ChatGPTに依頼：　CSVデータよりテキストデータマインドの分析をお願いします

ChatGPT回答

簡易的なトピック分析（LDA）を実施したところ、前処理が十分ではないため、URLや定型文（「こちら」「資料」「です」など）が多く抽出され、意味のあるトピックが得られませんでした。

	主なキーワード
1	https, youtube, watch, 動画, こちら
2	著者, 発行所, 資料, 紹介します
3	python, google, excel
4	資料, します, こちら
5	英語の一般単語（the, of, is…）

現在のCSVは一般的なテキストクリーニングだけでは十分ではありません。

「資料はこちら」「動画をご覧ください」「著者」「発行所」「〇〇を紹介します」英語のリンク由来の語（the, of, is など）

ChatGPTに依頼：　名詞のみで分析お願いします。

ChatGPT回答

順位	キーワード	出現回数
1	場合	2,888
2	説明	1,541
3	データ	1,541
4	クリック	1,232
5	問題	1,019
6	ブログ	952
7	必要	937
8	Excel	831

より精度の高い分析を行うには、MeCab や SudachiPy を用いて品詞を判定し、「名詞」のみを抽出する方法が適しています。これにより、「Python」「AI」「機械学習」「統計」「数学」「ChatGPT」などの専門用語をより正確に抽出でき、共起ネットワークやトピック分析の質も向上します。

