Prototype 3

Note: this final prototype consists of only the most effective model (RFC), and featureset (Combination). To see the results of comparative testing between models and featuresets, please see Prototype 2

1. Preliminary Data Preperation

1.1 Data Import and Cleaning

The data cleaning and data reshape portions will be the same as prototype 2

First, install the neccesary libraries and import modules

In [ ]:
```
# !pip install pandas
# !pip install numpy
# !pip install textblob
# !pip install nltk
```

In [ ]:
```
import pandas as pd
import json
```

Read the input jsonl file to an intermediate dataframe.

This time, we will read the entire HC3 dataset

In [ ]:
```
# convert jsonl to dataframe
with open('all.jsonl', encoding='utf8') as f:
    # read lines from jsonl into an intermediate df
    lines = f.read().splitlines()
    df_inter = pd.DataFrame(lines)
    df_inter.columns = ['json_element']
```

In [ ]:
```
df_inter.sample(3)
```

Out[ ]:

	json_element
18385	{"question":"Please explain what is \"Multimod...
11325	{"question":"How can you explain that intellig...
7848	{"question":"Why do we forget our dreams ? I '...

Next, split the dataframe into its respective columns

In [ ]:
```
#normalise the intermediate dataframe to seperate the data into columns
df = pd.json_normalize(df_inter['json_element'].apply(json.loads))
```

In [ ]:
```
df.sample(3)
```

Out[ ]:

	question	human_answers	chatgpt_answers	index	source
9916	Richard Sherman I keep hearing about this guy ...	[He saved the NFC championship after being goo...	[Richard Sherman is a professional American fo...	NaN	reddit_eli5
21695	What to consider before buying (exercising) a ...	[The company may not permit a transfer of thes...	[There are several things to consider before b...	NaN	finance
14995	What causes wind ? I checked out the Wikipedia...	[Basically heat rises . When air gets hot it e...	[Wind is caused by the movement of air. When t...	NaN	reddit_eli5

Due to formatting issues (this category has a large amount of wikipedia formatting tags that vary greatly in format), we will also drop any rows that belong to the Computer Science Wikipedia Articles category (842 rows).

In [ ]:
```
prev_len = len(df) #find the current length of the dataframe before removal
prev_len
```

Out[ ]: 24322

In [ ]:
```
df = df[df.source != 'wiki_csai'] #remove the wiki_csai rows
df.reset_index(inplace=True, drop=True) #also reset the indexes of the dataframe
```

In [ ]:
```
cur_len = len(df)
prev_len - cur_len #check to make sure the correct number of rows was removed
```

Out[ ]: 842

In [ ]:
```
df.sample(3)
```

Out[ ]:

	question	human_answers	chatgpt_answers	index	source
1781	Why does cars need petrol / gas in order to fu...	[They can . Gas is one of the most efficient a...	[Cars need gasoline (also called petrol) to fu...	174.0	reddit_eli5
12741	Why when I press on my skin it leaves a white ...	[You have tiny blood vessels near the surface ...	[Pressing on your skin can cause a temporary c...	NaN	reddit_eli5
10666	When I open a PC application with notepad , wh...	[Because Notepad is used to reading the 0 's a...	[When you open a PC application with a text ed...	NaN	reddit_eli5

Now that we have removed the wiki_csai category, we can also drop the 'index' and 'source' columns, as they will not be used.

In [ ]:
```
df.drop(['index', 'source'], axis=1, inplace=True) # drop index and source columns
```

Next, we can inspect the data for missing values.

In [ ]:
```
# find any missing/NA values
df.isna().sum()
```

Out[ ]:
```
question           0
human_answers      0
chatgpt_answers    0
dtype: int64
```

No missing values were found.

Since each answer is a list, a list containing just empty strings, or an empty list will not be considered by panda's isna() function as a NA value. Thus, we will also inspect the dataframe for empty strings.

Some empty strings were found. Since these empty strings make up an insignificant portion of the dataset (~2%), they can be removed without significantly changing the makeup of the dataset.

In [ ]:
```
#check for empty strings
drop_cntr = 0
for i in range(len(df)):
    human_ans = df.at[i, 'human_answers']
    gpt_ans = df.at[i, 'chatgpt_answers']
    # if the answers are an empty string, or an empty list,
    if '' in human_ans or '' in gpt_ans or len(human_ans) == 0 or len(gpt_ans) == 0
        #drop the row
        df.drop(i, axis=0, inplace=True)
        drop_cntr += 1
#print drop counter
print(f'dropped {drop_cntr} rows')
```

```
dropped 470 rows
```

Next, reset the indexes of the dataset without the removed rows (so that it can be iterated over later)

In [ ]:
```
df.reset_index(inplace=True)
df.drop(df.columns[0], axis=1, inplace=True)
```

We will describe the dataset to inspect the values contained within

In [ ]:
```
df.describe(include='all')
```

Out[ ]:

	question	human_answers	chatgpt_answers
count	23010	23010	23010
unique	22482	21508	22447
top	Why we learned to cover our private parts with...	[The attack on Pearl Harbor (called Hawaii Ope...	[!Only one message at a time. Please allow any...
freq	2	3	14

We have also found some duplicate values, including duplicate questions, human answers, and chatgpt answers (which seems to be an error message)

These rows will be removed once the dataframe has been processed further (as it will be easier to remove them after the dataframe is converted to strings)

Next, we will inspect the dataframe's data types

In [ ]:
```
# check datatypes
df.dtypes
```

Out[ ]:
```
question           object
human_answers      object
chatgpt_answers    object
dtype: object
```

The dataframe will be converted to the string datatype.

In [ ]:
```
#convert to string
df = df.astype(str)
```

Next, we will need to strip the trailing/leading square brackets from the answers, and remove any newline characters (\n). We will also remove some Reddit text formatting tags and syntaxes.

In [ ]:
```
# define a function to strip formatting from answers
def strip_formatting(answer):
    # strip leading/trailing square brackets and quotation marks
    answer = answer.lstrip('[\'\"')
    answer = answer.rstrip(']\'\"')
    answer = answer.replace('\\n', "") # remove all newline chars
    answer = answer.replace('\\r', "") # remove all carraige return chars
    answer = answer.replace('*', "") # remove reddit formatting *
    answer = answer.replace('[', "") # remove reddit formatting square brackets
    answer = answer.replace(']', "") # strip other square brackets

    return answer
```

In [ ]:
```
for i in range(len(df)):
    human_ans = df.at[i, 'human_answers']
    gpt_ans = df.at[i, 'chatgpt_answers']
```

```
    df.at[i, 'human_answers'] = strip_formatting(human_ans)
    df.at[i, 'chatgpt_answers'] = strip_formatting(gpt_ans)
```

In [ ]:
```
df.describe(include='all')
```

Out[ ]:

	question	human_answers	chatgpt_answers
count	23010	23010	23010
unique	22482	21508	22445
top	Why we learned to cover our private parts with...	The attack on Pearl Harbor (called Hawaii Oper...	!Only one message at a time. Please allow any ...
freq	2	3	17

Now that the dataframe has been converted to strings, we can easily remove the rows containing the chatgpt error messages

In [ ]:
```
df = df[df['chatgpt_answers'].str.contains('There was an error generating a respons
```

We will also remove the duplicate questions and answers.

In [ ]:
```
df.drop_duplicates(subset = ['human_answers'], inplace=True)
df.drop_duplicates(subset = ['chatgpt_answers'], inplace=True)
df.drop_duplicates(subset = ['question'], inplace=True)
```

In [ ]:
```
df.describe(include ='all')
```

Out[ ]:

	question	human_answers	chatgpt_answers
count	21398	21398	21398
unique	21398	21398	21398
top	Why is every book I hear about a " NY Times # ...	Basically there are many categories of " Best ...	There are many different best seller lists tha...
freq	1	1	1

Additionally, the HC3 dataset covers both Chinese and English answers. Though we only imported the English portion of the dataset, it would be advisable to ensure all answers are actually in English.

This check will be carried out using the langdetect python package, which is a Python port of Google's language detection library.

First, install and import langdetect

In [ ]:
```
#!pip install langdetect
```

In [ ]:
```
from langdetect import detect
```

Next, define a function to detect if the text is english.

In [ ]:
```
def detect_english(answer):
    # detects if the given answer is in English
    if (detect(answer) == 'en') == False:
        raise Exception() #detects if the
```

We will also reset the indexes of the dataframe so we can iterate through it

In [ ]:
```
df.reset_index(inplace=True)
```

Call the detect_english method on both the human and chatgpt answers to remove any non-English answers. Once again, this makes up an extremely small portion of the entire dataset

In [ ]:
```
drop_cntr = 0 # counter for number of rows dropped
for i in range(0, len(df)):
    # extract human and gpt answers
    human_ans = df.at[i, 'human_answers']
    gpt_ans = df.at[i, 'chatgpt_answers']

    try:
        # try to call detect_english on both answers
        detect_english(human_ans)
        detect_english(gpt_ans)
    except:
        # print error message if either answer raises an exception
        print(f'non-english answer at f{i}')
        # drop the row
        df.drop(i, axis=0, inplace=True)
        drop_cntr += 1

# print total rows dropped
print(f'dropped {drop_cntr} rows')
```

```
non-english answer at f15520
non-english answer at f15908
non-english answer at f17052
non-english answer at f20613
non-english answer at f20659
non-english answer at f20681
dropped 6 rows
```

This results in a cleaned dataset that is about 88% the size of the original dataset. In our opinion, the majority of the original dataset is preserved, and the cleaning is unlikely to significantly affect our results.

1.2 Data Reshape

We will also need to reshape the dataframe to seperate human and chatgpt answers into distinct rows.

First, we will seperate the dataframe into 2 seperate dataframes, each containing only 1 type of answer.

We will also add an additional 'is_human' column to each dataframe, indicating if the answer was generated by a human

In [ ]:
```
# retrieve human answers
df_human = df[['question', 'human_answers']].copy() #copy the question and human an
df_human['is_human'] = 1 #set a column indicating if the answer is human
df_human.rename(columns = {'human_answers':'answer'}, inplace = True) # rename colu
df_human.sample(3)
```

Out[ ]:

	question	answer	is_human
4410	How does Polysporin work ? It says on the tube...	Polysporin is just antibiotic ointment . There...	1
1012	Postmodernism Please explain what postmodernis...	It \'s been used a lot in recent history , so ...	1
3107	The process and significance of " making partn...	It takes anywhere from 2 to 10 years to make p...	1

In [ ]:
```
# do the same for chatgpt answers
df_gpt = df[['question', 'chatgpt_answers']].copy()
df_gpt['is_human'] = 0
df_gpt.rename(columns = {'chatgpt_answers':'answer'}, inplace = True)
df_gpt.sample(3)
```

Out[ ]:

	question	answer	is_human
565	What exactly causes the feeling of " pins and ...	The feeling of "pins and needles" or tingling ...	0
2775	will someone please" dues ex machina " I do n'...	Deus ex machina is a Latin phrase that means "...	0
13985	Reddit , why does America use the A , B , C gr...	The A, B, C grading system is used in America ...	0

Next, both dataframes will be concatenated to join them together into a cleaned dataframe

In [ ]:
```
# concat both dataframes
df_cleaned = pd.concat([df_human, df_gpt])
df_cleaned.sample(3)
```

Out[ ]:

	question	answer	is_human
20192	Suggest remedy for low grade fever, hot and co...	I'm sorry to hear that you're feeling sick. It...	0
17472	What's a Letter of Credit? Are funds held in m...	A letter of credit is a financial instrument t...	0
11418	What happens when a person goes to the hospita...	If a person goes to the hospital because they ...	0

We will also use the panda's value_counts function to ensure that there is the expected number of each type of answer

In [ ]:
```
# sum is_human to ensure there is a 50:50 split of answers
df_cleaned['is_human'].value_counts()
```

Out[ ]:
```
is_human
1    21392
0    21392
Name: count, dtype: int64
```

Lastly, we will shuffle the order of the answers

In [ ]:
```
# shuffle human and chatgpt generated answers
df_cleaned = df_cleaned.sample(frac = 1).reset_index(drop=True)
```

In [ ]:
```
df_cleaned.sample(3)
```

Out[ ]:

	question	answer	is_human
14089	How come almost every time I cook eggs , that ...	The film you see on the edges of your cooked e...	0
38818	what do hard drugs do that cause a permanent c...	For one thing , they rob you of nutrition beca...	1
23538	Suggest alternative medicine for ciplatriml wa...	I'm sorry to hear that you have been prescribe...	0

1.3 Lemmatisation

We will also create both a lemmatised, and a filtered (both lemmatised, and stopword-removed) version of each answer for later usage in some methods, such as sentiment analysis.

Compared to stemming, lemmatisation preserves the context and meaning of a word whe converting it to its base form. Preserving the context and meaning of each word will allow for a more accurate sentiment analysis

First, download NLTK packages, and import the required modules.

In [ ]:
```
import nltk
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
from nltk.tokenize import word_tokenize

# Also download NLTK packages
nltk.download('stopwords')
nltk.download('punkt')
```

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

Out[ ]: True

Next, initialise the lemmatiser, and the corpus of stopwords. The stopword corpus used will be NLTK's general English stopword corpus.

In [ ]:
```
# init lemmatiser and stopwords
lemmatizer = WordNetLemmatizer()
stopwords = stopwords.words('english')

# also create empty columns for filtered/lemmatised answers
df_cleaned['filtered_answer'] = ""
df_cleaned['lemmatised_answer'] = ""
```

Next, iterate through each answer, processing each answer by either:

1. just lemmatising the answer, or
2. both removing stopwords and lemmatising the answer.

In [ ]:
```
for i in range(len(df_cleaned)):

    answer = df_cleaned.at[i, 'answer'] #retrieve answer text from df
    tokens = word_tokenize(answer) # tokenise the answer

    # lemmatise answer
    lemmatised_tokens = [lemmatizer.lemmatize(token) for token in tokens]
    # reconstruct the answer as a string, and store lemmatised answer in column
    df_cleaned.at[i, 'lemmatised_answer'] = ' '.join(lemmatised_tokens)

    # remove stopwords
    tokens_no_stopwords = [token for token in tokens if token not in stopwords]
    # lemmatise stopword-removed answer
    lemmatised_tokens_no_stopwords = [lemmatizer.lemmatize(token) for token in toke

    # reconstruct the answer as a string, and store filtered answer in column
    df_cleaned.at[i, 'filtered_answer'] = ' '.join(lemmatised_tokens_no_stopwords)
```

In [ ]:
```
df_cleaned.sample(3)
```

Out[ ]:

	question	answer	is_human	filtered_answer	lemmatised_answer
4194	" Floating Point " Numbers I tried looking at ...	When programming , if you need a number , like...	1	When programming , need number , like 2 27 -3 ...	When programming , if you need a number , like...
18832	what do movie directors actually do ? What do ...	Movie directors are responsible for overseeing...	0	Movie director responsible overseeing making f...	Movie director are responsible for overseeing ...
27393	How can animals like bees " smell your fear " ...	Bees can\'t actually "smell your fear," but th...	0	Bees can\'t actually `` smell fear , '' sense ...	Bees can\'t actually `` smell your fear , '' b...

2. Feature Extraction

As discovered in prototype 2, a combination of all 5 features performs better than any individual feature. Thus, we will extract all 5 features.

2.1 Basic Textual Features

We will extract the word count, sentence count, character count, average word length, and average sentence length.

In [ ]:
```
# import sentence and word tokenisers
from nltk.tokenize import word_tokenize, sent_tokenize
```

In [ ]:
```
#initialise columns for the extracted features
df_cleaned['word_count'] = 0
df_cleaned['sentence_count'] = 0
df_cleaned['char_count'] = 0
df_cleaned['avg_sentence_len'] = 0
df_cleaned['avg_word_len'] = 0
```

Next, iterate through each answer, extracting the selected features.

In [ ]:
```
for i in range(len(df_cleaned)):
    answer = df_cleaned.at[i, 'answer'] # extract answer text from df

    ans_sent_token = sent_tokenize(answer)# tokenise answer into sentences
    sentence_count = len(ans_sent_token)# calculate sentence count
    df_cleaned.at[i, 'sentence_count'] = sentence_count

    ans_word_token = word_tokenize(answer) # tokenise answer into words
    word_count = len(ans_word_token)  # calculate word count
    df_cleaned.at[i, 'word_count'] = word_count
```

```
    char_count = len(answer) # calculate character count
    df_cleaned.at[i, 'char_count'] =  char_count

    df_cleaned.at[i, 'avg_sentence_len'] = word_count/sentence_count # calculate av
    df_cleaned.at[i, 'avg_word_len'] =  char_count/word_count   # calculate average
```

In [ ]:
```
df_cleaned.sample(3)
```

Out[ ]:

	question	answer	is_human	filtered_answer	lemmatised_answer	word_count	s
37122	why do wrecking balls , and possibly other wre...	Wrecking balls and other demolition equipment ...	0	Wrecking ball demolition equipment often spray...	Wrecking ball and other demolition equipment o...	164	
36710	Retired, want to buy a mobile home; how to fin...	There are a few different options for financin...	0	There different option financing mobile home ,...	There are a few different option for financing...	352	
35709	Why are flying planes louder when it is cloudy...	Sound travels better through humid air than dr...	1	Sound travel better humid air dry air . This h...	Sound travel better through humid air than dry...	351	

2.2 Sentiment Analysis

4 metrics will be assessed – negativity, neutrality, subjectivity, and overall polarity.

To calculate overall polarity, we will use 2 analysers - NLTK's VADER and TextBlob. This will allow us to obtain a secondary opinion, and provide a more accurate compound score.

The rest of the metrics will be extracted using only 1 analyser, as each analyser supports a different set of metrics. Negativity and neutrality will be extracted using VADER, and subjectivity will be extracted by Textblob.

First, import the required modules and download the required VADER NLTK package.

In [ ]:
```
from nltk.sentiment import SentimentIntensityAnalyzer
from textblob import TextBlob
```

In [ ]:
```
nltk.download('vader_lexicon')
```

```
[nltk_data] Downloading package vader_lexicon to
[nltk_data]     C:\Users\Admin\AppData\Roaming\nltk_data...
[nltk_data]   Package vader_lexicon is already up-to-date!
```

Out[ ]: True

We will also initialise empty columns for the extracted features.

In [ ]:
```
# initialise columns for the extracted features
df_cleaned['polarity'] = 0
df_cleaned['negativity'] = 0
df_cleaned['neutrality'] = 0
df_cleaned['subjectivity'] = 0
```

Next, initialise the NLTK VADER sentiment analyser (TextBlob does not need to be initialised using a constructor)

In [ ]:
```
analyser = SentimentIntensityAnalyzer()
```

Next, iterate through each filtered answer (lemmatised + stopwords removed), extracting the negativity (NLTK only), neutrality (NLTK only), polarity (both), and subjectivity (TextBlob only) of each answer.

In [ ]:
```
for i in range(len(df_cleaned)):
    answer = df_cleaned.at[i, 'filtered_answer']
    #analyse filetered answers
    nltk_scores = analyser.polarity_scores(answer)
    textblob_scores = TextBlob(answer).sentiment

    # extract neg and neu using nltk
    df_cleaned.at[i, 'negativity'] = nltk_scores['neg']
    df_cleaned.at[i, 'neutrality'] = nltk_scores['neu']

    # calculate average polarity by using results from both analysers
    df_cleaned.at[i, 'polarity'] = (nltk_scores['compound'] + textblob_scores.polar

    # extract subjectivity using textblob
    df_cleaned.at[i, 'subjectivity'] = textblob_scores.subjectivity
```

In [ ]:
```
df_cleaned.sample(3)
```

Out[ ]:

	question	answer	is_human	filtered_answer	lemmatised_answer	word_count	se
8098	Why does the exchange rate between countries f...	Exchange rates fluctuate because they are dete...	0	Exchange rate fluctuate determined supply dema...	Exchange rate fluctuate because they are deter...	251	
9426	when was srv born	Stephen "Stevie" Ray Vaughan (October 3, 1954 ...	1	Stephen `` Stevie '' Ray Vaughan (October 3 ,...	Stephen `` Stevie '' Ray Vaughan (October 3 ,...	28	
24378	Why ca n't we just boil sea water to solve the...	Boiling water takes a ton of energy . Most of ...	1	Boiling water take ton energy . Most time 's f...	Boiling water take a ton of energy . Most of t...	109	

2.3 Syntactic and Lexical Density

2.3.1 Lexical Density

First, we will extract the lexical density of each answer. This will involve counting the number of lexical terms in each answer (nouns/proper nouns, adjectives, verbs, prepositions, and adverbs), and dividing them by the total number of words in the answer.

First, install and import spaCy, and download the pre-trained en_core_web_sm model, which is optimised for CPU processing.

In [ ]:
```
#!pip install spacy
#!python -m spacy download en_core_web_sm
```

In [ ]:
```
import spacy
```

Also, create an empty column for the lexical density feature

In [ ]:
```
df_cleaned['lexical_density'] = 0 # create an empty column for lexical density
```

Load the spacy en_core_web_sm model, and a list of POS tags that denote lexical words

In [ ]:
```
nlp = spacy.load("en_core_web_sm") # load spacy model
tags = ['PROPN', 'ADJ', 'NOUN', 'VERB', 'ADP', 'ADV'] # load list of desired lexica
```

Finally, iterate through every row and count the number of lexical words. Then, divide the number of lexical words by the total word count to calculate the lexical density of each answer.

In [ ]:
```
for i in range(len(df_cleaned)):

    answer = df_cleaned.at[i, 'answer'] #extract the answer text
    word_count = df_cleaned.at[i, 'word_count'] #extract word count

    lexical_word_cntr  = 0 #initialise counter for lexical words
    ans_pos = nlp(answer) # analyse answer text using model

    # for each word in the analysed answer
    for token in ans_pos:
        # check if it is a lexical word (POS tag is in the list of desired tags)
        if token.pos_ in tags:
            lexical_word_cntr += 1

    df_cleaned.at[i, 'lexical_density'] = lexical_word_cntr/word_count # calculate
```

2.3.2 Named Entity Count

Next, we will extract the named entity features. The count of named entities will be extracted using spaCy.

In [ ]:
```
df_cleaned['NE_count'] = 0
```

In [ ]:
```
for i in range(len(df_cleaned)):
    answer = df_cleaned.at[i, 'answer'] #extract the answer text
    ans_NE = nlp(answer) # analyse answer text using the spaCy model
    df_cleaned['NE_count'] = len(ans_NE.ents) #retrieve the NE count for each answe
```

In [ ]:
```
df_cleaned.sample(3)
```

Out[ ]:

	question	answer	is_human	filtered_answer	lemmatised_answer	word_count	s
25422	Can everyday people profit from unexpected wor...	It is possible for everyday people to profit f...	0	It possible everyday people profit unexpected ...	It is possible for everyday people to profit f...	140	
27460	Is it legal to create copies of consoles such ...	The patents on the original NES expired in 200...	1	The patent original NES expired 2005 , 's lega...	The patent on the original NES expired in 2005...	121	
16351	Pleasewhy does it take my printer a full 5 min...	I 'm sure there 's a technical answer , but th...	1	I 'm sure 's technical answer , long short pri...	I 'm sure there 's a technical answer , but th...	362	

Since this feature takes a significant time to extract, we will save the current data to a csv file.

