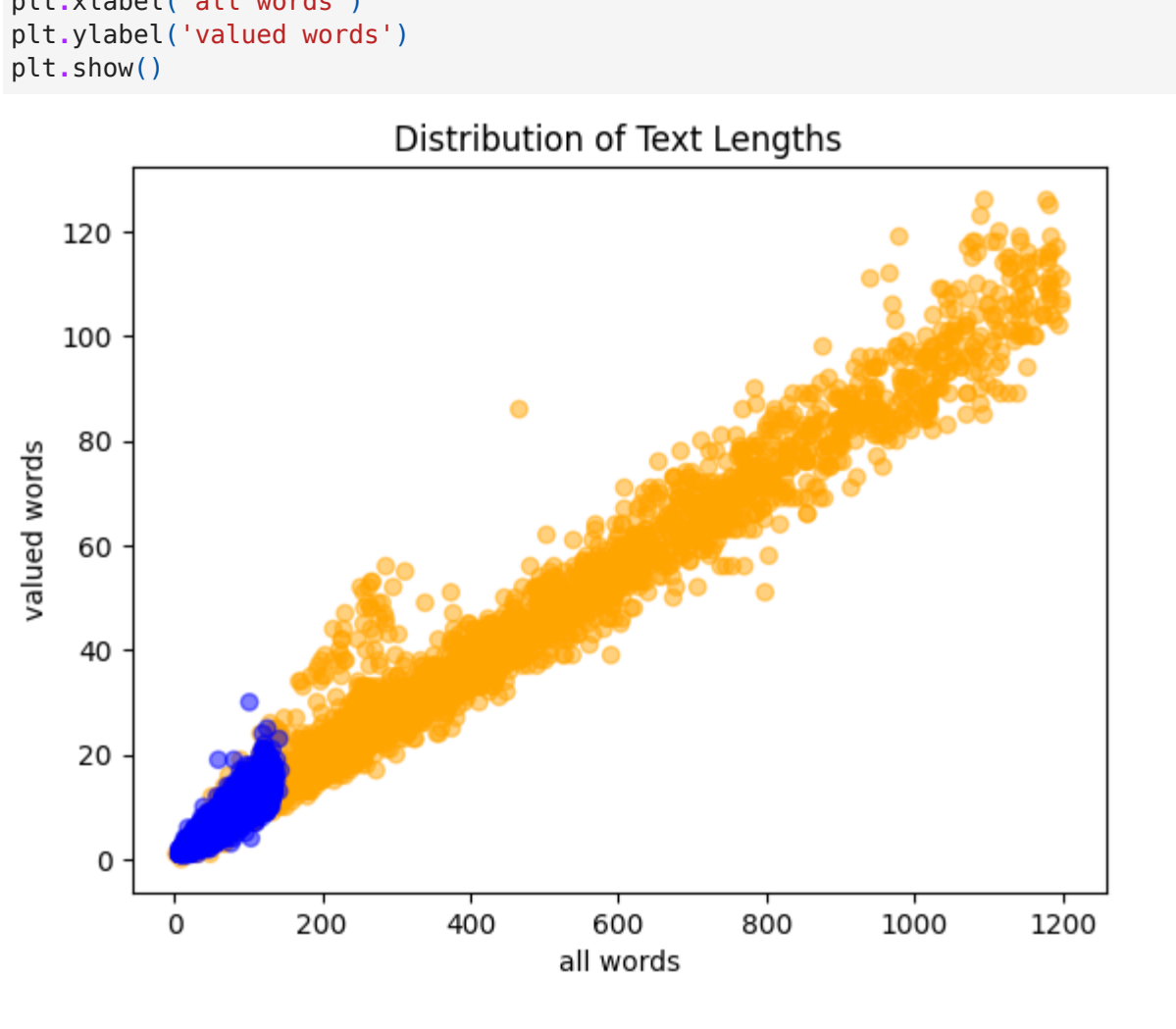
	text	is_depression	length	valueWords	valueLength
0	we understand that most people who reply immediately to an op with an invitation to talk privately mean only to help but this type of response usually lead to either disappointment or disaster it usually work out quite differently here than when you say pm me anytime in a casual social context we have huge admiration and appreciation for the goodwill and good citizenship of so many of you who support others here and flag inappropriate content even more so because we know that so many of you are struggling yourselves we re hard at work behind the scene on more information and resource to make it easier to give and get quality help here this is just a small start our new wiki page explains in detail why it s much better to respond in public comment at least until you ve gotten to know someone it will be maintained at r depression wiki private contact and the full text of the current version is below summary anyone who while acting a a helper invite or accepts private contact i e pm chat or any kind of offsite communication early in the conversion is showing either bad intention or bad judgement either way it s unwise to trust them pm me anytime seems like a kind and generous offer and it might be perfectly well meaning but unless and until a solid rapport ha been established it s just not a wise idea here are some point to consider before you offer or accept an invitation to communicate privately by posting supportive reply publicly you ll help more people than just the op if your response are of good quality you ll educate and inspire other helper the 9 90 rule http en wikipedia org wiki rule internet culture applies here a much a it doe anywhere else on the internet people who are struggling with serious mental health issue often justifiably have a low tolerance for disappointment and a high level of ever changing emotional need unless the helper is able to make a 00 commitment to be there for them in every way for a long a necessary offering a personal inbox a a resource is likely to do more harm than good this is why mental health crisis line responder usually don t give their name and caller aren t allowed to request specific responder it s much healthier and safer for the caller to develop a relationship with the agency a a whole analogously it s much safer and healthier for our ops to develop a relationship with the community a a whole even trained responder are generally not allowed to work high intensity situation alone it s partly about availability but it s mostly about wider perspective and preventing compassion fatigue if a helper get in over their head with someone whose mental health issue including suicidality which is often comorbid with depression escalate in a pm conversation it s much harder for others including the r depression and r suicidewatch moderator to help contrary to common assumption moderator can t see or police pm in our observation over many year the people who say pm me the most are consistently the one with the least understanding of mental health issue and mental health support we all have gap in our knowledge and in our ability to communicate effectively community input mitigates these limitation there s no reason why someone who s truly here to help would want to hide their response from community scrutiny if helper are concerned about their own privacy keep in mind that self disclosure when used supportively is more about the feeling than the detail and that we have no problem here with the use of alt throwaway account and have no restriction on account age or karma we all know the internet is used