In [1]: import pandas as pd import numpy as np import tensorflow import ast import re import matplotlib.pyplot as plt import seaborn as sns /home/Yusuf/anaconda3/lib/python3.6/site-packages/h5py/__init__.py:36: FutureWarning: Conversion of the second argument of issub dtype from `float` to `np.floating` is deprecated. In future, it will be treated as `np.float64 == np.dtype(float).type`. from ._conv import register_converters as _register_converters df = pd.read csv("Data/Merged Data.csv") df.head() Unnamed: GICS txt_link ticker text release_date items cik doc_name Sector 0001564590-0001564590-Health 18-006570.txt 2018-03-22 ['Item 0 0 1090872 18http://www.sec.gov/Archives/edgar/data/1090872... 0.2 Care : 20180322 16:22:07 5.07'] 006570.txt 0001564590... ['Item 0001090872-2.02', 0001090872-Health 18-000002.txt 2018-02-14 'Item Α 1090872 18http://www.sec.gov/Archives/edgar/data/1090872... 7.2 Care : 20180214 16:27:02 2.02', 000002.txt 0001090872... 'Item 9.01'] 0001564590-['Item 0001564590-18-000605.txt Health 2018-01-18 5.02', **2** 2 1090872 18http://www.sec.gov/Archives/edgar/data/1090872... 0.4 : 20180118 16:09:52 'Item 000605.txt 0001564590... 9.01'] ['Item 0001090872-2.02', 0001090872-Health 17-000015.txt 2017-11-20 'Item **3** 3 Α 1090872 17http://www.sec.gov/Archives/edgar/data/1090872... 0.3 Care : 20171120 16:09:02 2.02', 000015.txt 0001090872... 'Item 9.01'] ['Item 0001090872-2.02', 0001090872-Health 17-000011.txt 2017-08-15 'Item 1090872 17http://www.sec.gov/Archives/edgar/data/1090872... : 20170815 16:12:29 2.02', 000011.txt 0001090872... 'Item 9.01'] df['release date'] = df['release date'].map(lambda x: pd.to datetime(x)) df['items'] = df['items'].map(lambda x: ast.literal eval(x)) **Text PreProcessing** 1. Remove extra whitespace 2. Tokenize 3. Remove punctuation, stopwords, convert to lower case 4. Lemmatize 5. Load pre-trained word embeddings In [3]: # import spacy # from nltk.corpus import stopwords # import string # import matplotlib.pyplot as plt # import seaborn as sns # stop words = stopwords.words("english") # nlp = spacy.load("en core web sm") # nlp.max length = 76131683# punctuations = string.punctuation from nltk.corpus import stopwords stop words = stopwords.words("english") from nltk.stem import WordNetLemmatizer from nltk.tokenize import word tokenize wordnet lemmatizer = WordNetLemmatizer() import string punctuations = string.punctuation import matplotlib.pyplot as plt import seaborn as sns import dask.dataframe as dd from dask.multiprocessing import get from dask.diagnostics import ProgressBar def cleanup text(doc, logging=False): $doc = re.sub('\s+', ' ', doc).strip()$ doc = nlp(doc, disable=['parser', 'ner']) tokens = [tok.lemma .lower().strip() for tok in doc] tokens = [tok for tok in tokens if tok.isalpha()] tokens = [tok for tok in tokens if tok not in stop words and tok not in punctuations] tokens len = len(tokens) tokens = ' '.join(tokens) return tokens, tokens len def nltk tokenizer(text): try: tokens = [word for word in word tokenize(text) if word.isalpha()] tokens = list(filter(lambda t: t not in punctuations, tokens)) tokens = list(filter(lambda t: t.lower() not in stop words, tokens)) filtered tokens = [] for token in tokens: if re.search('[a-zA-Z]', token): filtered tokens.append(token) filtered tokens = list(map(lambda token: wordnet lemmatizer.lemmatize(token.lower()), filtered tokens)) filtered tokens = list(filter(lambda t: t not in punctuations, filtered tokens)) return filtered tokens except Exception as e: raise e def dask tokenizer(df): df['processed text'] = df['text'].map(nltk tokenizer) df['text len'] = df['processed text'].map(lambda x: len(x)) return df In [8]: pbar = ProgressBar() pbar.register() ddata = dd.from_pandas(df, npartitions=20) df = ddata.map_partitions(dask_tokenizer).compute(get=get)] | 0% Completed | 8min 40.5s IOPub message rate exceeded. The notebook server will temporarily stop sending output to the client in order to avoid crashing it. To change this limit, set the config variable `--NotebookApp.iopub_msg_rate_limit`. Current values: NotebookApp.iopub_msg_rate_limit=1000.0 (msgs/sec) NotebookApp.rate_limit_window=3.0 (secs)] | 0% Completed | 11min 28.6s [IOPub message rate exceeded. The notebook server will temporarily stop sending output to the client in order to avoid crashing it. To change this limit, set the config variable `--NotebookApp.iopub_msg_rate_limit`. Current values: NotebookApp.iopub_msg_rate_limit=1000.0 (msgs/sec) NotebookApp.rate_limit_window=3.0 (secs) [#########] | 22% Completed | 30min 14.2s IOPub message rate exceeded. The notebook server will temporarily stop sending output to the client in order to avoid crashing it. To change this limit, set the config variable `--NotebookApp.iopub_msg_rate_limit`. Current values: NotebookApp.iopub_msg_rate_limit=1000.0 (msgs/sec) NotebookApp.rate_limit_window=3.0 (secs) [#################################] | 70% Completed | 46min 56.1s IOPub message rate exceeded. The notebook server will temporarily stop sending output to the client in order to avoid crashing it. To change this limit, set the config variable `--NotebookApp.iopub_msg_rate_limit`. Current values: NotebookApp.iopub_msg_rate_limit=1000.0 (msgs/sec) NotebookApp.rate_limit_window=3.0 (secs)] | 80% Completed | 52min 58.2s IOPub message rate exceeded. The notebook server will temporarily stop sending output to the client in order to avoid crashing it. To change this limit, set the config variable `--NotebookApp.iopub_msg_rate_limit`. Current values: NotebookApp.iopub_msg_rate_limit=1000.0 (msgs/sec) NotebookApp.rate_limit_window=3.0 (secs) [################################] | 100% Completed | | 100% Completed | 1hr | 100% Completed | In [9]: df.head() **GICS Unnamed:** ticker txt_link cik doc_name text release_date items Sector 0001564590-0001564590-18-006570.txt Health 2018-03-22 ['Item 0 0 1090872 18http://www.sec.gov/Archives/edgar/data/1090872... 0.2 Care : 20180322 16:22:07 5.07'] 006570.txt 0001564590... ['Item 0001090872-2.02', 0001090872-18-000002.txt 2018-02-14 'Item Health 7.2 1 Α 1090872 http://www.sec.gov/Archives/edgar/data/1090872... Care : 20180214 16:27:02 2.02', 000002.txt 0001090872... 'Item 9.01'] 0001564590-['Item 0001564590-18-000605.txt 2018-01-18 Health 5.02', **2** 2 Α 1090872 18http://www.sec.gov/Archives/edgar/data/1090872... 0.4 Care 'Item 16:09:52 : 20180118 000605.txt 0001564590... 9.01'] ['Item 0001090872-2.02', 0001090872-2017-11-20 17-000015.txt Health 'Item 1090872 **3** 3 Α 17http://www.sec.gov/Archives/edgar/data/1090872... 0.3 Care 16:09:02 : 20171120 2.02', 000015.txt 0001090872... 'Item 9.01'] ['Item 0001090872-2.02', 0001090872-'Item Health 17-000011.txt 2017-08-15 Α 1090872 17http://www.sec.gov/Archives/edgar/data/1090872... 5.1 Care 20170815 16:12:29 2.02', 000011.txt 0001090872... 'Item 9.01'] In []: df.to csv("Data/lemmatized text.csv", chunksize=1000) In [120] df = pd.read csv("Data/lemmatized text.csv") df.head() **Unnamed: Unnamed: GICS** ticker txt_link cik doc_name text release_date Sector 0.1 0001564590-0001564590-18-006570.txt 2018-03-22 Health 0 Α 1090872 http://www.sec.gov/Archives/edgar/data/1090872.. 18-Care : 20180322 16:22:07 006570.txt 0001564590... 0001090872-0001090872-18-000002.txt 2018-02-14 Health http://www.sec.gov/Archives/edgar/data/1090872... 1090872 **1** 1 Α 18-Care : 20180214 16:27:02 000002.txt 0001090872... 0001564590-0001564590-18-000605.txt 2018-01-18 Health 1090872 **2** 2 2 Α http://www.sec.gov/Archives/edgar/data/1090872... 18-Care : 20180118 16:09:52 000605.txt 0001564590... 0001090872-0001090872-17-000015.txt 2017-11-20 Health **3** 3 3 1090872 17http://www.sec.gov/Archives/edgar/data/1090872... Care : 20171120 16:09:02 000015.txt 0001090872... 0001090872-0001090872-Health 17-000011.txt 2017-08-15 1090872 17http://www.sec.gov/Archives/edgar/data/1090872... : 20170815 16:12:29 000011.txt 0001090872... In [121] df.drop(['Unnamed: 0','Unnamed: 0.1',"doc name",'txt link','text'],axis=1,inplace=True) df['items'] = df['items'].map(lambda x: ast.literal eval(x)) df['items'] = df['items'].map(lambda items: [' '.join(x.split()) for x in items]) plt.style.use("ggplot") df['text len'].plot.hist(bins=50, normed=True) plt.xlabel("Document Length") plt.show() /home/Yusuf/anaconda3/lib/python3.6/site-packages/matplotlib/axes/ axes.py:6462: UserWarning: The 'normed' kwarg is deprecated, and has been replaced by the 'density' kwarg. warnings.warn("The 'normed' kwarg is deprecated, and has been " 0.000016 0.000014 0.000012 Preducts 0.000000 0.000000 0.000000 0.000004 0.000002 0.000000 500000 1000000 1500000 2000000 2500000 3000000 Document Length int(df['text len'].quantile(.9)) 34603 df['text len'].describe() count 1.671000e+04 1.951087e+04 mean 7.462709e+04 std 9.500000e+01 6.800000e+02 1.902500e+03 75% 8.176750e+03 2.934190e+06 Name: text_len, dtype: float64 In [126] sns.countplot(y=df['GICS Sector']) plt.savefig("Graphs/sectors.png", format="png") plt.show() Health Care Industrials Consumer Discretionary Information Technology g Consumer Staples Utilities Financials Real Estate Materials Telecommunication Services 500 1000 1500 2000 2500 count In [127] #Count plot of signals sns.countplot(df['signal']) plt.show() 10000 8000 6000 count 4000 2000 0 down stay signal Classes are imbalanced, will need to adjust In [128] df['release date'] = df['release date'].map(lambda x: pd.to datetime(x)) sns.countplot(pd.DatetimeIndex(df['release date']).year) plt.savefig("Graphs/year_balances.png", format="png") plt.show() 6000 5000 4000 count 3000 2000 1000 2013 2014 2015 2016 2017 2012 release_date