Análise de Cluster com texthero

Motivadores

A área de governância possui um sistema de registro de chamados de incidêntes. Esses sistema é responsável por gerenciar todo o ciclo de vida dos chamados. Apesar disso o sistema não possui um processo para analisar os chamados encerrados e observar padrões nos dados que possam trazer insights para melhorar a resolução dos top incidêntes.

A proposta é analisar a descrição dos chamados e encontrar um padrão para identificar macro tópicos de problemas. Um grande desafio é a necessidade de processamento de linguagem natural, haja vista, a grande quantidade de chamados e a forma como as descrições estão armazenadas. Esse estudo visa analisar a viabilidade para esse tipo de problema e que pode ser escalado para outras análises textuais de tópicos.

Import de bibliotecas

A principal biblioteca usada é a 'texthero' para tratamento textual.

In [2]:
```
import pandas as pd
import texthero as hero
import matplotlib.pyplot as plt
from texthero import stopwords
import wordcloud
import numpy as np
from math import sqrt
from sklearn.cluster import KMeans
```

Import da base de dados

In [3]:
```
tickets_db = pd.read_csv('file_db.csv')
```

Filtro de categorias

O estudo de viabilidade está considerando descrições da categoria 2 de Empréstimo Pessoa Física, mais especificamente os registros de erros funcionais.

In [4]:
```
epf_filter = tickets_db['Categoria_2'] == 'EPF - Emprestimo Pessoa Fisica'
erro_filter = tickets_db['Categoria_3'] == 'Erro Funcional'
indisponivel_filter = tickets_db['Categoria_3'] == 'Sistema indisponivel'
aplicativo_df = tickets_db[epf_filter & erro_filter]
aplicativo_df.shape
```

Out[4]: (3896, 33)

Pré-processamento das informações, retirada de stopwords e stemmezation(raiz da palavra)

o hero.clean() executa sete funções quando você passa uma série de pandas. Essas sete funções são:

- minúsculas: minúsculas todo o texto.
- remove_diacritics (): remove todos os acentos das strings.
- remove_stopwords (): remove todas as palavras de parada.
- remove_digits (): remove todos os blocos de dígitos.
- remove*punctuation (): Remove toda a string.punctuation (! “# $% & ‘() * +, -. / :; <=>? @[]^ ` {|} ~).*
- fillna (s): substitui valores não atribuídos por espaços vazios.
- remove_whitespace (): remove todo o espaço em branco entre as palavras

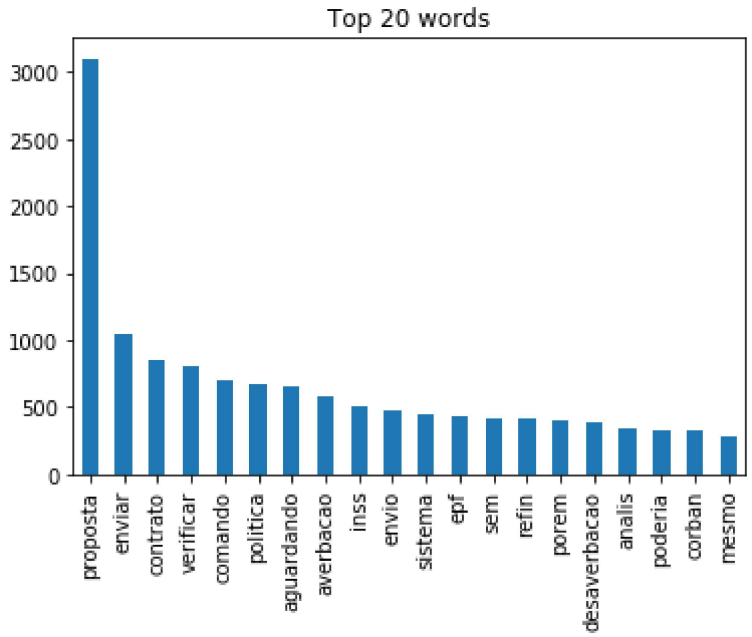
In [6]:
```
problems_content = aplicativo_df[["descricao_problema"]]
problems_content['clean_content'] = hero.clean(problems_content['descricao_
problema'])
problems_content
#tirando palavras específicas
default_stopwords = stopwords.DEFAULT
custom_stopwords = default_stopwords.union(set(['os','ao','nao','noite','ob
rigado', 'att'
                                                'boa','bom','em','foi','n
a','se','com','esta',
                                                'dia','tarde','no','do','d
e','da','o','a','favor',
                                                'que','e','gentileza','bom
dia','favor', 'boa noite','para','por']))

clean_text = hero.remove_stopwords(problems_content['clean_content'], custo
m_stopwords)
clean_text = hero.preprocessing.stem(clean_text,language='portuguese',stem
='porter')
clean_text
```

```
C:\Users\Natale\anaconda3\lib\site-packages\ipykernel_launcher.py:2: Setti
ngWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
```