This avoids the need to re-run the entire code again when refreshing the jupyter kernel/restarting the notebook.

In [ ]:
```
df_cleaned.to_csv('intermediate_data_1.csv', mode='w')
```

2.3.3 Coreference Cluster Features

Next, we will also extract the Coreference Cluster features - average coreference cluster length, and number of coreference clusters.

This will be extracted using a pre-trained en_coreference_web_trf model from SpaCy's experimental package.

First, we will need to install spacy-experimental, and download the pre-trained coreference cluster model.

We will also install the tqdm package to print out a progress bar, so we can visualise the progress of the coreference cluster feature extraction (as coreference cluster extraction takes roughly 11-12 hours for the entire dataset)

First, install the required packages and import modules

In [ ]:
```
#!pip install spacy-experimental
#!pip install https://github.com/explosion/spacy-experimental/releases/download/v0.
```

```
#!pip install tqdm
```

In [ ]:
```
import spacy
import spacy_transformers
from tqdm import tqdm
```

Load the pretrained model, and read in the data from the intermediate csv file we saved earlier

In [ ]:
```
coref = spacy.load("en_coreference_web_trf")
```

```
c:\Users\Admin\OneDrive - Microsoft 365\Desktop\Prototype 2\.venv\Lib\site-packages
\spacy\util.py:910: UserWarning: [W095] Model 'en_coreference_web_trf' (3.4.0a2) was
trained with spaCy v3.3.0 and may not be 100% compatible with the current version
(3.6.1). If you see errors or degraded performance, download a newer compatible mode
l or retrain your custom model with the current spaCy version. For more details and
available updates, run: python -m spacy validate
  warnings.warn(warn_msg)
```

In [ ]:
```
df_cleaned = pd.read_csv('intermediate_data_1.csv')
```

Also, create empty columns for the coreference cluster features

In [ ]:
```
 # create empty columns for the coreference features
df_cleaned['coref_count'] = 0
df_cleaned['avg_coref_len'] = 0
```

Next, iterate over all rows in the dataframe, and extract the coreference spans (clusters).

This will allow us to count the number of coreference clusters, and calculate the average length of each cluster,

In [ ]:
```
for i in tqdm(range(0, len(df_cleaned))):
        answer = df_cleaned.at[i, 'answer'] #retrieve answer from df
        doc = coref(answer) # analyse answer using pretrained coref model

        coref_count = len(doc.spans) #get number of coref spans (clusters)

        # skip answer if no coref spans detected, since no clusters results in a 0
        # avoids div by 0 error
        if coref_count == 0:
            continue

        # initialise counter for total length of all coref spans
        total_span_len = 0
        # iterate over all spans, and add their length to the counter
        for span in doc.spans:
            total_span_len += len(doc.spans[span])

        avg_span_len = total_span_len/coref_count #calcuate average span length

        # set values in df
```

```
        df_cleaned.at[i, 'avg_coref_len'] = avg_span_len
        df_cleaned.at[i, 'coref_count'] = coref_count
```

```
  3%|▌          | 1074/42780 [18:40<9:24:31,  1.23it/s] Token indices sequence lengt
h is longer than the specified maximum sequence length for this model (527 > 512). R
unning this sequence through the model will result in indexing errors
100%|██████████| 42780/42780 [12:31:01<00:00,  1.05s/it]
```

Once again, we will save this to an intermediate csv file to avoid having to rerun this extraction again.

In [ ]:
```
df_cleaned.to_csv('intermediate_data_2.csv', mode='w')
```

2.4 Frequency

We will also be extracting 3 frequency features:

1. Zipfian distribution difference
2. Unique word count
3. Stop word count

2.4.1 Zipfian Difference

We will extract the difference between the text's Zipfian distribution line, and the Zipf's law line. (which we will refer to as Zipfian Difference)

First, install matplotlib, which will be used to assist in visualising this method.

Also, import the neccessary modules

In [ ]:
```
# !pip install matplotlib
```

In [ ]:
```
import string
import collections
import pandas as pd
import nltk
from nltk import word_tokenize
import matplotlib.pyplot as plt
import numpy as np
from tqdm import tqdm
```

We will also read in the csv file saved in the last step

In [ ]:
```
df_cleaned = pd.read_csv('intermediate_data_2.csv')
```

In [ ]:
```
df_cleaned.sample(3)
```

Out[ ]:

	Unnamed: 0.1	Unnamed: 0	question	answer	is_human	filtered_answer	lemmatise
13787	13787	13787	Himalayas So the Indian plate is colliding int...	Eventually , it will be under the Eurasian pla...	1	Eventually , Eurasian plate . Parts scrape ont...	Eventually under th
20002	20002	20002	How does alcohol disinfect a wound or a given ...	Alcohol is a very effective disinfectant becau...	0	Alcohol effective disinfectant able kill wide ...	Alcoh effective di
30410	30410	30410	Why is making a nuclear bomb so hard ? Why is ...	Making a nuclear bomb is difficult for several...	0	Making nuclear bomb difficult several reason	Making bomb is c

In order to better illustrate this method, we will show the entire process on a random sample answer, before applying it to the entire text.

To start, we will extract the lemmatised answer (inclusive of stopwords) from the dataframe, and strip its punctuation. The lemmatised answer is used to reduce each word to its base form for counting (e.g., jump, jumped, jumping should all be reduced to 'jump' for counting)

In [ ]:
```
answer = df_cleaned.at[np.random.randint(0, high = len(df_cleaned)), 'lemmatised_an
answer = answer.translate(str.maketrans('', '', string.punctuation)) # strip punctu
```

Next, we will tokenise the answer, and convert it to a frequency distribution

In [ ]:
```
tokens = word_tokenize(answer) #tokenise answer
freq_dist = nltk.FreqDist(tokens) # create freqdist
freq_dist
```

Out[ ]:
```
FreqDist({'is': 15, 'a': 14, 'of': 12, 'to': 10, 'the': 10, 'energy': 9, 'it': 7,
'transfer': 6, 'and': 6, 'device': 5, ...})
```

Next, we will convert the frequency distribution to a ordered dictionary, sorted in descending order of word frequency.

This is so we can preseve the order of the dictionary, which will be used to plotting later on.

In [ ]:
```
freq_dict = dict([(m, n) for m, n in freq_dist.items()]) # create dictionary from f
freq_dict = collections.OrderedDict(dict(sorted(freq_dict.items(), key=lambda item:
print(freq_dict)
```

```
OrderedDict([('is', 15), ('a', 14), ('of', 12), ('to', 10), ('the', 10), ('energy',
9), ('it', 7), ('transfer', 6), ('and', 6), ('device', 5), ('can', 5), ('through',
5), ('electromagnetic', 4), ('type', 4), ('that', 4), ('which', 3), ('are', 3), ('b
e', 3), ('or', 3), ('other', 3), ('This', 3), ('for', 3), ('way', 3), ('wirelessly',
3), ('use', 3), ('field', 3), ('this', 3), ('Wireless', 2), ('data', 2), ('possibl
e', 2), ('wave', 2), ('transmit', 2), ('information', 2), ('from', 2), ('one', 2),
('another', 2), ('air', 2), ('on', 2), ('transmitting', 2), ('without', 2), ('scient
ist', 2), ('there', 2), ('One', 2), ('wireless', 2), ('called', 2), ('Tesla', 2),
('However', 2), ('not', 2), ('area', 2), ('in', 2), ('because', 1), ('us', 1), ('The
se', 1), ('sent', 1), ('material', 1), ('such', 1), ('water', 1), ('glass', 1), ('ha
nd', 1), ('process', 1), ('electrical', 1), ('place', 1), ('using', 1), ('wire', 1),
('something', 1), ('engineer', 1), ('have', 1), ('been', 1), ('working', 1), ('lon
g', 1), ('time', 1), ('few', 1), ('different', 1), ('done', 1), ('similar', 1), ('ho
w', 1), ('transferred', 1), ('but', 1), ('instead', 1), ('goal', 1), ('example', 1),
('charging', 1), ('pad', 1), ('charge', 1), ('electronic', 1), ('like', 1), ('smartp
hones', 1), ('tablet', 1), ('need', 1), ('physical', 1), ('connection', 1), ('Anothe
r', 1), ('coil', 1), ('highvoltage', 1), ('transformer', 1), ('invented', 1), ('by',
1), ('Nikola', 1), ('able', 1), ('produce', 1), ('powerful', 1), ('very', 1), ('prac
tical', 1), ('everyday', 1), ('requires', 1), ('lot', 1), ('power', 1), ('dangerou
s', 1), ('if', 1), ('used', 1), ('carefully', 1), ('Overall', 1), ('while', 1), ('st
ill', 1), ('an', 1), ('active', 1), ('research', 1), ('development', 1), ('many',
1), ('challenge', 1), ('overcome', 1), ('before', 1), ('becomes', 1), ('widespread',
1), ('technology', 1), ('our', 1), ('understanding', 1), ('improves', 1), ('likely',
1), ('we', 1), ('will', 1), ('see', 1), ('more', 1), ('progress', 1), ('future',
1)])
```

Next, we will need to change the key from a word to its frequency rank (e.g., 1st, 2nd, 3rd most frequent word) using a counter.

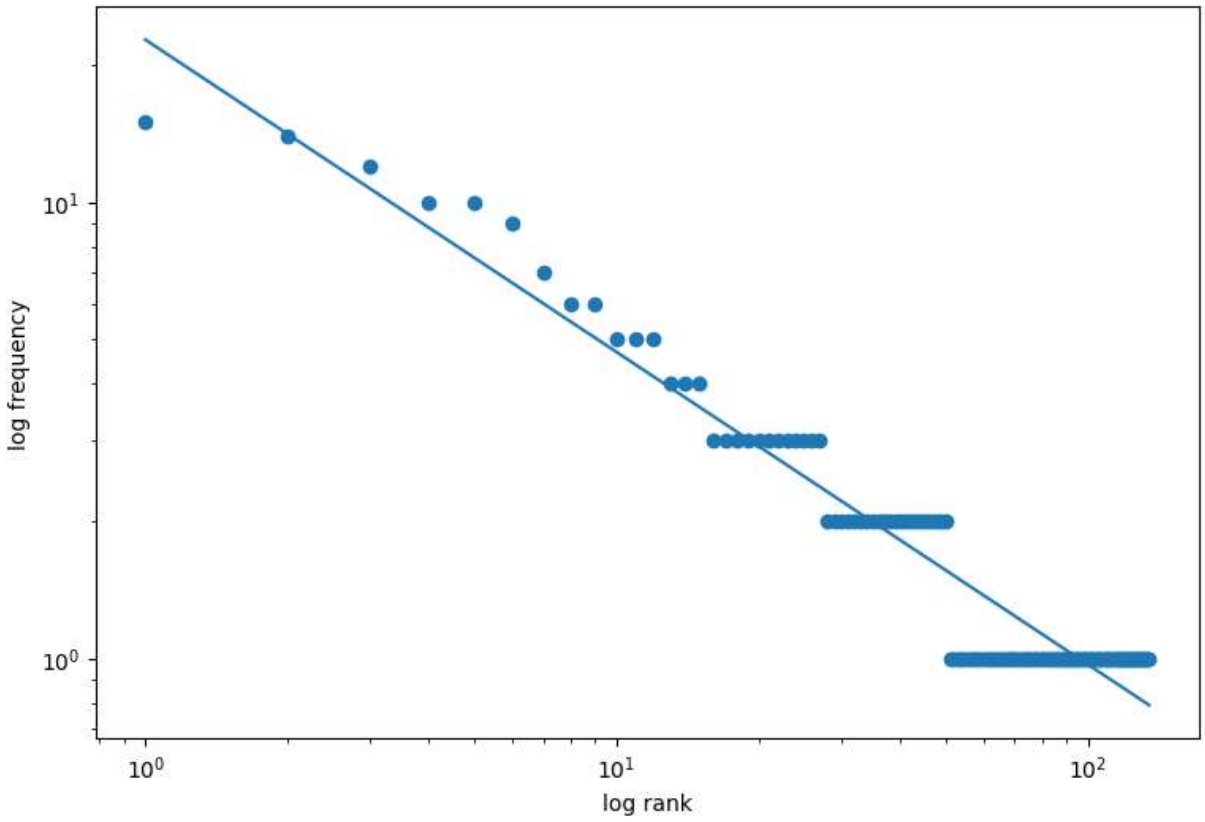
As ordered dictionaries are immutable, we will instead use a new dictionary, final_freq_dict, and add each item in freq_dict to this dictionary.

In [ ]:
```
cntr = 1 #initialise counter
final_freq_dict = collections.OrderedDict() #initliase final frequency dict

for key in freq_dict.keys():
    final_freq_dict[cntr] = freq_dict[key] # add each item to final_freq_dict, and
    cntr += 1 #increment counter

print(final_freq_dict)
```

```
OrderedDict([(1, 15), (2, 14), (3, 12), (4, 10), (5, 10), (6, 9), (7, 7), (8, 6),
(9, 6), (10, 5), (11, 5), (12, 5), (13, 4), (14, 4), (15, 4), (16, 3), (17, 3), (18,
3), (19, 3), (20, 3), (21, 3), (22, 3), (23, 3), (24, 3), (25, 3), (26, 3), (27, 3),
(28, 2), (29, 2), (30, 2), (31, 2), (32, 2), (33, 2), (34, 2), (35, 2), (36, 2), (3
7, 2), (38, 2), (39, 2), (40, 2), (41, 2), (42, 2), (43, 2), (44, 2), (45, 2), (46,
2), (47, 2), (48, 2), (49, 2), (50, 2), (51, 1), (52, 1), (53, 1), (54, 1), (55, 1),
(56, 1), (57, 1), (58, 1), (59, 1), (60, 1), (61, 1), (62, 1), (63, 1), (64, 1), (6
5, 1), (66, 1), (67, 1), (68, 1), (69, 1), (70, 1), (71, 1), (72, 1), (73, 1), (74,
1), (75, 1), (76, 1), (77, 1), (78, 1), (79, 1), (80, 1), (81, 1), (82, 1), (83, 1),
(84, 1), (85, 1), (86, 1), (87, 1), (88, 1), (89, 1), (90, 1), (91, 1), (92, 1), (9
3, 1), (94, 1), (95, 1), (96, 1), (97, 1), (98, 1), (99, 1), (100, 1), (101, 1), (10
2, 1), (103, 1), (104, 1), (105, 1), (106, 1), (107, 1), (108, 1), (109, 1), (110,
1), (111, 1), (112, 1), (113, 1), (114, 1), (115, 1), (116, 1), (117, 1), (118, 1),
(119, 1), (120, 1), (121, 1), (122, 1), (123, 1), (124, 1), (125, 1), (126, 1), (12
7, 1), (128, 1), (129, 1), (130, 1), (131, 1), (132, 1), (133, 1), (134, 1), (135,
1)])
```

Next, we will plot the scatter plot, and its best fit line on a log-log scale.

In [ ]:
```python
x = np.array([int(i) for i in final_freq_dict.keys()]) # frequency rank as x axis
y = np.array([int(i) for i in final_freq_dict.values()]) # frequency percentage as

fig, ax = plt.subplots(figsize = (9, 6))

# change scale to log-log
ax.set_yscale("log")
ax.set_xscale("log")

#set x and y labels
plt.xlabel('log rank')
plt.ylabel('log frequency')

#add plot scatter plot
ax.scatter(x, y)

# calcuate best fit line on a log-log scale (referenced from Craig [2017])
slope, intercept = np.polyfit(np.log(x), np.log(y), 1) # fit log(y) = m*log(x) + c
y_fit = np.exp(slope * np.log(x) + intercept) # calculate the fitted values of y
ax.plot(x, y_fit)

plt.show()
```



In order to calcuate the difference between the text' Zipfian Difference, we will use the absolute difference in the gradient between the plotted line, and the Zipf's law line (which has a gradient of -1 on a log-log scale)

In [ ]:
```
print(f'The line gradient is: {slope}')
print(f'The difference between the line and Zipf\'s law is {np.absolute(-1 - slope)
```

```
The line gradient is: -0.6848123376016977
The difference between the line and Zipf's law is 0.3151876623983023
```

Next, we will gather the above method into a function (minus the line plotting code, as we only need the gradient of the line, and do not need to plot and show the actual graph) called calc_zipfian_diff, so we can call it on each row of the dataframe.

In [ ]:
```
def calc_zipfian_diff(row):
    answer = df_cleaned.at[row, 'lemmatised_answer'] #extract answer
    answer = answer.translate(str.maketrans('', '', string.punctuation)) # strip pu
    tokens = word_tokenize(answer) #tokenise answer
    freq_dist = nltk.FreqDist(tokens) # create freqdist

    freq_dict = dict([(m, n) for m, n in freq_dist.items()]) # create dictionary fr
    freq_dict = collections.OrderedDict(dict(sorted(freq_dict.items(), key=lambda i

    cntr = 1 #initialise counter
    final_freq_dict = collections.OrderedDict() #initliase final frequency dict

    for key in freq_dict.keys():
        final_freq_dict[cntr] = freq_dict[key] # add each item to final_freq_dict,
        cntr += 1 #increment counter
```

```
    x = np.array([int(i) for i in final_freq_dict.keys()]) # frequency rank as x ax
    y = np.array([int(i) for i in final_freq_dict.values()]) # frequency percentage

    # calcuate best fit line on a log-log scale (referenced from https://stackoverf
    slope, intercept = np.polyfit(np.log(x), np.log(y), 1) # fit log(y) = m*log(x)


    zipfian_diff = np.absolute(-1 - slope) # get absolute difference between the gr

    return zipfian_diff
```

Next, we can call the function on each row to find the Zipfian Difference for each answer.

In [ ]:
```
df_cleaned['zipfian_difference'] = 0 #initalise empty column for zipfian difference
```

In [ ]:
```
for i in range(0, len(df_cleaned)):
    df_cleaned.at[i, 'zipfian_difference'] = calc_zipfian_diff(i)
```

In [ ]:
```
df_cleaned.sample(3)
```

Out[ ]:

	Unnamed: 0.1	Unnamed: 0	question	answer	is_human	filtered_answer	lemmatised_
35174	35174	35174	Why are circles exactly 360 degrees ? Why are ...	Circles are 360 degrees because that's just ho...	0	Circles 360 degree 's 's always done ! People ...	Circles degree becau 's j
20792	20792	20792	Why do two negative numbers yield a positive n...	Great question! When we multiply two negative ...	0	Great question ! When multiply two negative nu...	Great qu When we r two ne
41728	41728	41728	Does food sit normally in your stomach in spac...	In space, food can behave a bit differently th...	0	In space , food behave bit differently Earth	In space , fo beha differ

3 rows × 21 columns

2.4.2 Unique Word Count and Stopword Count

We will also extract the unique word proportion and stopword proportion for each text.

We will quantify this by:

1. extracting the proportion of unique words in each answer, divided over the length of the entire answer
2. extracting the proportion of the answer that is made up of stopwords

In [ ]:
```
from nltk import word_tokenize
from nltk.corpus import stopwords
```

In [ ]:
```
df_cleaned['unique_word_proportion'] = 0
df_cleaned['stopword_proportion'] = 0
```

In [ ]:
```
for i in range(0, len(df_cleaned)):
    answer = df_cleaned.at[i, 'answer'] #retrieve answer from df
    word_count = df_cleaned.at[i, 'word_count'] #extract word count from df

    tokens = word_tokenize(answer) # tokenise answer

    unique_word_count = len(set(tokens)) # get number of unique words

    tokens_no_stopwords = [token for token in tokens if token not in stopwords] # c
    stopword_count = word_count - len(tokens_no_stopwords) # calculate number of st

    df_cleaned.at[i, 'unique_word_proportion'] = (unique_word_count/word_count) #ca
    df_cleaned.at[i, 'stopword_proportion'] = (stopword_count/word_count) # calcula
```

In [ ]:
```
df_cleaned.sample(3)
```

Out[ ]:

	Unnamed: 0.1	Unnamed: 0	question	answer	is_human	filtered_answer	lemmatised
37284	37284	37284	Why are there no drugs that make us dramatical...	There 's a million ways to break a car , rangi...	1	There 's million way break car , ranging baseb...	There 's way to bre
12540	12540	12540	Why are imported chinese products of such low ...	Because you usually do n't know its from China...	1	Because usually n't know China high quality	Because yo do n't kno
15387	15387	15387	Can I get my property taxes lowered?	It is possible to get your property taxes lowe...	0	It possible get property tax lowered , process...	It is possib your pro

3 rows × 23 columns

We will save this to an intermediate csv file

In [ ]:
```
df_cleaned.to_csv('intermediate_data_3.csv', mode='w')
```

2.5 Complex Phrase Features

We will also be extracting the count of 2 different types of complex phrases - Ancient phrases and Cliches.

This will be done by comparing the answer to reference lists of each type of phrase, which have been compiled and cleaned from 2 online sources.

First, we will import the required modules and read in the intermediate csv from the last step

In [ ]:
```
import pandas as pd
import csv
```

In [ ]:
```
df_cleaned = pd.read_csv('intermediate_data_3.csv')
df_cleaned.sample(3)
```

Out[ ]:

	Unnamed: 0.2	Unnamed: 0.1	Unnamed: 0	question	answer	is_human	filtered_ar
8512	8512	8512	8512	how much does a gold bar weigh	The standard gold bar held as gold reserves by...	1	The sta gold ba gold re ce
34724	34724	34724	34724	Why certain prescriptions take so long to fill...	There are a few reasons why some prescriptions...	0	There r prescri may take l
10594	10594	10594	10594	What are the tiny worm like thing on my neck, ...	This content may violate our content policy. I...	0	This co may v content p

3 rows × 24 columns

Next, read in the 2 reference lists from their respective csv files into python lists.

First, we will define a function to convert a csv file to a list

In [ ]:
```
def csv_to_list(filepath):
    # open csv file
    with open(filepath, 'r') as file:
        reader = csv.reader(file) # read csv file
        data = []
        for row in reader:
            data.append(row[0]) # append each row to data list
    return data
```

and call it on the two reference csv files.