by some people to exploit or abuse others these people do want to hide their deceptive and manipulative response from everyone except their victim there are many of them who specifically target those who are vulnerable because of mental health issue if a helper invite an op to talk privately and give them a good supportive experience they ve primed that person to be more vulnerable to abuser this sort of cognitive priming tends to be particularly effective when someone s in a state of mental health crisis when people rely more on heuristic than critical reasoning if ops want to talk privately posting on a wide open anonymous forum like reddit might not be the best option although we don t recommend it we do allow ops to request private contact when asking for support if you want to do this please keep your expectation realistic and to have a careful look at the history of anyone who offer to pm before opening up to them	1	4535	[understand, people, reply, immediately, op, invitation, talk, privately, mean, help, type, response, usually, lead, either, disappointment, disaster, usually, work, quite, differently, say, pm, anytime, casual, social, context, huge, admiration, appreciation, goodwill, good, citizenship, many, support, others, flag, inappropriate, content, even, know, many, struggling, hard, work, behind, scene, information, resource, make, easier, give, get, quality, help, small, start, new, wiki, page, explains, detail, much, better, respond, public, comment, least, gotten, know, someone, maintained, r, depression, wiki, private, contact, full, text, current, version, summary, anyone, acting, helper, invite, accepts, private, contact, e, pm, chat, kind, offsite, communication, early, conversion, showing, either, bad, ...]	419
1	welcome to r depression s check in post a place to take a moment and share what is going on and how you are doing if you have an accomplishment you want to talk about these shouldn t be standalone post in the sub a they violate the role model rule but are welcome here or are having a tough time but prefer not to make your own post this is a place you can share our subreddit rule are located in the sidebar you can also always access them at r depression about rule since all of them exist for important safety reason we ask everyone here to read and follow them please click report on any harmful content you see here we always want to know and deal a soon a we can we also have several wikis there for help with finding and giving support r depression wiki what is depression provides guidance about what is and isn t a depressive disorder guidance on the complex nature of the illness that are usually grouped under the depression label and redirect information for common off topic issue r depression wiki giving help offer information on the nature and value of peer support for mental health issue in general and lot of guidance for learning what is and isn t usually helpful in giving peer support ysk that the type of rule violation that we most frequently see interfering with people getting safe and relevant support here are people breaking the private contact rule you should never trust anyone who try to get you into a private conversation in response to a post here see r depression wiki private contact i m here to help post this show that you don t understand the most basic principle of peer support especially selectivity the giving help wiki explains more about this role modelling i e achievement or advice post this is an expert free zone that s what peer support mean rule we know that internet culture celebrate not just bragging about your achievement but bragging about your good intention nothing