Out[6]:
```
127       usuario acesso privilegiado banco dado epf est...
151       mutuo extrato corban aba conta vinculada corre...
235       servico safra financeira host safra financeira...
288       acessar menu wf control qualidad saindo seguin...
673       login cbu32311 corban gabriel pere abrindo edi...
                              ...
109669          processar retorno reimplant parado desd att
109677    senhor estamo problema comando desaverbacao do...
109749    retirar pendencia digit da proposta abaixo dig...
109793    peco enviar comando bf exclusao do desconto ju...
109803    proposta encontra liquidada porem ainda encont...
Name: clean_content, Length: 3896, dtype: object
```

Vizualização das palavras mais frequentes

In [7]:
```
NUM_TOP_WORDS = 20

top_20 = hero.visualization.top_words(clean_text).head(NUM_TOP_WORDS)

# Draw the bar chart

top_20.plot.bar(rot=90, title="Top 20 words");

plt.show(block=True);
```



Term frequency-inverse document frequency e Clusterização

TF-IDF é uma medida estatística que avalia a relevância de uma palavra para um documento em uma coleção de documentos. Isso é feito multiplicando duas métricas: quantas vezes uma palavra aparece em um documento e a frequência inversa da palavra em um conjunto de documentos.

Ele tem muitos usos, o mais importante na análise automatizada de texto, e é muito útil para pontuar palavras em algoritmos de aprendizado de máquina para Processamento de Linguagem Natural (PNL).

In [8]:
```
column_names = ["content","tfidf", "kmeans_labels"]

descricao_problemas_df = pd.DataFrame (columns = column_names)

descricao_problemas_df["content"] = clean_text
# convert them into tf-idf features.
descricao_problemas_df['tfidf'] = (
    descricao_problemas_df['content']
    .pipe(hero.tfidf)
)

# perform clustering algorithm by using kmeans()
descricao_problemas_df['kmeans_labels'] = (
    descricao_problemas_df['tfidf']
    .pipe(hero.kmeans, n_clusters=5)
    .astype(str)
)
```

In [9]:
```
descricao_problemas_df.head()
```

Out[9]:

	content	tfidf	kmeans_labels
127	usuario acesso privilegiado banco dado epf est...	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, ...	4
151	mutuo extrato corban aba conta vinculada corre...	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, ...	4
235	servico safra financeira host safra financeira...	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, ...	4
288	acessar menu wf control qualidad saindo seguin...	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, ...	4
673	login cbu32311 corban gabriel pere abrindo edi...	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, ...	4

