S12_T01_Pipelines_grid_search_and_text_mining_w

July 11, 2022

S12 T01: Pipelines, grid search and text mining

Libraries

[95]:
```
#Python library
import requests
from urllib.request import urlopen
import string


#Data Manipulation
import pandas as pd
import numpy as np
from geopandas import read_file
from sklearn import preprocessing
from imblearn.over_sampling import SMOTE
from imblearn.under_sampling import RandomUnderSampler
from sklearn.ensemble import RandomForestClassifier
from imblearn.pipeline import Pipeline as imbPipeline
from imblearn.pipeline import Pipeline
import category_encoders as ce
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import (train_test_split,
                                      StratifiedKFold,
                                    GridSearchCV)
from sklearn.metrics import (classification_report,
                             multilabel_confusion_matrix)

#Text
from bs4 import BeautifulSoup
from nltk.tokenize import word_tokenize
import nltk
nltk.download('punkt')
from nltk.probability import FreqDist
from nltk.corpus import stopwords
nltk.download('stopwords')
from nltk.stem import PorterStemmer
from nltk.tokenize import sent_tokenize
from nltk.sentiment.vader import SentimentIntensityAnalyzer
```

```
nltk.download('vader_lexicon')



#Data Visualization
import seaborn as sns
import matplotlib.pyplot as plt
plt.style.use('ggplot')
import folium
from sklearn import set_config
import scikitplot as skplt
```

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

Notebook Functions

[384]:
```
#1: Percent of nan values
def percent_nan(df):
    return round((df.isnull().sum()/df.shape[0])*100,2)
```

[385]:
```
#2: group by df
def grouped_df(df,col):
    return df.groupby(col).sum()
```

[386]:
```
#3: isolate a column
def isolate_col(df,col):
    return df[col]
```

[387]:
```
#4 function to the data type to another
def astype_convertion(df,cols,dtype):
    df[cols] = df[cols].astype(dtype)
```

[388]:
```
#5 filter by type of veh
def veh_type(df,col,veh):
    return df.loc[df[col] ==  veh ]
```

[389]:
```
#6 plot map with num of vehicles
def plot_veh(df1,df2,col,label1 ,max_val ):
    fig, ax = plt.subplots(figsize=(12,12))

    df1.plot(col,
                  markersize = 1000 * df1[col]/df1[col].max()     ,
                  ax=ax,color="red")

    df2.plot('Nombre_vehicles',ax=ax, cmap="OrRd" , edgecolor='k',
             alpha= .4,legend=True,legend_kwds=
             {'label': "Total Vehicles by district",
                        'orientation': "horizontal", 'shrink':.8, 'pad':.1}  )

    ax.set_title(col+" "+"by Districts")

    for x, y, labels1  in zip(df1.geometry.x,df1.geometry.y , df1[col] ):
        ax.annotate(labels1, xy=(x , y ), xytext=(3, 3),
                    textcoords="offset points",
                    ha='center'
                   )



    plt.show()
```

[390]:
```
#7 function to select dtype
def select_dtype(df,feat):
    return df.select_dtypes(include = feat).columns
```

[391]:
```
#8 function to drop columns
def drop_cols(df,cols):
    return df.drop(columns=cols)
```

[392]:
```
#9 function to plot the confusion matrix for each class (multiclass)
def plot_conf_matrix(matrix,title):
    fig,ax = plt.subplots(1,1,figsize=(5,5))
    group_counts = ["{0:0.0f}".format(v) for v in matrix.flatten()]
    group_percentages = ["{0:.2%}".format(value)
                         for value in matrix.flatten()/np.sum(matrix)]
    labels = [f"{ant1}\n{ant2}" for ant1, ant2 in␣
 ↪zip(group_counts,group_percentages)]
    labels = np.asarray(labels).reshape(2,2)
    ax = sns.heatmap(matrix, annot=labels, fmt='',
                 cmap='Blues', ax=ax)
    ax.set_title(title)
    ax.set_xlabel('\nPredicted Values')
    ax.set_ylabel('Actual Values ')
    ax.xaxis.set_ticklabels(['False','True'])
    ax.yaxis.set_ticklabels(['False','True'])
    plt.show()
```

Level 1

EDA

```
[393]: #url of the data for the exercise
       url = "https://opendata-ajuntament.barcelona.cat/data/dataset/
        ↪90dc3d6e-1c9a-4136-aed8-74bebb43e052/resource/
        ↪227e5ef8-7ada-44f9-bbbc-d167c0173889/download/2021_tipologia_parc_vehicles.
        ↪csv"
```

```
[394]: #the data shows the type of vehicles by Barcelona district
       data = pd.read_csv(url)
       data
```

```
[394]:        Any Codi_Districte Nom_Districte Codi_Barri  Nom_Barri Seccio_censal  \
       0     2021             01  Ciutat Vella         01   el Raval             1
       1     2021             01  Ciutat Vella         01   el Raval             1
       2     2021             01  Ciutat Vella         01   el Raval             1
       3     2021             01  Ciutat Vella         01   el Raval             1
       4     2021             01  Ciutat Vella         01   el Raval             1
       …     …               …            …          …          …           …
       6399  2021             NC     No consta         NC  No consta            NC
       6400  2021             NC     No consta         NC  No consta            NC
       6401  2021             NC     No consta         NC  No consta            NC
       6402  2021             NC     No consta         NC  No consta            NC
       6403  2021             NC     No consta         NC  No consta            NC

             Tipologia_parc  Nombre_vehicles
       0           Turismes              338
       1              Motos              128
       2        Ciclomotors              682
       3         Furgonetes               51
       4            Camions               23
       …                 …               …
       6399           Motos              101
       6400     Ciclomotors               66
       6401      Furgonetes               22
       6402         Camions                9
       6403  Altres vehicles               4

       [6404 rows x 8 columns]
```

```
[395]: #there are missing values but as a string
       No_Consta_Data = data.loc[data['Codi_Barri'] ==  "NC" ]
       No_Consta_Data
```

```
[395]:        Any Codi_Districte Nom_Districte Codi_Barri  Nom_Barri Seccio_censal  \
       6398  2021             NC     No consta         NC  No consta            NC
       6399  2021             NC     No consta         NC  No consta            NC
       6400  2021             NC     No consta         NC  No consta            NC
       6401  2021             NC     No consta         NC  No consta            NC
```

```
6402  2021            NC     No consta         NC  No consta               NC
6403  2021            NC     No consta         NC  No consta               NC

       Tipologia_parc  Nombre_vehicles
6398         Turismes              192
6399            Motos              101
6400      Ciclomotors               66
6401       Furgonetes               22
6402          Camions                9
6403  Altres vehicles                4
```

[396]:
```
df = data.copy()
```

[397]:
```
#replace the string with a NaN
df.replace(['No consta',"NC"], np.nan,inplace=True)
```

[398]:
```
#% of the NaN value
percent_nan(df)
```

[398]:
```
Any                0.00
Codi_Districte     0.09
Nom_Districte      0.09
Codi_Barri         0.09
Nom_Barri          0.09
Seccio_censal      0.09
Tipologia_parc     0.00
Nombre_vehicles    0.00
dtype: float64
```

[399]:
```
#the missing data is a small percent of the total data and I will drop it
df.dropna(axis=0,inplace=True)
```

[400]:
```
percent_nan(df)
```

[400]:
```
Any                0.0
Codi_Districte     0.0
Nom_Districte      0.0
Codi_Barri         0.0
Nom_Barri          0.0
Seccio_censal      0.0
Tipologia_parc     0.0
Nombre_vehicles    0.0
dtype: float64
```

[401]:
```
 df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
```

```
Int64Index: 6398 entries, 0 to 6397
Data columns (total 8 columns):
 #   Column           Non-Null Count  Dtype 
---  ------           --------------  ----- 
 0   Any              6398 non-null   int64 
 1   Codi_Districte   6398 non-null   object
 2   Nom_Districte    6398 non-null   object
 3   Codi_Barri       6398 non-null   object
 4   Nom_Barri        6398 non-null   object
 5   Seccio_censal    6398 non-null   object
 6   Tipologia_parc   6398 non-null   object
 7   Nombre_vehicles  6398 non-null   int64 
dtypes: int64(2), object(6)
memory usage: 449.9+ KB
```

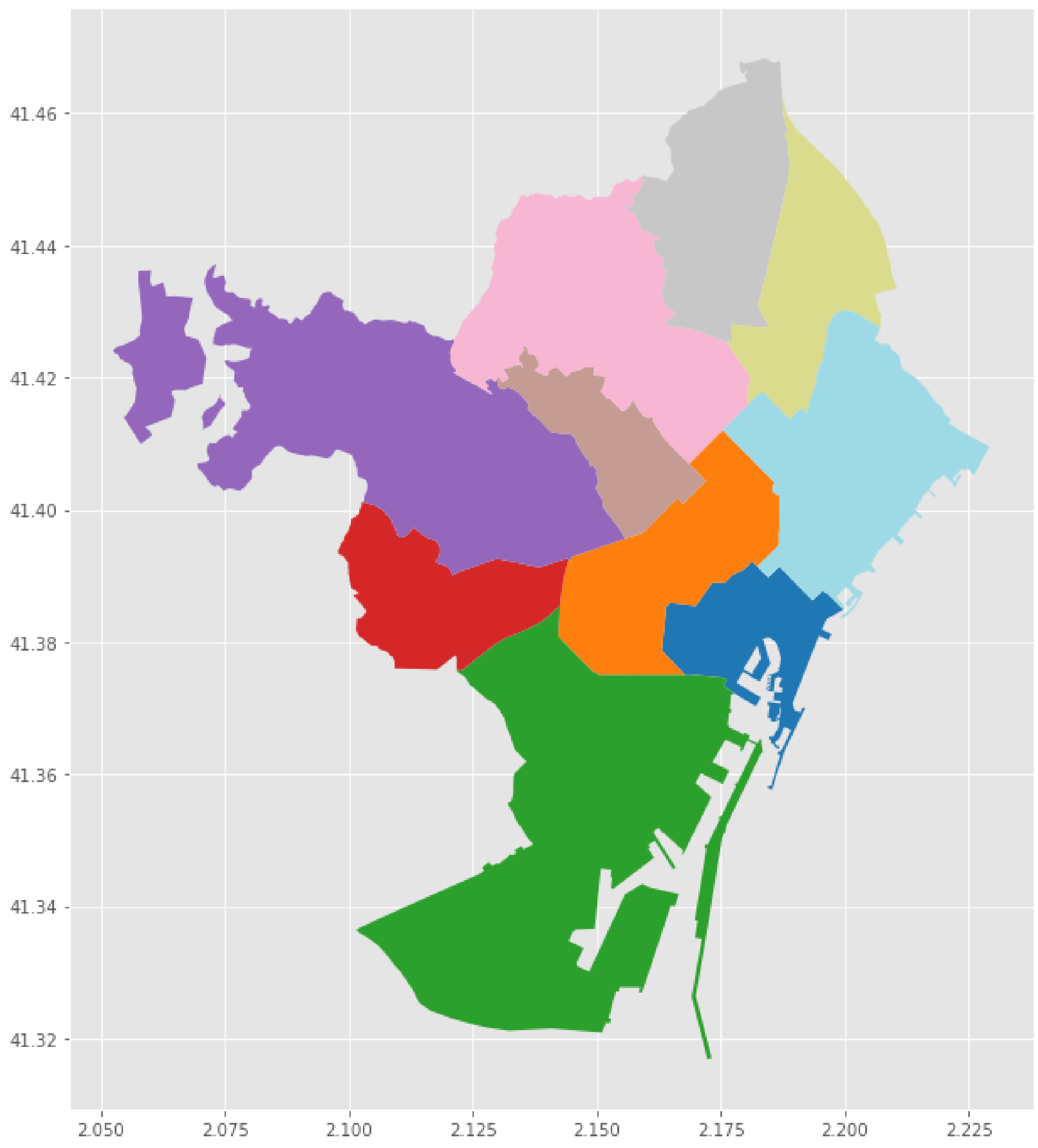
[402]:
```
#transform the feaures as categorical values
astype_convertion(df,["Nom_Districte","Nom_Barri","Tipologia_parc"],"category")
```