like that is ever acceptable here content that s more about making a statement or casually polling the sub than seeking personal support or in a comment giving it rule and 0 off topic post about difficult situation or circumstance including interpersonal loss grief sadness anger and other difficult emotion are not mental illness the what is depression wiki ha suggestion for other place to post about these issue which are 00 valid and serious but inappropriate here	1	2394	[welcome, r, depression, check, post, place, take, moment, share, going, accomplishment, want, talk, standalone, post, sub, violate, role, model, rule, welcome, tough, time, prefer, make, post, place, share, subreddit, rule, located, sidebar, also, always, access, r, depression, rule, since, exist, important, safety, reason, ask, everyone, read, follow, please, click, report, harmful, content, see, always, want, know, deal, soon, also, several, wikis, help, finding, giving, support, r, depression, wiki, depression, provides, guidance, depressive, disorder, guidance, complex, nature, illness, usually, grouped, depression, label, redirect, information, common, topic, issue, r, depression, wiki, giving, help, offer, information, nature, value, peer, support, mental, health, issue, ...]	220
2	anyone else instead of sleeping more when depressed stay up all night to avoid the next day from coming sooner may be the social anxiety in me but life is so much more peaceful when everyone else is asleep and not expecting thing of you	1	236	[anyone, else, instead, sleeping, depressed, stay, night, avoid, next, day, coming, sooner, may, social, anxiety, life, much, peaceful, everyone, else, asleep, expecting, thing]	23
3	i ve kind of stuffed around a lot in my life delaying the inevitable of having to work a job and be a responsible adult and i m but the longest i ve ever held a job wa 9 month it wasn t that i m lazy i wa always doing other thing i enjoy but i know now unemployment ha caused most of my depression recently i just feel utterly hopeless when i think soon enough i ll have to move out on my own in some shitty house working a job i couldn t care le about to me it just seems like the perfect recipe to depression	1	510	[kind, stuffed, around, lot, life, delaying, inevitable, work, job, responsible, adult, longest, ever, held, job, wa, 9, month, lazy, wa, always, thing, enjoy, know, unemployment, ha, caused, depression, recently, feel, utterly, hopeless, think, soon, enough, move, shitty, house, working, job, care, le, seems, like, perfect, recipe, depression]	47
4	sleep is my greatest and most comforting escape whenever i wake up these day the literal very first emotion i feel is just misery and reminding myself of all my problem i can t even have a single second to myself it s like waking up everyday is just welcoming yourself back to hell	1	281	[sleep, greatest, comforting, escape, whenever, wake, day, literal, first, emotion, feel, misery, reminding, problem, even, single, second, like, waking, everyday, welcoming, back, hell]	23
...	...	...	...	...	...
7726	is that snow	0	12	[snow]	1
7727	moulin rouge mad me cry once again	0	34	[moulin, rouge, mad, cry]	4
7728	trying to shout but can t find people on the list	0	49	[trying, shout, find, people, list]	5
7729	ughh can t find my red sox hat got ta wear this creepy nick pirro version	0	73	[ughh, find, red, sox, hat, got, ta, wear, creepy, nick, pirro, version]	12
7730	slept wonderfully finally tried swatching for new project classic line cardi from stash but don t like color must wait for now	0	126	[slept, wonderfully, finally, tried, swatching, new, project, classic, line, cardi, stash, like, color, must, wait]	15