ChatGPT回答

Jupyter Notebookでブログ記事のテキストマイニングを一通り実施する流れとプログラム

1. CSV読込
2. 前処理
3. SudachiPyによる名詞抽出
4. 頻出語ランキング
5. WordCloud
6. 共起ネットワーク
7. LDAによるトピック分析

Jupyter Notebookで
エラーが出ると
Geminiが手直ししてくれる

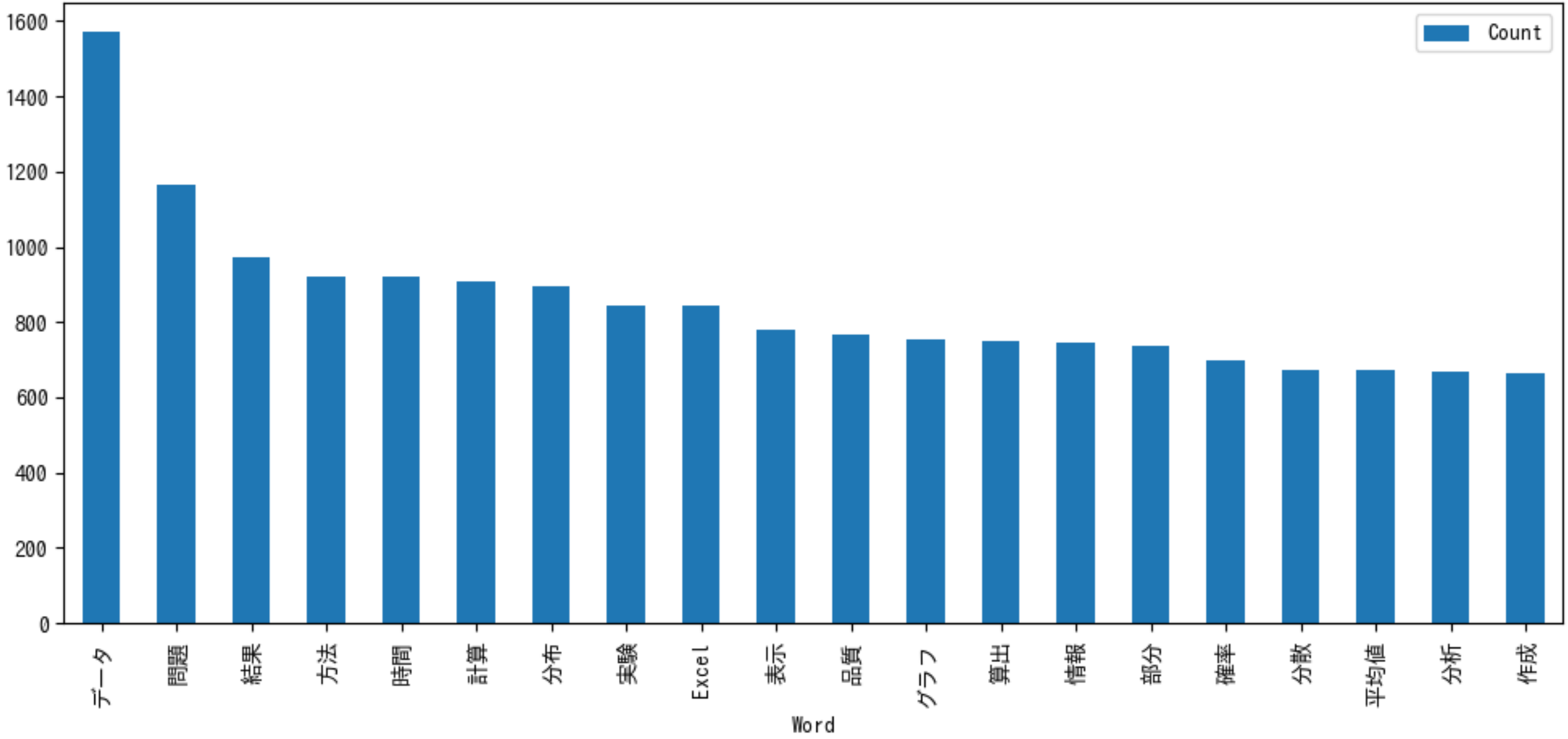
```
# ライブラリのインストール
!pip install pandas sudachipy sudachidict-core
!pip install wordcloud
!pip install networkx
!pip install matplotlib
!pip install scikit-learn

# 日本語フォントのインストール
!apt-get -y install fonts-ipafont-gothic

import os
import matplotlib.font_manager as fm
import matplotlib.pyplot as plt

# Note: fm.get_cachedir() is not available in this matplotlib version.
# Relying on fm.fontManager.addfont() to register new fonts.

font_path = "/usr/share/fonts/opentype/ipafont-gothic/ipag.ttf"

# フォントマネージャーにフォントを追加
fm.fontManager.addfont(font_path)
# fm.fontManager.rebuild() # フォントキャッシュを再構築（このバージョンでは利用不可）

# フォントファイルから正確なフォントファミリー名を取得
# これにより、MatplotlibがTTFファイル内で認識する名前を確実に使用します。
prop = fm.FontProperties(fname=font_path)
actual_font_name = prop.get_name()

# matplotlibの日本語フォント設定
# 取得した正確なフォントファミリー名を優先して設定
plt.rcParams['font.family'] = [actual_font_name, 'sans-serif']
plt.rcParams['axes.unicode_minus'] = False # マイナス記号の表示を修正
```

```
# ライブラリ読込
import pandas as pd
import re

from collections import Counter
from itertools import combinations

from sudachipy import dictionary

from wordcloud import WordCloud

import networkx as nx

from sklearn.feature_extraction.text import CountVectorizer
from sklearn.decomposition import LatentDirichletAllocation

# CSV読込
df = pd.read_csv("cleaned_blog_posts.csv")

df.head()
# 前処理
def clean_text(text):

text = str(text)

text = re.sub(r"http\S+", "", text)
text = re.sub(r"www\S+", "", text)
text = re.sub(r"[0-9]", "", text)
text = re.sub(r"[!-/:-@[{-~]", " ", text)

return text

df["text"] = df["Cleaned_Content"].apply(clean_text)
# 名詞抽出
tokenizer = dictionary.Dictionary().create()