[403]:
```
#this dataset is for mapping barcelona
barcelona_shape = read_file("/Volumes/GoogleDrive/Mi unidad/Barcelona Activa/
 ↪Itinerario Data Science/S12/shapefiles_barcelona_distrito/
 ↪shapefiles_barcelona_distrito.shp"
                           ).set_index('c_distri').sort_index()
barcelona_shape.head()
```

[403]:
```
          cartodb_id             n_distri   homes   dones          area  \
c_distri
01                 1         Ciutat Vella   53968   48379  4.368465e+06
02                 2             Eixample  123571  142906  7.476392e+06
03                 3       Sants-MontjuÃ¯c   87877   95243  2.294042e+07
04                 4            Les Corts   38331   43939  6.017532e+06
05                 5  SarriÃ -Sant Gervasi   67799   80113  2.009280e+07

                 perim        coord_x       coord_y  \
c_distri
01        21035.207261  431616.773990  4.581564e+06
02        13902.573980  430243.353657  4.582773e+06
03        47125.925905  428562.773279  4.578163e+06
04        12481.472647  426369.646389  4.582295e+06
05        37563.642237  425388.507986  4.585170e+06

                                   web_1  \
c_distri
01             http://www.bcn.cat/ciutatvella
02                http://www.bcn.cat/eixample
03           http://www.bcn.cat/sants-montjuic
04                http://www.bcn.cat/lescorts
```

```
05        http://www.bcn.cat/sarria-santgervasi

                                                     web_2  \
c_distri
01        http://www.bcn.cat/estadistica/catala/dades/in…
02        http://www.bcn.cat/estadistica/catala/dades/in…
03        http://www.bcn.cat/estadistica/catala/dades/in…
04        http://www.bcn.cat/estadistica/catala/dades/in…
05        http://www.bcn.cat/estadistica/catala/dades/in…

                                                     web_3  \
c_distri
01        http://www.bcn.cat/estadistica/catala/dades/gu…
02        http://www.bcn.cat/estadistica/catala/dades/gu…
03        http://www.bcn.cat/estadistica/catala/dades/gu…
04        http://www.bcn.cat/estadistica/catala/dades/gu…
05        http://www.bcn.cat/estadistica/catala/dades/gu…

                                                  geometry
c_distri
01        POLYGON ((2.18239 41.39143, 2.18346 41.39061, …
02        POLYGON ((2.18239 41.39143, 2.18135 41.39222, …
03        POLYGON ((2.16785 41.37498, 2.16856 41.37495, …
04        POLYGON ((2.10291 41.40109, 2.10295 41.40110, …
05        MULTIPOLYGON (((2.07212 41.41270, 2.07050 41.4…
```

[404]:
```
#map of barcelona by distric
fig, ax = plt.subplots(figsize=(12, 12))
barcelona_shape.plot('cartodb_id', cmap='tab20',ax=ax)
plt.show()
```



```
[405]: #use function (2 & 3) to create a pandas pipe for group and isolate the
       #veh by neighborhood
       veh_by_distric = df.pipe(grouped_df,col="Codi_Districte").
        ↪pipe(isolate_col,col="Nombre_vehicles")
       veh_by_distric
```

```
[405]: Codi_Districte
       01      41530
       02     136061
       03      86682
```

```
04     50340
05    104854
06     59286
07     83433
08     69527
09     68245
10    111321
Name: Nombre_vehicles, dtype: int64
```

[406]:
```
#rename the feature
df["Tipologia_parc"] = df.Tipologia_parc.cat.rename_categories({"Altres␣
 ↪vehicles":"Altres Vehicles"},
                                                  )
```

[407]:
```
#make a list with the name categories
veh_types = df.Tipologia_parc.cat.categories
veh_types
```

[407]:
```
Index(['Altres Vehicles', 'Camions', 'Ciclomotors', 'Furgonetes', 'Motos',
       'Turismes'],
      dtype='object')
```

[408]:
```
# find the number of vehicles by district

Altres_vehicles_by_district = df.pipe(veh_type,"Tipologia_parc","Altres␣
 ↪Vehicles").pipe(grouped_df,"Codi_Districte").
 ↪pipe(isolate_col,col="Nombre_vehicles").
 ↪rename("Num_Altres_vehicles",inplace=True)
Camions_by_district = df.pipe(veh_type,"Tipologia_parc","Camions").
 ↪pipe(grouped_df,"Codi_Districte").pipe(isolate_col,col="Nombre_vehicles").
 ↪rename("Num_Camions",inplace=True)
Ciclomotors_by_district = df.pipe(veh_type,"Tipologia_parc","Ciclomotors").
 ↪pipe(grouped_df,"Codi_Districte").pipe(isolate_col,col="Nombre_vehicles").
 ↪rename("Num_Ciclomotors",inplace=True)
Furgonetes_by_district = df.pipe(veh_type,"Tipologia_parc","Furgonetes").
 ↪pipe(grouped_df,"Codi_Districte").pipe(isolate_col,col="Nombre_vehicles").
 ↪rename("Num_Furgonetes",inplace=True)
Motos_by_district = df.pipe(veh_type,"Tipologia_parc","Motos").
 ↪pipe(grouped_df,"Codi_Districte").pipe(isolate_col,col="Nombre_vehicles").
 ↪rename("Num_Motos",inplace=True)
Turismes_by_district = df.pipe(veh_type,"Tipologia_parc","Turismes").
 ↪pipe(grouped_df,"Codi_Districte").pipe(isolate_col,col="Nombre_vehicles").
 ↪rename("Num_Turismes",inplace=True)
```

[409]:
```
#number of other vehicles by district
Altres_vehicles_by_district
```

```
[409]: Codi_Districte
       01     774
       02    3996
       03    4577
       04     691
       05    1343
       06     763
       07    1130
       08    1006
       09    1142
       10    2135
       Name: Num_Altres_vehicles, dtype: int64
```

```
[410]: #concat the data in the barcelona dataset
       df_bcn_veh = pd.concat([barcelona_shape,
                               veh_by_distric,
                               Altres_vehicles_by_district,
                              Camions_by_district,
                              Ciclomotors_by_district,
                               Furgonetes_by_district,
                               Motos_by_district,
                               Turismes_by_district],axis=1)
       df_bcn_veh
```

```
[410]:     cartodb_id             n_distri   homes   dones          area  \
       01           1         Ciutat Vella   53968   48379  4.368465e+06
       02           2             Eixample  123571  142906  7.476392e+06
       03           3        Sants-MontjuÃ¯c   87877   95243  2.294042e+07
       04           4           Les Corts   38331   43939  6.017532e+06
       05           5  SarriÃ -Sant Gervasi   67799   80113  2.009280e+07
       06           6             GrÃ cia   55611   65891  4.185517e+06
       07           7        Horta-GuinardÃ³   79017   89075  1.194708e+07
       08           8           Nou Barris   78448   87862  8.041439e+06
       09           9          Sant Andreu   70151   77581  6.565322e+06
       10          10           Sant MartÃ  113572  122147  1.052376e+07

                  perim        coord_x       coord_y  \
       01  21035.207261  431616.773990  4.581564e+06
       02  13902.573980  430243.353657  4.582773e+06
       03  47125.925905  428562.773279  4.578163e+06
       04  12481.472647  426369.646389  4.582295e+06
       05  37563.642237  425388.507986  4.585170e+06
       06  12280.060880  429253.013001  4.584840e+06
       07  20413.187364  429117.618770  4.586950e+06
       08  14698.411907  431185.040621  4.588829e+06
       09  15132.450209  432697.846739  4.587572e+06
       10  20736.527911  433330.835564  4.584520e+06
```

```
                                 web_1  \
01        http://www.bcn.cat/ciutatvella
02           http://www.bcn.cat/eixample
03     http://www.bcn.cat/sants-montjuic
04           http://www.bcn.cat/lescorts
05 http://www.bcn.cat/sarria-santgervasi
06             http://www.bcn.cat/gracia
07     http://www.bcn.cat/horta-guinardo
08          http://www.bcn.cat/noubarris
09         http://www.bcn.cat/santandreu
10          http://www.bcn.cat/santmarti

                                              web_2  \
01 http://www.bcn.cat/estadistica/catala/dades/in…
02 http://www.bcn.cat/estadistica/catala/dades/in…
03 http://www.bcn.cat/estadistica/catala/dades/in…
04 http://www.bcn.cat/estadistica/catala/dades/in…
05 http://www.bcn.cat/estadistica/catala/dades/in…
06 http://www.bcn.cat/estadistica/catala/dades/in…
07 http://www.bcn.cat/estadistica/catala/dades/in…
08 http://www.bcn.cat/estadistica/catala/dades/in…
09 http://www.bcn.cat/estadistica/catala/dades/in…
10 http://www.bcn.cat/estadistica/catala/dades/in…

                                              web_3  \
01 http://www.bcn.cat/estadistica/catala/dades/gu…
02 http://www.bcn.cat/estadistica/catala/dades/gu…
03 http://www.bcn.cat/estadistica/catala/dades/gu…
04 http://www.bcn.cat/estadistica/catala/dades/gu…
05 http://www.bcn.cat/estadistica/catala/dades/gu…
06 http://www.bcn.cat/estadistica/catala/dades/gu…
07 http://www.bcn.cat/estadistica/catala/dades/gu…
08 http://www.bcn.cat/estadistica/catala/dades/gu…
09 http://www.bcn.cat/estadistica/catala/dades/gu…
10 http://www.bcn.cat/estadistica/catala/dades/gu…

                                          geometry  Nombre_vehicles  \
01 POLYGON ((2.18239 41.39143, 2.18346 41.39061, …            41530
02 POLYGON ((2.18239 41.39143, 2.18135 41.39222, …           136061
03 POLYGON ((2.16785 41.37498, 2.16856 41.37495, …            86682
04 POLYGON ((2.10291 41.40109, 2.10295 41.40110, …            50340
05 MULTIPOLYGON (((2.07212 41.41270, 2.07050 41.4…           104854
06 POLYGON ((2.16865 41.40696, 2.16979 41.40608, …            59286
07 POLYGON ((2.17616 41.42544, 2.17654 41.42500, …            83433
08 POLYGON ((2.18745 41.46213, 2.18744 41.46212, …            69527
09 POLYGON ((2.20719 41.42756, 2.20558 41.42831, …            68245
10  POLYGON ((2.20719 41.42756, 2.20715 41.42747, …          111321

    Num_Altres_vehicles  Num_Camions  Num_Ciclomotors  Num_Furgonetes  \
01                  774         1343             7424            3201
02                 3996         2524             7987            6280
03                 4577         3157             5024            4759
04                  691          627             2395            1706
05                 1343         1038             4265            3175
06                  763          702             3240            2332
07                 1130         1053             4555            3813
08                 1006         1109             3720            3844
09                 1142         1282             2927            3534
10                 2135         1806             7738            5654

    Num_Motos  Num_Turismes
01      11645         17143
02      39201         76073
03      20593         48572
04      14069         30852
05      36830         58203
06      19040         33209
07      23726         49156
08      14850         44998
09      15293         44067
10      25308         68680
```

[411]:
```
#get the centroids from the poligons of the districts
centroids_bcn_veh_df =  df_bcn_veh["geometry"].centroid
centroids_bcn_veh_df.rename("geometry",inplace=True)
```

```
/var/folders/sq/fggxzcn90p73_7hm62_tglm40000gn/T/ipykernel_1295/725556170.py:2:
UserWarning: Geometry is in a geographic CRS. Results from 'centroid' are likely
incorrect. Use 'GeoSeries.to_crs()' to re-project geometries to a projected CRS
before this operation.

  centroids_bcn_veh_df =  df_bcn_veh["geometry"].centroid
```

[411]:
```
01    POINT (2.18105 41.38084)
02    POINT (2.16449 41.39162)
03    POINT (2.14493 41.34995)
04    POINT (2.11821 41.38697)
05    POINT (2.10612 41.41277)
06    POINT (2.15240 41.41015)
07    POINT (2.15053 41.42914)
08    POINT (2.17506 41.44624)
09    POINT (2.19331 41.43505)
```

```
10    POINT (2.20122 41.40761)
Name: geometry, dtype: geometry
```