7731 rows × 5 columns

Text Lengths vs. Valued Words

In [7]:

```
words = data[data["length"] < 1200]

colorMap = ["blue", "orange"]
colors = [colorMap[val] for val in words["is_depression"]]

plt.scatter(words["length"],words["valueLength"], color=colors, marker='o', alpha=0.5)


plt.title('Distribution of Text Lengths')
plt.xlabel('all words')
plt.ylabel('valued words')
plt.show()
```



In [8]:

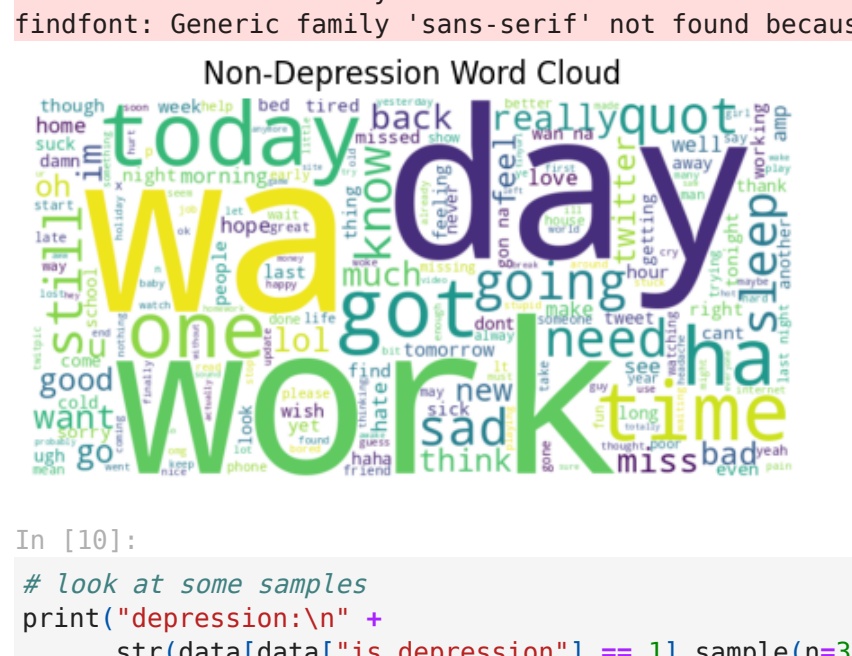
```
data['valueText'] = data["valueWords"].apply(lambda  x: ' '.join(x))
```

Creating word clouds to show most frequently occurring words

In [9]:

```
ndText = ' '.join(data['valueText'].loc[data['is_depression']==0])
dText = ' '.join(data['valueText'].loc[data['is_depression']==1])

# Create WordCloud objects for each category
ndWordcloud = WordCloud(width=800, height=400, background_color='white').generate(ndText)
dWordcloud = WordCloud(width=800, height=400, background_color='white').generate(dText)


plt.rcParams['font.sans-serif'] = 'Arial'


# Display the non-depression word cloud
plt.figure(figsize=(12, 6))
plt.subplot(1, 2, 1)
plt.imshow(ndWordcloud, interpolation='bilinear')
plt.title('Non-Depression Word Cloud')
plt.axis('off')

# Display the depression word cloud
plt.subplot(1, 2, 2)
plt.imshow(dWordcloud, interpolation='bilinear')
plt.title('Depression Word Cloud')
plt.axis('off')

plt.show()
```

```
findfont: Generic family 'sans-serif' not found because none of the following families were found: Arial
findfont: Generic family 'sans-serif' not found because none of the following families were found: Arial
findfont: Generic family 'sans-serif' not found because none of the following families were found: Arial
findfont: Generic family 'sans-serif' not found because none of the following families were found: Arial
findfont: Generic family 'sans-serif' not found because none of the following families were found: Arial
findfont: Generic family 'sans-serif' not found because none of the following families were found: Arial
findfont: Generic family 'sans-serif' not found because none of the following families were found: Arial
findfont: Generic family 'sans-serif' not found because none of the following families were found: Arial
findfont: Generic family 'sans-serif' not found because none of the following families were found: Arial
findfont: Generic family 'sans-serif' not found because none of the following families were found: Arial
findfont: Generic family 'sans-serif' not found because none of the following families were found: Arial
findfont: Generic family 'sans-serif' not found because none of the following families were found: Arial
findfont: Generic family 'sans-serif' not found because none of the following families were found: Arial
findfont: Generic family 'sans-serif' not found because none of the following families were found: Arial
findfont: Generic family 'sans-serif' not found because none of the following families were found: Arial
findfont: Generic family 'sans-serif' not found because none of the following families were found: Arial
findfont: Generic family 'sans-serif' not found because none of the following families were found: Arial
findfont: Generic family 'sans-serif' not found because none of the following families were found: Arial
```