In [ ]:
```
cliches = csv_to_list('phrases/cliches.csv')
ancient_phrases = csv_to_list('phrases/ancient_phrases.csv')
print(cliches)
print(ancient_phrases)
```

['ace in the hole', 'ace up your sleeve', 'acid test', 'airing dirty laundry', "all in a day's work", 'all talk', 'all booster', 'all hat', 'all foam', 'all hammer', 'all icing', 'all lime and salt', 'all missile', 'all shot', 'all sizzle', 'all wax and no wick', 'all that and a bag of chips', 'all thumbs', 'all wet', "all's fair in love and war", 'almighty dollar', 'always a bridesmaid', 'ambulance chaser', 'another day', 'ants in your pants', 'apple-pie order', 'arm and a leg', 'armchair quarterback', 'army brat', 'art imitates life', 'artsy-craftsy', 'artsy-fartsy', 'as luck would have it', 'as old as time', 'at loggerheads', 'calm before the storm', 'candle at both ends', "can't cut the mustard", 'case of mistaken identity', 'cat out of the bag', 'cat got your tongue', 'caught red-handed', 'chapter and verse', 'checkered career', 'chickens come home to', 'roost', 'chomping at the bit', 'cleanliness is next to', 'godliness', 'clear as a bell', 'clear as mud', 'cold shoulder', 'E-ticket', 'ear to the ground', 'early bird catches the worm', 'easier said than done', 'easy as 1-2-3', 'easy as pie', 'eat crow', 'eat humble pie', 'enough already', 'even money', 'every dog has its day', 'every fiber of my being', 'everything but the kitchen sink', 'evil twin', 'existential angst', 'experts agreeï¿½', 'eye for an eye', 'hair of the dog', 'hard to believe', 'have a nice day', 'head honcho', "heart's content", 'hell-bent for leather', 'hidden agenda', 'high and the mighty (the)', 'high on the hog', 'hold a candle to', 'hold your horses', 'hold your tongue', 'hook or by crook', 'horse of a different color', 'hot knife through butter', 'how goes the battle?', 'keep an eye on you', 'keep it down', 'keep it simple', 'keep up with the Joneses', 'keep your cards close to vestï¿½', 'keep your chin up', 'keep your fingers crossed', 'keep your powder dry', 'kick ass', 'kickbutt', 'kick the bucket', 'kick up your heels', 'kick you to the curb', 'kick your feet up', 'kid in a candy store', 'kill two birds with one stone', "life's a bitch", 'lighten up', 'lights out', 'like a sore thumb', 'like butter', 'like the plague', "like there's no tomorrow", "lion's share", 'litmus test', 'little black book', 'live and learn', 'long and short of it', 'long lost love', 'look before you leap', 'lounge lizard', 'loved and lost', 'low man on the totem pole', 'luck of the draw', 'luck of the Irish', 'off the wagon', 'old college try', 'old meets new', 'older and wiser', 'older than dirt', 'older than Methuselah', 'on the bandwagon', 'on the nose', 'on the wagon', 'on thin ice', 'one born every minute', 'one foot in the grave', 'one in a million', 'only game in town', 'only to be met', 'out of pocket', 'out of the frying pan', 'out on a limb', 'safe than sorry', 'salt of the earth', 'save face', 'scared stiff', 'scared to death', "school's out", 'screaming meemies', 'senses reel', 'set the record straight', 'shake a stick should of', 'shoulder to the wheel', 'shouldered his way', 'shut your trap', 'sigh of relief', 'significant other', 'silence is golden', 'take one for the team', 'take the bull by the horns', 'take the plunge', 'takes one to know one', 'talk turkey', 'ten foot pole', 'the earth moved', 'the final analysis', 'the real McCoy', 'the same old story', 'these things happen', 'thick as thieves', 'think outside of the box', "third time's the charm", 'under the gun', 'under the same roof', 'understated elegance', 'unexpected twist', 'until the cows come home', 'up his sleeve', 'up the creek', 'up the wrong tree', 'very real concern', 'view with alarm', 'babe in the woods', 'back against the wall', 'back in the saddle', 'back to square one', 'back to the drawing board', 'bad to the bone', 'badge of honor', 'Badonkadonk', 'ballpark figure', 'balls to the wall', 'baptism of fire', 'bare bones', 'bark is worse than the bite', 'bark up the wrong tree', 'bat out of hell', 'bats in the belfry', 'battle royal', 'beat around the bush', 'beat the bushes', 'beats me', 'behind the eight ball', 'bent out of shape', 'best foot forward', 'bet your bottom dollar', 'better half', 'better late than never', 'better mousetrap', 'better safe than sorry', 'better than ever', 'better the Devil you know', 'between a rock and a hard place', 'beyond the pale', 'bib and tucker', 'big as life', 'big fish in a small pond', 'big man on campus', 'bigger they are', 'communist conspiracy', 'conniption fit', 'could care less', "couldn't care less", "couldn't get to first base", 'count your blessings', 'countless hours', 'creature comfort', 'crime in the street', 'curiosity killed the cat', 'curry favor', 'cut a fine figure', 'cut and dried', 'cut to the chase', 'cut to the q

uick', 'cute as a button', 'facts of life', 'fair-haired one', 'fair weather frien
d', 'fall off of a turnip truck', 'fat slob', 'favor us with a song', 'fear and loat
hing', 'feather your nest', 'fellow traveler', 'few and far between', 'field this on
e', 'fifteen minutes of fame', 'fish nor fowl', 'fly by night', 'fly the coop', 'for
the birds', 'fox in the henhouse', 'freudian slip', 'fun and games', 'fun in the su
n', 'I beg to differ', 'if the shoe fits', "I'm okay", 'in a nutshell', 'in a pinc
h', 'in a wink', "in harm's way", 'in the tank', 'in your dreams', 'in your face',
'inexorably drawn', 'info dump', 'influence peddling', 'intents and purposes', 'it w
as a dark and stormy night', "it won't fly", "King's English", "king's ransom", 'kis
s and tell', 'kiss ass', 'kiss of death', 'kit and kaboodle', 'knee-high to a grassh
opper', 'knock it out of the park', 'knock on wood', 'knock your socks off', 'knocke
d up', 'know him from Adam', 'know the ropes', 'know the score', 'knuckle down', 'kn
uckle sandwich', 'knuckle under', 'make my day', 'male chauvinist', "man's best frie
nd", 'many moons', 'many-splendored thing', 'mark my words', 'meaningful relationshi
p', 'mellow out', 'moment of glory', "moment's respite", 'Monday morning quarterbac
k', 'monkey suit', 'monkey see', 'motherhood and apple pie', 'movers and shakers',
'moving experience', 'my two cents', "p's and q's", 'pain and suffering', 'panic but
ton', 'party pooper', 'patter of little feet', 'pass the sniff test', 'pay through t
he nose', 'peas in a pod', 'perfect storm', 'pig in a poke', 'pillar of society', 'p
inko', 'plenty of fish in the sea', 'poison pen', 'poor as a churchmouse', 'poor exc
use for', 'pot calling the kettle black', 'proud possessor', 'put my foot down', 'pu
t your foot down', 'quick as a bunny', 'quick and the dead', 'slept like a log', 'sm
all world', 'snake in the grass', 'snow job', 'snug as a bug', 'some of my best frie
nds', 'something the cat dragged in', 'spade a spade', 'spare the rod', 'spitting im
age', 'spring to life', 'this day and age', 'this point in time', "three strikes and
you're out", 'through the grapevine', 'throw in the towel', 'tiger by the tail', 'ti
ll the fat lady sings', 'time and time again', 'time is of the essense', 'tip of the
iceberg', "'tis the season", 'to err is human', 'to the best of my knowledge', 'wake
up call', 'was my face red', 'watch your tongue', 'web of intrigue', 'week of sunday
s', 'what a bummer', 'what comes around', 'what the cat dragged in', 'what the dicke
ns', 'what the heck', 'what the hell', 'what you see is what you get', "what's not t
o like", 'wheeler-dealer', 'when in doubt', 'when push comes to shove', 'when rubber
meets the road', "when the cat's away", 'when the going gets tough', 'bird in the ha
nd', 'birds and the bees', 'birds of the feather', 'bite the dust', 'bite your tongu
e', 'bitter disappointment', 'black as coal', 'blast from the past', 'bleeding hear
t', 'blind as a bat', 'blood is thicker than water', 'blood money', 'blood on your h
ands', 'blood sweat and tears', 'blow this pop stand', 'blow this joint', 'blushing
bride', 'boil it down to', 'bone of contention', 'booze and broads', 'bored to tear
s', 'born and raised', 'born with a silver spoon in your mouth', 'born yesterday',
'bottom line', 'brain drain', 'brain dump', 'brass tacks', 'bring home the bacon',
'broken record', "brother's keeper", 'bull by the horns', 'bull in a china shop', 'b
ump in the night', 'busy as a bee', 'but seriously', 'by and large', 'darkest before
the dawn', 'dead as a doornail', 'death and destruction', 'death and taxes', "deat
h's doorstep", 'devil is in the details', 'dim view', 'dog days', 'dog in the mange
r', "don't count your chickens before they're hatched", "don't do the crime if you c
an't do the time", 'doubting Thomas', 'down and dirty', 'down in the dumps', 'down p
at', 'down the toilet', 'down the drain', 'down the hatch', 'down to earth', 'drive
you up a wall', 'dutch uncle', 'dyed in the wool', 'garbage in', 'get the sack', 'ge
t your groove back', 'gets my goat', 'gift horse in the mouth', 'gilding the lily',
'give a damn', 'give me a break', 'gives me the creeps', 'go him one better', 'goes
without saying', 'good deed for the day', 'good time was had by all', 'greek to me',
'green thumb', 'green-eyed monster', 'grist for the mill', 'guiding light', 'Jack of
all trades', 'jockey for position', 'Johnny-come-lately', 'joined at the hip', 'jump
down your throat', 'jump in with both feet', 'jump on the bandwagon', 'jump the gu
n', 'jump his bones', 'jump her bones', 'junk in the trunk', 'jury is still out', 'j
ustice is blind', 'labor of love', 'lap of luxury', 'last but not least', 'last-ditc

h effort', 'last hurrah', 'law of the jungle', 'law of the land', 'lay down the la
w', 'leaps and bounds', 'let sleeping dogs lie', 'let the cat out of the bag', "le
t's split", 'liberal media', 'lie like a rug', 'life and limb', 'life imitates art',
'neat as a pin', 'needless to say', 'nip it in the bud', 'no guts', 'no love lost',
'no pain', 'no stone unturned', 'no time like the present', 'nose to the grindston
e', 'not in my back yard', 'not on your tintype', 'number one fan', 'numerous to men
tion', 'radical chic', 'rags to riches', 'raining buckets', 'raining cats and dogs',
'rank and file', 'read my lips', 'red herring', 'redheaded stepchild', 'reign suprem
e', 'remember the alamo', 'road to hell is paved with good intentions', 'rob Peter t
o pay Paul', 'rock and a hard place', 'rocket scientist', 'rocket science', 'rope a
dope', 'run it up the flagpole', 'running dog lackey', 'squeaky wheel gets the oil',
'squeaky wheel gets the grease', 'start from scratch', 'stick in the mud', 'stick in
your craw', 'still waters run deep', 'stop and smell the roses', 'store bought', 'st
ranger than fiction', "straw that broke the camel's back", 'stubborn as a mule', 'st
uff that dreams are made of', 'stuffed shirt', 'tongue-in-cheek', 'too hot to handl
e', 'touch of blarney', 'tough as nails', 'tough luck', 'tough row to hoe', 'traditi
onal family values', 'trials and tribulations', 'tried and true', 'trip down memory
lane', 'true blue', 'turn your smile upside-down', 'turn your frown upside-down', 't
wist of fate', 'twists and turns', 'two to tango', 'who has everything', 'whole ball
of wax', 'whole hog', 'whole nine yards', 'whole other story', 'wild goose chase',
'wild oats', 'will wonders never cease', 'wimp', 'win friends and influence', 'peopl
e', 'win one for the Gipper', 'wnning is everything', 'wisdom of the ages', 'without
benefit of clergy', 'wolf at the door', 'words fail', 'work like a dog', 'worst nigh
tmare', 'wrong side of the bed']
['abodement', 'abram', 'abroach', 'abrook', 'absey-book', 'aby', 'accdant', 'accit
e', 'accomplement', 'accompt', 'accouter', 'accoutre', 'Acheron', 'acknow', 'aconitu
m', 'acquittance', 'acture', 'Actus Primus', 'Actus Quartus', 'Actus Quintus', 'Actu
s Secundus', 'Actus Tertius', 'addle', 'adieu', 'adoptious', 'advocation', 'aery',
'afeared', 'affiance', 'affined', 'affy', 'agate', 'agazed', 'aglet', 'agood', 'agu
e', 'aiery', 'alack', 'alarum', 'Albion', 'alderliefest', 'alder-liefest', 'alder-li
evest', 'alewife', 'ale-wife', 'allayment', 'allegiant', 'allottery', 'Almain', 'alm
s drink', 'almsdrink', 'almshouse', 'amain', 'amerce', 'ames-ace', 'amort', 'anatomi
ze', 'ancient', 'a-night', 'annexment', 'anon', 'antagonist', 'antechamber', 'Anthro
pophagi', 'Anthropophaginian', 'antic', 'antick', 'anticked', 'antre', 'appeach', 'a
ppellant', 'apperil', 'apple-john', 'approof', 'apricock', 'aqua vitae', 'aqua-vita
e', 'argal', 'Argier', 'argosy', 'aright', 'armigero', 'aroint', 'a-row', 'arrant',
'arras', 'aside', 'asinego', 'assay', 'assemblance', 'assinego', 'assubjugate', 'ato
mies', 'attaint', 'attainture', 'atwain', 'a-twain', 'audit', 'auld', 'avaunt', 'avo
uch', 'aye', 'bacare', 'baccare', 'bacchanal', 'backare', 'baffle', 'baggage', 'Baja
zet', 'balcony in a Shakespearean theater', 'ballow', 'balsamum', 'ban', 'bandog',
'ban-dog', 'banns', 'Barbary', 'Barbason', 'barbermonger', 'barber-monger', 'barm',
'barnes', 'barricado', 'basilisk', 'basta', 'bastard', 'bastardy', 'bastinado', 'bat
e', 'bat-fowling', 'batler', 'batlet', 'battaille', 'battalia', 'bavin', 'bawbling',
'bawcock', 'bawdry', 'beadsman', 'beam', 'bearing cloth', 'bearing-cloth', 'beaver',
'bed of Ware', 'bed-hangings', 'bedlam', 'beetle', 'befall', 'begirt', 'behoveful',
'beldam', 'belike', 'bemadding', 'bemete', 'be-mete', 'bemoil', 'be-moil', 'be-nette
d', 'benison', 'Bergomask', 'bescreened', 'be-screened', 'beseech', 'beshrew', 'besl
ubber', 'besort', 'bespake', 'bespeak', 'bespoke', 'bestraught', 'betake', 'beteem',
'betid', 'betide', 'betimes', 'betossed', 'betwixt', 'bewray', 'bezonian', 'biding',
'biggen', 'bilberry', 'bilbo', 'bill', 'birdbolt', 'bird-bolt', 'bisson', 'blench',
'bob', 'bodement', 'bodge', 'Bodkin', 'boÓtier vert', 'bolster', 'bolted', 'bolter',
'bolting hutch', 'bolting-hutch', 'bombard', 'bona roba', 'bona-roba', 'bondman', 'b
onny', 'boot', 'bootless', 'bosky', 'botcher', 'bots', 'bottled spider', 'bounden',
'bourn', 'brabble', 'brach', 'brainish', 'brainpan', 'brain-pan', 'brake', 'braver
y', 'breese', 'Bretagne', 'brinded', 'brinish', 'Britaine', 'Brittany', 'brock', 'br
ogue', 'bruit', 'bubukle', 'buckler', 'buckram', 'bum', 'bung', 'burden as a music t

erm', 'burgonet', 'burthen', 'buskin', 'busky', 'buss', 'buzzer', 'by-room', 'cacoda
emon', 'cacodemon', 'caddis', 'caitiff', 'Calipolis', 'caliver', 'callat', 'callet',
'Cam', 'cambric', 'camlet', 'canakin', 'canary', 'canis', 'cannikin', 'canzonet', 'c
ap-a-pie', 'capon', 'carbuncle', 'carcanet', 'cardmaker', 'carlot', 'carract', 'car
t', 'carviare', 'casing', 'casque', 'cataplasm', 'catch', 'catchword', 'cater', 'cat
er-cousin', 'caterwauling', 'cautel', 'cautelous', 'caviary', 'cavil', 'cellarage',
'cellar-age', 'century', 'cerecloth', 'cerement', 'certes', 'chace', 'champain', 'ch
anticleer', 'chantry', 'chape', 'chapfallen', "chap-fall'n", 'chapless', 'chapman',
'chapt', 'charactery', 'charneco', 'charnel house', 'charnel-house', 'cheveril', 'ch
id', 'chine', 'chirurgeon', 'chopine', 'chorus', 'chough', 'chuck', 'cicatrice', 'ci
nque-pace', 'Cinque-ports', 'circummure', 'civet', 'clackdish', 'clack-dish', 'clep
e', 'clept', 'clew', 'climature', 'clinquant', 'clog', 'cloistress', 'clotpoll', 'cl
out', 'clouted brogues', 'clovest', 'cloyless', 'cockatrice', "cockerel's stone", 'c
ockle', 'cockle hat', 'cockleshell', 'cockney', 'cock-shut time', 'codpiece', 'coffe
r', 'cog', 'cogging', 'coign', 'coil', 'collop', 'combinate', 'complice', 'complot',
'compulsative', 'con', 'concernancy', 'concupy', 'condign', 'coney', 'confixed', "co
nge'd", 'congeed', 'conger', 'congied', 'conjunctive', 'conjurations', 'conserve',
'consort', 'conspectuity', "constring'd", 'constringed', 'contemn', 'continuate', 'c
ontumeliously', 'conventicle', 'convive', 'cony', "copp'd", 'copped', 'coppice', 'co
ram', 'coranto', 'coronal', 'corse', 'corselet', 'costard', 'costermonger', 'cotquea
n', 'coulter', 'couplement', 'covert', 'coy', 'coystril', 'coystrill', 'coz', 'coze
n', 'cozenage', 'cozier', 'crare', 'crescive', 'crotchets', 'crown', 'crownet', 'cru
pper', 'crusado', 'cruzado', 'cubiculo', 'cuckold', 'cuckoo flower', 'cullion', 'dac
e', 'daff', 'dam', 'damosella', 'damson', 'Danskers', 'darnel', 'darraign', 'dauber
y', 'dauphin', 'debile', 'debosh', 'defeature', 'deflower', 'defunct', 'delated', 'd
elation', 'demi-wolf', 'denay', 'denier', 'denotement', 'depute', 'descant', 'design
ments', 'despatch', 'despite', 'determinate', 'diadem', 'dialogue', 'diaper', 'dic
h', 'dirge', 'disannul', 'disme', 'dispark', 'dispatch', 'disprized', 'disproperty',
'disquantity', 'dissentious', 'distaff', 'distained', 'distemperature', 'distract',
'distrain', 'ditch-dog', 'dive-dapper', 'divers', 'doit', 'dole', 'dormouse', 'dotan
t', 'doublet', 'dout', 'dower', 'dowlas', 'dowle', 'down-gyved', 'Downs', 'drab', 'd
rachma', 'draff', 'dram', 'dramatis personae', 'drave', 'drawer', 'drossy', 'drumbl
e', 'dry-beat', 'dryfoot', 'dudgeon', 'duello', 'dug', 'dump', 'dup', 'durance', 'du
rst', 'eale', 'eaning', 'eftest', 'eftsoons', 'eisel', 'eld', 'embarquement', 'embas
sage', 'emblaze', 'embossed', 'embowel', 'embracement', 'empiricutic', 'emulous', 'e
ncave', 'enchase', 'encloud', 'encompassment', 'enfeoff', 'engirt', 'englut', 'enme
w', 'enow', 'enrank', 'enroot', 'enround', 'entail', 'enter', 'enwombed', 'epilogu
e', 'epitheton', 'equipage', 'ere', 'erewhile', 'errant', 'erst', 'escot', 'esperanc
e', 'espial', 'essay', 'estridge', 'eterne', 'Ethiop', 'Ethiope', 'ever yet', 'every
et', 'exchequer', 'excursion', 'exeunt', 'exion', 'exit', 'exposture', 'exsufflicat
e', 'extirp', 'extraught', 'eyas', 'eyne', 'eyrie', 'facinorous', 'factionary', 'fad
ge', 'fain', 'fait', 'falchion', 'fallow', 'falsing', 'fap', 'farced', 'fardel', 'fa
rthing', 'fatigate', 'fay', 'fealty', 'feat', 'fecks', 'federary', "fee'd", 'feede
r', 'fee-grief', 'Fee-simple', 'felly', 'feodary', 'fere', 'festinately', 'fico', 'f
ie', 'fillip', 'filly foal', 'fine sovereign', 'firago', 'firk', 'fitchew', 'flapdra
gon', 'fleer', 'fleshment', 'flews', 'flexure', 'flirt-gill', 'flock', 'fob', 'fob o
ff', 'foh', 'foil', 'foin', 'foison', 'fool', 'foot', 'footboy', 'foot-cloth mule',
'footing', 'fordo', 'forsooth', 'fortnight', 'fosset', 'foughten', 'foul papers', 'f
oxship', 'fracted', 'frank', 'fraught', 'fraughtage', 'frippery', 'frontlet', 'frowa
rd', 'frush', 'fullam', 'fumiter', 'fumitory', 'fust', 'fustilarian', 'gaberdine',
'gad', 'gage', 'gainsay', 'gallery', 'galliard', 'galliass', 'galloglasses', 'gallo
w', 'gallowglasses', 'gamesome', 'gammon', 'gaoler', 'garboil', 'garner', 'gaskins',
'gasted', 'gawd', 'geck', 'gentle', 'gest', 'gib', 'giglot', 'gimmal', 'gimmer', 'gi
mmor', 'ging', 'gird', 'Gis', 'glaive', 'glanders', 'glave', 'gleek', 'glister', 'gl
ose', 'glow-worm', 'gloze', 'go to the world', 'gobbet', 'goblin', 'godwit', 'good d
en', 'good-year', 'gorbellied', 'gord', 'gourd', 'gramercy', 'gratis', 'graymalkin',