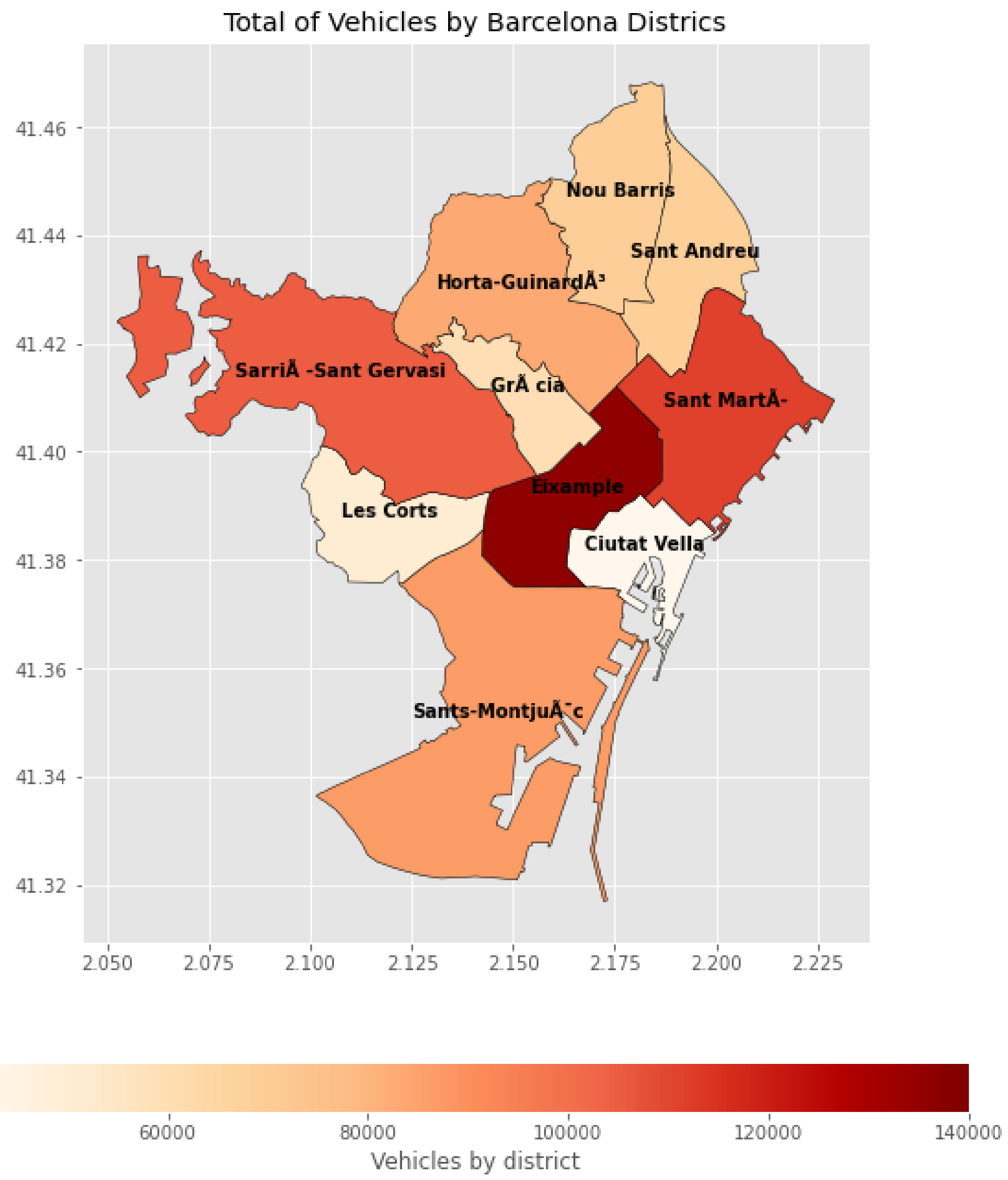
[412]:
```
#concat the data and declare it as a dataset point

df_bcn_veh_p = pd.concat([df_bcn_veh.iloc[:,:8],
                          centroids_bcn_veh_df,
                        veh_by_distric,
                        Altres_vehicles_by_district,
                       Camions_by_district,
                       Ciclomotors_by_district,
                        Furgonetes_by_district,
                        Motos_by_district,
                        Turismes_by_district],axis=1)
df_bcn_veh_p
```

[412]:
```
    cartodb_id            n_distri   homes   dones          area  \
01           1        Ciutat Vella   53968   48379  4.368465e+06
02           2            Eixample  123571  142906  7.476392e+06
03           3       Sants-MontjuÃ¯c   87877   95243  2.294042e+07
04           4           Les Corts   38331   43939  6.017532e+06
05           5  SarriÃ -Sant Gervasi   67799   80113  2.009280e+07
06           6             GrÃ cia   55611   65891  4.185517e+06
07           7       Horta-GuinardÃ³   79017   89075  1.194708e+07
08           8          Nou Barris   78448   87862  8.041439e+06
09           9         Sant Andreu   70151   77581  6.565322e+06
10          10          Sant MartÃ  113572  122147  1.052376e+07

           perim       coord_x       coord_y                  geometry  \
01  21035.207261  431616.773990  4.581564e+06  POINT (2.18105 41.38084)
02  13902.573980  430243.353657  4.582773e+06  POINT (2.16449 41.39162)
03  47125.925905  428562.773279  4.578163e+06  POINT (2.14493 41.34995)
04  12481.472647  426369.646389  4.582295e+06  POINT (2.11821 41.38697)
05  37563.642237  425388.507986  4.585170e+06  POINT (2.10612 41.41277)
06  12280.060880  429253.013001  4.584840e+06  POINT (2.15240 41.41015)
07  20413.187364  429117.618770  4.586950e+06  POINT (2.15053 41.42914)
08  14698.411907  431185.040621  4.588829e+06  POINT (2.17506 41.44624)
09  15132.450209  432697.846739  4.587572e+06  POINT (2.19331 41.43505)
10  20736.527911  433330.835564  4.584520e+06  POINT (2.20122 41.40761)

    Nombre_vehicles  Num_Altres_vehicles  Num_Camions  Num_Ciclomotors  \
01            41530                  774         1343             7424
02           136061                 3996         2524             7987
03            86682                 4577         3157             5024
04            50340                  691          627             2395
05           104854                 1343         1038             4265
06            59286                  763          702             3240
```

```
07             83433                     1130         1053             4555
08             69527                     1006         1109             3720
09             68245                     1142         1282             2927
10            111321                     2135         1806             7738

    Num_Furgonetes  Num_Motos  Num_Turismes
01            3201      11645         17143
02            6280      39201         76073
03            4759      20593         48572
04            1706      14069         30852
05            3175      36830         58203
06            2332      19040         33209
07            3813      23726         49156
08            3844      14850         44998
09            3534      15293         44067
10            5654      25308         68680
```

[413]:
```
#make some correction in the name districts but in some... is not posible
df_bcn_veh.replace({"n_distri":{"Sants-MontjuÃ¯c":"Sants-Montjuic",
                                 "SarriÃ-SantGervasi":"Sarria-Sant Gervasi",
                                 "GrÃ cia":"Gracia",
                              "Horta-GuinardÃ³":"Horta-Guinardo",
                                 "Sant MartÃ":"Sant Marti"}},
                     inplace=True
                    )
```

[414]:
```
fig, ax = plt.subplots(figsize=(12, 12))

df_bcn_veh.plot('Nombre_vehicles', legend=True, edgecolor='k',
                ax=ax ,legend_kwds={'label': "Vehicles by district",
                        'orientation': "horizontal", 'shrink':.8, 'pad':.1},
                vmax=140e3,cmap="OrRd")

ax.set_title("Total of Vehicles by Barcelona Districs")

for x, y, labels1  in zip(df_bcn_veh_p.geometry.x,df_bcn_veh_p.geometry.y ,
                          df_bcn_veh_p['n_distri'] ):
        ax.annotate(labels1, xy=(x , y ), xytext=(3, 3), color="k",
                    fontsize=10,
                    weight="bold",
                    textcoords="offset points",
                    horizontalalignment='center'
                    )
plt.show()
```



The districts with the highest number of vehicles are Eixample, Sant Marti, Sarria-Sant Gervarsi, the least Ciutat Vella, Lest Corts and Gracia.

```
[415]: #list with the number of veh features
       veh_list_cols = df_bcn_veh.columns[13:]
       veh_list_cols
```

```
[415]: Index(['Num_Altres_vehicles', 'Num_Camions', 'Num_Ciclomotors',
              'Num_Furgonetes', 'Num_Motos', 'Num_Turismes'],
      dtype='object')
```

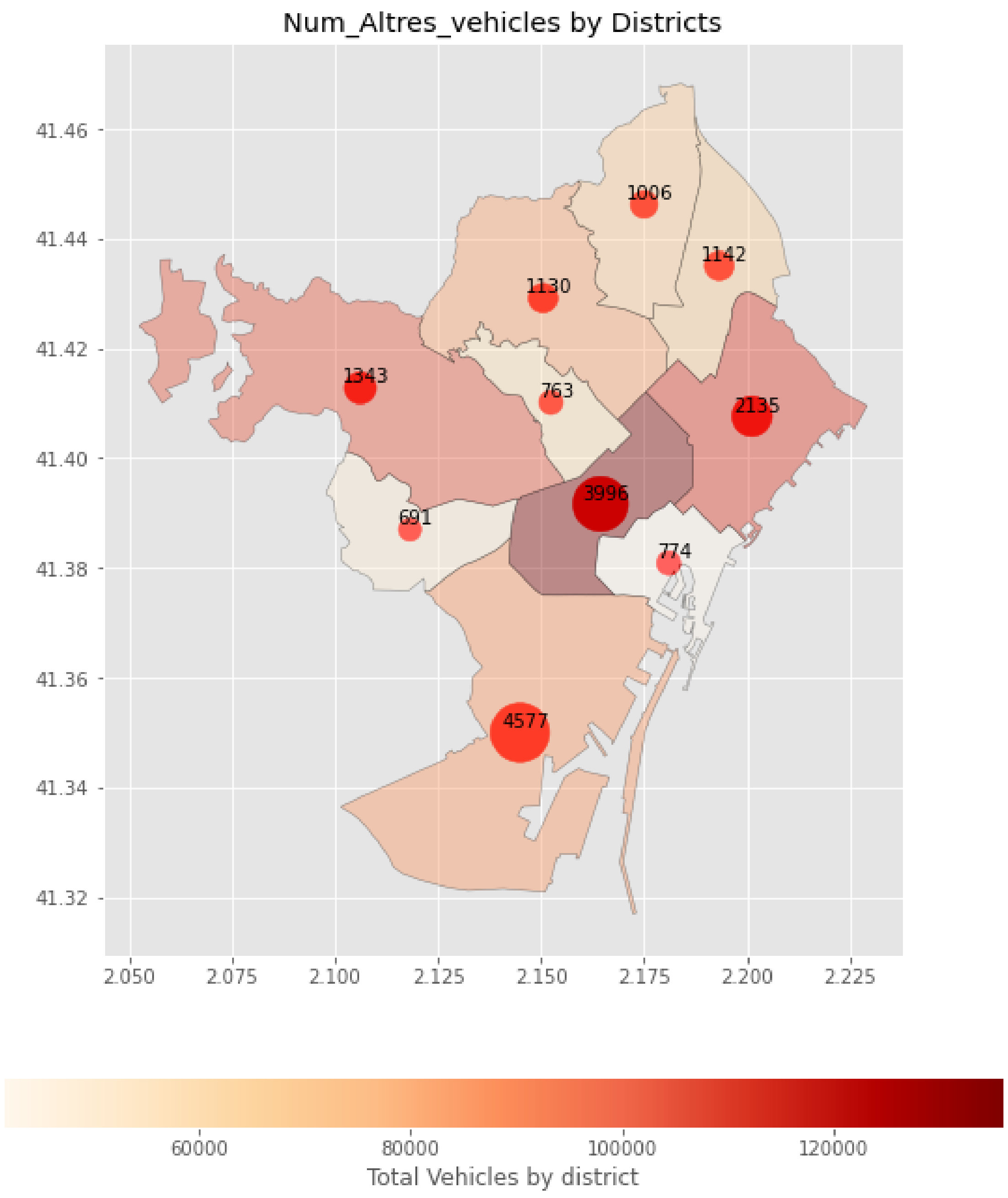
[416]:
```
districts_layer = df_bcn_veh.explore(column= "Nombre_vehicles",
                  tooltip="Nombre_vehicles",
                  popup= True,
                  cmap="OrRd",
                  style_kwds=dict(color="black"),
                                     name = "BCN Districts",
                                      width= 750,height=500,
                                     )



districts_layer
```