Visualização dos Clusters por quantidade

In [9]:
```
descricao_problemas_df['kmeans_labels'].value_counts()
```

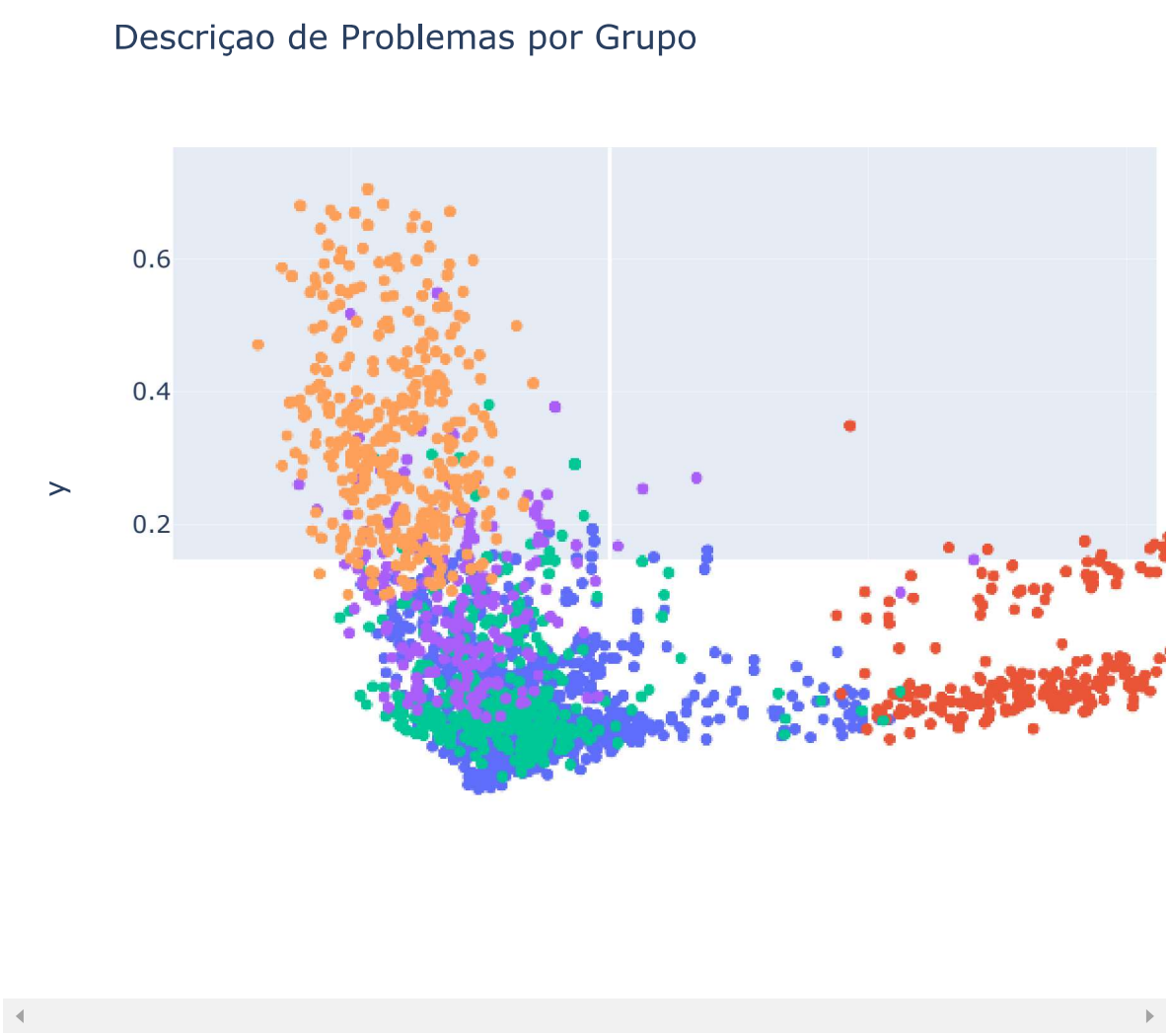
Out[9]:
```
1    2128
4     605
0     448
3     446
2     269
Name: kmeans_labels, dtype: int64
```

Performance de PCA e visualização dos clusters

O objetivo do PCA é encontrar um meio de condensar a informação contida em várias variáveis originais em um conjunto menor de variáveis estatísticas (componentes) com uma perda mínima de informação.

Verifica-se a formação de 5 clusters de descrição de problemas. É necessário ainda a análise de cada grupo para identificar o tópico de cada um.

In [10]:
```
#perform pca
descricao_problemas_df['pca'] = descricao_problemas_df['tfidf'].pipe(hero.pca)

#show scatterplot
hero.scatterplot(descricao_problemas_df, 'pca', color="kmeans_labels", title="Descriçao de Problemas por Grupo")
```

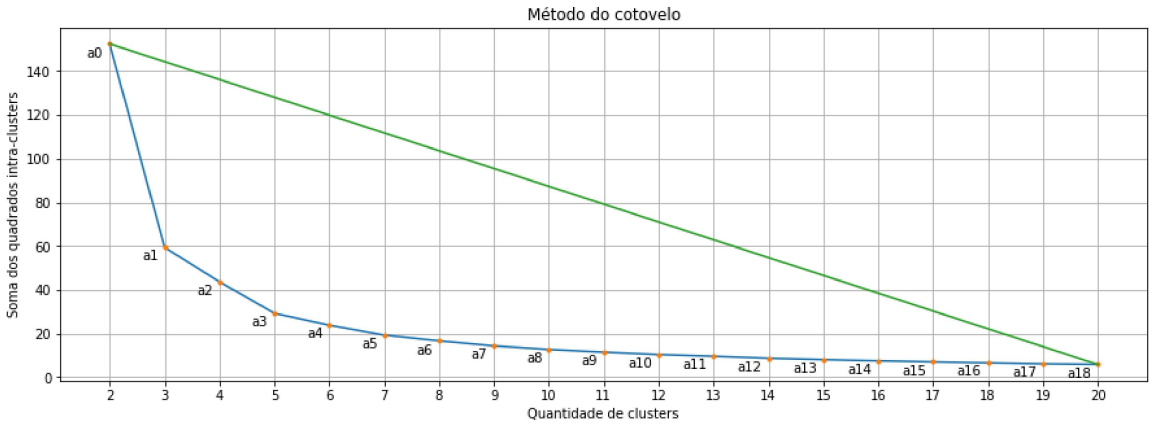


Análise da quantidade ideal de clusters por soma dos quadrados intra-clusters

O ponto que indica o equilíbrio entre maior homogeneidade dentro do cluster e a maior diferença entre clusters, é o ponto da curva mais distante de uma reta traçada entre os pontos P0 = a0 e P1 = a18.