In [10]:

```
# look at some samples
print("depression:\n" +
      str(data[data["is_depression"] == 1].sample(n=3).text.tolist()))
print("not depression:\n" +
      str(data[data["is_depression"] == 0].sample(n=3).text.tolist()))
```

```
depression:
['don t let depression slow you down we got this', 'it s been five year and a couple day now since my first m
ajor attempt i m not too sure how i feel to be honest ha life improved well not really of course after that f
irst attempt i tried a couple more time i think if we look at the net value i m relatively in the same spot a
i wa five year ago vibing on the bridge thankfully i m not too upset about that i think sadness and my inabil
ity to get out of said sadness is something that i will have to live with i m getting better at fighting it t
hat s for sure are some day tiring and fucking unbearable of course but now i m just content with existing sa
d day are hard some day are easy i think i m just shocked that i m still alive well i definitely know that yo
unger me would be shocked i m still kicking around maybe he d be happy that i stayed for this long there real
ly isn t any point to this post just a bit of selfish self congratulation and the ability to say i m still al
ive guy i hope everyone ha a good day it s raining where i am but it s really nice to look at', 'i feel a if
life is winning i simply dont want to be here anymore i cant tell if im loved by my parent because it certain
ly doesnt feel like it i feel like they just use me a a maid and thats just how ive been taught since i wa yo
unger so there not and reason to think that im unable to every week feel the same monday friday is homework w
ith study group and im struggling in class because i cant remember shit it not that i dont pay attention in c
lass it that i am mentally unable to rememebr shit i know back in high school i never had the problem then tw
o year of covid hit and now im aa dumb a rock in a engineering major ive wanted to switch so many time every
semester but each time someone talk me out of it ive told my parent twoce already but they insist that i fini
sh school before trying to learn other thing i cant tell if my girlfriend love me anymore after the argument
we had a week or two ago about me playing clash of clan and not giving her attention even though i feel like
im doing everything i can to give her attention i cant sleep anymore and i just find myself staying up late a
nd repeating the cycle it not like im not trying to better myself either i go to the gym with my friend at sc
hool and ive been doing it for week now i just thought it would help me get into the right mindset and help m
e focus more on school and maybe bring me out of this year depressive episode ive been in since covid started
but nothing ive done work now im afraid that i have been pushing my gf and my close friend away and i just wa
nt to end it all i just want to jump off a bridge into oncoming traffic where i will have no chance of recove
ring i just want to go so badly but i cant im trapped not by others but by myself cry doesnt make me le of a
man right']
not depression:
['trying to find the motivation to write some essay and finding it sad that my life revolves around essay wri
ting', 'tried to fix hubby s computer it didn t come with disk and now window key won t work might have to bu
y it like we can afford that', 'drinking a nuun lemon amp lime electrolyte tab drink it s not sweet enough']
```

In [11]:

```
# class balance
print(data[data["is_depression"] == 1].count())
print(data[data["is_depression"] == 0].count())
```

```
text             3831
is_depression    3831
length           3831
valueWords       3831
valueLength      3831
valueText        3831
dtype: int64
text             3900
is_depression    3900
length           3900
valueWords       3900
valueLength      3900
valueText        3900
dtype: int64
```