[416]:
```
<folium.folium.Map at 0x1491df1f0>
```

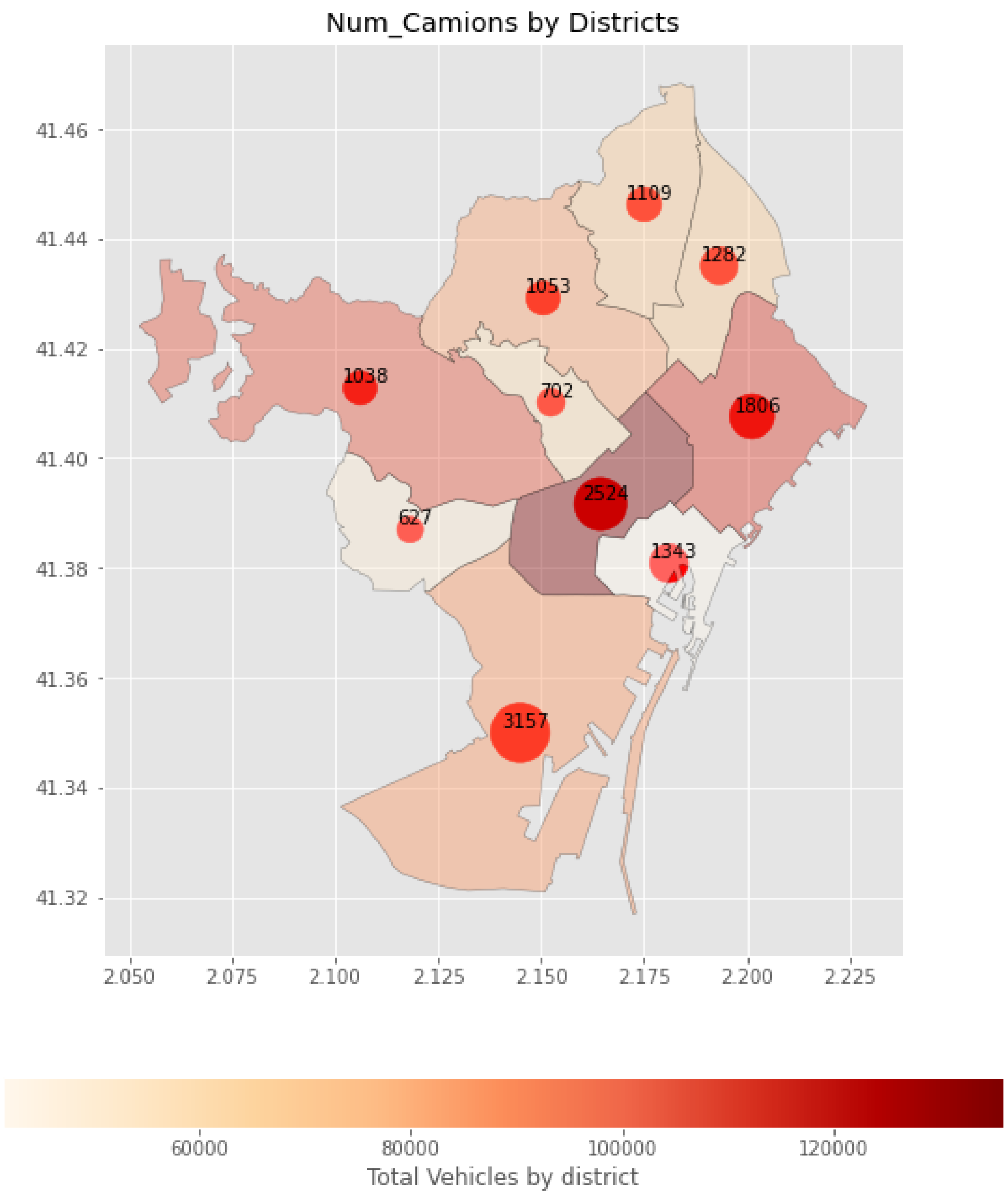
The interactive map show barcelona districts the number of vehicles when you click the maps shows the more detailed info of the district. Sarria - Sant Gervasi have a bigger area than Eixample but the former have less urban area, so have more vehicles.

[417]:
```
#individual map of number of vehicles by district
for i,j in zip(veh_list_cols,veh_types):
    plot_veh(df_bcn_veh_p,df_bcn_veh,i,j,"Nombre_vehicles")
```

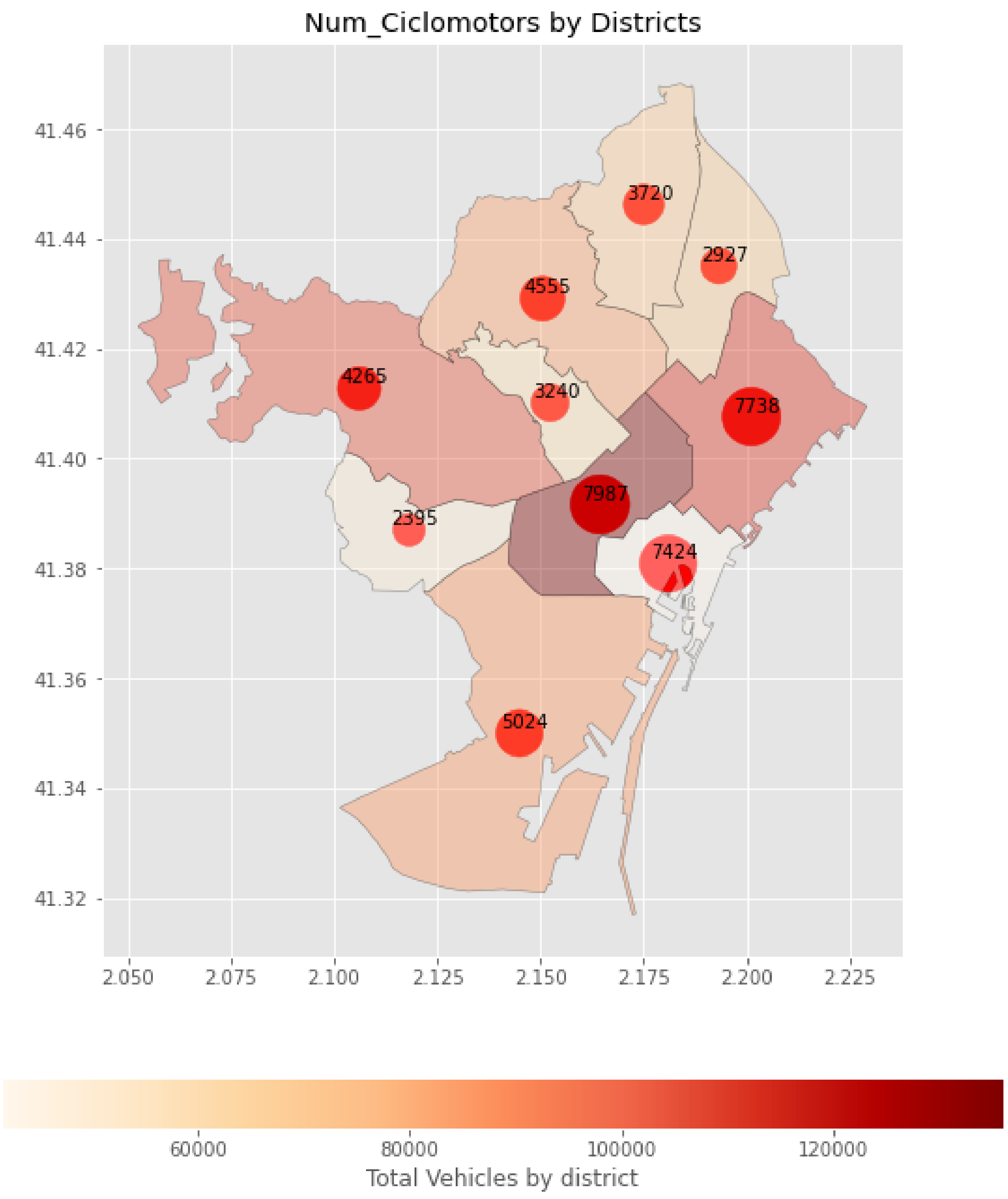
```
/Users/franciscoregalado/opt/anaconda3/envs/geo_env/lib/python3.10/site-
packages/geopandas/plotting.py:644: UserWarning: Only specify one of 'column' or
'color'. Using 'color'.
  warnings.warn(
```



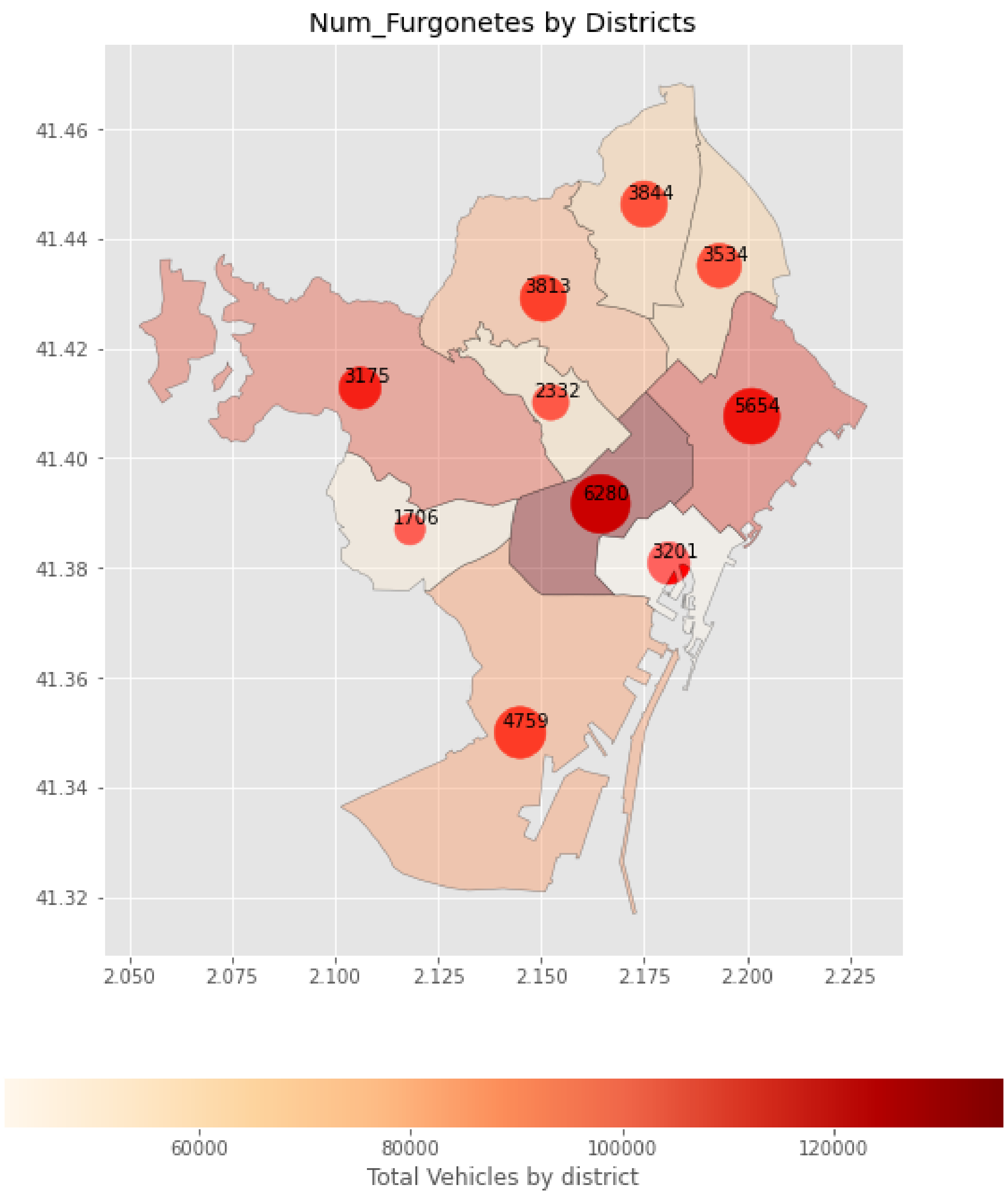
```
/Users/franciscoregalado/opt/anaconda3/envs/geo_env/lib/python3.10/site-
packages/geopandas/plotting.py:644: UserWarning: Only specify one of 'column' or
'color'. Using 'color'.
  warnings.warn(
```