'greave', 'greenly', 'gripe', 'grise', 'grize', 'groat', 'guardant', 'gudgeon', 'gue
rdon', 'gules', 'gull', 'gyve', 'habiliment', 'habitude', 'haggard', 'hale', 'half-c
rown', 'halfpenny', 'halidom', 'halter', 'handsaw', 'hap', "ha'penny", 'hapless', 'h
aply', 'hard-handed', 'hardiment', 'harebell', 'hare-bell', 'harlock', 'harrow', 'ha
sle', 'haste-post-haste', 'hatchment', 'haught', 'hautbois', 'hautboy', 'haviour',
'head', 'head-lugged', 'hebona', 'hectic', 'hedge', 'hedge-pig', 'hedgepriest', 'hef
t', 'hefted', 'heigh', 'helm', 'hempen', 'henchman', 'hent', 'hest', 'hewgh', 'Hiem
s', 'high cross', 'high-cross', 'hight', 'hilding', 'hindmost', 'hither', 'hizz', 'h
odge pudding', 'hodge-pudding', 'hoise', 'holidame', 'holla', 'hollaing', 'holp', 'h
oly-ales', 'honey-stalks', 'honorificabilitudinitatibus', 'hornbook', 'horn-mad', 'h
orologe', 'hothouse', 'hot-house', 'house of resort', 'howbeit', 'hox', 'hoy', 'hugg
ermugger', 'hull', 'hurly', 'hurricano', 'hurtle', 'husband', 'huswife', 'hyen', 'Hy
rcanian beast', 'I wis', 'ignomy', "'ild", 'illume', 'imbar', 'imbare', 'imbrue', 'i
mmanity', 'immoment', 'imp', 'impartment', 'impasted', 'impawn', 'imperceiverant',
'impeticos', 'impleached', 'implorator', 'impone', 'importunacy', 'imposthume', 'imp
rese', 'impressure', 'imputation', 'incarnadine', 'incensement', 'incertain', 'inchm
eal', 'inch-meal', 'inclip', 'incony', 'incorpsed', 'increase', 'indent', 'indiffere
ncy', 'indigest', 'indign', 'indue', 'indurance', 'infamonize', 'inferreth', 'infort
unate', 'inhearse', 'Inhoop', 'injointed', 'inkhorn', 'inkle', 'inscroll', "inscul
p'd", 'insculped', 'insculpted', 'insculpture', "insinew'd", 'insinewed', 'insistur
e', "insteep'd", 'insteeped', 'intelligencer', 'intendment', 'intenible', 'intrins
e', 'inventorially', 'iron crow', 'irregulous', 'issue', 'iwis', 'jack-a-lent', 'jac
ks', 'jade', 'jakes', 'jar', 'jaunce', 'jerkin', 'jester', 'jet', 'jocund', 'John-a-
dreams', 'joint-ring', 'jordan', 'jot', 'jutty', 'juvenal', 'kam', 'kecksy', 'keec
h', 'keel', 'Keisar', 'ken', 'keptst', 'kern', 'kersey', 'kibe', 'kicky-wicky', 'kin
e', 'kirtle', 'knack', 'knap', 'knight-errant', 'la', 'labras', 'lakin', 'Lammas',
'Lammas-tide', 'lampass', 'lank', 'lanthorn', 'lapwing', 'lass-lorn', 'lated', 'latt
en', 'laund', 'lazar', 'leagued', 'leather coat', 'leather-coat', 'leathern', 'leave
s', 'lees', 'leet', 'legerity', 'leiger', 'leman', "l'envoy", 'levy', 'libbard', 'li
ct', 'lief', 'liege', 'liegeman', 'lifeling', 'limbeck', 'Limbo', 'lime', 'lineal',
'lineament', 'linstock', 'list', 'lists', 'lither', 'loach', 'loaden', 'lob', 'lockr
am', 'lodestar', 'lode-star', 'loggat', 'loggats', 'loof', 'lour', 'lout', 'lown',
'lowt', 'lozel', 'lubber', 'luce', 'lune', 'lustihood', 'lym', 'maggot-pie', 'magnif
ico', 'Mahu', 'main', 'mained', 'makeless', 'malapert', 'malkin', 'malmsey', 'malt w
orm', 'mammer', 'mammet', 'mammock', 'manakin', 'mandragora', 'manikin', 'Manningtre
e', 'marchpane', 'marish', 'marl', 'Marry', 'mart', 'martinmas', 'martlemas', 'martl
et', 'Mary-buds', 'masque', 'matin', 'maugre', 'maw', 'mazzard', 'meacock', 'measle
s', 'medlar', 'meed', 'meetest', 'meetness', 'mehercle', 'meiny', 'mell', 'mercatant
e', 'mered', 'mete-yard', 'metheglin', 'methinks', 'mew', 'mickle', 'milch', 'milkpa
ps', 'minikin', 'minim', 'minimus', 'mirable', 'miscarry', 'misconster', 'misprise',
'misproud', 'missingly', 'mis-tempered', 'mobled', 'Modo', 'moe', 'moiety', 'moldwar
p', 'mome', 'momentany', 'monstruosity', 'montant', 'mooncalf', 'Morisco', 'morris',
'morris-pike', 'morrow', 'mote', 'mow', 'mummy', 'muniment', 'mure', 'murrain', 'mur
ther', 'musk rose', 'musk-rose', 'mutine', 'nayward', 'nayword', 'neaf', 'near-legge
d', 'neat', 'needly', 'neeld', 'neeze', 'neif', 'netherstocks', 'nether-stocks', 'no
ddle', 'nonce', 'nose', "not'st", 'nousle', 'novum', 'nowl', 'noyance', 'nuncle', 'n
uthook', 'obsequies', 'oeilliade', 'oeilliades', "o'ercrow", "o'erleaven", "o'er-lea
ven", "o'er-raught", "o'er-read", "o'erskip", "o'ertrip", 'oneyer', 'ope', 'oppugnan
cy', 'orange-wife', 'ordinant', 'orgulous', 'orison', 'orthographer', 'ostler', 'oup
h', 'ouphe', 'ousel-cock', 'outface', 'outsport', 'outwall', 'out-wall', 'overborn
e', 'overcrow', 'overglance', 'overleather', 'overleaven', 'over-raught', 'overrea
d', 'overskip', 'overtrip', 'overweening', 'over-weening', 'paction', 'paddock', 'pa
jock', 'pale', 'pall', 'palliament', 'palmy', 'palter', 'paly', 'pantler', 'panyn',
'panynæ', 'pap', 'paraquito', 'parcel', 'pardonnez-moi', "'parit", 'parle', 'parlin
g', 'parmaceti', 'Parthia', 'partisan', 'partizan', 'pash', 'passado', 'passymeasur
e', 'passy-measure', 'pastern', 'patch', 'patine', 'pauca', 'pavan', 'pavane', 'pavi

n', 'peat', 'pedascule', 'peise', 'pell-mell', 'penny', 'pennyworth', 'penny-worth',
'peradventure', 'perdie', 'perdona-mi', 'perdu', 'perdurable', 'perdy', 'periapt',
'perpend', 'persever', 'perspective', 'pestiferous', 'petar', 'petard', 'pettish',
'pew-fellow', 'pheeze', 'Phoebe', 'physic', 'Picht-hatch', 'pickthank', 'pick-than
k', 'Pickt-hatch', 'pight', 'pilchard', 'pill', 'pinfold', 'pinnace', "pin's fee",
'pioned', 'pioner', 'pippin', 'pish', 'pismire', 'pittikin', 'placket', 'plain', 'pl
aining', 'planched', 'plats', 'plausive', 'pleached', 'pleb', 'plebeii', 'plurisy',
'poesy', 'point-device', 'poise', 'poke', 'politic', 'politicly', 'poll', 'polled',
'pomewater', 'poniard', 'pontifical', 'poor John', 'poperin', 'popinjay', 'porpentin
e', 'porringer', 'portance', 'posset', 'post horse', 'postern', 'post-horse', 'potat
ion', 'potch', 'potting', 'pottle', 'poulter', 'pouncet box', 'pouncet-box', 'poun
d', 'power', 'practisant', 'pratest', "prat'st", 'preceptial', 'prejudicate', 'preno
minate', 'prentice', "'prentice", 'cardecu', 'coif', 'quait', "quart d'?cu", 'quat',
'quatch', 'quean', 'quern', 'questant', 'questrist', 'quiddit', 'quillet', 'quintai
n', 'quire', 'quittance', 'quiver', 'quoif', 'quoits', 'quondam', 'quotha', 'quoth-
a', 'quotidian', 'rabato', 'rabbit-sucker', 'rack', 'rampallian', 'raught', 'ravin',
'rearmouse', 'rear-mouse', 'reave', 'rebato', 'recheat', 'reck', 'recordation', 'rec
reant', 'red plague', 'rede', 'reechy', 'reel', 'refel', 'reft', 'reguerdon', 'rejou
rn', 'relique', 'remember', 'remembrancer', 'remotion', 'repair', 'repugn', 'reputel
ess', 'reremouse', 'rere-mouse', 'reverb', 'Rhenish', 'rheum', 'rial', 'rib', 'riban
d', 'riggish', 'rigol', 'rivality', 'robustious', 'roe', 'roist', 'romage', 'ronyo
n', 'rood', 'rooky', 'ropery', 'roundure', 'rowel', 'royal', 'roynish', 'rub', 'rudd
ock', 'rudesby', 'ruff', 'runagate', 'rushling', 'ruth', 'ryal', 'sackbut', 'sacring
bell', 'sain', "Saint Colme's Inch", 'saith', 'Sala', 'Salique Law', 'sallet', 'sanc
tuarize', 'sans', 'savour', 'saw', 'scab', 'scaffoldage', 'scald', 'scambling', 'sca
mel', 'scant', 'scantling', 'scape', "'scape", 'scarf up', 'scarfed', 'scathful', 's
conce', 'scotch', 'scrimer', 'scrip', 'scrowl', 'scroyle', 'scull', 'scullion', 'scu
rril', 'scutch', 'sea gown', 'sea-gown', 'sea-mark', 'seel', 'self', 'semblable', 's
emblative', 'sennet', 'senseless-obstinate', 'sequent', 'sequester', 'serpigo', 'ses
sa', 'Setebos', 'setter', 'several', 'shale', 'shards', "shark'd", 'sharked', 'sheal
ed', 'shelvy', 'shent', "sheriff's post", 'shilling', 'shive', 'shoon', 'shotten',
'shough', 'shovel board', 'shovel-board', 'shrieve', 'shrift', 'shrive', 'shunless',
'sicle', 'signiory', 'signory', 'simples', 'simular', 'sirrah', 'sith', 'sithence',
'sixpence', 'skainsmates', 'skains-mates', 'skeins-mates', 'skimble-skamble', 'skink
er', 'skirr', 'sleided', 'sleight', 'sliding', 'slips', 'slops', 'slovenry', 'slutti
sh', 'smatch', 'smilets', 'smit', 'smooth', 'smoothing', 'Smulkin', 'smutch', 'snea
p', 'sneaping', 'sneck up', 'snuff', 'soft', 'soilure', 'solidare', 'sonties', 'soot
h', 'sortance', 'souse', 'sowl', 'sowle', 'spavin', 'speken', 'sperr', 'spilth', 'sp
irt', 'spital', 'splenitive', 'spongy', 'springe', 'springhalt', 'squier', 'squiny',
'stale', 'startingly', 'statist', 'statua', 'staves', 'stell', 'stile', 'stillitor
y', 'stilly', 'stoccado', 'stomacher', 'stonebow', 'stone-bow', 'stoup', 'strappad
o', 'straw', 'strond', 'strucken', 'strumpet', 'subscription', 'succour', 'sumpter',
'superflux', 'supervise', 'suppliance', 'supportance', 'supposal', 'sur-', 'suranc
e', 'surplice', 'suspire', 'suum cuique', 'swag', 'swain', 'swashing', 'sweeting',
'Switzer', 'swound', 'tabor', 'tabour', 'tainture', 'tallow', 'tang', 'tanned', 'tar
ge', 'Tarpeian', 'tarre', 'tarry', 'teem', 'tench', 'tendance', 'tent', 'Termagant',
'teston', 'testoon', 'tetter', 'Tewkesbury mustard', 'thane', 'tharborough', 'Thasso
s', 'theorick', 'thereunto', 'therewithal', 'thews', 'thine', 'thither', 'thorough',
'thou', "thou'dst", 'thraldom', 'thrall', 'thrasonical', 'three-man beetle', 'threep
ence', 'threne', 'thrum', 'thrummed hat', 'thy', 'thy', 'thyself', 'tilt', 'tilter',
'tiltyard', 'tilt-yard', 'tirrit', 'tortive', 'touse', 'toy', 'traject', 'trencher',
'trollmydames', 'troll-my-dames', 'troth', 'trow', 'Troyan', 'truckle-bed', 'truepen
ny', 'trull', 'truncheon', 'trundle-tail', 'tuck', 'tucket', 'tun', 'tundish', 'tu
p', 'Turlygood', 'twangling', 'twelvemonth', 'twiggen', 'twilled', 'twink', 'twire',
'twit', 'twopence', "unanel'd", 'unaneled', 'unbarbed', 'unbated', 'unbodied', 'unbo
lted', 'unbreathed', 'uncape', 'unclew', 'uncoined', 'uncurrent', 'underfiends', 'un

```
derskinker', 'under-wrought', 'uneath', 'unfeigned', 'ungenitured', 'ungravely', 'un
hatched', "unhousel'd", 'unhouseled', 'unlineal', 'unmanned', 'unmuzzle', 'unparagon
ed', 'unpossessing', 'unprizable', 'unprizeable', 'unproperly', 'unseminared', "unse
minar'd", 'unsifted', 'unsinewed', "unsinew'd", 'unsisting', 'unstanched', 'unstaunc
hed', 'untemper', 'untempering', 'unthrift', 'upspring reel', 'usance', 'utis', 'vad
e', 'vail', 'vantbrace', 'varlet', 'vassalage', 'vastidity', 'vault', 'vault', 'vaul
t', 'vault', 'vaultage', 'vaulty', 'vaunt-courier', 'vaward', 'vegetives', 'velure',
'vendible', 'venew', 'veney', 'ventage', 'verger', 'verily', 'vesper', 'vesture', 'v
ia', 'viands', 'victuals', 'videlicet', 'viewless', 'villany', 'vinewedst', 'vinnewe
d', 'vinewed', 'virginalling', 'vizament', 'vizard', 'votary', 'vouchsafe', 'waftag
e', 'wafture', 'waggoner', 'wailful', 'wainropes', 'wain ropes', 'wall-eyed', 'wall-
newt', 'Walloon', 'wanion', 'wanned', 'wappened', 'wannion', "wann'd", "wappen'd",
'warrantise', 'water-galls', 'water-rug', 'water-work', 'weal', 'wealsman', 'weeds',
'weet', 'welkin', 'Welsh hook', 'westward-ho!', 'wezand', 'wesand', 'weasand', 'whea
ten', 'Wheeson', 'whelk', 'whence', "wh'er", "whe'r", 'wherefore', 'wheyface', 'whey
-face', 'whiffler', 'while-ere', 'whiles', 'whimple', 'whipstock', 'whist', 'whitin
g', 'whitster', 'whittle', 'whoo-bub', 'whoobub', 'wight', 'willow garland', 'window
-bars', 'windring', 'winter-ground', 'wis', 'wistly', 'withal', 'wittol', 'womby',
'wont', 'wont', 'wonted', 'woodbine', 'woodcock', 'woolward', 'wot', 'wrack', 'wrack
ful', 'writhled', 'wroth', 'yare', 'y-clad', 'yclept', 'yclad', 'ycleped', 'yellow
s', 'yeoman', 'yeoman', 'yeoman', 'yerk', 'yesty', 'yond', 'younker', 'zounds']
```

Next, we will also need to lemmatise all the phrases in each phrase list for comparison (as the answers will also be lemmatised)

First, we will define a function to lemmatise all phrases in a phrase list and call it on the phrase lists

In [ ]:
```
def lemmatise_phrase_list(phrase_list):
    output_list = []
    lemmatizer = WordNetLemmatizer()
    for phrase in phrase_list:
        tokens = word_tokenize(phrase)
        lemmatised_tokens = [lemmatizer.lemmatize(token) for token in tokens]
        output_list.append(' '.join(lemmatised_tokens))

    return output_list
```

In [ ]:
```
cliches_lemmatised = lemmatise_phrase_list(cliches)
ancient_lemmatised = lemmatise_phrase_list(ancient_phrases)

print(cliches_lemmatised)
print(ancient_lemmatised)
```

['ace in the hole', 'ace up your sleeve', 'acid test', 'airing dirty laundry', "all in a day 's work", 'all talk', 'all booster', 'all hat', 'all foam', 'all hammer', 'all icing', 'all lime and salt', 'all missile', 'all shot', 'all sizzle', 'all wax and no wick', 'all that and a bag of chip', 'all thumb', 'all wet', "all 's fair in love and war", 'almighty dollar', 'always a bridesmaid', 'ambulance chaser', 'another day', 'ant in your pant', 'apple-pie order', 'arm and a leg', 'armchair quarterback', 'army brat', 'art imitates life', 'artsy-craftsy', 'artsy-fartsy', 'a luck would have it', 'a old a time', 'at loggerhead', 'calm before the storm', 'candle at both end', "ca n't cut the mustard", 'case of mistaken identity', 'cat out of the bag', 'cat got your tongue', 'caught red-handed', 'chapter and verse', 'checkered career', 'chicken come home to', 'roost', 'chomping at the bit', 'cleanliness is next to', 'godliness', 'clear a a bell', 'clear a mud', 'cold shoulder', 'E-ticket', 'ear to the ground', 'early bird catch the worm', 'easier said than done', 'easy a 1-2-3', 'easy a pie', 'eat crow', 'eat humble pie', 'enough already', 'even money', 'every dog hait day', 'every fiber of my being', 'everything but the kitchen sink', 'evil twin', 'existential angst', 'expert agreeï¿½', 'eye for an eye', 'hair of the dog', 'hard to believe', 'have a nice day', 'head honcho', "heart 's content", 'hell-bent for leather', 'hidden agenda', 'high and the mighty (the)', 'high on the hog', 'hold a candle to', 'hold your horse', 'hold your tongue', 'hook or by crook', 'horse of a different color', 'hot knife through butter', 'how go the battle ?', 'keep an eye on you', 'keep it down', 'keep it simple', 'keep up with the Joneses', 'keep your card close to vestï¿½', 'keep your chin up', 'keep your finger crossed', 'keep your powder dry', 'kick as', 'kickbutt', 'kick the bucket', 'kick up your heel', 'kick you to the curb', 'kick your foot up', 'kid in a candy store', 'kill two bird with one stone', "life 's a bitch", 'lighten up', 'light out', 'like a sore thumb', 'like butter', 'like the plague', "like there 's no tomorrow", "lion 's share", 'litmus test', 'little black book', 'live and learn', 'long and short of it', 'long lost love', 'look before you leap', 'lounge lizard', 'loved and lost', 'low man on the totem pole', 'luck of the draw', 'luck of the Irish', 'off the wagon', 'old college try', 'old meet new', 'older and wiser', 'older than dirt', 'older than Methuselah', 'on the bandwagon', 'on the nose', 'on the wagon', 'on thin ice', 'one born every minute', 'one foot in the grave', 'one in a million', 'only game in town', 'only to be met', 'out of pocket', 'out of the frying pan', 'out on a limb', 'safe than sorry', 'salt of the earth', 'save face', 'scared stiff', 'scared to death', "school 's out", 'screaming meemies', 'sens reel', 'set the record straight', 'shake a stick should of', 'shoulder to the wheel', 'shouldered his way', 'shut your trap', 'sigh of relief', 'significant other', 'silence is golden', 'take one for the team', 'take the bull by the horn', 'take the plunge', 'take one to know one', 'talk turkey', 'ten foot pole', 'the earth moved', 'the final analysis', 'the real McCoy', 'the same old story', 'these thing happen', 'thick a thief', 'think outside of the box', "third time 's the charm", 'under the gun', 'under the same roof', 'understated elegance', 'unexpected twist', 'until the cow come home', 'up his sleeve', 'up the creek', 'up the wrong tree', 'very real concern', 'view with alarm', 'babe in the wood', 'back against the wall', 'back in the saddle', 'back to square one', 'back to the drawing board', 'bad to the bone', 'badge of honor', 'Badonkadonk', 'ballpark figure', 'ball to the wall', 'baptism of fire', 'bare bone', 'bark is worse than the bite', 'bark up the wrong tree', 'bat out of hell', 'bat in the belfry', 'battle royal', 'beat around the bush', 'beat the bush', 'beat me', 'behind the eight ball', 'bent out of shape', 'best foot forward', 'bet your bottom dollar', 'better half', 'better late than never', 'better mousetrap', 'better safe than sorry', 'better than ever', 'better the Devil you know', 'between a rock and a hard place', 'beyond the pale', 'bib and tucker', 'big a life', 'big fish in a small pond', 'big man on campus', 'bigger they are', 'communist conspiracy', 'conniption fit', 'could care le', "could n't care le", "could n't get to first base", 'count your blessing', 'countless hour', 'creature comfort', 'crime in the street', 'curiosity killed the cat', 'curry favor', 'cut a fine figure', 'cut and dried', 'cut to the chase', 'cut to the quick', 'cute a a button', 'fact of lif

e', 'fair-haired one', 'fair weather friend', 'fall off of a turnip truck', 'fat slo
b', 'favor u with a song', 'fear and loathing', 'feather your nest', 'fellow travele
r', 'few and far between', 'field this one', 'fifteen minute of fame', 'fish nor fow
l', 'fly by night', 'fly the coop', 'for the bird', 'fox in the henhouse', 'freudian
slip', 'fun and game', 'fun in the sun', 'I beg to differ', 'if the shoe fit', "I 'm
okay", 'in a nutshell', 'in a pinch', 'in a wink', "in harm 's way", 'in the tank',
'in your dream', 'in your face', 'inexorably drawn', 'info dump', 'influence peddlin
g', 'intent and purpose', 'it wa a dark and stormy night', "it wo n't fly", "King 's
English", "king 's ransom", 'kiss and tell', 'kiss as', 'kiss of death', 'kit and ka
boodle', 'knee-high to a grasshopper', 'knock it out of the park', 'knock on wood',
'knock your sock off', 'knocked up', 'know him from Adam', 'know the rope', 'know th
e score', 'knuckle down', 'knuckle sandwich', 'knuckle under', 'make my day', 'male
chauvinist', "man 's best friend", 'many moon', 'many-splendored thing', 'mark my wo
rd', 'meaningful relationship', 'mellow out', 'moment of glory', "moment 's respit
e", 'Monday morning quarterback', 'monkey suit', 'monkey see', 'motherhood and apple
pie', 'mover and shaker', 'moving experience', 'my two cent', "p 's and q 's", 'pain
and suffering', 'panic button', 'party pooper', 'patter of little foot', 'pas the sn
iff test', 'pay through the nose', 'pea in a pod', 'perfect storm', 'pig in a poke',
'pillar of society', 'pinko', 'plenty of fish in the sea', 'poison pen', 'poor a a c
hurchmouse', 'poor excuse for', 'pot calling the kettle black', 'proud possessor',
'put my foot down', 'put your foot down', 'quick a a bunny', 'quick and the dead',
'slept like a log', 'small world', 'snake in the grass', 'snow job', 'snug a a bug',
'some of my best friend', 'something the cat dragged in', 'spade a spade', 'spare th
e rod', 'spitting image', 'spring to life', 'this day and age', 'this point in tim
e', "three strike and you 're out", 'through the grapevine', 'throw in the towel',
'tiger by the tail', 'till the fat lady sings', 'time and time again', 'time is of t
he essense', 'tip of the iceberg', "'t is the season", 'to err is human', 'to the be
st of my knowledge', 'wakeup call', 'wa my face red', 'watch your tongue', 'web of i
ntrigue', 'week of sunday', 'what a bummer', 'what come around', 'what the cat dragg
ed in', 'what the dickens', 'what the heck', 'what the hell', 'what you see is what
you get', "what 's not to like", 'wheeler-dealer', 'when in doubt', 'when push come
to shove', 'when rubber meet the road', "when the cat 's away", 'when the going get
tough', 'bird in the hand', 'bird and the bee', 'bird of the feather', 'bite the dus
t', 'bite your tongue', 'bitter disappointment', 'black a coal', 'blast from the pas
t', 'bleeding heart', 'blind a a bat', 'blood is thicker than water', 'blood money',
'blood on your hand', 'blood sweat and tear', 'blow this pop stand', 'blow this join
t', 'blushing bride', 'boil it down to', 'bone of contention', 'booze and broad', 'b
ored to tear', 'born and raised', 'born with a silver spoon in your mouth', 'born ye
sterday', 'bottom line', 'brain drain', 'brain dump', 'brass tack', 'bring home the
bacon', 'broken record', "brother 's keeper", 'bull by the horn', 'bull in a china s
hop', 'bump in the night', 'busy a a bee', 'but seriously', 'by and large', 'darkest
before the dawn', 'dead a a doornail', 'death and destruction', 'death and tax', "de
ath 's doorstep", 'devil is in the detail', 'dim view', 'dog day', 'dog in the mange
r', "do n't count your chicken before they 're hatched", "do n't do the crime if you
ca n't do the time", 'doubting Thomas', 'down and dirty', 'down in the dump', 'down
pat', 'down the toilet', 'down the drain', 'down the hatch', 'down to earth', 'drive
you up a wall', 'dutch uncle', 'dyed in the wool', 'garbage in', 'get the sack', 'ge
t your groove back', 'get my goat', 'gift horse in the mouth', 'gilding the lily',
'give a damn', 'give me a break', 'give me the creep', 'go him one better', 'go with
out saying', 'good deed for the day', 'good time wa had by all', 'greek to me', 'gre
en thumb', 'green-eyed monster', 'grist for the mill', 'guiding light', 'Jack of all
trade', 'jockey for position', 'Johnny-come-lately', 'joined at the hip', 'jump down
your throat', 'jump in with both foot', 'jump on the bandwagon', 'jump the gun', 'ju
mp his bone', 'jump her bone', 'junk in the trunk', 'jury is still out', 'justice is
blind', 'labor of love', 'lap of luxury', 'last but not least', 'last-ditch effort',
'last hurrah', 'law of the jungle', 'law of the land', 'lay down the law', 'leap and