Looking at the length of the texts

In [12]:

```
# the samples suggest, the is_depression texts are longer than the others. Is that really so?
# print average length, min length and max length of elements shows that length could be an important factor
print("depression:\n"
      "avg: " + str(sum(map(len, data[data["is_depression"] == 1].text))/float(len(data[data["is_depression"]
      "min: " + str(min(map(len, data[data["is_depression"] == 1].text))) + "\n" +
      "max: " + str(max(map(len, data[data["is_depression"] == 1].text))) + "\n"
      )
print("not depression:\n"
      "avg: " + str(sum(map(len, data[data["is_depression"] == 0].text))/float(len(data[data["is_depression"]
      "min: " + str(min(map(len, data[data["is_depression"] == 0].text))) + "\n" +
      "max: " + str(max(map(len, data[data["is_depression"] == 0].text))) + "\n"
      )
```

```
depression:
avg: 658.299138606108
min: 3
max: 19822

not depression:
avg: 70.97572435395458
min: 7
max: 144
```

pre-process and explore some more

In [14]:

```
def preprocess(text):
    # remove extra blanks
    re.sub(r'\t{2,}', ' ', text)
    # lowercase
    text = text.lower()
    # TODO other stuff
    text = remove_stopwords(text)
    text = word_tokenize(text)
    return text

def preprocess_extensive(text):
     # remove extra blanks
    re.sub(r'\t{2,}', ' ', text)
    # remove numbers
    text = ''.join(i for i in text if not i.isdigit())
    # lowercase
    text = text.lower()
    # remove stopwords with gensim
    text = remove_stopwords(text)
    # tokenize
    text = word_tokenize(text)
    # lemmatize tokens
    lemmatizer = WordNetLemmatizer()
    lemm_text = [lemmatizer.lemmatize(word) for word in text]
    text = ' '.join(lemm_text)
    return text

data["text"] = data["text"].apply(preprocess_extensive)
```

finding the most common words

In [15]:

```
word_list = [word_tokenize(text) for text in data["text"]]
# Flatten the list of lists into a single list of words (excluding words shorter that 3 chars as many single
flat_word_list = [word for sublist in word_list for word in sublist if len(word) > 2]
# Use Counter to count the occurrences of each word
word_counts = Counter(flat_word_list)
# Get the 10 most common words
most_common_words = word_counts.most_common(10)
com_words_df = pd.DataFrame(most_common_words, columns=['Word', 'Count'])
print(com_words_df)
```

```
         Word  Count
0        like   3584
1        feel   3227
2        want   2292
3        know   2283
4        life   2147
5        time   2070
6         day   1688
7      people   1660
8  depression   1569
9     anxiety   1548
```

In [16]:

```
# try word2vec on data
# Build Word2Vec model
model = Word2Vec(sentences=word_list, vector_size=50, window=3, min_count=1, workers=4)
```

In [17]:

```
# Find the top 10 words most similar to "depression"
similar_words = model.wv.most_similar("depression", topn=10)
sim_words_df = pd.DataFrame(similar_words, columns=['Word', 'Similarity'])
print(sim_words_df)
```

```
        Word  Similarity
0  theekween    0.969143
1     social    0.964523
2      dozen    0.958801
3   physical    0.957774
4      issue    0.954665
5   disorder    0.953881
6     severe    0.951071
7     trauma    0.948063
8     stress    0.947296
9    symptom    0.944013
```

feature extraction

bag of words

In [18]:

```
# vectorizer does tokenization, data already lowercased
# https://scikit-learn.org/stable/modules/feature_extraction.html
# https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html#skle
X = data['text'].to_numpy()
y = data['is_depression'].to_numpy()
# Bag of words
vectorizer = CountVectorizer(analyzer='word', ngram_range=(1, 1))
X_BOW = vectorizer.fit_transform(X)
```

splitting test & train

In [19]:

```
X_train, X_test, y_train, y_test = train_test_split(X_BOW, y, stratify=y)
```

train and predict with naive bayes

In [20]:

```
# train
# https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.MultinomialNB.html#sklearn.naive_baye
clf = MultinomialNB()
clf.fit(X_train, y_train)
# predict
y_hat = clf.predict(X_test)
```

evaluation of the prediction

In [21]:

```
# https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html
# https://scikit-learn.org/stable/modules/generated/sklearn.metrics.accuracy_score.html#sklearn.metrics.accur
# https://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html#sklearn.metrics.f1_score
tn, fp, fn, tp = confusion_matrix(y_test, y_hat).ravel()
accuracy = accuracy_score(y_test, y_hat)
f1 = f1_score(y_test, y_hat, zero_division=1.0)
print(tn, fp, fn, tp, accuracy, f1)
```

```
763 212 54 904 0.8623900672529746 0.8717454194792672
```