# 不要語辞書
stopwords = {

"こちら","資料","動画","紹介","著者","発行所","場合",
"説明","ご覧","そう","クリック","発行","THE","ブログ","必要","ところ","以下
","ファイル",
"こと","もの","ため","よう","これ","それ","右下","右上","左下","左上",
"上述","最近","本日","サイト","以上","ページ","全て",
"さん","ます","です","ある","いる","する",
"今回","以前","今日","昨日"
}
```

```
# 名詞抽出関数
def extract_nouns(text):

words=[]

for token in tokenizer.tokenize(text):

if token.part_of_speech()[0]=="名詞":

word=token.dictionary_form()

if len(word)>1 and word not in stopwords:

words.append(word)

return words
# 実行
df["nouns"]=df["text"].apply(extract_nouns)
# 頻出語ランキング
counter=Counter()

for words in df["nouns"]:

counter.update(words)

freq=pd.DataFrame(counter.most_common(50),
columns=["Word","Count"])

freq
# 棒グラフ
ax = freq.head(20).plot.bar(
x="Word",
y="Count",
figsize=(12,5)
)

# X軸のラベルに日本語フォントを適用
for label in ax.get_xticklabels():
label.set_fontproperties(fm.FontProperties(fname=font_path))

plt.show()
# WordCloud
# Colab環境に存在する日本語フォントのパスを指定
# fonts-ipafont-gothic パッケージに含まれる IPA Gothic フォントのパス
# 正しいパスはlsコマンドで確認後、更新してください。
wc=WordCloud(
font_path=font_path,
width=1600,
height=900,
background_color="white"
)
# WordCloud
wc.generate_from_frequencies(counter)

plt.figure(figsize=(14,8))
plt.imshow(wc)
plt.axis("off")
plt.show()
# 共起ネットワーク
cooccur=Counter()

for words in df["nouns"]:

words=list(set(words))

for pair in combinations(words,2):

cooccur[pair]+=1
# 上位100本だけ描画
G=nx.Graph()

for pair,count in cooccur.most_common(100):

if count>=5:

G.add_edge(pair[0],pair[1],weight=count)
# 描画
plt.figure(figsize=(15,15))

pos=nx.spring_layout(G,k=0.5)

nx.draw_networkx_nodes(G,pos,node_size=400)

nx.draw_networkx_labels(G,pos,font_family=actual_font_name) # font_familyを正確
な名前に変更

nx.draw_networkx_edges(G,pos,alpha=0.3)

plt.axis("off")
plt.show()
# LDAトピック分析,文章を名詞列に変換
documents=[" ".join(x) for x in df["nouns"]]
```