bound', 'let sleeping dog lie', 'let the cat out of the bag', "let 's split", 'liber
al medium', 'lie like a rug', 'life and limb', 'life imitates art', 'neat a a pin',
'needle to say', 'nip it in the bud', 'no gut', 'no love lost', 'no pain', 'no stone
unturned', 'no time like the present', 'nose to the grindstone', 'not in my back yar
d', 'not on your tintype', 'number one fan', 'numerous to mention', 'radical chic',
'rag to rich', 'raining bucket', 'raining cat and dog', 'rank and file', 'read my li
p', 'red herring', 'redheaded stepchild', 'reign supreme', 'remember the alamo', 'ro
ad to hell is paved with good intention', 'rob Peter to pay Paul', 'rock and a hard
place', 'rocket scientist', 'rocket science', 'rope a dope', 'run it up the flagpol
e', 'running dog lackey', 'squeaky wheel get the oil', 'squeaky wheel get the greas
e', 'start from scratch', 'stick in the mud', 'stick in your craw', 'still water run
deep', 'stop and smell the rose', 'store bought', 'stranger than fiction', "straw th
at broke the camel 's back", 'stubborn a a mule', 'stuff that dream are made of', 's
tuffed shirt', 'tongue-in-cheek', 'too hot to handle', 'touch of blarney', 'tough a
nail', 'tough luck', 'tough row to hoe', 'traditional family value', 'trial and trib
ulation', 'tried and true', 'trip down memory lane', 'true blue', 'turn your smile u
pside-down', 'turn your frown upside-down', 'twist of fate', 'twist and turn', 'two
to tango', 'who ha everything', 'whole ball of wax', 'whole hog', 'whole nine yard',
'whole other story', 'wild goose chase', 'wild oat', 'will wonder never cease', 'wim
p', 'win friend and influence', 'people', 'win one for the Gipper', 'wnning is every
thing', 'wisdom of the age', 'without benefit of clergy', 'wolf at the door', 'word
fail', 'work like a dog', 'worst nightmare', 'wrong side of the bed']
['abodement', 'abram', 'abroach', 'abrook', 'absey-book', 'aby', 'accdant', 'accit
e', 'accomplement', 'accompt', 'accouter', 'accoutre', 'Acheron', 'acknow', 'aconitu
m', 'acquittance', 'acture', 'Actus Primus', 'Actus Quartus', 'Actus Quintus', 'Actu
s Secundus', 'Actus Tertius', 'addle', 'adieu', 'adoptious', 'advocation', 'aery',
'afeared', 'affiance', 'affined', 'affy', 'agate', 'agazed', 'aglet', 'agood', 'agu
e', 'aiery', 'alack', 'alarum', 'Albion', 'alderliefest', 'alder-liefest', 'alder-li
evest', 'alewife', 'ale-wife', 'allayment', 'allegiant', 'allottery', 'Almain', 'alm
s drink', 'almsdrink', 'almshouse', 'amain', 'amerce', 'ames-ace', 'amort', 'anatomi
ze', 'ancient', 'a-night', 'annexment', 'anon', 'antagonist', 'antechamber', 'Anthro
pophagi', 'Anthropophaginian', 'antic', 'antick', 'anticked', 'antre', 'appeach', 'a
ppellant', 'apperil', 'apple-john', 'approof', 'apricock', 'aqua vitae', 'aqua-vita
e', 'argal', 'Argier', 'argosy', 'aright', 'armigero', 'aroint', 'a-row', 'arrant',
'arras', 'aside', 'asinego', 'assay', 'assemblance', 'assinego', 'assubjugate', 'ato
mies', 'attaint', 'attainture', 'atwain', 'a-twain', 'audit', 'auld', 'avaunt', 'avo
uch', 'aye', 'bacare', 'baccare', 'bacchanal', 'backare', 'baffle', 'baggage', 'Baja
zet', 'balcony in a Shakespearean theater', 'ballow', 'balsamum', 'ban', 'bandog',
'ban-dog', 'banns', 'Barbary', 'Barbason', 'barbermonger', 'barber-monger', 'barm',
'barnes', 'barricado', 'basilisk', 'basta', 'bastard', 'bastardy', 'bastinado', 'bat
e', 'bat-fowling', 'batler', 'batlet', 'battaille', 'battalia', 'bavin', 'bawbling',
'bawcock', 'bawdry', 'beadsman', 'beam', 'bearing cloth', 'bearing-cloth', 'beaver',
'bed of Ware', 'bed-hangings', 'bedlam', 'beetle', 'befall', 'begirt', 'behoveful',
'beldam', 'belike', 'bemadding', 'bemete', 'be-mete', 'bemoil', 'be-moil', 'be-nette
d', 'benison', 'Bergomask', 'bescreened', 'be-screened', 'beseech', 'beshrew', 'besl
ubber', 'besort', 'bespake', 'bespeak', 'bespoke', 'bestraught', 'betake', 'beteem',
'betid', 'betide', 'betimes', 'betossed', 'betwixt', 'bewray', 'bezonian', 'biding',
'biggen', 'bilberry', 'bilbo', 'bill', 'birdbolt', 'bird-bolt', 'bisson', 'blench',
'bob', 'bodement', 'bodge', 'Bodkin', 'boÓtier vert', 'bolster', 'bolted', 'bolter',
'bolting hutch', 'bolting-hutch', 'bombard', 'bona roba', 'bona-roba', 'bondman', 'b
onny', 'boot', 'bootless', 'bosky', 'botcher', 'bot', 'bottled spider', 'bounden',
'bourn', 'brabble', 'brach', 'brainish', 'brainpan', 'brain-pan', 'brake', 'braver
y', 'breese', 'Bretagne', 'brinded', 'brinish', 'Britaine', 'Brittany', 'brock', 'br
ogue', 'bruit', 'bubukle', 'buckler', 'buckram', 'bum', 'bung', 'burden a a music te
rm', 'burgonet', 'burthen', 'buskin', 'busky', 'bus', 'buzzer', 'by-room', 'cacodaem
on', 'cacodemon', 'caddis', 'caitiff', 'Calipolis', 'caliver', 'callat', 'callet',

'Cam', 'cambric', 'camlet', 'canakin', 'canary', 'canis', 'cannikin', 'canzonet', 'c
ap-a-pie', 'capon', 'carbuncle', 'carcanet', 'cardmaker', 'carlot', 'carract', 'car
t', 'carviare', 'casing', 'casque', 'cataplasm', 'catch', 'catchword', 'cater', 'cat
er-cousin', 'caterwauling', 'cautel', 'cautelous', 'caviary', 'cavil', 'cellarage',
'cellar-age', 'century', 'cerecloth', 'cerement', 'certes', 'chace', 'champain', 'ch
anticleer', 'chantry', 'chape', 'chapfallen', "chap-fall'n", 'chapless', 'chapman',
'chapt', 'charactery', 'charneco', 'charnel house', 'charnel-house', 'cheveril', 'ch
id', 'chine', 'chirurgeon', 'chopine', 'chorus', 'chough', 'chuck', 'cicatrice', 'ci
nque-pace', 'Cinque-ports', 'circummure', 'civet', 'clackdish', 'clack-dish', 'clep
e', 'clept', 'clew', 'climature', 'clinquant', 'clog', 'cloistress', 'clotpoll', 'cl
out', 'clouted brogue', 'clovest', 'cloyless', 'cockatrice', "cockerel 's stone", 'c
ockle', 'cockle hat', 'cockleshell', 'cockney', 'cock-shut time', 'codpiece', 'coffe
r', 'cog', 'cogging', 'coign', 'coil', 'collop', 'combinate', 'complice', 'complot',
'compulsative', 'con', 'concernancy', 'concupy', 'condign', 'coney', 'confixed', "co
nge 'd", 'congeed', 'conger', 'congied', 'conjunctive', 'conjuration', 'conserve',
'consort', 'conspectuity', "constring 'd", 'constringed', 'contemn', 'continuate',
'contumeliously', 'conventicle', 'convive', 'cony', "copp 'd", 'copped', 'coppice',
'coram', 'coranto', 'coronal', 'corse', 'corselet', 'costard', 'costermonger', 'cotq
uean', 'coulter', 'couplement', 'covert', 'coy', 'coystril', 'coystrill', 'coz', 'co
zen', 'cozenage', 'cozier', 'crare', 'crescive', 'crotchet', 'crown', 'crownet', 'cr
upper', 'crusado', 'cruzado', 'cubiculo', 'cuckold', 'cuckoo flower', 'cullion', 'da
ce', 'daff', 'dam', 'damosella', 'damson', 'Danskers', 'darnel', 'darraign', 'dauber
y', 'dauphin', 'debile', 'debosh', 'defeature', 'deflower', 'defunct', 'delated', 'd
elation', 'demi-wolf', 'denay', 'denier', 'denotement', 'depute', 'descant', 'design
ments', 'despatch', 'despite', 'determinate', 'diadem', 'dialogue', 'diaper', 'dic
h', 'dirge', 'disannul', 'disme', 'dispark', 'dispatch', 'disprized', 'disproperty',
'disquantity', 'dissentious', 'distaff', 'distained', 'distemperature', 'distract',
'distrain', 'ditch-dog', 'dive-dapper', 'diver', 'doit', 'dole', 'dormouse', 'dotan
t', 'doublet', 'dout', 'dower', 'dowlas', 'dowle', 'down-gyved', 'Downs', 'drab', 'd
rachma', 'draff', 'dram', 'dramatis persona', 'drave', 'drawer', 'drossy', 'drumbl
e', 'dry-beat', 'dryfoot', 'dudgeon', 'duello', 'dug', 'dump', 'dup', 'durance', 'du
rst', 'eale', 'eaning', 'eftest', 'eftsoons', 'eisel', 'eld', 'embarquement', 'embas
sage', 'emblaze', 'embossed', 'embowel', 'embracement', 'empiricutic', 'emulous', 'e
ncave', 'enchase', 'encloud', 'encompassment', 'enfeoff', 'engirt', 'englut', 'enme
w', 'enow', 'enrank', 'enroot', 'enround', 'entail', 'enter', 'enwombed', 'epilogu
e', 'epitheton', 'equipage', 'ere', 'erewhile', 'errant', 'erst', 'escot', 'esperanc
e', 'espial', 'essay', 'estridge', 'eterne', 'Ethiop', 'Ethiope', 'ever yet', 'every
et', 'exchequer', 'excursion', 'exeunt', 'exion', 'exit', 'exposture', 'exsufflicat
e', 'extirp', 'extraught', 'eyas', 'eyne', 'eyrie', 'facinorous', 'factionary', 'fad
ge', 'fain', 'fait', 'falchion', 'fallow', 'falsing', 'fap', 'farced', 'fardel', 'fa
rthing', 'fatigate', 'fay', 'fealty', 'feat', 'fecks', 'federary', "fee 'd", 'feede
r', 'fee-grief', 'Fee-simple', 'felly', 'feodary', 'fere', 'festinately', 'fico', 'f
ie', 'fillip', 'filly foal', 'fine sovereign', 'firago', 'firk', 'fitchew', 'flapdra
gon', 'fleer', 'fleshment', 'flews', 'flexure', 'flirt-gill', 'flock', 'fob', 'fob o
ff', 'foh', 'foil', 'foin', 'foison', 'fool', 'foot', 'footboy', 'foot-cloth mule',
'footing', 'fordo', 'forsooth', 'fortnight', 'fosset', 'foughten', 'foul paper', 'fo
xship', 'fracted', 'frank', 'fraught', 'fraughtage', 'frippery', 'frontlet', 'frowar
d', 'frush', 'fullam', 'fumiter', 'fumitory', 'fust', 'fustilarian', 'gaberdine', 'g
ad', 'gage', 'gainsay', 'gallery', 'galliard', 'gallia', 'galloglasses', 'gallow',
'gallowglasses', 'gamesome', 'gammon', 'gaoler', 'garboil', 'garner', 'gaskin', 'gas
ted', 'gawd', 'geck', 'gentle', 'gest', 'gib', 'giglot', 'gimmal', 'gimmer', 'gimmo
r', 'ging', 'gird', 'Gis', 'glaive', 'glanders', 'glave', 'gleek', 'glister', 'glos
e', 'glow-worm', 'gloze', 'go to the world', 'gobbet', 'goblin', 'godwit', 'good de
n', 'good-year', 'gorbellied', 'gord', 'gourd', 'gramercy', 'gratis', 'graymalkin',
'greave', 'greenly', 'gripe', 'grise', 'grize', 'groat', 'guardant', 'gudgeon', 'gue
rdon', 'gules', 'gull', 'gyve', 'habiliment', 'habitude', 'haggard', 'hale', 'half-c

rown', 'halfpenny', 'halidom', 'halter', 'handsaw', 'hap', "ha'penny", 'hapless', 'h
aply', 'hard-handed', 'hardiment', 'harebell', 'hare-bell', 'harlock', 'harrow', 'ha
sle', 'haste-post-haste', 'hatchment', 'haught', 'hautbois', 'hautboy', 'haviour',
'head', 'head-lugged', 'hebona', 'hectic', 'hedge', 'hedge-pig', 'hedgepriest', 'hef
t', 'hefted', 'heigh', 'helm', 'hempen', 'henchman', 'hent', 'hest', 'hewgh', 'Hiem
s', 'high cross', 'high-cross', 'hight', 'hilding', 'hindmost', 'hither', 'hizz', 'h
odge pudding', 'hodge-pudding', 'hoise', 'holidame', 'holla', 'hollaing', 'holp', 'h
oly-ales', 'honey-stalks', 'honorificabilitudinitatibus', 'hornbook', 'horn-mad', 'h
orologe', 'hothouse', 'hot-house', 'house of resort', 'howbeit', 'hox', 'hoy', 'hugg
ermugger', 'hull', 'hurly', 'hurricano', 'hurtle', 'husband', 'huswife', 'hyen', 'Hy
rcanian beast', 'I wi', 'ignomy', "'ild", 'illume', 'imbar', 'imbare', 'imbrue', 'im
manity', 'immoment', 'imp', 'impartment', 'impasted', 'impawn', 'imperceiverant', 'i
mpeticos', 'impleached', 'implorator', 'impone', 'importunacy', 'imposthume', 'impre
se', 'impressure', 'imputation', 'incarnadine', 'incensement', 'incertain', 'inchmea
l', 'inch-meal', 'inclip', 'incony', 'incorpsed', 'increase', 'indent', 'indifferenc
y', 'indigest', 'indign', 'indue', 'indurance', 'infamonize', 'inferreth', 'infortun
ate', 'inhearse', 'Inhoop', 'injointed', 'inkhorn', 'inkle', 'inscroll', "insculp
'd", 'insculped', 'insculpted', 'insculpture', "insinew 'd", 'insinewed', 'insistur
e', "insteep 'd", 'insteeped', 'intelligencer', 'intendment', 'intenible', 'intrins
e', 'inventorially', 'iron crow', 'irregulous', 'issue', 'iwis', 'jack-a-lent', 'jac
k', 'jade', 'jakes', 'jar', 'jaunce', 'jerkin', 'jester', 'jet', 'jocund', 'John-a-d
reams', 'joint-ring', 'jordan', 'jot', 'jutty', 'juvenal', 'kam', 'kecksy', 'keech',
'keel', 'Keisar', 'ken', 'keptst', 'kern', 'kersey', 'kibe', 'kicky-wicky', 'kine',
'kirtle', 'knack', 'knap', 'knight-errant', 'la', 'labras', 'lakin', 'Lammas', 'Lamm
as-tide', 'lampass', 'lank', 'lanthorn', 'lapwing', 'lass-lorn', 'lated', 'latten',
'laund', 'lazar', 'leagued', 'leather coat', 'leather-coat', 'leathern', 'leaf', 'le
e', 'leet', 'legerity', 'leiger', 'leman', "l'envoy", 'levy', 'libbard', 'lict', 'li
ef', 'liege', 'liegeman', 'lifeling', 'limbeck', 'Limbo', 'lime', 'lineal', 'lineame
nt', 'linstock', 'list', 'list', 'lither', 'loach', 'loaden', 'lob', 'lockram', 'lod
estar', 'lode-star', 'loggat', 'loggats', 'loof', 'lour', 'lout', 'lown', 'lowt', 'l
ozel', 'lubber', 'luce', 'lune', 'lustihood', 'lym', 'maggot-pie', 'magnifico', 'Mah
u', 'main', 'mained', 'makeless', 'malapert', 'malkin', 'malmsey', 'malt worm', 'mam
mer', 'mammet', 'mammock', 'manakin', 'mandragora', 'manikin', 'Manningtree', 'march
pane', 'marish', 'marl', 'Marry', 'mart', 'martinmas', 'martlemas', 'martlet', 'Mary
-buds', 'masque', 'matin', 'maugre', 'maw', 'mazzard', 'meacock', 'measles', 'medla
r', 'meed', 'meetest', 'meetness', 'mehercle', 'meiny', 'mell', 'mercatante', 'mere
d', 'mete-yard', 'metheglin', 'methinks', 'mew', 'mickle', 'milch', 'milkpaps', 'min
ikin', 'minim', 'minimus', 'mirable', 'miscarry', 'misconster', 'misprise', 'misprou
d', 'missingly', 'mis-tempered', 'mobled', 'Modo', 'moe', 'moiety', 'moldwarp', 'mom
e', 'momentany', 'monstruosity', 'montant', 'mooncalf', 'Morisco', 'morris', 'morris
-pike', 'morrow', 'mote', 'mow', 'mummy', 'muniment', 'mure', 'murrain', 'murther',
'musk rose', 'musk-rose', 'mutine', 'nayward', 'nayword', 'neaf', 'near-legged', 'ne
at', 'needly', 'neeld', 'neeze', 'neif', 'netherstocks', 'nether-stocks', 'noddle',
'nonce', 'nose', "not'st", 'nousle', 'novum', 'nowl', 'noyance', 'nuncle', 'nuthoo
k', 'obsequies', 'oeilliade', 'oeilliades', "o'ercrow", "o'erleaven", "o'er-leaven",
"o'er-raught", "o'er-read", "o'erskip", "o'ertrip", 'oneyer', 'ope', 'oppugnancy',
'orange-wife', 'ordinant', 'orgulous', 'orison', 'orthographer', 'ostler', 'ouph',
'ouphe', 'ousel-cock', 'outface', 'outsport', 'outwall', 'out-wall', 'overborne', 'o
vercrow', 'overglance', 'overleather', 'overleaven', 'over-raught', 'overread', 'ove
rskip', 'overtrip', 'overweening', 'over-weening', 'paction', 'paddock', 'pajock',
'pale', 'pall', 'palliament', 'palmy', 'palter', 'paly', 'pantler', 'panyn', 'panyn
æ', 'pap', 'paraquito', 'parcel', 'pardonnez-moi', "'parit", 'parle', 'parling', 'pa
rmaceti', 'Parthia', 'partisan', 'partizan', 'pash', 'passado', 'passymeasure', 'pas
sy-measure', 'pastern', 'patch', 'patine', 'pauca', 'pavan', 'pavane', 'pavin', 'pea
t', 'pedascule', 'peise', 'pell-mell', 'penny', 'pennyworth', 'penny-worth', 'peradv
enture', 'perdie', 'perdona-mi', 'perdu', 'perdurable', 'perdy', 'periapt', 'perpen