TF-IDF Vectorization & Multinomial Naive Bayes

In [22]:

```
# https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfTransformer.html#skl
# encode
vectorizer = TfidfVectorizer()
X_TFIDF = vectorizer.fit_transform(X)
# split
X_train, X_test, y_train, y_test = train_test_split(X_TFIDF, y, stratify=y)
# train
clf = MultinomialNB()
clf.fit(X_train, y_train)
# predict
y_hat = clf.predict(X_test)
# evaluate
tn, fp, fn, tp = confusion_matrix(y_test, y_hat).ravel()
accuracy = accuracy_score(y_test, y_hat)
f1 = f1_score(y_test, y_hat, zero_division=1.0)
print(tn, fp, fn, tp, accuracy, f1)
```

```
731 244 22 936 0.8623900672529746 0.8755846585594013
```

both are very similar
F1 score little bit better using TF-IDF
accuracy is exactly the same

a little experiment

because the text length was so different in the two classes

In [23]:

```
# How well would a model perform, which is only trained on length of text?
X = data["text"].apply(len).to_numpy().reshape(-1,1)
y = data["is_depression"].to_numpy()
# train test split
X_train, X_test, y_train, y_test = train_test_split(X, y, stratify=y)
# train
clf = MultinomialNB()
clf.fit(X_train, y_train)
# predict
y_hat = clf.predict(X_test)
# evaluate
tn, fp, fn, tp = confusion_matrix(y_test, y_hat).ravel()
accuracy = accuracy_score(y_test, y_hat)
f1 = f1_score(y_test, y_hat, zero_division=1.0)
# it just thinks everything is not depression. so 50%. works horrible.
# we could try adding some more features if we feel like it, but not right now.
print(tn, fp, fn, tp, accuracy, f1)
print(np.unique(y_hat, return_counts=True))
```

```
975 0 958 0 0.5043973098810139 0.0
(array([0]), array([1933]))
```

This predicts everything as not depression
Accuracy 50% => random guessing\

LSTM

using this resource https://towardsdatascience.com/naive-bayes-and-lstm-based-classifier-models-63d521a48c20

In [26]:

```
X = data['text'].to_numpy()
y = data['is_depression'].to_numpy()
# train test split
X_train, X_test, y_train, y_test = train_test_split(X, y, stratify=y)
```

Prepering the data for NN

In [27]:

```
# Hyperparameters of the model
oov_tok = '<OOK>'
embedding_dim = 100
max_length = 150
padding_type='post'
trunc_type='post'

# tokenizes sentences
tokenizer = Tokenizer()
tokenizer.fit_on_texts(X_train)

# vocabulary size
word_index = tokenizer.word_index
vocab_size = len(tokenizer.word_index) + 1

# converts train dataset to sequence and pads sequences
train_sequences = tokenizer.texts_to_sequences(X_train)
train_padded = pad_sequences(train_sequences, padding='post', maxlen=max_length)

# converts Test dataset to sequence and pads sequences
test_sequences = tokenizer.texts_to_sequences(X_test)
test_padded = pad_sequences(test_sequences, padding='post', maxlen=max_length)
```

initialize sequential model with multiple layers

In [28]:

```
model = keras.Sequential([
    keras.layers.Embedding(vocab_size, embedding_dim, input_length=max_length),
    keras.layers.Bidirectional(keras.layers.LSTM(64)),
    keras.layers.Dense(24, activation='relu'),
    keras.layers.Dense(1, activation='sigmoid')
])

# compiles model
model.compile(loss='binary_crossentropy',
              optimizer='adam',
              metrics=['accuracy'])

# model summary
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, 150, 100)          1566500

 bidirectional (Bidirection  (None, 128)               84480
 al)

 dense (Dense)               (None, 24)                3096

 dense_1 (Dense)             (None, 1)                 25

=================================================================
Total params: 1654101 (6.31 MB)
Trainable params: 1654101 (6.31 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

training the model

In [29]:

```
num_epochs = 5
history = model.fit(train_padded, y_train,
                    epochs=num_epochs, verbose=1,
                    validation_split=0.1)
```

```
Epoch 1/5
164/164 [==============================] - 21s 112ms/step - loss: 0.2663 - accuracy: 0.8821 - val_loss: 0.128
2 - val_accuracy: 0.9603
Epoch 2/5
164/164 [==============================] - 18s 107ms/step - loss: 0.0651 - accuracy: 0.9780 - val_loss: 0.156
1 - val_accuracy: 0.9448
Epoch 3/5
164/164 [==============================] - 18s 111ms/step - loss: 0.0302 - accuracy: 0.9895 - val_loss: 0.152
2 - val_accuracy: 0.9569
Epoch 4/5
164/164 [==============================] - 20s 119ms/step - loss: 0.0126 - accuracy: 0.9950 - val_loss: 0.183
6 - val_accuracy: 0.9448
Epoch 5/5
164/164 [==============================] - 18s 108ms/step - loss: 0.0068 - accuracy: 0.9971 - val_loss: 0.221
0 - val_accuracy: 0.9448
```

very high accuracy towards the end

validation accuracy stays around the same

validation loss gets a bit larger

Model Evaluation & Performance Metrics

In [30]:

```
# Gets probabilities
prediction = model.predict(test_padded)
print("The probabilities are - ", prediction, sep='\n')

# Gets labels based on probability 1 if p>= 0.5 else 0
for each in prediction:
    if each[0] >=0.5:
        each[0] = 1
    else:
        each[0] = 0
prediction = prediction.astype('int32')
print("\nThe labels are - ", prediction, sep='\n')

# Calculates accuracy on Test data
print("\nThe accuracy of the model is ", accuracy_score(y_test, prediction))
print("\nThe accuracy and other metrics are \n", classification_report(y_test, prediction, labels=[0, 1]),sep
```

```
61/61 [==============================] - 3s 23ms/step
The probabilities are -
[[3.3173394e-05]
 [1.1964569e-04]
 [3.4760064e-01]
 ...
 [9.9993342e-01]
 [9.9996924e-01]
 [4.0248659e-01]]

The labels are -
[[0]
 [0]
 [0]
 ...
 [1]
 [1]
 [0]]

The accuracy of the model is  0.9368856699430936

The accuracy and other metrics are

              precision    recall  f1-score   support

           0       0.94      0.93      0.94       975
           1       0.93      0.94      0.94       958

    accuracy                           0.94      1933
   macro avg       0.94      0.94      0.94      1933
weighted avg       0.94      0.94      0.94      1933
```

all accuracies are very high with around 94%

=> identifies both depression & non depression posts very good

Model Prediction on New Sentences

In [31]:

```
# trying some recent posts I am getting from r/depression and non depression related subreddits
sentence = ["Our most-broken and least-understood rules is \"helpers may not invite private contact as a firs
            "Idk why but everyone seems depressed these days (including me).. Is it a phase that everyone goe
            "For everyone that needs to hear it I love you no matter what and just keep up the hard work. Stu
            "Guys I have a 2 in 1 laptop, Fedora was working incredibly from a USB Flashdrive compared to oth

# converts to a sequence
test_sequences = tokenizer.texts_to_sequences(sentence)

# pads the sequence
test_padded = pad_sequences(test_sequences, padding='post', maxlen=max_length)

# Gets probabilities
prediction = model.predict(test_padded)
print("The probabilities are - ", prediction, sep='\n')

# Gets labels based on probability 1 if p>= 0.5 else 0
for each in prediction:
    if each[0] >=0.5:
        each[0] = 1
    else:
        each[0] = 0
prediction = prediction.astype('int32')
print("\nThe labels are - ", prediction, sep='\n')
```

```
1/1 [==============================] - 0s 33ms/step
The probabilities are -
[[6.3732604e-04]
 [9.9949104e-01]
 [9.4455516e-01]
 [9.6468029e-05]]

The labels are -
[[0]
 [1]
 [1]
 [0]]
```