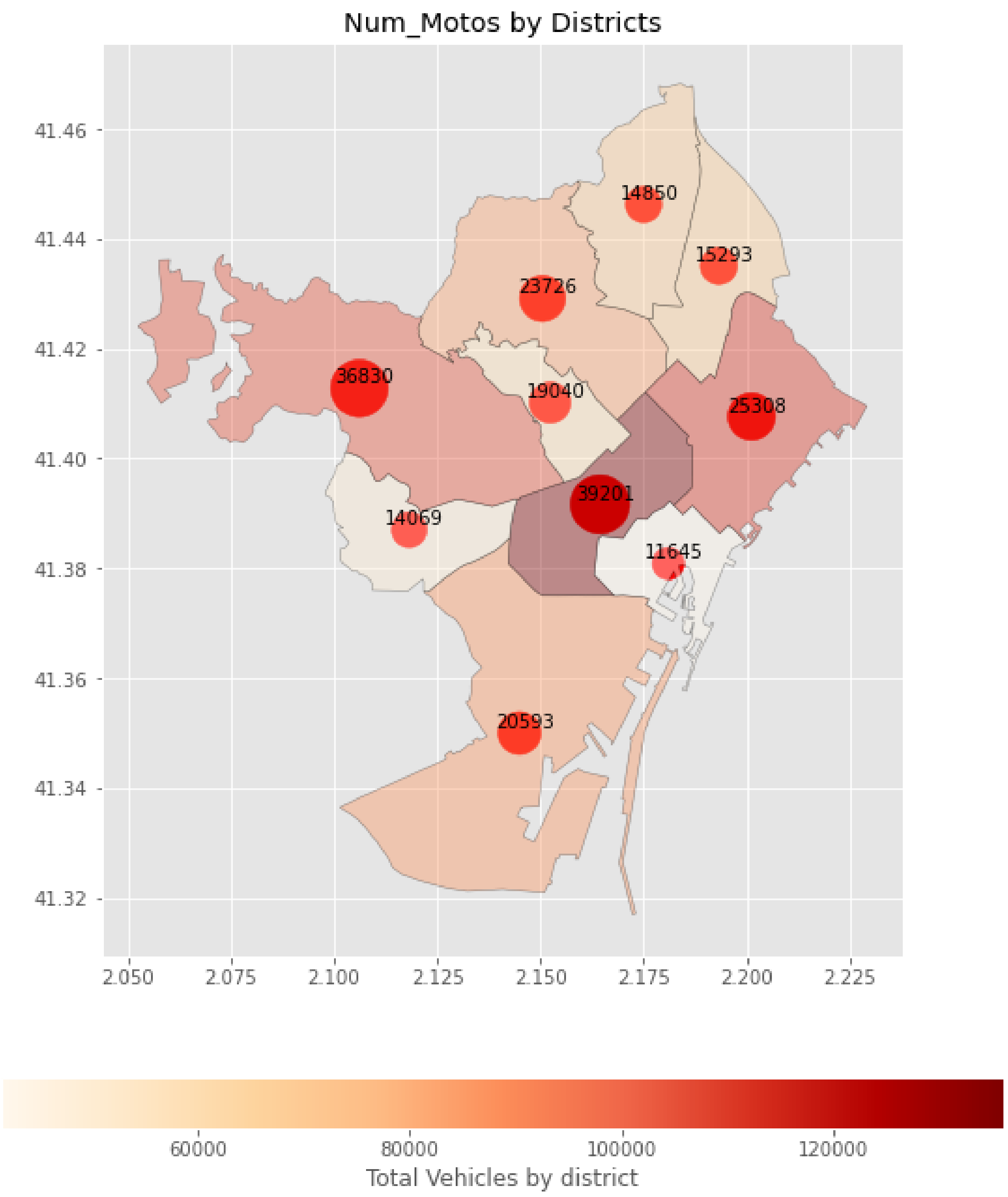
```
/Users/franciscoregalado/opt/anaconda3/envs/geo_env/lib/python3.10/site-
packages/geopandas/plotting.py:644: UserWarning: Only specify one of 'column' or
'color'. Using 'color'.
  warnings.warn(
```



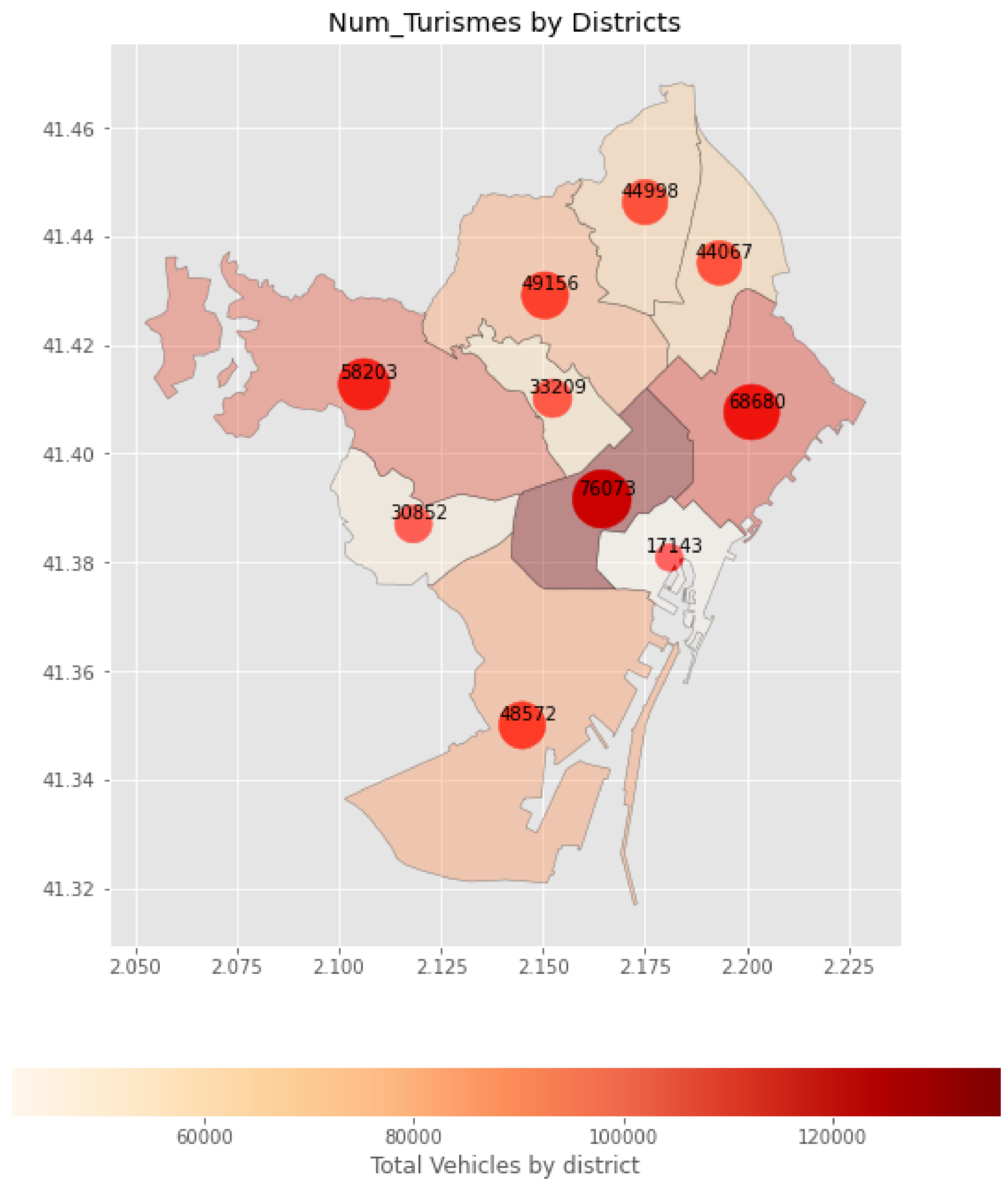
```
/Users/franciscoregalado/opt/anaconda3/envs/geo_env/lib/python3.10/site-
packages/geopandas/plotting.py:644: UserWarning: Only specify one of 'column' or
'color'. Using 'color'.
  warnings.warn(
```



```
/Users/franciscoregalado/opt/anaconda3/envs/geo_env/lib/python3.10/site-
packages/geopandas/plotting.py:644: UserWarning: Only specify one of 'column' or
'color'. Using 'color'.
  warnings.warn(
```



```
/Users/franciscoregalado/opt/anaconda3/envs/geo_env/lib/python3.10/site-
packages/geopandas/plotting.py:644: UserWarning: Only specify one of 'column' or
'color'. Using 'color'.
  warnings.warn(
```



The plot shows the number of vechicles by districts with a point sized by the type number of vehicles, the graphs show a trend toward the use of vehicles in the different districts.

Preprocessing

I will drop some features that will not add information for the model as the coordinates or webpages.

[418]:
```
df2 = df.copy()
df2.head()
```

[418]:
```
    Any Codi_Districte Nom_Districte Codi_Barri Nom_Barri Seccio_censal  \
0  2021             01  Ciutat Vella         01  el Raval             1
1  2021             01  Ciutat Vella         01  el Raval             1
2  2021             01  Ciutat Vella         01  el Raval             1
3  2021             01  Ciutat Vella         01  el Raval             1
4  2021             01  Ciutat Vella         01  el Raval             1

  Tipologia_parc  Nombre_vehicles
0       Turismes              338
1          Motos              128
2    Ciclomotors              682
3     Furgonetes               51
4        Camions               23
```

[419]:
```
df2.info()
```

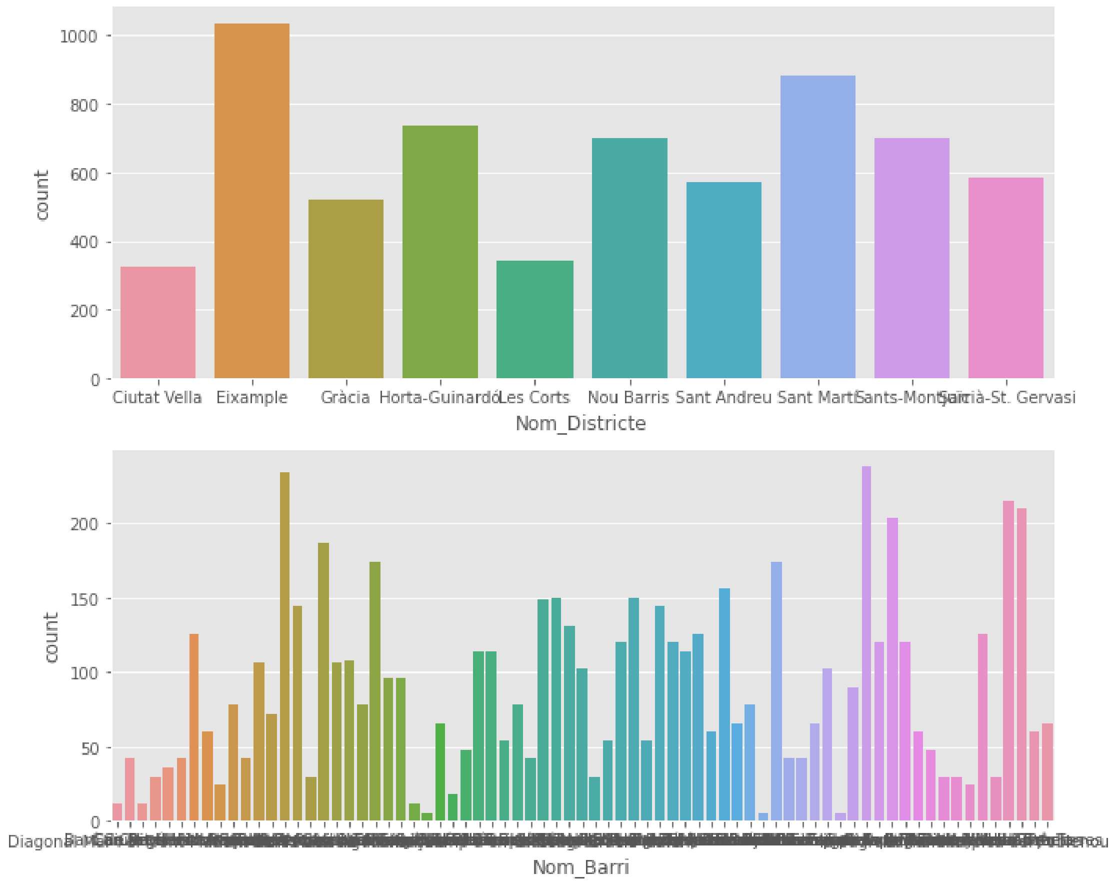
```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 6398 entries, 0 to 6397
Data columns (total 8 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   Any              6398 non-null   int64
 1   Codi_Districte   6398 non-null   object
 2   Nom_Districte    6398 non-null   category
 3   Codi_Barri       6398 non-null   object
 4   Nom_Barri        6398 non-null   category
 5   Seccio_censal    6398 non-null   object
 6   Tipologia_parc   6398 non-null   category
 7   Nombre_vehicles  6398 non-null   int64
dtypes: category(3), int64(2), object(3)
memory usage: 321.9+ KB
```