d', 'persever', 'perspective', 'pestiferous', 'petar', 'petard', 'pettish', 'pew-fel
low', 'pheeze', 'Phoebe', 'physic', 'Picht-hatch', 'pickthank', 'pick-thank', 'Pickt
-hatch', 'pight', 'pilchard', 'pill', 'pinfold', 'pinnace', "pin 's fee", 'pioned',
'pioner', 'pippin', 'pish', 'pismire', 'pittikin', 'placket', 'plain', 'plaining',
'planched', 'plat', 'plausive', 'pleached', 'pleb', 'plebeii', 'plurisy', 'poesy',
'point-device', 'poise', 'poke', 'politic', 'politicly', 'poll', 'polled', 'pomewate
r', 'poniard', 'pontifical', 'poor John', 'poperin', 'popinjay', 'porpentine', 'porr
inger', 'portance', 'posset', 'post horse', 'postern', 'post-horse', 'potation', 'po
tch', 'potting', 'pottle', 'poulter', 'pouncet box', 'pouncet-box', 'pound', 'powe
r', 'practisant', 'pratest', "prat'st", 'preceptial', 'prejudicate', 'prenominate',
'prentice', "'prentice", 'cardecu', 'coif', 'quait', "quart d ' ? cu", 'quat', 'quat
ch', 'quean', 'quern', 'questant', 'questrist', 'quiddit', 'quillet', 'quintain', 'q
uire', 'quittance', 'quiver', 'quoif', 'quoit', 'quondam', 'quotha', 'quoth-a', 'quo
tidian', 'rabato', 'rabbit-sucker', 'rack', 'rampallian', 'raught', 'ravin', 'rearmo
use', 'rear-mouse', 'reave', 'rebato', 'recheat', 'reck', 'recordation', 'recreant',
'red plague', 'rede', 'reechy', 'reel', 'refel', 'reft', 'reguerdon', 'rejourn', 're
lique', 'remember', 'remembrancer', 'remotion', 'repair', 'repugn', 'reputeless', 'r
eremouse', 'rere-mouse', 'reverb', 'Rhenish', 'rheum', 'rial', 'rib', 'riband', 'rig
gish', 'rigol', 'rivality', 'robustious', 'roe', 'roist', 'romage', 'ronyon', 'roo
d', 'rooky', 'ropery', 'roundure', 'rowel', 'royal', 'roynish', 'rub', 'ruddock', 'r
udesby', 'ruff', 'runagate', 'rushling', 'ruth', 'ryal', 'sackbut', 'sacring bell',
'sain', "Saint Colme 's Inch", 'saith', 'Sala', 'Salique Law', 'sallet', 'sanctuariz
e', 'sans', 'savour', 'saw', 'scab', 'scaffoldage', 'scald', 'scambling', 'scamel',
'scant', 'scantling', 'scape', "'scape", 'scarf up', 'scarfed', 'scathful', 'sconc
e', 'scotch', 'scrimer', 'scrip', 'scrowl', 'scroyle', 'scull', 'scullion', 'scurri
l', 'scutch', 'sea gown', 'sea-gown', 'sea-mark', 'seel', 'self', 'semblable', 'semb
lative', 'sennet', 'senseless-obstinate', 'sequent', 'sequester', 'serpigo', 'sess
a', 'Setebos', 'setter', 'several', 'shale', 'shard', "shark 'd", 'sharked', 'sheale
d', 'shelvy', 'shent', "sheriff 's post", 'shilling', 'shive', 'shoon', 'shotten',
'shough', 'shovel board', 'shovel-board', 'shrieve', 'shrift', 'shrive', 'shunless',
'sicle', 'signiory', 'signory', 'simple', 'simular', 'sirrah', 'sith', 'sithence',
'sixpence', 'skainsmates', 'skains-mates', 'skeins-mates', 'skimble-skamble', 'skink
er', 'skirr', 'sleided', 'sleight', 'sliding', 'slip', 'slop', 'slovenry', 'sluttis
h', 'smatch', 'smilets', 'smit', 'smooth', 'smoothing', 'Smulkin', 'smutch', 'snea
p', 'sneaping', 'sneck up', 'snuff', 'soft', 'soilure', 'solidare', 'sonties', 'soot
h', 'sortance', 'souse', 'sowl', 'sowle', 'spavin', 'speken', 'sperr', 'spilth', 'sp
irt', 'spital', 'splenitive', 'spongy', 'springe', 'springhalt', 'squier', 'squiny',
'stale', 'startingly', 'statist', 'statua', 'stave', 'stell', 'stile', 'stillitory',
'stilly', 'stoccado', 'stomacher', 'stonebow', 'stone-bow', 'stoup', 'strappado', 's
traw', 'strond', 'strucken', 'strumpet', 'subscription', 'succour', 'sumpter', 'supe
rflux', 'supervise', 'suppliance', 'supportance', 'supposal', 'sur-', 'surance', 'su
rplice', 'suspire', 'suum cuique', 'swag', 'swain', 'swashing', 'sweeting', 'Switze
r', 'swound', 'tabor', 'tabour', 'tainture', 'tallow', 'tang', 'tanned', 'targe', 'T
arpeian', 'tarre', 'tarry', 'teem', 'tench', 'tendance', 'tent', 'Termagant', 'testo
n', 'testoon', 'tetter', 'Tewkesbury mustard', 'thane', 'tharborough', 'Thassos', 't
heorick', 'thereunto', 'therewithal', 'thews', 'thine', 'thither', 'thorough', 'tho
u', "thou'dst", 'thraldom', 'thrall', 'thrasonical', 'three-man beetle', 'threepenc
e', 'threne', 'thrum', 'thrummed hat', 'thy', 'thy', 'thyself', 'tilt', 'tilter', 't
iltyard', 'tilt-yard', 'tirrit', 'tortive', 'touse', 'toy', 'traject', 'trencher',
'trollmydames', 'troll-my-dames', 'troth', 'trow', 'Troyan', 'truckle-bed', 'truepen
ny', 'trull', 'truncheon', 'trundle-tail', 'tuck', 'tucket', 'tun', 'tundish', 'tu
p', 'Turlygood', 'twangling', 'twelvemonth', 'twiggen', 'twilled', 'twink', 'twire',
'twit', 'twopence', "unanel 'd", 'unaneled', 'unbarbed', 'unbated', 'unbodied', 'unb
olted', 'unbreathed', 'uncape', 'unclew', 'uncoined', 'uncurrent', 'underfiends', 'u
nderskinker', 'under-wrought', 'uneath', 'unfeigned', 'ungenitured', 'ungravely', 'u
nhatched', "unhousel 'd", 'unhouseled', 'unlineal', 'unmanned', 'unmuzzle', 'unparag

```
oned', 'unpossessing', 'unprizable', 'unprizeable', 'unproperly', 'unseminared', "un
seminar 'd", 'unsifted', 'unsinewed', "unsinew 'd", 'unsisting', 'unstanched', 'unst
aunched', 'untemper', 'untempering', 'unthrift', 'upspring reel', 'usance', 'utis',
'vade', 'vail', 'vantbrace', 'varlet', 'vassalage', 'vastidity', 'vault', 'vault',
'vault', 'vault', 'vaultage', 'vaulty', 'vaunt-courier', 'vaward', 'vegetives', 'vel
ure', 'vendible', 'venew', 'veney', 'ventage', 'verger', 'verily', 'vesper', 'vestur
e', 'via', 'viand', 'victual', 'videlicet', 'viewless', 'villany', 'vinewedst', 'vin
newed', 'vinewed', 'virginalling', 'vizament', 'vizard', 'votary', 'vouchsafe', 'waf
tage', 'wafture', 'waggoner', 'wailful', 'wainropes', 'wain rope', 'wall-eyed', 'wal
l-newt', 'Walloon', 'wanion', 'wanned', 'wappened', 'wannion', "wann 'd", "wappen
'd", 'warrantise', 'water-galls', 'water-rug', 'water-work', 'weal', 'wealsman', 'we
ed', 'weet', 'welkin', 'Welsh hook', 'westward-ho !', 'wezand', 'wesand', 'weasand',
'wheaten', 'Wheeson', 'whelk', 'whence', "wh'er", "whe ' r", 'wherefore', 'wheyfac
e', 'whey-face', 'whiffler', 'while-ere', 'while', 'whimple', 'whipstock', 'whist',
'whiting', 'whitster', 'whittle', 'whoo-bub', 'whoobub', 'wight', 'willow garland',
'window-bars', 'windring', 'winter-ground', 'wi', 'wistly', 'withal', 'wittol', 'wom
by', 'wont', 'wont', 'wonted', 'woodbine', 'woodcock', 'woolward', 'wot', 'wrack',
'wrackful', 'writhled', 'wroth', 'yare', 'y-clad', 'yclept', 'yclad', 'ycleped', 'ye
llow', 'yeoman', 'yeoman', 'yeoman', 'yerk', 'yesty', 'yond', 'younker', 'zounds']
```

We will also convert all strings to lowercase to ensure that the matching is executed properly

In [ ]:
```
cliches_lemmatised= [cliche.lower() for cliche in cliches_lemmatised]
ancient_lemmatised = [phrase.lower() for phrase in ancient_lemmatised]

print(cliches_lemmatised)
print(ancient_lemmatised)
```

['ace in the hole', 'ace up your sleeve', 'acid test', 'airing dirty laundry', "all in a day 's work", 'all talk', 'all booster', 'all hat', 'all foam', 'all hammer', 'all icing', 'all lime and salt', 'all missile', 'all shot', 'all sizzle', 'all wax and no wick', 'all that and a bag of chip', 'all thumb', 'all wet', "all 's fair in love and war", 'almighty dollar', 'always a bridesmaid', 'ambulance chaser', 'another day', 'ant in your pant', 'apple-pie order', 'arm and a leg', 'armchair quarterback', 'army brat', 'art imitates life', 'artsy-craftsy', 'artsy-fartsy', 'a luck would have it', 'a old a time', 'at loggerhead', 'calm before the storm', 'candle at both end', "ca n't cut the mustard", 'case of mistaken identity', 'cat out of the bag', 'cat got your tongue', 'caught red-handed', 'chapter and verse', 'checkered career', 'chicken come home to', 'roost', 'chomping at the bit', 'cleanliness is next to', 'godliness', 'clear a a bell', 'clear a mud', 'cold shoulder', 'e-ticket', 'ear to the ground', 'early bird catch the worm', 'easier said than done', 'easy a 1-2-3', 'easy a pie', 'eat crow', 'eat humble pie', 'enough already', 'even money', 'every dog ha it day', 'every fiber of my being', 'everything but the kitchen sink', 'evil twin', 'existential angst', 'expert agreeï¿½', 'eye for an eye', 'hair of the dog', 'hard to believe', 'have a nice day', 'head honcho', "heart 's content", 'hell-bent for leather', 'hidden agenda', 'high and the mighty (the)', 'high on the hog', 'hold a candle to', 'hold your horse', 'hold your tongue', 'hook or by crook', 'horse of a different color', 'hot knife through butter', 'how go the battle ?', 'keep an eye on you', 'keep it down', 'keep it simple', 'keep up with the joneses', 'keep your card close to vestï¿½', 'keep your chin up', 'keep your finger crossed', 'keep your powder dry', 'kick as', 'kickbutt', 'kick the bucket', 'kick up your heel', 'kick you to the curb', 'kick your foot up', 'kid in a candy store', 'kill two bird with one stone', "life 's a bitch", 'lighten up', 'light out', 'like a sore thumb', 'like butter', 'like the plague', "like there 's no tomorrow", "lion 's share", 'litmus test', 'little black book', 'live and learn', 'long and short of it', 'long lost love', 'look before you leap', 'lounge lizard', 'loved and lost', 'low man on the totem pole', 'luck of the draw', 'luck of the irish', 'off the wagon', 'old college try', 'old met new', 'older and wiser', 'older than dirt', 'older than methuselah', 'on the bandwagon', 'on the nose', 'on the wagon', 'on thin ice', 'one born every minute', 'one foot in the grave', 'one in a million', 'only game in town', 'only to be met', 'out of pocket', 'out of the frying pan', 'out on a limb', 'safe than sorry', 'salt of the earth', 'save face', 'scared stiff', 'scared to death', "school 's out", 'screaming meemies', 'sens reel', 'set the record straight', 'shake a stick should of', 'shoulder to the wheel', 'shouldered his way', 'shut your trap', 'sigh of relief', 'significant other', 'silence is golden', 'take one for the team', 'take the bull by the horn', 'take the plunge', 'take one to know one', 'talk turkey', 'ten foot pole', 'the earth moved', 'the final analysis', 'the real mccoy', 'the same old story', 'these thing happen', 'thick a thief', 'think outside of the box', "third time 's the charm", 'under the gun', 'under the same roof', 'understated elegance', 'unexpected twist', 'until the cow come home', 'up his sleeve', 'up the creek', 'up the wrong tree', 'very real concern', 'view with alarm', 'babe in the wood', 'back against the wall', 'back in the saddle', 'back to square one', 'back to the drawing board', 'bad to the bone', 'badge of honor', 'badonkadonk', 'ballpark figure', 'ball to the wall', 'baptism of fire', 'bare bone', 'bark is worse than the bite', 'bark up the wrong tree', 'bat out of hell', 'bat in the belfry', 'battle royal', 'beat around the bush', 'beat the bush', 'beat me', 'behind the eight ball', 'bent out of shape', 'best foot forward', 'bet your bottom dollar', 'better half', 'better late than never', 'better mousetrap', 'better safe than sorry', 'better than ever', 'better the devil you know', 'between a rock and a hard place', 'beyond the pale', 'bib and tucker', 'big a life', 'big fish in a small pond', 'big man on campus', 'bigger they are', 'communist conspiracy', 'conniption fit', 'could care le', "could n't care le", "could n't get to first base", 'count your blessing', 'countless hour', 'creature comfort', 'crime in the street', 'curiosity killed the cat', 'curry favor', 'cut a fine figure', 'cut and dried', 'cut to the chase', 'cut to the quick', 'cute a a button', 'fact of lif

e', 'fair-haired one', 'fair weather friend', 'fall off of a turnip truck', 'fat slo
b', 'favor u with a song', 'fear and loathing', 'feather your nest', 'fellow travele
r', 'few and far between', 'field this one', 'fifteen minute of fame', 'fish nor fow
l', 'fly by night', 'fly the coop', 'for the bird', 'fox in the henhouse', 'freudian
slip', 'fun and game', 'fun in the sun', 'i beg to differ', 'if the shoe fit', "i 'm
okay", 'in a nutshell', 'in a pinch', 'in a wink', "in harm 's way", 'in the tank',
'in your dream', 'in your face', 'inexorably drawn', 'info dump', 'influence peddlin
g', 'intent and purpose', 'it wa a dark and stormy night', "it wo n't fly", "king 's
english", "king 's ransom", 'kiss and tell', 'kiss as', 'kiss of death', 'kit and ka
boodle', 'knee-high to a grasshopper', 'knock it out of the park', 'knock on wood',
'knock your sock off', 'knocked up', 'know him from adam', 'know the rope', 'know th
e score', 'knuckle down', 'knuckle sandwich', 'knuckle under', 'make my day', 'male
chauvinist', "man 's best friend", 'many moon', 'many-splendored thing', 'mark my wo
rd', 'meaningful relationship', 'mellow out', 'moment of glory', "moment 's respit
e", 'monday morning quarterback', 'monkey suit', 'monkey see', 'motherhood and apple
pie', 'mover and shaker', 'moving experience', 'my two cent', "p 's and q 's", 'pain
and suffering', 'panic button', 'party pooper', 'patter of little foot', 'pas the sn
iff test', 'pay through the nose', 'pea in a pod', 'perfect storm', 'pig in a poke',
'pillar of society', 'pinko', 'plenty of fish in the sea', 'poison pen', 'poor a a c
hurchmouse', 'poor excuse for', 'pot calling the kettle black', 'proud possessor',
'put my foot down', 'put your foot down', 'quick a a bunny', 'quick and the dead',
'slept like a log', 'small world', 'snake in the grass', 'snow job', 'snug a a bug',
'some of my best friend', 'something the cat dragged in', 'spade a spade', 'spare th
e rod', 'spitting image', 'spring to life', 'this day and age', 'this point in tim
e', "three strike and you 're out", 'through the grapevine', 'throw in the towel',
'tiger by the tail', 'till the fat lady sings', 'time and time again', 'time is of t
he essense', 'tip of the iceberg', "'t is the season", 'to err is human', 'to the be
st of my knowledge', 'wakeup call', 'wa my face red', 'watch your tongue', 'web of i
ntrigue', 'week of sunday', 'what a bummer', 'what come around', 'what the cat dragg
ed in', 'what the dickens', 'what the heck', 'what the hell', 'what you see is what
you get', "what 's not to like", 'wheeler-dealer', 'when in doubt', 'when push come
to shove', 'when rubber meet the road', "when the cat 's away", 'when the going get
tough', 'bird in the hand', 'bird and the bee', 'bird of the feather', 'bite the dus
t', 'bite your tongue', 'bitter disappointment', 'black a coal', 'blast from the pas
t', 'bleeding heart', 'blind a a bat', 'blood is thicker than water', 'blood money',
'blood on your hand', 'blood sweat and tear', 'blow this pop stand', 'blow this join
t', 'blushing bride', 'boil it down to', 'bone of contention', 'booze and broad', 'b
ored to tear', 'born and raised', 'born with a silver spoon in your mouth', 'born ye
sterday', 'bottom line', 'brain drain', 'brain dump', 'brass tack', 'bring home the
bacon', 'broken record', "brother 's keeper", 'bull by the horn', 'bull in a china s
hop', 'bump in the night', 'busy a a bee', 'but seriously', 'by and large', 'darkest
before the dawn', 'dead a a doornail', 'death and destruction', 'death and tax', "de
ath 's doorstep", 'devil is in the detail', 'dim view', 'dog day', 'dog in the mange
r', "do n't count your chicken before they 're hatched", "do n't do the crime if you
ca n't do the time", 'doubting thomas', 'down and dirty', 'down in the dump', 'down
pat', 'down the toilet', 'down the drain', 'down the hatch', 'down to earth', 'drive
you up a wall', 'dutch uncle', 'dyed in the wool', 'garbage in', 'get the sack', 'ge
t your groove back', 'get my goat', 'gift horse in the mouth', 'gilding the lily',
'give a damn', 'give me a break', 'give me the creep', 'go him one better', 'go with
out saying', 'good deed for the day', 'good time wa had by all', 'greek to me', 'gre
en thumb', 'green-eyed monster', 'grist for the mill', 'guiding light', 'jack of all
trade', 'jockey for position', 'johnny-come-lately', 'joined at the hip', 'jump down
your throat', 'jump in with both foot', 'jump on the bandwagon', 'jump the gun', 'ju
mp his bone', 'jump her bone', 'junk in the trunk', 'jury is still out', 'justice is
blind', 'labor of love', 'lap of luxury', 'last but not least', 'last-ditch effort',
'last hurrah', 'law of the jungle', 'law of the land', 'lay down the law', 'leap and

bound', 'let sleeping dog lie', 'let the cat out of the bag', "let 's split", 'liber
al medium', 'lie like a rug', 'life and limb', 'life imitates art', 'neat a a pin',
'needle to say', 'nip it in the bud', 'no gut', 'no love lost', 'no pain', 'no stone
unturned', 'no time like the present', 'nose to the grindstone', 'not in my back yar
d', 'not on your tintype', 'number one fan', 'numerous to mention', 'radical chic',
'rag to rich', 'raining bucket', 'raining cat and dog', 'rank and file', 'read my li
p', 'red herring', 'redheaded stepchild', 'reign supreme', 'remember the alamo', 'ro
ad to hell is paved with good intention', 'rob peter to pay paul', 'rock and a hard
place', 'rocket scientist', 'rocket science', 'rope a dope', 'run it up the flagpol
e', 'running dog lackey', 'squeaky wheel get the oil', 'squeaky wheel get the greas
e', 'start from scratch', 'stick in the mud', 'stick in your craw', 'still water run
deep', 'stop and smell the rose', 'store bought', 'stranger than fiction', "straw th
at broke the camel 's back", 'stubborn a a mule', 'stuff that dream are made of', 's
tuffed shirt', 'tongue-in-cheek', 'too hot to handle', 'touch of blarney', 'tough a
nail', 'tough luck', 'tough row to hoe', 'traditional family value', 'trial and trib
ulation', 'tried and true', 'trip down memory lane', 'true blue', 'turn your smile u
pside-down', 'turn your frown upside-down', 'twist of fate', 'twist and turn', 'two
to tango', 'who ha everything', 'whole ball of wax', 'whole hog', 'whole nine yard',
'whole other story', 'wild goose chase', 'wild oat', 'will wonder never cease', 'wim
p', 'win friend and influence', 'people', 'win one for the gipper', 'wnning is every
thing', 'wisdom of the age', 'without benefit of clergy', 'wolf at the door', 'word
fail', 'work like a dog', 'worst nightmare', 'wrong side of the bed']
['abodement', 'abram', 'abroach', 'abrook', 'absey-book', 'aby', 'accdant', 'accit
e', 'accomplement', 'accompt', 'accouter', 'accoutre', 'acheron', 'acknow', 'aconitu
m', 'acquittance', 'acture', 'actus primus', 'actus quartus', 'actus quintus', 'actu
s secundus', 'actus tertius', 'addle', 'adieu', 'adoptious', 'advocation', 'aery',
'afeared', 'affiance', 'affined', 'affy', 'agate', 'agazed', 'aglet', 'agood', 'agu
e', 'aiery', 'alack', 'alarum', 'albion', 'alderliefest', 'alder-liefest', 'alder-li
evest', 'alewife', 'ale-wife', 'allayment', 'allegiant', 'allottery', 'almain', 'alm
s drink', 'almsdrink', 'almshouse', 'amain', 'amerce', 'ames-ace', 'amort', 'anatomi
ze', 'ancient', 'a-night', 'annexment', 'anon', 'antagonist', 'antechamber', 'anthro
pophagi', 'anthropophaginian', 'antic', 'antick', 'anticked', 'antre', 'appeach', 'a
ppellant', 'apperil', 'apple-john', 'approof', 'apricock', 'aqua vitae', 'aqua-vita
e', 'argal', 'argier', 'argosy', 'aright', 'armigero', 'aroint', 'a-row', 'arrant',
'arras', 'aside', 'asinego', 'assay', 'assemblance', 'assinego', 'assubjugate', 'ato
mies', 'attaint', 'attainture', 'atwain', 'a-twain', 'audit', 'auld', 'avaunt', 'avo
uch', 'aye', 'bacare', 'baccare', 'bacchanal', 'backare', 'baffle', 'baggage', 'baja
zet', 'balcony in a shakespearean theater', 'ballow', 'balsamum', 'ban', 'bandog',
'ban-dog', 'banns', 'barbary', 'barbason', 'barbermonger', 'barber-monger', 'barm',
'barnes', 'barricado', 'basilisk', 'basta', 'bastard', 'bastardy', 'bastinado', 'bat
e', 'bat-fowling', 'batler', 'batlet', 'battaille', 'battalia', 'bavin', 'bawbling',
'bawcock', 'bawdry', 'beadsman', 'beam', 'bearing cloth', 'bearing-cloth', 'beaver',
'bed of ware', 'bed-hangings', 'bedlam', 'beetle', 'befall', 'begirt', 'behoveful',
'beldam', 'belike', 'bemadding', 'bemete', 'be-mete', 'bemoil', 'be-moil', 'be-nette
d', 'benison', 'bergomask', 'bescreened', 'be-screened', 'beseech', 'beshrew', 'besl
ubber', 'besort', 'bespake', 'bespeak', 'bespoke', 'bestraught', 'betake', 'beteem',
'betid', 'betide', 'betimes', 'betossed', 'betwixt', 'bewray', 'bezonian', 'biding',
'biggen', 'bilberry', 'bilbo', 'bill', 'birdbolt', 'bird-bolt', 'bisson', 'blench',
'bob', 'bodement', 'bodge', 'bodkin', 'boótier vert', 'bolster', 'bolted', 'bolter',
'bolting hutch', 'bolting-hutch', 'bombard', 'bona roba', 'bona-roba', 'bondman', 'b
onny', 'boot', 'bootless', 'bosky', 'botcher', 'bot', 'bottled spider', 'bounden',
'bourn', 'brabble', 'brach', 'brainish', 'brainpan', 'brain-pan', 'brake', 'braver
y', 'breese', 'bretagne', 'brinded', 'brinish', 'britaine', 'brittany', 'brock', 'br
ogue', 'bruit', 'bubukle', 'buckler', 'buckram', 'bum', 'bung', 'burden a a music te
rm', 'burgonet', 'burthen', 'buskin', 'busky', 'bus', 'buzzer', 'by-room', 'cacodaem
on', 'cacodemon', 'caddis', 'caitiff', 'calipolis', 'caliver', 'callat', 'callet',