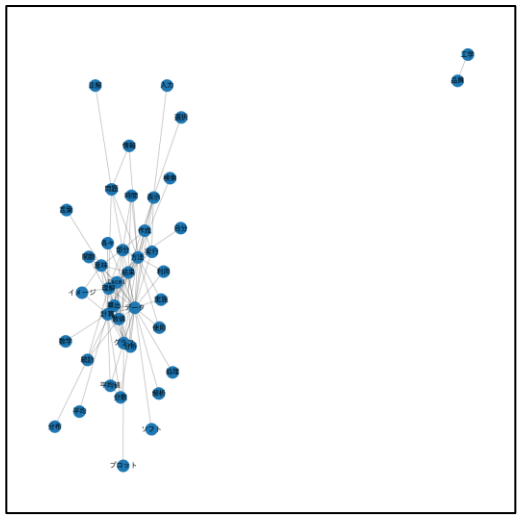
```
# 文書ベクトル
vectorizer=CountVectorizer()

X=vectorizer.fit_transform(documents)
# LDA
lda=LatentDirichletAllocation(
n_components=5,
random_state=0
)

lda.fit(X)
# トピック表示
feature_names=vectorizer.get_feature_names_out()

for topic_idx,topic in enumerate(lda.components_):

print("Topic",topic_idx+1)

print([feature_names[i]
for i in topic.argsort()[:-11:-1]])

print()
# 保存,頻出語
freq.to_csv("frequency.csv",index=False,encoding="utf-8-sig")
# WordCloud
plt.savefig("wordcloud.png",dpi=300)
```