[420]:
```
#use function (8) to drop columns
df_veh1 = drop_cols(df2,select_dtype(df2,["object"]))
df_veh = drop_cols(df_veh1,"Any")
df_veh
```

[420]:
```
      Nom_Districte            Nom_Barri   Tipologia_parc  Nombre_vehicles
0      Ciutat Vella             el Raval         Turismes              338
1      Ciutat Vella             el Raval            Motos              128
2      Ciutat Vella             el Raval      Ciclomotors              682
3      Ciutat Vella             el Raval       Furgonetes               51
4      Ciutat Vella             el Raval          Camions               23
…               …                    …                …                …
6393    Sant Martí  la Verneda i la Pau            Motos              127
6394    Sant Martí  la Verneda i la Pau      Ciclomotors               55
6395    Sant Martí  la Verneda i la Pau       Furgonetes               31
```

```
6396    Sant Martí  la Verneda i la Pau         Camions                16
6397    Sant Martí  la Verneda i la Pau  Altres Vehicles                7

[6398 rows x 4 columns]
```

[421]:
```
fig , axes = plt.subplots(2,1,figsize=(10,8))
plt.style.use('ggplot')

for i,val in enumerate(select_dtype(df_veh,["category"])[:2]):
    sns.countplot(x=val,data=df_veh,ax=axes[i])
    plt.tight_layout()
plt.show()
```



The plots show us a imbalanced dataset

[422]:
```
df_veh.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 6398 entries, 0 to 6397
Data columns (total 4 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   Nom_Districte    6398 non-null   category
 1   Nom_Barri        6398 non-null   category
 2   Tipologia_parc   6398 non-null   category
 3   Nombre_vehicles  6398 non-null   int64
dtypes: category(3), int64(1)
memory usage: 380.0 KB
```

[423]:
```
#select the categorical features to encode
objects_to_encoding = df_veh.select_dtypes(include = ["category"]).columns
objects_to_encoding
```

[423]:
```
Index(['Nom_Districte', 'Nom_Barri', 'Tipologia_parc'], dtype='object')
```

[424]:
```
#I will use a label encoder to transform the string to a int
LabelEncoder = preprocessing.LabelEncoder()

for i in objects_to_encoding:
    df_veh[i] = LabelEncoder.fit_transform(df_veh[i])
```

[425]:
```
df_veh
```

[425]:
```
      Nom_Districte  Nom_Barri  Tipologia_parc  Nombre_vehicles
0                 0         45               5              338
1                 0         45               4              128
2                 0         45               2              682
3                 0         45               3               51
4                 0         45               1               23
…               …          …               …               …
6393              7         67               4              127
6394              7         67               2               55
6395              7         67               3               31
6396              7         67               1               16
6397              7         67               0                7

[6398 rows x 4 columns]
```

Exercise 1

Gather the set of data you use and perform a pipeline and a gridsearch applying the Random Forest algorithm.

[426]:
```
#The dataset if imbalanced so I will use a smote to balanced it

smote = SMOTE(random_state=7,sampling_strategy='not majority')
under = RandomUnderSampler(sampling_strategy='not majority')
rf = RandomForestClassifier()
```

```
[427]:  # Hyperparameter Tuning
        parameters = {
            'rf__n_estimators': [10,100, 200],
            'rf__max_depth' : [2, 4, 6, 8, 10],
            'rf__min_samples_split': [12,20,40],
            'rf__criterion':['gini', 'entropy'],
            'rf__bootstrap': [True, False]
        }
```

```
[428]: #define my X and y
       X= df_veh.drop(["Tipologia_parc"], axis=1)
       y = df_veh["Tipologia_parc"]
       X.head()
```

```
[428]:    Nom_Districte  Nom_Barri  Nombre_vehicles
       0              0         45              338
       1              0         45              128
       2              0         45              682
       3              0         45               51
       4              0         45               23
```

```
[429]: # Split in train and test
       X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                            test_size= 0.2,
                                                            random_state=7,
                                                           shuffle = True)
```

```
[430]:  # Shape of split
       X_train.shape, y_train.shape, X_test.shape, y_test.shape
```

```
[430]: ((5118, 3), (5118,), (1280, 3), (1280,))
```

```
[431]: #make the pipeline
       pipeline =   Pipeline([
                            ('smote', smote),
                            ('under',under),
                            ('rf', rf)

       ])
```

```
[432]: set_config(display='diagram')

       pipeline

       #the figure show the structure of the pipeline
```

[432]:
```
Pipeline(steps=[('smote',
                 SMOTE(random_state=7, sampling_strategy='not majority')),
                ('under', RandomUnderSampler(sampling_strategy='not majority')),
                ('rf', RandomForestClassifier())])
```

[433]:
```
skfold = StratifiedKFold(n_splits=5,
                         shuffle=True,
                         random_state=11)
```

[451]:
```
grid = GridSearchCV(pipeline, parameters,cv=skfold)
```

[474]:
```
y_score = grid.fit(X_train, y_train)
```

[462]:
```
grid.best_score_
```

[462]:
```
0.6129347401328644
```

The best score of rhe GridSearch tells us that the models with the best parameters have a fit of 61.29%.

[455]:
```
#best parameters
grid.best_params_
```

[455]:
```
{'rf__bootstrap': True,
 'rf__criterion': 'entropy',
 'rf__max_depth': 10,
 'rf__min_samples_split': 12,
 'rf__n_estimators': 200}
```

[454]:
```
y_pred = grid.predict(X_test)
```

[457]:
```
classes_names = data.Tipologia_parc.unique()
classes_lista = pd.Series(classes_names).sort_values(ascending=True).to_list()
classes_lista
```

[457]:
```
['Altres vehicles',
 'Camions',
 'Ciclomotors',
 'Furgonetes',
 'Motos',
 'Turismes']
```

[458]:
```
report_rf = classification_report(y_test , y_pred,output_dict=True␣
 ↪,target_names=classes_lista)
df_report_rf = pd.DataFrame(report_rf).T
df_report_rf
```

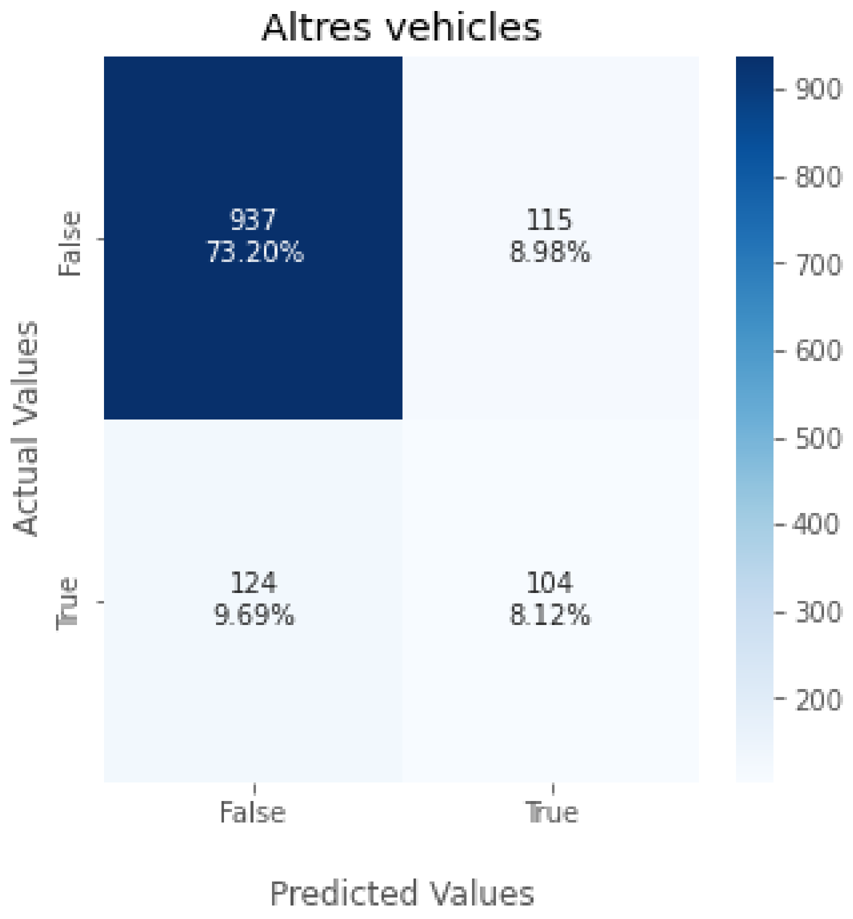
[458]:
```
                 precision    recall  f1-score    support
Altres vehicles   0.474886  0.456140  0.465324   228.0000
Camions           0.480663  0.424390  0.450777   205.0000
Ciclomotors       0.455556  0.600000  0.517895   205.0000
Furgonetes        0.348101  0.279188  0.309859   197.0000
Motos             0.933333  0.907407  0.920188   216.0000
Turismes          0.904959  0.956332  0.929936   229.0000
accuracy          0.612500  0.612500  0.612500     0.6125
macro avg         0.599583  0.603910  0.598997  1280.0000
weighted avg      0.607508  0.612500  0.607367  1280.0000
```

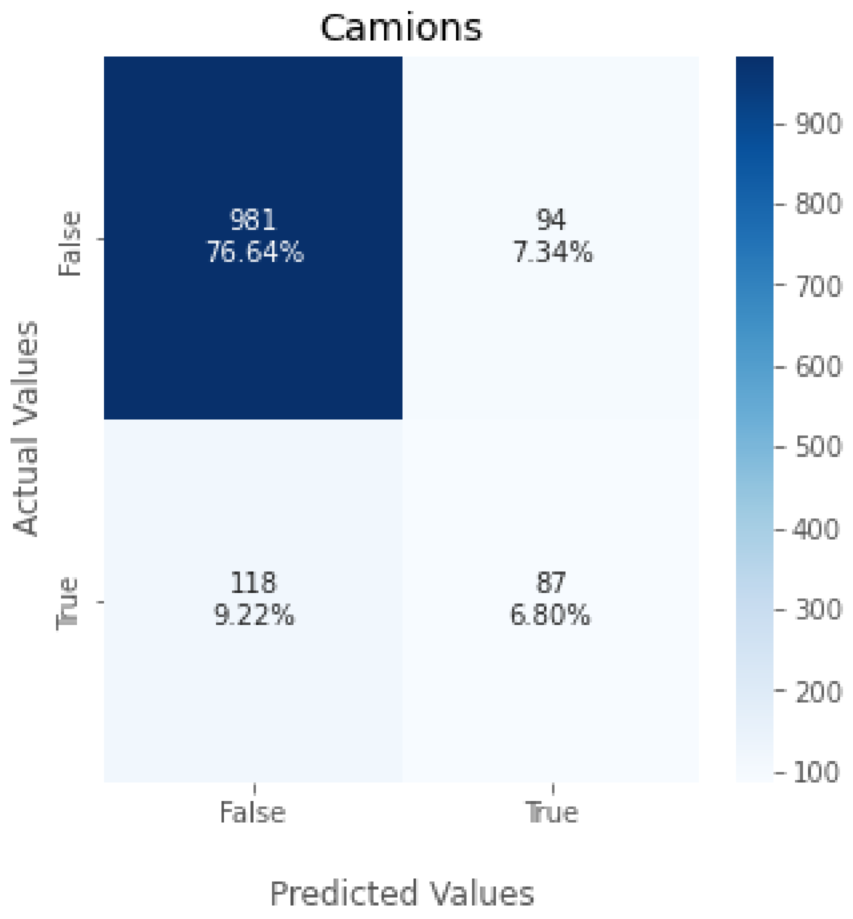
The report of the different vehicle types shows that the model predicts better the Tourism and Bikes and drops sharply for the rest of vehicles.