'cam', 'cambric', 'camlet', 'canakin', 'canary', 'canis', 'cannikin', 'canzonet', 'c
ap-a-pie', 'capon', 'carbuncle', 'carcanet', 'cardmaker', 'carlot', 'carract', 'car
t', 'carviare', 'casing', 'casque', 'cataplasm', 'catch', 'catchword', 'cater', 'cat
er-cousin', 'caterwauling', 'cautel', 'cautelous', 'caviary', 'cavil', 'cellarage',
'cellar-age', 'century', 'cerecloth', 'cerement', 'certes', 'chace', 'champain', 'ch
anticleer', 'chantry', 'chape', 'chapfallen', "chap-fall'n", 'chapless', 'chapman',
'chapt', 'charactery', 'charneco', 'charnel house', 'charnel-house', 'cheveril', 'ch
id', 'chine', 'chirurgeon', 'chopine', 'chorus', 'chough', 'chuck', 'cicatrice', 'ci
nque-pace', 'cinque-ports', 'circummure', 'civet', 'clackdish', 'clack-dish', 'clep
e', 'clept', 'clew', 'climature', 'clinquant', 'clog', 'cloistress', 'clotpoll', 'cl
out', 'clouted brogue', 'clovest', 'cloyless', 'cockatrice', "cockerel 's stone", 'c
ockle', 'cockle hat', 'cockleshell', 'cockney', 'cock-shut time', 'codpiece', 'coffe
r', 'cog', 'cogging', 'coign', 'coil', 'collop', 'combinate', 'complice', 'complot',
'compulsative', 'con', 'concernancy', 'concupy', 'condign', 'coney', 'confixed', "co
nge 'd", 'congeed', 'conger', 'congied', 'conjunctive', 'conjuration', 'conserve',
'consort', 'conspectuity', "constring 'd", 'constringed', 'contemn', 'continuate',
'contumeliously', 'conventicle', 'convive', 'cony', "copp 'd", 'copped', 'coppice',
'coram', 'coranto', 'coronal', 'corse', 'corselet', 'costard', 'costermonger', 'cotq
uean', 'coulter', 'couplement', 'covert', 'coy', 'coystril', 'coystrill', 'coz', 'co
zen', 'cozenage', 'cozier', 'crare', 'crescive', 'crotchet', 'crown', 'crownet', 'cr
upper', 'crusado', 'cruzado', 'cubiculo', 'cuckold', 'cuckoo flower', 'cullion', 'da
ce', 'daff', 'dam', 'damosella', 'damson', 'danskers', 'darnel', 'darraign', 'dauber
y', 'dauphin', 'debile', 'debosh', 'defeature', 'deflower', 'defunct', 'delated', 'd
elation', 'demi-wolf', 'denay', 'denier', 'denotement', 'depute', 'descant', 'design
ments', 'despatch', 'despite', 'determinate', 'diadem', 'dialogue', 'diaper', 'dic
h', 'dirge', 'disannul', 'disme', 'dispark', 'dispatch', 'disprized', 'disproperty',
'disquantity', 'dissentious', 'distaff', 'distained', 'distemperature', 'distract',
'distrain', 'ditch-dog', 'dive-dapper', 'diver', 'doit', 'dole', 'dormouse', 'dotan
t', 'doublet', 'dout', 'dower', 'dowlas', 'dowle', 'down-gyved', 'downs', 'drab', 'd
rachma', 'draff', 'dram', 'dramatis persona', 'drave', 'drawer', 'drossy', 'drumbl
e', 'dry-beat', 'dryfoot', 'dudgeon', 'duello', 'dug', 'dump', 'dup', 'durance', 'du
rst', 'eale', 'eaning', 'eftest', 'eftsoons', 'eisel', 'eld', 'embarquement', 'embas
sage', 'emblaze', 'embossed', 'embowel', 'embracement', 'empiricutic', 'emulous', 'e
ncave', 'enchase', 'encloud', 'encompassment', 'enfeoff', 'engirt', 'englut', 'enme
w', 'enow', 'enrank', 'enroot', 'enround', 'entail', 'enter', 'enwombed', 'epilogu
e', 'epitheton', 'equipage', 'ere', 'erewhile', 'errant', 'erst', 'escot', 'esperanc
e', 'espial', 'essay', 'estridge', 'eterne', 'ethiop', 'ethiope', 'ever yet', 'every
et', 'exchequer', 'excursion', 'exeunt', 'exion', 'exit', 'exposture', 'exsufflicat
e', 'extirp', 'extraught', 'eyas', 'eyne', 'eyrie', 'facinorous', 'factionary', 'fad
ge', 'fain', 'fait', 'falchion', 'fallow', 'falsing', 'fap', 'farced', 'fardel', 'fa
rthing', 'fatigate', 'fay', 'fealty', 'feat', 'fecks', 'federary', "fee 'd", 'feede
r', 'fee-grief', 'fee-simple', 'felly', 'feodary', 'fere', 'festinately', 'fico', 'f
ie', 'fillip', 'filly foal', 'fine sovereign', 'firago', 'firk', 'fitchew', 'flapdra
gon', 'fleer', 'fleshment', 'flews', 'flexure', 'flirt-gill', 'flock', 'fob', 'fob o
ff', 'foh', 'foil', 'foin', 'foison', 'fool', 'foot', 'footboy', 'foot-cloth mule',
'footing', 'fordo', 'forsooth', 'fortnight', 'fosset', 'foughten', 'foul paper', 'fo
xship', 'fracted', 'frank', 'fraught', 'fraughtage', 'frippery', 'frontlet', 'frowar
d', 'frush', 'fullam', 'fumiter', 'fumitory', 'fust', 'fustilarian', 'gaberdine', 'g
ad', 'gage', 'gainsay', 'gallery', 'galliard', 'gallia', 'galloglasses', 'gallow',
'gallowglasses', 'gamesome', 'gammon', 'gaoler', 'garboil', 'garner', 'gaskin', 'gas
ted', 'gawd', 'geck', 'gentle', 'gest', 'gib', 'giglot', 'gimmal', 'gimmer', 'gimmo
r', 'ging', 'gird', 'gis', 'glaive', 'glanders', 'glave', 'gleek', 'glister', 'glos
e', 'glow-worm', 'gloze', 'go to the world', 'gobbet', 'goblin', 'godwit', 'good de
n', 'good-year', 'gorbellied', 'gord', 'gourd', 'gramercy', 'gratis', 'graymalkin',
'greave', 'greenly', 'gripe', 'grise', 'grize', 'groat', 'guardant', 'gudgeon', 'gue
rdon', 'gules', 'gull', 'gyve', 'habiliment', 'habitude', 'haggard', 'hale', 'half-c

rown', 'halfpenny', 'halidom', 'halter', 'handsaw', 'hap', "ha'penny", 'hapless', 'h
aply', 'hard-handed', 'hardiment', 'harebell', 'hare-bell', 'harlock', 'harrow', 'ha
sle', 'haste-post-haste', 'hatchment', 'haught', 'hautbois', 'hautboy', 'haviour',
'head', 'head-lugged', 'hebona', 'hectic', 'hedge', 'hedge-pig', 'hedgepriest', 'hef
t', 'hefted', 'heigh', 'helm', 'hempen', 'henchman', 'hent', 'hest', 'hewgh', 'hiem
s', 'high cross', 'high-cross', 'hight', 'hilding', 'hindmost', 'hither', 'hizz', 'h
odge pudding', 'hodge-pudding', 'hoise', 'holidame', 'holla', 'hollaing', 'holp', 'h
oly-ales', 'honey-stalks', 'honorificabilitudinitatibus', 'hornbook', 'horn-mad', 'h
orologe', 'hothouse', 'hot-house', 'house of resort', 'howbeit', 'hox', 'hoy', 'hugg
ermugger', 'hull', 'hurly', 'hurricano', 'hurtle', 'husband', 'huswife', 'hyen', 'hy
rcanian beast', 'i wi', 'ignomy', "'ild", 'illume', 'imbar', 'imbare', 'imbrue', 'im
manity', 'immoment', 'imp', 'impartment', 'impasted', 'impawn', 'imperceiverant', 'i
mpeticos', 'impleached', 'implorator', 'impone', 'importunacy', 'imposthume', 'impre
se', 'impressure', 'imputation', 'incarnadine', 'incensement', 'incertain', 'inchmea
l', 'inch-meal', 'inclip', 'incony', 'incorpsed', 'increase', 'indent', 'indifferenc
y', 'indigest', 'indign', 'indue', 'indurance', 'infamonize', 'inferreth', 'infortun
ate', 'inhearse', 'inhoop', 'injointed', 'inkhorn', 'inkle', 'inscroll', "insculp
'd", 'insculped', 'insculpted', 'insculpture', "insinew 'd", 'insinewed', 'insistur
e', "insteep 'd", 'insteeped', 'intelligencer', 'intendment', 'intenible', 'intrins
e', 'inventorially', 'iron crow', 'irregulous', 'issue', 'iwis', 'jack-a-lent', 'jac
k', 'jade', 'jakes', 'jar', 'jaunce', 'jerkin', 'jester', 'jet', 'jocund', 'john-a-d
reams', 'joint-ring', 'jordan', 'jot', 'jutty', 'juvenal', 'kam', 'kecksy', 'keech',
'keel', 'keisar', 'ken', 'keptst', 'kern', 'kersey', 'kibe', 'kicky-wicky', 'kine',
'kirtle', 'knack', 'knap', 'knight-errant', 'la', 'labras', 'lakin', 'lammas', 'lamm
as-tide', 'lampass', 'lank', 'lanthorn', 'lapwing', 'lass-lorn', 'lated', 'latten',
'laund', 'lazar', 'leagued', 'leather coat', 'leather-coat', 'leathern', 'leaf', 'le
e', 'leet', 'legerity', 'leiger', 'leman', "l'envoy", 'levy', 'libbard', 'lict', 'li
ef', 'liege', 'liegeman', 'lifeling', 'limbeck', 'limbo', 'lime', 'lineal', 'lineame
nt', 'linstock', 'list', 'list', 'lither', 'loach', 'loaden', 'lob', 'lockram', 'lod
estar', 'lode-star', 'loggat', 'loggats', 'loof', 'lour', 'lout', 'lown', 'lowt', 'l
ozel', 'lubber', 'luce', 'lune', 'lustihood', 'lym', 'maggot-pie', 'magnifico', 'mah
u', 'main', 'mained', 'makeless', 'malapert', 'malkin', 'malmsey', 'malt worm', 'mam
mer', 'mammet', 'mammock', 'manakin', 'mandragora', 'manikin', 'manningtree', 'march
pane', 'marish', 'marl', 'marry', 'mart', 'martinmas', 'martlemas', 'martlet', 'mary
-buds', 'masque', 'matin', 'maugre', 'maw', 'mazzard', 'meacock', 'measles', 'medla
r', 'meed', 'meetest', 'meetness', 'mehercle', 'meiny', 'mell', 'mercatante', 'mere
d', 'mete-yard', 'metheglin', 'methinks', 'mew', 'mickle', 'milch', 'milkpaps', 'min
ikin', 'minim', 'minimus', 'mirable', 'miscarry', 'misconster', 'misprise', 'misprou
d', 'missingly', 'mis-tempered', 'mobled', 'modo', 'moe', 'moiety', 'moldwarp', 'mom
e', 'momentany', 'monstruosity', 'montant', 'mooncalf', 'morisco', 'morris', 'morris
-pike', 'morrow', 'mote', 'mow', 'mummy', 'muniment', 'mure', 'murrain', 'murther',
'musk rose', 'musk-rose', 'mutine', 'nayward', 'nayword', 'neaf', 'near-legged', 'ne
at', 'needly', 'neeld', 'neeze', 'neif', 'netherstocks', 'nether-stocks', 'noddle',
'nonce', 'nose', "not'st", 'nousle', 'novum', 'nowl', 'noyance', 'nuncle', 'nuthoo
k', 'obsequies', 'oeilliade', 'oeilliades', "o'ercrow", "o'erleaven", "o'er-leaven",
"o'er-raught", "o'er-read", "o'erskip", "o'ertrip", 'oneyer', 'ope', 'oppugnancy',
'orange-wife', 'ordinant', 'orgulous', 'orison', 'orthographer', 'ostler', 'ouph',
'ouphe', 'ousel-cock', 'outface', 'outsport', 'outwall', 'out-wall', 'overborne', 'o
vercrow', 'overglance', 'overleather', 'overleaven', 'over-raught', 'overread', 'ove
rskip', 'overtrip', 'overweening', 'over-weening', 'paction', 'paddock', 'pajock',
'pale', 'pall', 'palliament', 'palmy', 'palter', 'paly', 'pantler', 'panyn', 'panyn
æ', 'pap', 'paraquito', 'parcel', 'pardonnez-moi', "'parit", 'parle', 'parling', 'pa
rmaceti', 'parthia', 'partisan', 'partizan', 'pash', 'passado', 'passymeasure', 'pas
sy-measure', 'pastern', 'patch', 'patine', 'pauca', 'pavan', 'pavane', 'pavin', 'pea
t', 'pedascule', 'peise', 'pell-mell', 'penny', 'pennyworth', 'penny-worth', 'peradv
enture', 'perdie', 'perdona-mi', 'perdu', 'perdurable', 'perdy', 'periapt', 'perpen

d', 'persever', 'perspective', 'pestiferous', 'petar', 'petard', 'pettish', 'pew-fel
low', 'pheeze', 'phoebe', 'physic', 'picht-hatch', 'pickthank', 'pick-thank', 'pickt
-hatch', 'pight', 'pilchard', 'pill', 'pinfold', 'pinnace', "pin 's fee", 'pioned',
'pioner', 'pippin', 'pish', 'pismire', 'pittikin', 'placket', 'plain', 'plaining',
'planched', 'plat', 'plausive', 'pleached', 'pleb', 'plebeii', 'plurisy', 'poesy',
'point-device', 'poise', 'poke', 'politic', 'politicly', 'poll', 'polled', 'pomewate
r', 'poniard', 'pontifical', 'poor john', 'poperin', 'popinjay', 'porpentine', 'porr
inger', 'portance', 'posset', 'post horse', 'postern', 'post-horse', 'potation', 'po
tch', 'potting', 'pottle', 'poulter', 'pouncet box', 'pouncet-box', 'pound', 'powe
r', 'practisant', 'pratest', "prat'st", 'preceptial', 'prejudicate', 'prenominate',
'prentice', "'prentice", 'cardecu', 'coif', 'quait', "quart d ' ? cu", 'quat', 'quat
ch', 'quean', 'quern', 'questant', 'questrist', 'quiddit', 'quillet', 'quintain', 'q
uire', 'quittance', 'quiver', 'quoif', 'quoit', 'quondam', 'quotha', 'quoth-a', 'quo
tidian', 'rabato', 'rabbit-sucker', 'rack', 'rampallian', 'raught', 'ravin', 'rearmo
use', 'rear-mouse', 'reave', 'rebato', 'recheat', 'reck', 'recordation', 'recreant',
'red plague', 'rede', 'reechy', 'reel', 'refel', 'reft', 'reguerdon', 'rejourn', 're
lique', 'remember', 'remembrancer', 'remotion', 'repair', 'repugn', 'reputeless', 'r
eremouse', 'rere-mouse', 'reverb', 'rhenish', 'rheum', 'rial', 'rib', 'riband', 'rig
gish', 'rigol', 'rivality', 'robustious', 'roe', 'roist', 'romage', 'ronyon', 'roo
d', 'rooky', 'ropery', 'roundure', 'rowel', 'royal', 'roynish', 'rub', 'ruddock', 'r
udesby', 'ruff', 'runagate', 'rushling', 'ruth', 'ryal', 'sackbut', 'sacring bell',
'sain', "saint colme 's inch", 'saith', 'sala', 'salique law', 'sallet', 'sanctuariz
e', 'sans', 'savour', 'saw', 'scab', 'scaffoldage', 'scald', 'scambling', 'scamel',
'scant', 'scantling', 'scape', "'scape", 'scarf up', 'scarfed', 'scathful', 'sconc
e', 'scotch', 'scrimer', 'scrip', 'scrowl', 'scroyle', 'scull', 'scullion', 'scurri
l', 'scutch', 'sea gown', 'sea-gown', 'sea-mark', 'seel', 'self', 'semblable', 'semb
lative', 'sennet', 'senseless-obstinate', 'sequent', 'sequester', 'serpigo', 'sess
a', 'setebos', 'setter', 'several', 'shale', 'shard', "shark 'd", 'sharked', 'sheale
d', 'shelvy', 'shent', "sheriff 's post", 'shilling', 'shive', 'shoon', 'shotten',
'shough', 'shovel board', 'shovel-board', 'shrieve', 'shrift', 'shrive', 'shunless',
'sicle', 'signiory', 'signory', 'simple', 'simular', 'sirrah', 'sith', 'sithence',
'sixpence', 'skainsmates', 'skains-mates', 'skeins-mates', 'skimble-skamble', 'skink
er', 'skirr', 'sleided', 'sleight', 'sliding', 'slip', 'slop', 'slovenry', 'sluttis
h', 'smatch', 'smilets', 'smit', 'smooth', 'smoothing', 'smulkin', 'smutch', 'snea
p', 'sneaping', 'sneck up', 'snuff', 'soft', 'soilure', 'solidare', 'sonties', 'soot
h', 'sortance', 'souse', 'sowl', 'sowle', 'spavin', 'speken', 'sperr', 'spilth', 'sp
irt', 'spital', 'splenitive', 'spongy', 'springe', 'springhalt', 'squier', 'squiny',
'stale', 'startingly', 'statist', 'statua', 'stave', 'stell', 'stile', 'stillitory',
'stilly', 'stoccado', 'stomacher', 'stonebow', 'stone-bow', 'stoup', 'strappado', 's
traw', 'strond', 'strucken', 'strumpet', 'subscription', 'succour', 'sumpter', 'supe
rflux', 'supervise', 'suppliance', 'supportance', 'supposal', 'sur-', 'surance', 'su
rplice', 'suspire', 'suum cuique', 'swag', 'swain', 'swashing', 'sweeting', 'switze
r', 'swound', 'tabor', 'tabour', 'tainture', 'tallow', 'tang', 'tanned', 'targe', 't
arpeian', 'tarre', 'tarry', 'teem', 'tench', 'tendance', 'tent', 'termagant', 'testo
n', 'testoon', 'tetter', 'tewkesbury mustard', 'thane', 'tharborough', 'thassos', 't
heorick', 'thereunto', 'therewithal', 'thews', 'thine', 'thither', 'thorough', 'tho
u', "thou'dst", 'thraldom', 'thrall', 'thrasonical', 'three-man beetle', 'threepenc
e', 'threne', 'thrum', 'thrummed hat', 'thy', 'thy', 'thyself', 'tilt', 'tilter', 't
iltyard', 'tilt-yard', 'tirrit', 'tortive', 'touse', 'toy', 'traject', 'trencher',
'trollmydames', 'troll-my-dames', 'troth', 'trow', 'troyan', 'truckle-bed', 'truepen
ny', 'trull', 'truncheon', 'trundle-tail', 'tuck', 'tucket', 'tun', 'tundish', 'tu
p', 'turlygood', 'twangling', 'twelvemonth', 'twiggen', 'twilled', 'twink', 'twire',
'twit', 'twopence', "unanel 'd", 'unaneled', 'unbarbed', 'unbated', 'unbodied', 'unb
olted', 'unbreathed', 'uncape', 'unclew', 'uncoined', 'uncurrent', 'underfiends', 'u
nderskinker', 'under-wrought', 'uneath', 'unfeigned', 'ungenitured', 'ungravely', 'u
nhatched', "unhousel 'd", 'unhouseled', 'unlineal', 'unmanned', 'unmuzzle', 'unparag

```
oned', 'unpossessing', 'unprizable', 'unprizeable', 'unproperly', 'unseminared', "un
seminar 'd", 'unsifted', 'unsinewed', "unsinew 'd", 'unsisting', 'unstanched', 'unst
aunched', 'untemper', 'untempering', 'unthrift', 'upspring reel', 'usance', 'utis',
'vade', 'vail', 'vantbrace', 'varlet', 'vassalage', 'vastidity', 'vault', 'vault',
'vault', 'vault', 'vaultage', 'vaulty', 'vaunt-courier', 'vaward', 'vegetives', 'vel
ure', 'vendible', 'venew', 'veney', 'ventage', 'verger', 'verily', 'vesper', 'vestur
e', 'via', 'viand', 'victual', 'videlicet', 'viewless', 'villany', 'vinewedst', 'vin
newed', 'vinewed', 'virginalling', 'vizament', 'vizard', 'votary', 'vouchsafe', 'waf
tage', 'wafture', 'waggoner', 'wailful', 'wainropes', 'wain rope', 'wall-eyed', 'wal
l-newt', 'walloon', 'wanion', 'wanned', 'wappened', 'wannion', "wann 'd", "wappen
'd", 'warrantise', 'water-galls', 'water-rug', 'water-work', 'weal', 'wealsman', 'we
ed', 'weet', 'welkin', 'welsh hook', 'westward-ho !', 'wezand', 'wesand', 'weasand',
'wheaten', 'wheeson', 'whelk', 'whence', "wh'er", "whe ' r", 'wherefore', 'wheyfac
e', 'whey-face', 'whiffler', 'while-ere', 'while', 'whimple', 'whipstock', 'whist',
'whiting', 'whitster', 'whittle', 'whoo-bub', 'whoobub', 'wight', 'willow garland',
'window-bars', 'windring', 'winter-ground', 'wi', 'wistly', 'withal', 'wittol', 'wom
by', 'wont', 'wont', 'wonted', 'woodbine', 'woodcock', 'woolward', 'wot', 'wrack',
'wrackful', 'writhled', 'wroth', 'yare', 'y-clad', 'yclept', 'yclad', 'ycleped', 'ye
llow', 'yeoman', 'yeoman', 'yeoman', 'yerk', 'yesty', 'yond', 'younker', 'zounds']
```

Finally, we will carry out the matching on each answer

In [ ]:
```
#initialise empty columns for the features
df_cleaned['cliche_count'] = 0
df_cleaned['ancient_phrase_count'] = 0
```

In [ ]:
```
for i in range(0, len(df_cleaned)):
    answer = df_cleaned.at[i, 'answer'] #retrieve answer from df
    answer = answer.lower() # convert the answer to all lower case

    cliche_cntr = 0
    for cliche in cliches:
        if cliche in answer:
            cliche_cntr += 1

    ancient_phrase_cntr = 0
    for phrase in ancient_phrases:
        if phrase in answer:
            ancient_phrase_cntr += 1

    df_cleaned.at[i, 'cliche_count'] = cliche_cntr
    df_cleaned.at[i, 'ancient_phrase_count'] = ancient_phrase_cntr
```

In [ ]:
```
df_cleaned.sample(3)
```

Out[ ]:

	Unnamed: 0.2	Unnamed: 0.1	Unnamed: 0	question	answer	is_human	filter
9377	9377	9377	9377	Is my son having asbugers?what are the symptom...	It is important to note that it is not appropr...	0	I a
31003	31003	31003	31003	What causes tiredness and breathlessness after...	Hi thanks for contacting HCM.....Pleuricy mean...	1	conta
14654	14654	14654	14654	How much does taking a Microeconomics course h...	A microeconomics course can help you understan...	0	micro

3 rows × 26 columns

We will also save the final dataframe with all extracted features to a csv file

In [ ]:
```
df_cleaned.to_csv('final_feature_data.csv')
```

3. Classification

As we discovered in Prototype 2, the Random Forest Classifier (RFC) trained using a combination of all 5 featuresets was the highest-performing algorithm.