$$\text{distance}(P_0, P_1, (x, y)) = \frac{|(y_1 - y_0)x - (x_1 - x_0)y + x_1 y_0 - y_1 x_0|}{\sqrt{(y_1 - y_0)^2 + (x_1 - x_0)^2}}$$

In [382]:
```
data = np.array(descricao_problemas_df['pca'].to_list())

def calculate_wcss(data):
    wcss = []
    for n in range(2, 21):
        kmeans = KMeans(n_clusters=n)
        kmeans.fit(X=data)
        wcss.append(kmeans.inertia_)

    return wcss

def optimal_number_of_clusters(wcss):
    x1, y1 = 2, wcss[0]
    x2, y2 = 20, wcss[len(wcss)-1]

    distances = []
    for i in range(len(wcss)):
        x0 = i+2
        y0 = wcss[i]
        numerator = abs((y2-y1)*x0 - (x2-x1)*y0 + x2*y1 - y2*x1)
        denominator = sqrt((y2 - y1)**2 + (x2 - x1)**2)
        distances.append(numerator/denominator)

    return distances.index(max(distances)) + 2

df = data
sum_of_squares = calculate_wcss(df)

# calculando a quantidade ótima de clusters
n = optimal_number_of_clusters(sum_of_squares)

x1, x2 = 2, 20
intervalo = range(x1,x2+1)
plt.figure(figsize=(15,5))
plt.title('Método do cotovelo')
plt.xlabel('Quantidade de clusters')
plt.ylabel('Soma dos quadrados intra-clusters')
plt.grid()
plt.xticks(intervalo)
plt.plot(intervalo, sum_of_squares) # pontos laranjas
plt.plot(intervalo, sum_of_squares, '.') # linha azul

y2 = sum_of_squares[len(sum_of_squares)-1]
y1 = sum_of_squares[0]

plt.plot([x2, x1], [y2,y1]) # linha verde

for x,y in zip(intervalo,sum_of_squares): # colocando nome nos pontos
    label = "a{}".format(x-2)
    plt.annotate(label,
                 (x,y),
                 textcoords="offset points",
                 xytext=(-5,-10),
                 ha='right')

plt.show()
print('Número ideal de clusters: ' + str(n))
```




```
Número ideal de clusters: 5
```

Algumas descrições do Cluster 1

In [384]:
```
kmeans_filter = descricao_problemas_df['kmeans_labels'] == '2'
pd.DataFrame(descricao_problemas_df[kmeans_filter]['content'].head(20))
```

Out[384]:

	content
1149	cpf constam dua proposta aguardando politica desd processar ficha
1702	proposta aguardando politica destravar att
2098	proposta parada fase aguardando politica desd
2268	proposta aguardando politica
2483	proposta parada aguardando politica desd cancelar proposta
5039	solicitamo habilitar ferramenta cancelamento proposta poi aguardando politica desd
6096	proposta aguardando politica destravar
7144	cancelar proposta aguardando politica desd
7278	cancelar proposta parada aguardando politica desd print tela enivado email
7281	cancelar proposta parada aguardando politica desd
8839	cancelar proposta aguardando politica
9950	processar ficha aguardando politica
13746	proposta travada aguardando politica desd processar ficha
13920	solicitado cancelamento contrato abaixo aguardando politica
13970	cancelar proposta parada aguardando politica
14067	pp travada aguardando politica desd processar ficha
14516	proposta aguardando politica liberar
14673	verificar proposta parada aguardando politica
14872	proposta aguardando politica desd verificar obrigada
14958	proposta travada aguardando politica processar ficha

WordCloud

In [385]:
```
hero.wordcloud(descricao_problemas_df[kmeans_filter]['content'], max_words=
10) #, width=100, height=100
```

Análise de viabilidade

Analisando o aplicativo 'EPF - Emprestimo Pessoa Física' e olhando os erros funcionais, verificou-se o agrupamento de diferentes informações. Essas informações podem guiar o rumo das entrevistas com as áreas, como:

- Proposta travada/parada aguardando politicas
- proposta cancelada com desaverbação/averbação
- reenvio de arquivo inss / reenviar propostas

A proposta da separação por grupos(clusters) é identificar similaridades e assim identificar tópicos recorrentes de problemas enfrentados pelas áreas.