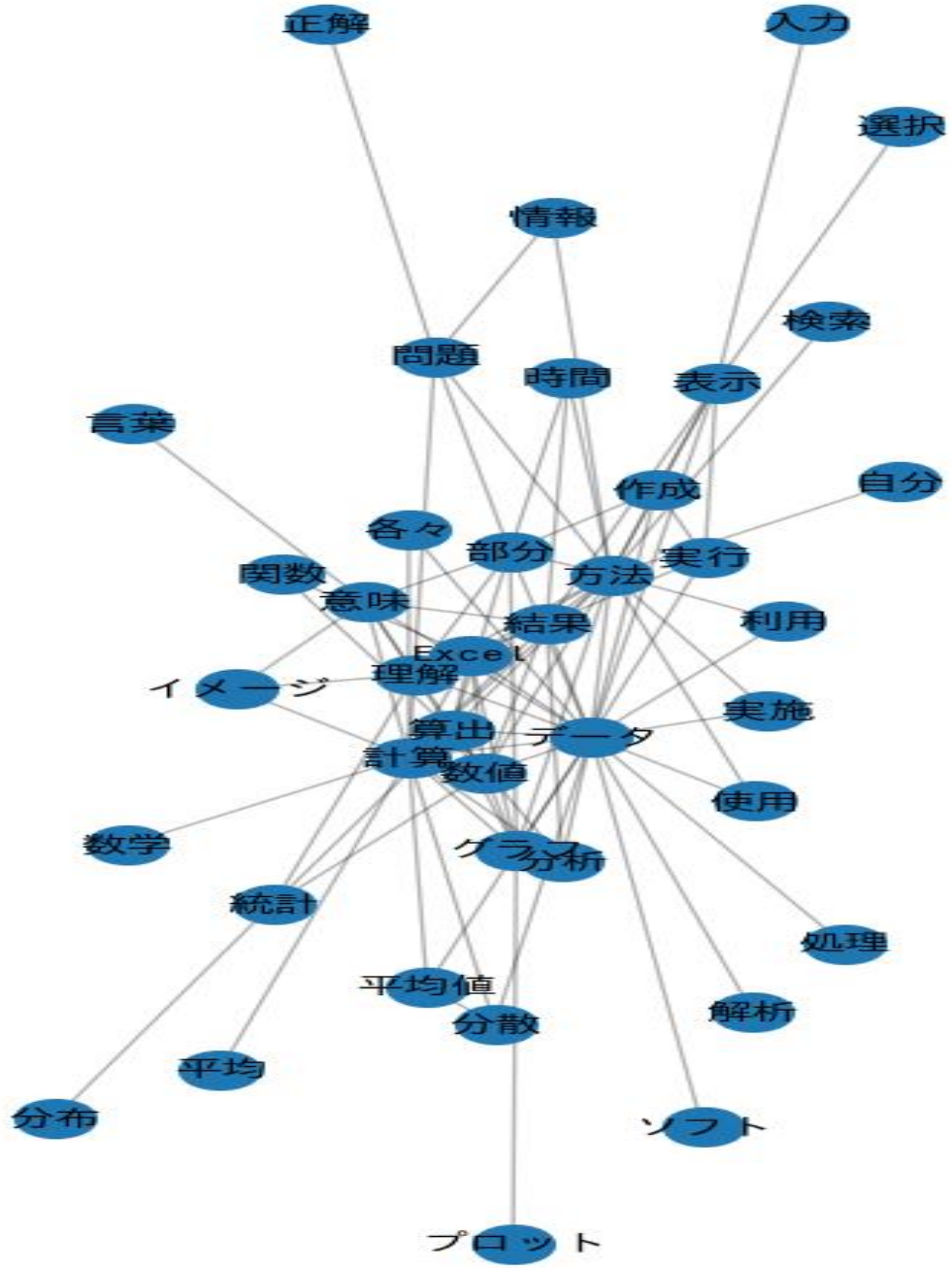
頻出語ランキング

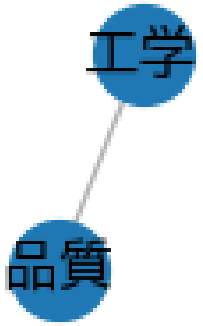

Count
1600
1400
1200
1000
800
600
400
200
0
データ
問題
結果
方法
時間
計算
分布
実験
Excel
表示
品質
グラフ
算出
情報
部分
確率
分散
平均値
分析
作成
Word


データ
結果
問題
Excel
計算
算出
確率
分布
表示
方法
品質
時間
集計
部分
情報
平均値
写真
グラフ
分析
作成
数値
分散
意味
統計
検定
行列
関係
位置
効果
関数
距離
方向
事例
実行
言葉
理解
工学
日本
研修
機能
人間
登場
変化
状態
評価
数学
研究
比較
最初
係数
実施
解析
選択
検索
自分
利用
設定
因子
ソフト
イメージ
管理
使用
of
is
AI
SN
to
Python
Google
In
AND
YouTube
website
プログラム
プロット
エネルギー
ツール
スマホ
パラメータ
保存
手法
右図
面積
電子
数字
計画
作製
確認
先日
購入
原因
植物
温度
わけ
勉強
次元
ロボット
思考
環境
興味
実際
予測
入力
技術
最後
誤差
処理
移動
内容
分類
標準偏差
曲線
範囲
標本
判断
先生
現在
質問
製品
順番
各々
画面
作業
学習
過去
直交
今まで
ベクトル
変数
種々
存在
正解
文字
認識
水準
手順
条件
世界
回転
話題
変換
動物
画像
シート
コード
タイトル
平均
反応
工程
以外
記事
子供
影響
原理
数式
システム
英語
経験
特性
横軸
種類
仕事
通り
違い
形状
推定
直線
回帰
セル
インストール
ダウンロード
アクセス
項目
左図
細胞
生産
信号
答え
場所
理由
物質
時代
目的
表現
縦軸
参考
アイコン
真中
図書館
チェック
長さ
ばらつき
判定
要因
正規
検査
変動

共起ネットワーク




正解
入力
選択
情報
検索
問題
時間
表示
言葉
作成
自分
各々
部分
実行
関数
方法
意味
結果
利用
Excel
理解
イメージ
実施
算出
データ
計算
数値
使用
数学
グラフ
分析
統計
処理
平均値
解析
分散
平均
分布
ソフト
プロット


工学
品質

トピックごとの単語の出現頻度を詳しく分析するコードを作成しましょうか？

はい、LDAトピック分析の結果をグラフで可視化しました。 各トピックの上位キーワードの重要度を棒グラフで確認できます。

トピックごとの単語の出現頻度をさらに詳しく分析するコードを作成しましょうか？

トピックごとの単語出現頻度を分析するコードを作成してください

LDAの各トピックに関連する元記事を抽出する方法はありますか？

何を構築しますか？


Gemini 2.5 Flash

トピックごとの単語出現頻度を分析するコードを作成してください

はい、承知いたしました。LDAで割り当てられた各文書の優勢なトピックを特定し、そのトピックに属する文書内での単語出現頻度を詳細に分析するコードを作成します。これにより、各トピックがどのような単語で構成されているかをより深く理解できます。


承認して実行


許可


キャンセル

何を構築しますか？


Gemini 2.5 Flash

LDAトピック分析結果の可視化

各トピックの上位キーワードを棒グラフで可視化します。これにより、各トピックがどのような内容を扱っているかを直感的に把握できます。