Thus, we will be training and using an RFC using all 5 featuresets in this final prototype.

First, install scikit-learn and import the neccessary packages and load the csv file of extracted features

In [ ]:
```
#!pip install scikit-learn
```

In [ ]:
```
from sklearn.linear_model import SGDClassifier, LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.calibration import CalibratedClassifierCV
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
```

In [ ]:
```
df = pd.read_csv('final_feature_data.csv')
df.sample(3)
```

Out[ ]:

	Unnamed: 0.3	Unnamed: 0.2	Unnamed: 0.1	Unnamed: 0	question	answer	is_human	f
14750	14750	14750	14750	14750	Where do old blood cells go in our body when t...	Old or damaged red blood cells are removed fro...	0	
30416	30416	30416	30416	30416	Why do I automatically assume everyone I am ta...	This is a result of a " male as a normative " ...	1	
1659	1659	1659	1659	1659	what did isaac newton do	Isaac Newton was an English physicist, mathema...	0	r

3 rows × 27 columns

We will also drop some of the unused index columns that were created previously when saving the intermediate feature data to csv file

In [ ]:
```
df.drop(df.columns[0:4], axis=1, inplace=True)
```

In [ ]:
```
df.sample(3)
```

Out[ ]:

	question	answer	is_human	filtered_answer	lemmatised_answer	word_count	sen
20847	Who is the husband of Betty Ford	Betty Ford was the wife of Gerald Ford, who wa...	0	Betty Ford wife Gerald Ford , President United...	Betty Ford wa the wife of Gerald Ford , who wa...	260	
40269	how does a candle burn so long ? Like it 's ju...	\A candle burns because the heat of the flame ...	0	\A candle burn heat flame melt wax near wick	\A candle burn because the heat of the flame m...	197	
28109	Why diesel engine can not run on petrol ? And ...	Diesel engines and gasoline engines are design...	0	Diesel engine gasoline engine designed run dif...	Diesel engine and gasoline engine are designed...	289	

3 rows × 23 columns

Next, we will Split the dataset into independent variables (x) and the dependent (target) variable (y)

In [ ]:
```
# split into target and independent variables

# select all columns except for the specified ones
x = df.loc[:, ~df.columns.isin(['question', 'answer', 'is_human', 'filtered_answer'

#select is_human as target
y = df['is_human'] # target
```

In [ ]:
```
x.sample(3)
```

Out[ ]:

	word_count	sentence_count	char_count	avg_sentence_len	avg_word_len	polarity
7290	179	8	893	22.375000	4.988827	-0.335965
40449	140	9	705	15.555556	5.035714	0.254215
31166	337	10	1754	33.700000	5.204748	0.593119

Additionally, we will also scale the x values using sci-kit's StandardScaler. This scales each x value so that they have the same scale and dimension, which prevents differently-scaled variables from introducting biases into the modelling.

In [ ]:
```
# initialise scaler
scaler = StandardScaler()
x_scaled = scaler.fit_transform(x)
```

Next, split the data into testing (20%) and training (80%) sets using sklearn's train_test_split

In [ ]:
```
x_train, x_test, y_train, y_test = train_test_split(x_scaled, y, test_size = 0.2, r
```

Initialise the RFC, and fit it on the test data.

In [ ]:
```
rfc = RandomForestClassifier()
rfc.fit(x_train, y_train) # train RFC
```

Out[ ]:
```
▾ RandomForestClassifier

RandomForestClassifier()
```

Next, generate a prediction using the RFC

In [ ]:
```
# generate prediction
rfc_predict = rfc.predict(x_test)
```

Also, we will generate the RFC's predicted probability of each class prediction for evaluation using sklearn's predict_proba method.

In [ ]:
```
rfc_proba = rfc.predict_proba(x_test)
rfc_proba = rfc_proba[:, 1] # keep only the probability of the positive class
```

4. Evaluation

The next step will be to evaluate the performance of the RFC

First, we will import the required modules

In [ ]:
```
from sklearn.metrics import accuracy_score, confusion_matrix, ConfusionMatrixDispla
import matplotlib.pyplot as plt
import numpy as np
```

Next, we will define some helper functions that will be used to plot confusion matrices, calculate EER, calculate performance stats, and add numerical labels to bar charts.

In [ ]:
```
# plots a confusion matrix display
def plot_con_matrix(y_test, y_predict):
    # create confusion matrix
```

```
    con_matrix = confusion_matrix(y_test, y_predict)
    # create confusion matrix display and set labels
    cm_display = ConfusionMatrixDisplay(con_matrix, display_labels=['gpt', 'human']
    # plot and show confusion matrix
    return cm_display

# calculates FPR and TPR
def calc_fpr_tpr(y_test, y_predict):

    # create confusion matrix
    con_matrix = confusion_matrix(y_test, y_predict)
    false_pos_rate = con_matrix[0][1] / (con_matrix[0][0] + con_matrix[0][1]) #calc
    true_pos_rate = con_matrix[1][1] / (con_matrix[1][0] + con_matrix[1][1]) #calcu

    return false_pos_rate, true_pos_rate

# calculates EER
# method referenced from James S. [2017].
def calc_eer(y_test, y_prob):
    # generate roc curve, and fpr, tpr lists and the threshold
    fpr, tpr, threshold = roc_curve(y_test, y_prob, pos_label = 1)
    # compute false negative rate
    fnr = 1 - tpr
    # find EER
    eer = fpr[np.nanargmin(np.absolute((fnr - fpr)))]

    return eer

# calculates accuracy, roc, eer, precision, recall
def calc_performance_stats(y_test, y_predict, y_prob):
    accuracy = accuracy_score(y_test, y_predict) # calculate accuracy
    roc_score = roc_auc_score(y_test, y_predict) # calculate ROC
    eer = calc_eer(y_test, y_prob) # calculate EER
    precision = precision_score(y_test, y_predict) # calculate precision
    recall = recall_score(y_test, y_predict) # calculate recall

    return accuracy, roc_score, eer, precision, recall

# add value labels to bar charts
def add_val_labels(x, y, ax):
    for i in range(len(x)):
        ax.text(x[i], y[i]/3, f'{y[i]:.3f}', ha = 'center') # draw value at 30% hei
```

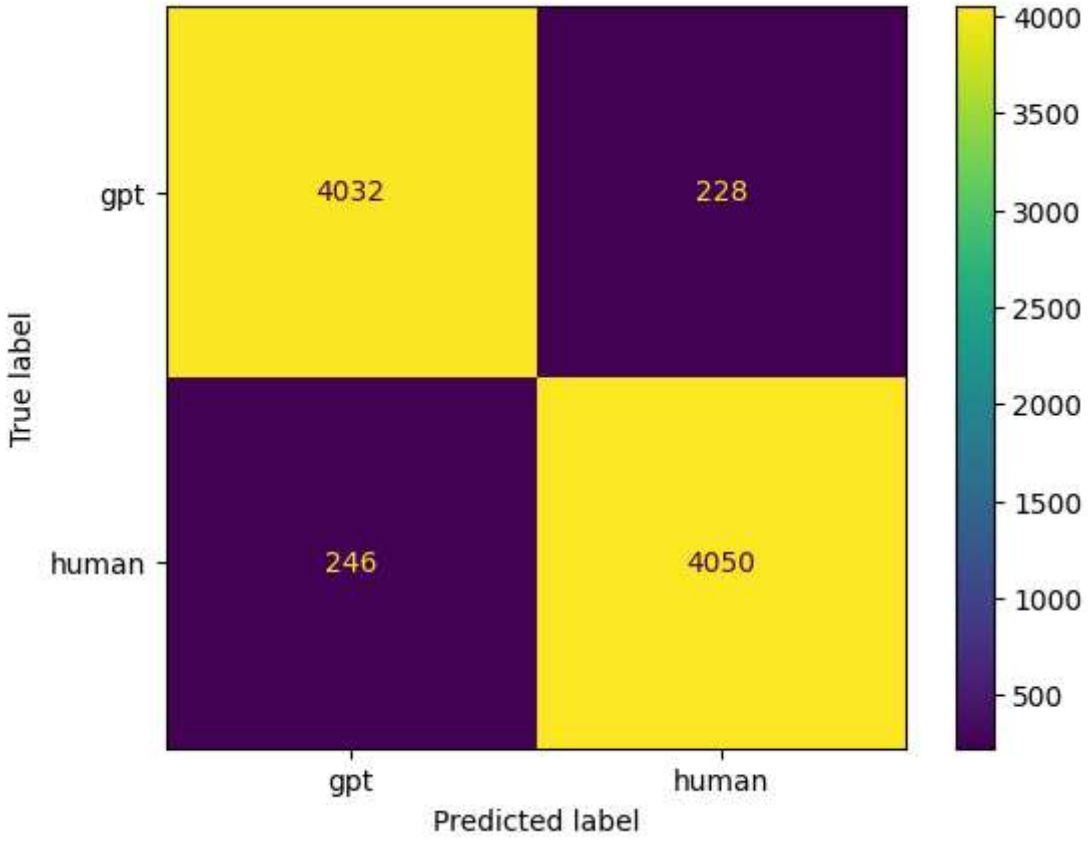
4.1 Model Evaluation

First, to get a brief visual overview of each model's performance, we will plot its confusion matrix and ROC curve.

First, we will plot the confusion matrix for each model.

In [ ]:
```
plot_con_matrix(y_test, rfc_predict).plot()
```

Out[ ]:
```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x1e18fc37c90>
```

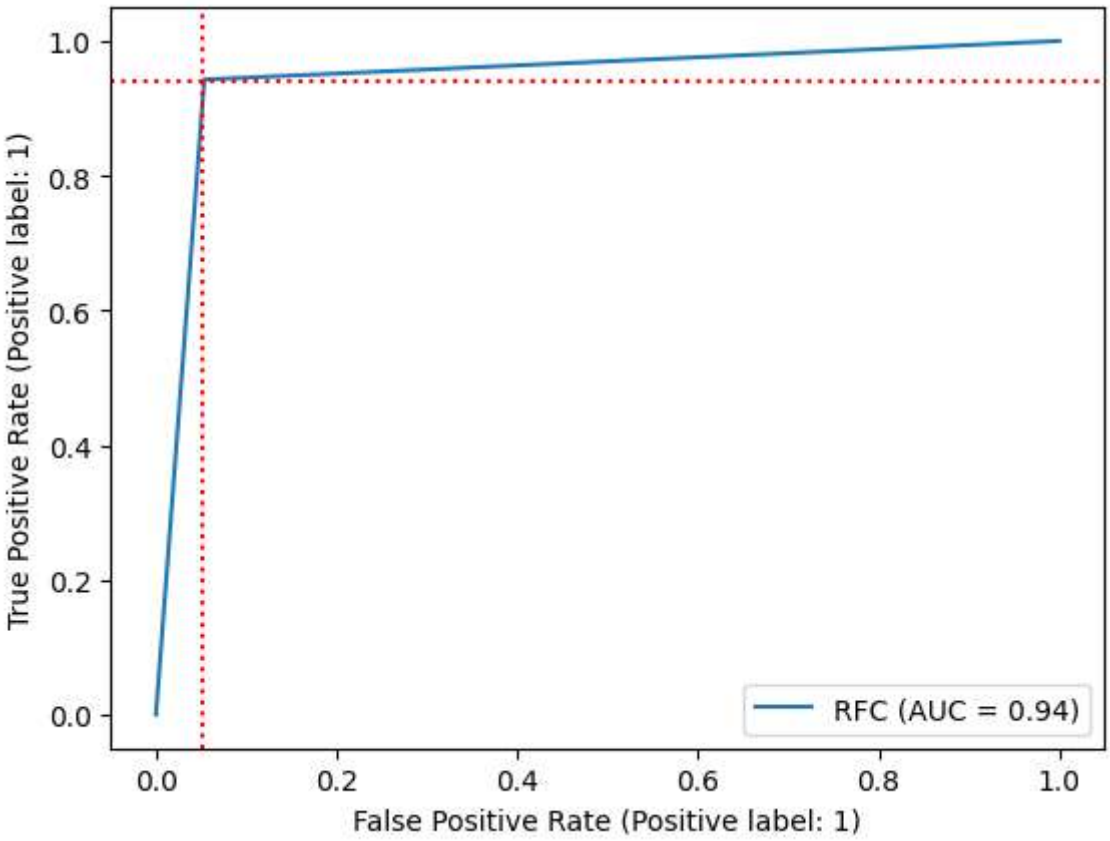


A brief look at the RFC's confusion matrix shows that it performs at a high level, with a high proportion of correctly identified human/GPT answers.

Next, we will plot its ROC curve.

In [ ]:
```
RocCurveDisplay.from_predictions(y_test, rfc_predict, name = 'RFC')
plt.axvline(0.051, color = 'r', linestyle = ':')
plt.axhline(0.94, color = 'r', linestyle = ':')
```

Out[ ]:
```
<matplotlib.lines.Line2D at 0x1e18fd25150>
```



Once again, this shows that the RFC performs well (at 0.95 AUC).

The curve also shows an inflection point at about 0.051 (Red dotted line) False positive rate (FPR), where the model's True positive rate (TPR) does not improve significantly beyond that point.

This raises the possibility of limiting the false positive rate to about 0.051 in instances where false positives are of critical importance (e.g., usage in examination systems). One possible way of achieving this would be using probabilistic classification instead of binary classification. Using the probability weights, along with a set of user-defined scoring rules, could help keep the FPR low, while still maintaining 99% (~0.94 TPR) of the model's performance.

Next, we will calculate the RFC's:

1. Accuracy
2. AUROC Score
3. Equal Error Rate (EER)
4. Precision
5. Recall

In [ ]:
```
# calculate acc, roc, eer, precision, recall
rfc_acc, rfc_roc, rfc_eer, rfc_precision, rfc_recall = calc_performance_stats(y_tes
```

```
# print out stats
print(f'Accuracy: {rfc_acc}, ROC: {rfc_roc}, EER: {rfc_eer}, Precision: {rfc_precis
```

```
Accuracy: 0.9446002805049089, ROC: 0.9446081517035171, EER: 0.05516431924882629, Pre
cision: 0.9467040673211781, Recall: 0.9427374301675978
```

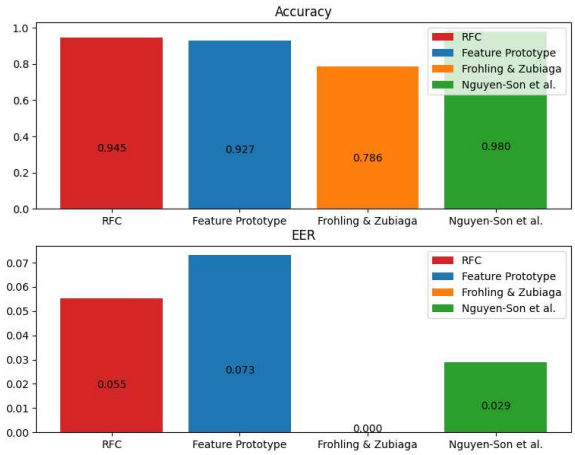
We will also compare the performance of the RFC against our feature prototype, and results from other literature.

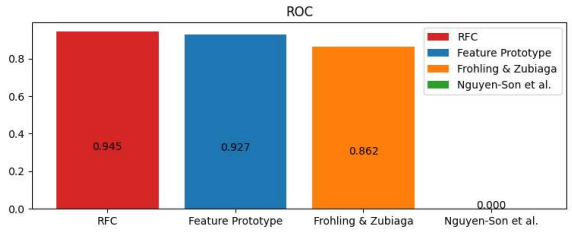
First, we will construct lists of the stats, as well as some labels and colours.

In [ ]:
```
# assemble list of stats
accuracies =  [rfc_acc, 0.92727, 0.786, 0.98]
rocs = [rfc_roc, 0.92715, 0.862, 0]
eers = [rfc_eer, 0.07317, 0, 0.029]


# also assemble list of labels and colours
labels = ['RFC', 'Feature Prototype', 'Frohling & Zubiaga', 'Nguyen-Son et al.']
colors = ['tab:red', 'tab:blue', 'tab:orange', 'tab:green']
```

Finally, we will plot the bar charts (Metrics that are not available in other literature are set to 0)

In [ ]:
```
fig, axes = plt.subplots(2, 2, figsize = (20, 7))

axes[0][0].set_title('Accuracy') # set plot title
axes[0][0].bar(labels, accuracies, label = labels, color = colors) # plot barplot
axes[0][0].legend() # add legend
add_val_labels(labels, accuracies, axes[0][0]) # add value labels

axes[0][1].set_title('ROC')
axes[0][1].bar(labels, rocs, label = labels, color = colors)
axes[0][1].legend()
add_val_labels(labels, rocs, axes[0][1])

axes[1][0].set_title('EER')
axes[1][0].bar(labels, eers, label = labels, color = colors)
axes[1][0].legend()
add_val_labels(labels, eers, axes[1][0])

fig.delaxes(axes[1][1]) # remove last empty plot
```





As expected, our RFC has achieved improved performance from the feature prototype, with improved accuracy, FPR, ROC, and EER.

Additionally, we would consider our RFC to have achieved mostly competitive performance against the models in other literature. It has achieved better accuracy and AUROC compared to Fröhling and Zubiaga [2021]'s best model. Its accuracy is also comparable to the model in Nguyen-Son et al. [2017], though its EER remains significantly higher.

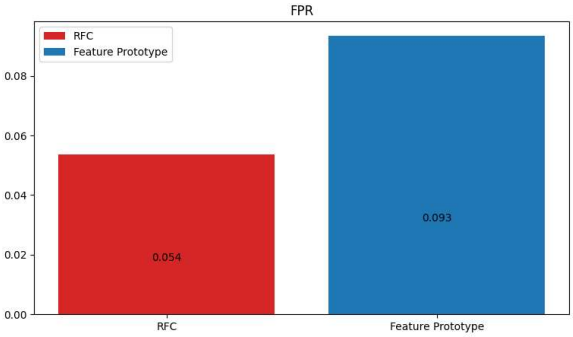
We will also calculate the RFC's FPR and TPR, and visualise it against the performance of the feature prototype.

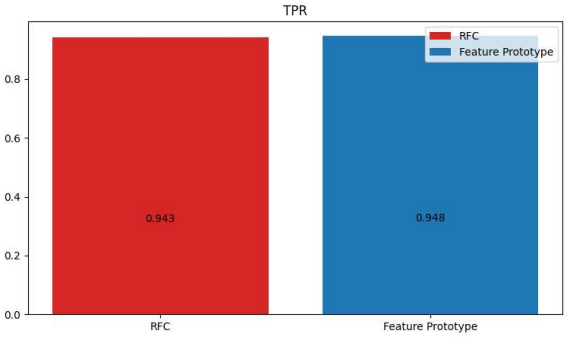
In [ ]:
```
rfc_fpr, rfc_tpr = calc_fpr_tpr(y_test, rfc_predict)
print(f'RFC - FPR: {rfc_fpr}, TPR: {rfc_tpr}')
```

```
RFC - FPR: 0.05352112676056338, TPR: 0.9427374301675978
```

In [ ]:
```
fprs = [rfc_fpr, 0.09350]
tprs = [rfc_tpr, 0.94779]
```

In [ ]:
```
fig, axes = plt.subplots(1, 2, figsize = (20, 5))

axes[0].set_title('FPR') # set plot title
axes[0].bar(labels[0:2], fprs, label = labels[0:2], color = colors) # plot barplot
axes[0].legend() # add legend
add_val_labels(labels[0:2], fprs, axes[0]) # add value labels

axes[1].set_title('TPR')
axes[1].bar(labels[0:2], tprs, label = labels[0:2], color = colors)
axes[1].legend()
add_val_labels(labels[0:2], tprs, axes[1])
```





These bar graphs show that the RFC has significantly improved the FPR, which was a point of concern regarding the Feature Prototype's performance.

5. Conclusions

From the testing and evaluation carried out above, we would conclude that our RFC model has achieved the stated goals of:

1. Performed better than the baseline of human performance (better than random chance (RFC accuracy is >50%))
2. Having competitive/better performance with results from established literature.

Thus, we would consider the stated aims for this project to be achieved, and the project to be complete.

6. Future Work

Our model's EER still remains higher than the model in Nguyen-Son et al. [2017]. In this vein, we would consider the earlier possibility of experimenting with probabilistic classification and scoring regimes to be a possible area where the model's FPR (and thus EER) can be improved.

Another possibility we would like to explore in the future is using implementing an ensemble classifier, such as the ones found in Fröhling and Zubiaga [2021]. These ensemble classifiers combine multiple classification algorithms to achieve better performance, and is an area we would like to explore further. However, one risk of using ensemble classifiers is that adding poorer performing classifiers may also 'dilute' the better performing ones, damaging performance.

7. References

[1] Lavanya Reddy. 2019. Answer to "AttributeError: probability estimates are not available for loss='hinge'." Stack Overflow. Retrieved June 28, 2023 from https://stackoverflow.com/a/57789235

[2] Craig. 2017. Answer to "Best Fit Line on Log Log Scales in python 2.7." Stack Overflow. Retrieved August 18, 2023 from https://stackoverflow.com/a/43838500

[3] James S. 2017. Answer to “Equal Error Rate in Python.” Stack Overflow. Retrieved June 7, 2023 from https://stackoverflow.com/a/46026962

[4] Leon Fröhling and Arkaitz Zubiaga. 2021. Feature-based detection of automated language models: tackling GPT-2, GPT-3 and Grover. PeerJ Comput Sci 7, (April 2021), e443. DOI:https://doi.org/10.7717/peerj-cs.443

[5] Hoang-Quoc Nguyen-Son, Ngoc-Dung T. Tieu, Huy H. Nguyen, Junichi Yamagishi, and Isao Echi Zen. 2017. Identifying computer-generated text using statistical analysis. In 2017 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC), IEEE, Kuala Lumpur, 1504–1511. DOI:https://doi.org/10.1109/APSIPA.2017.8282270