Esse estudo mostrou a viabilidade de prosseguir com essa abordagem e encontrar mais temas que possam auxiliar nas entrevistas.

Referências

https://cibersistemas.pt/tecnologia/como-usar-o-texthero-para-preparar-um-conjunto-de-dados-baseado-em-texto-para-o-seu-projeto-de-pnl/ (https://cibersistemas.pt/tecnologia/como-usar-o-texthero-para-preparar-um-conjunto-de-dados-baseado-em-texto-para-o-seu-projeto-de-pnl/)

https://medium.com/pizzadedados/kmeans-e-metodo-do-cotovelo-94ded9fdf3a9 (https://medium.com/pizzadedados/kmeans-e-metodo-do-cotovelo-94ded9fdf3a9)

https://medium.com/programadores-ajudando-programadores/k-means-o-que-%C3%A9-como-funciona-aplica%C3%A7%C3%B5es-e-exemplo-em-python-6021df6e2572 (https://medium.com/programadores-ajudando-programadores/k-means-o-que-%C3%A9-como-funciona-aplica%C3%A7%C3%B5es-e-exemplo-em-python-6021df6e2572)

https://medium.com/somos-tera/como-modelar-t%C3%B3picos-atrav%C3%A9s-de-latent-dirichlet-allocation-lda-atrav%C3%A9s-da-biblioteca-gensim-1fa17357ad4b (https://medium.com/somos-tera/como-modelar-t%C3%B3picos-atrav%C3%A9s-de-latent-dirichlet-allocation-lda-atrav%C3%A9s-da-biblioteca-gensim-1fa17357ad4b)

https://leportella.com/pt-br/npl-com-spacy/ (https://leportella.com/pt-br/npl-com-spacy/)

Extra - Análise do texto pré-processado com Top2Vec

In [13]:
```
from top2vec import Top2Vec
```

In [14]:
```
descr_problem_list = descricao_problemas_df['content'].to_list()
```

In [15]:
```
#model = Top2Vec(documents, embedding_model='universal-sentence-encoder')
model = Top2Vec(documents=descr_problem_list, speed="learn", workers=8)
```

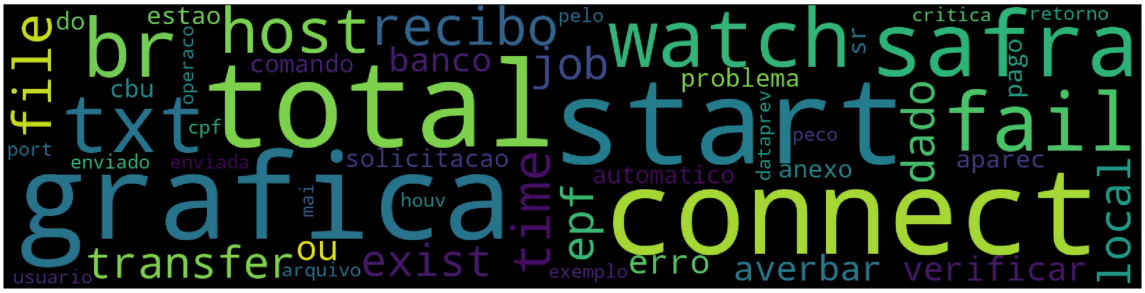
```
2020-12-24 12:17:40,061 - top2vec - INFO - Pre-processing documents for tr
aining
2020-12-24 12:17:40,307 - top2vec - INFO - Creating joint document/word em
bedding
2020-12-24 12:17:55,125 - top2vec - INFO - Creating lower dimension embedd
ing of documents
2020-12-24 12:18:28,027 - top2vec - INFO - Finding dense areas of document
s
2020-12-24 12:18:28,199 - top2vec - INFO - Finding topics
```

In [17]:
```
model.get_num_topics()
topic_words, word_scores, topic_nums = model.get_topics(2)
```

In [18]:
```
for topic in topic_nums:
    model.generate_topic_wordcloud(topic)
```

Topic 0

Topic 1



Top2Vec apresentou resultados confusos e não conclusivos. Carece de análise mais profunda para entender esse comportamento.

Referência

https://github.com/ddangelov/Top2Vec (https://github.com/ddangelov/Top2Vec)