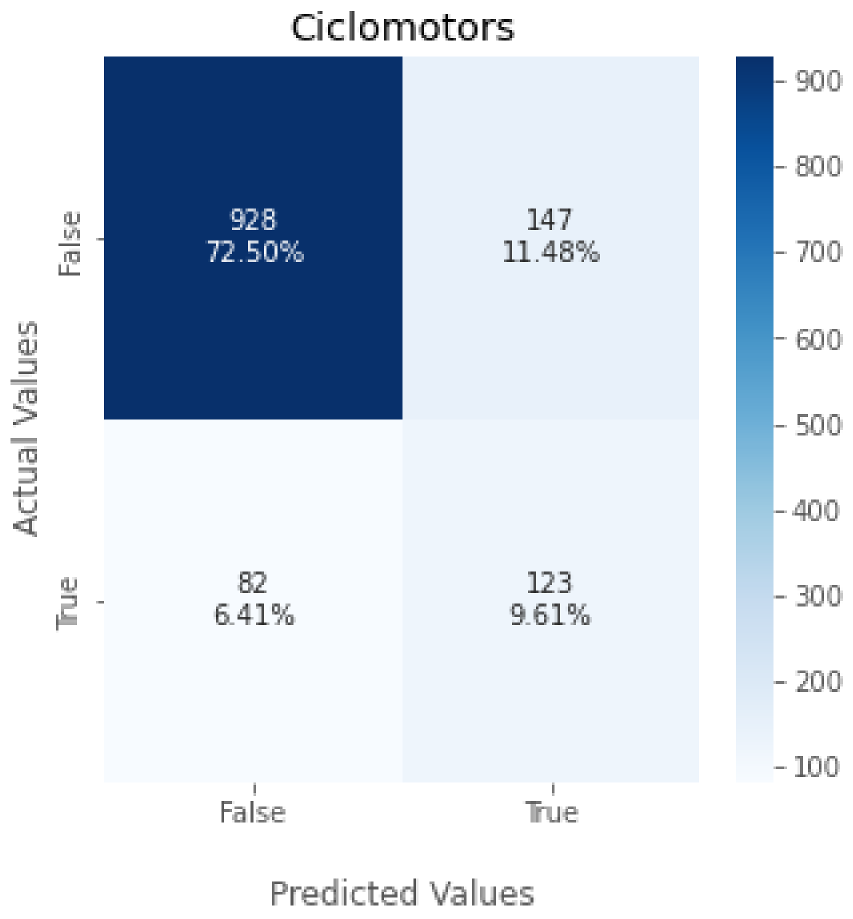
[459]:
```
#confusion matrix for each veh type
matrix_multi = multilabel_confusion_matrix(y_test, y_pred )
matrix_multi
```

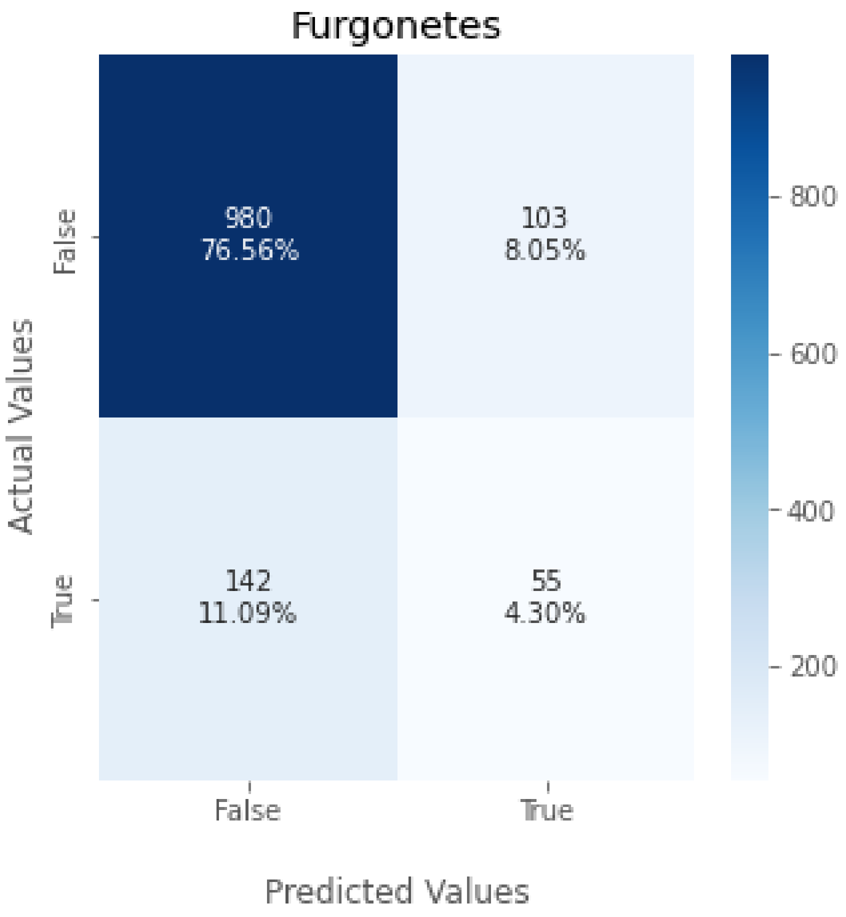
[459]:
```
array([[[ 937,  115],
        [ 124,  104]],

       [[ 981,   94],
        [ 118,   87]],

       [[ 928,  147],
        [  82,  123]],

       [[ 980,  103],
        [ 142,   55]],

       [[1050,   14],
        [  20,  196]],

       [[1028,   23],
        [  10,  219]]])
```

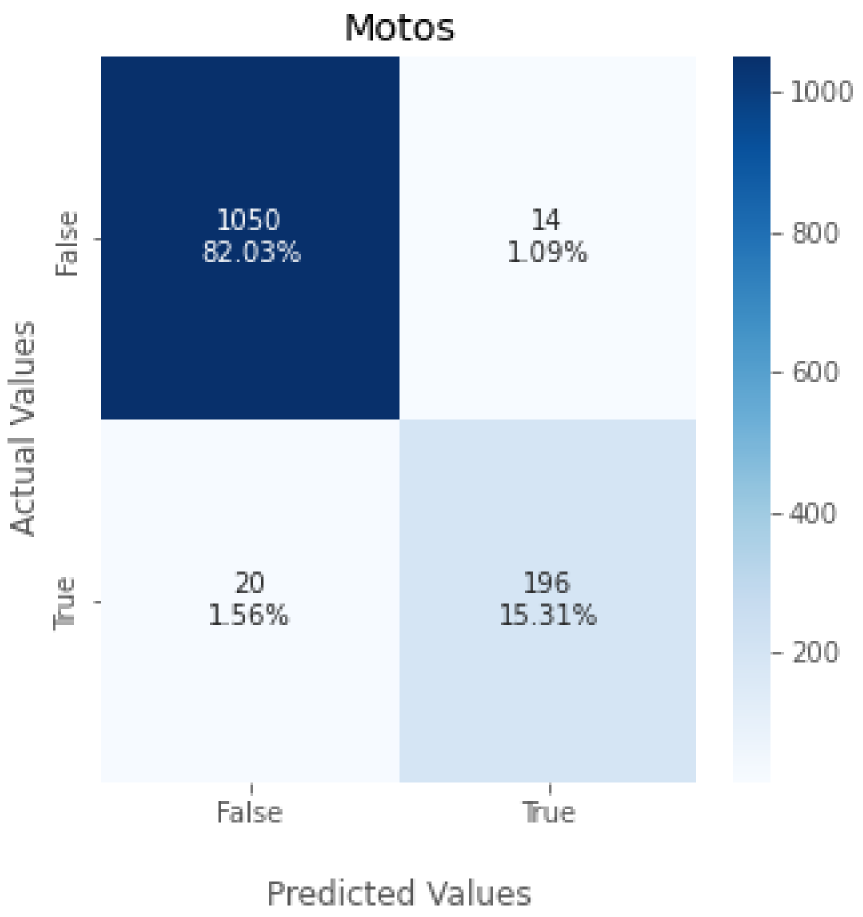
[460]:
```
#use the function (9) to plot each class
for matrix,title in zip(matrix_multi,classes_lista):
    plot_conf_matrix(matrix,title)
```

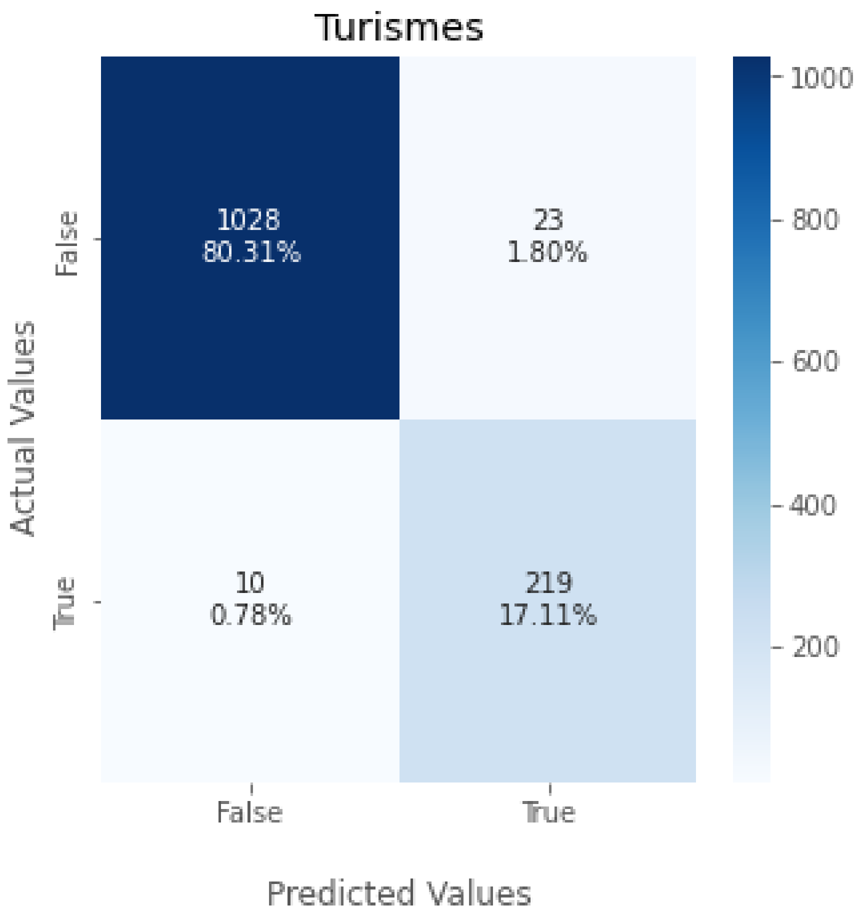

Altres vehicles
937
73.20%
115
8.98%
124
9.69%
104
8.12%
Actual Values
False
True
Predicted Values
False
True
900
800
700
600
500
400
300
200


Camions
981
76.64%
94
7.34%
118
9.22%
87
6.80%
Actual Values
False
True
Predicted Values
False
True
900
800
700
600
500
400
300
200
100


Ciclomotors
928
72.50%
147
11.48%
82
6.41%
123
9.61%
False
True
Actual Values
False
True
Predicted Values
900
800
700
600
500
400
300
200
100

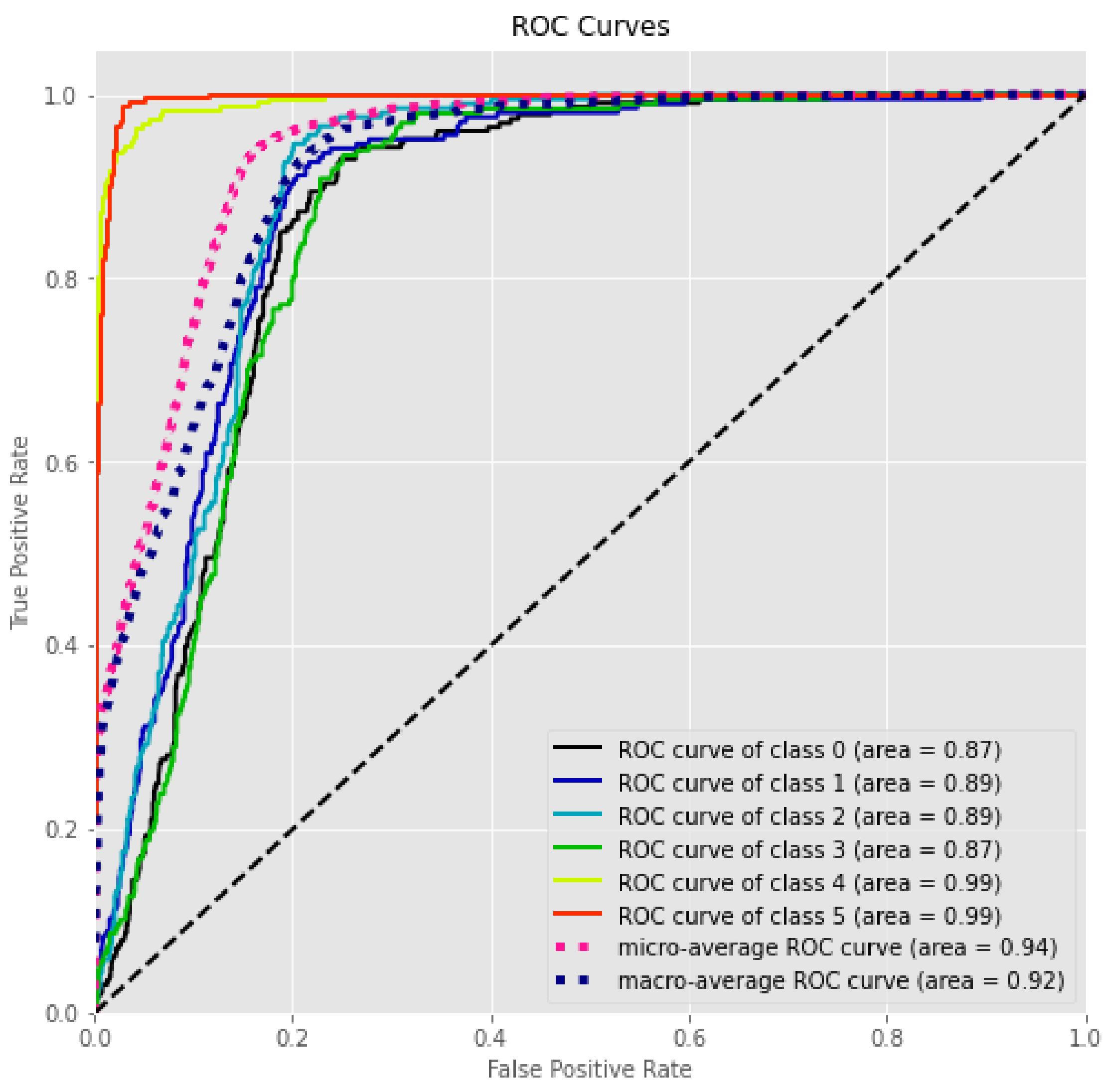

Furgonetes
980
76.56%
103
8.05%
142
11.09%
55
4.30%
False
True
Actual Values
False
True
Predicted Values
800
600
400
200


Motos
Actual Values
False
True
1050
82.03%
14
1.09%
20
1.56%
196
15.31%
Predicted Values
1000
800
600
400
200



The confusion matrix plots show what the classification report shows the low number of false positive of tourism and bikes.

```
[493]: grid.predict_proba(X_test)
       skplt.metrics.plot_roc(y_test, y_probas,title='ROC Curves',figsize=(8,8),)
       plt.show()
```



The ROC curve shows that the Random Forest Classifier doesn't model well the data, it can model very well only 2 classes, in general the model can be used but its important to know which classes are have a better fit.

Exercise 2

Take a text in English that you want, and calculate the frequency of the words

[73]:

```
raw_text = "Fragments Within the Works of Tolkien When writing The Lord of the␣
  ↪Rings, Tolkien strove to build a mythos of England. To do so, Tolkien drew␣
  ↪from the fragments in the world around him. Over the course of the past two␣
  ↪weeks, we have used our understanding of Tolkien's fascination with the␣
  ↪fragments in our world to better understand where Tolkien's stories come␣
  ↪from; however, The Notion Club Papers have demonstrated to me that we have␣
  ↪neglected to consider how the idea of "fragments" has affected the telling␣
  ↪of the stories themselves. The Notion Club Papers provide a glimpse of␣
  ↪Tolkien's creating process in the form of Ramer's ramblings: The act of␣
  ↪Creation is one of finding fragments and giving them context. Looking back␣
  ↪now at Tolkien's stories -The Lord of the Rings in particular - I find that␣
  ↪fragments lie at their very hearts. Through this examination, I hope to␣
  ↪achieve better knowledge of Tolkien's use of fragments so as to better␣
  ↪understand the stories themselves. First, I will prove a brief account of␣
  ↪fragments as we understand them. The fragments that Tolkien was most often␣
  ↪interested in were those of language - for example, the remaining snippets␣
  ↪of a long lost poem or an oddity in a word's modern meaning. Being the␣
  ↪philologist that he was, Tolkien would take these fragments and try to␣
  ↪provide a history, a context, for them. Effectively, Tolkien was using these␣
  ↪fragments to work backwards through time by developing for the fragment an␣
  ↪etymology of sorts. The class identified a clear example of this in Book I␣
  ↪of The Lord of the Rings where Frodo sings a full version of "Hey diddle␣
  ↪diddle" and establishes it as a song that was old even when Frodo sang it in␣
  ↪Bree. By creating a history for the poem, Tolkien creates a tangible␣
  ↪connection between the modern day and his Creation. Through examination of␣
  ↪Ramer's ramblings in Tolkien's Notion Club Papers we can gain insight as to␣
  ↪how Tolkien understood and used fragments. In The Notion Club Papers, Ramer␣
```

```python
print(raw_text)
```

```
Fragments Within the Works of Tolkien When writing The Lord of the Rings,
Tolkien strove to build a mythos of England. To do so, Tolkien drew from the
fragments in the world around him. Over the course of the past two weeks, we
have used our understanding of Tolkien's fascination with the fragments in our
world to better understand where Tolkien's stories come from; however, The
Notion Club Papers have demonstrated to me that we have neglected to consider
how the idea of "fragments" has affected the telling of the stories themselves.
The Notion Club Papers provide a glimpse of Tolkien's creating process in the
form of Ramer's ramblings: The act of Creation is one of finding fragments and
giving them context. Looking back now at Tolkien's stories -The Lord of the
Rings in particular - I find that fragments lie at their very hearts. Through
this examination, I hope to achieve better knowledge of Tolkien's use of
fragments so as to better understand the stories themselves. First, I will prove
a brief account of fragments as we understand them. The fragments that Tolkien
was most often interested in were those of language - for example, the remaining
snippets of a long lost poem or an oddity in a word's modern meaning. Being the
philologist that he was, Tolkien would take these fragments and try to provide a
history, a context, for them. Effectively, Tolkien was using these fragments to
work backwards through time by developing for the fragment an etymology of
sorts. The class identified a clear example of this in Book I of The Lord of the
Rings where Frodo sings a full version of "Hey diddle diddle" and establishes it
as a song that was old even when Frodo sang it in Bree. By creating a history
for the poem, Tolkien creates a tangible connection between the modern day and
his Creation. Through examination of Ramer's ramblings in Tolkien's Notion Club
Papers we can gain insight as to how Tolkien understood and used fragments. In
The Notion Club Papers, Ramer expresses the idea that the identity of an object
or place is the union of its physical self with its history; fragments preserve
this identity even after the subject has lost its physical and historical
presence by continuing to tell the story. However, these fragments are
meaningless without context, much in the way that a meteorite - a fragment of
what was once a large object with a long history - is just another stone if we
do not already know its history. Language fragments are unique in this aspect in
that the fragment's history can often be found within the fragment itself. What
enables Tolkien to find the story within these fragments is the heredity of
language: Human beings have a "native language" which is stored within our
"incarnate beings." By linking modern fragments to the languages that he had
been developing since childhood, Tolkien was able to provide a context for the
fragments - he was able to move back in time. Interestingly, we can find the
motif of fragments within Tolkien's stories themselves and The Lord of the Rings
in particular. The One Ring was a powerful artifact of the Second Age, an object
of great power - but when Sauron was cast down and the Ring lost, its context
was lost with it. When Bilbo found it an Age later, it was nothing more than a
ring with interesting capabilities. The Ring was a mere fragment of its dark
master, a fragment of an Age, a fragment without context. Nor does it seem that
Bilbo was particularly concerned with the history of the Ring, for he had made
```

```
no progress as to its identity by the time he had passed it on to Frodo.
However, it is only when the Ring is passed on to Frodo that the Ring's true
nature is revealed and Frodo can begin to move towards truly understanding the
Ring and its history. Frodo completes the history of the Ring by means of a
heredity "native language" of adventure shared between him and Bilbo. And when
the Ring is finally destroyed, it takes with it the remnants of the Second Age.
In this way I believe that Frodo has a lot in common with Ramer and Tolkien. At
the conclusion of The Lord of the Rings, Merry and Pippin and Sam have
successfully reintegrated into hobbit society, but not so Frodo. As bearer of
the One Ring, Frodo had pursued its story and completed the tale of the Lord of
the Rings, but he had also been exposed to its power. I suspect this exposure to
be similar in effect to Ramer's lucid dreams: His dream-wanderings and the
pursuit of fragments through time and space have broadened his perspective in a
way that other members of the Notion Club seem unable to really identify with,
even if they have some experience in setting their own dreams free. Like Ramer,
Frodo has experienced the vastness of the world in the sense of time and space,
and his experience has marked him. As a result, Frodo is unable in reintegrate
into his old life and is ultimately forced to depart. In the above I have
attempted to explore the importance of fragments not only as tools of creation
but as motifs within the Lord of the Rings. A more complete analysis should give
more attention to those stories in which the fragment in question was not given
a context outside of the plot, as in The Hobbit or Smith of Wootton Major. I
believe it would be interesting to compare elements of both types of stories -
those in which context is given to the fragment and those where it is not - to
determine how the narrative is affected. Author: N. Malaqai Vasquez
```

[74]:
```python
#remove punctuation from text
translator = str.maketrans('', '', string.punctuation)
clean_text = raw_text.translate(translator)
print(clean_text)
```

```
Fragments Within the Works of Tolkien When writing The Lord of the Rings Tolkien
strove to build a mythos of England To do so Tolkien drew from the fragments in
the world around him Over the course of the past two weeks we have used our
understanding of Tolkien's fascination with the fragments in our world to better
understand where Tolkien's stories come from however The Notion Club Papers have
demonstrated to me that we have neglected to consider how the idea of
"fragments" has affected the telling of the stories themselves The Notion Club
Papers provide a glimpse of Tolkien's creating process in the form of Ramer's
ramblings The act of Creation is one of finding fragments and giving them
context Looking back now at Tolkien's stories -The Lord of the Rings in
particular - I find that fragments lie at their very hearts Through this
examination I hope to achieve better knowledge of Tolkien's use of fragments so
as to better understand the stories themselves First I will prove a brief
account of fragments as we understand them The fragments that Tolkien was most
often interested in were those of language - for example the remaining snippets
of a long lost poem or an oddity in a word's modern meaning Being the
philologist that he was Tolkien would take these fragments and try to provide a
```

history a context for them Effectively Tolkien was using these fragments to work backwards through time by developing for the fragment an etymology of sorts The class identified a clear example of this in Book I of The Lord of the Rings where Frodo sings a full version of "Hey diddle diddle" and establishes it as a song that was old even when Frodo sang it in Bree By creating a history for the poem Tolkien creates a tangible connection between the modern day and his Creation Through examination of Ramer's ramblings in Tolkien's Notion Club Papers we can gain insight as to how Tolkien understood and used fragments In The Notion Club Papers Ramer expresses the idea that the identity of an object or place is the union of its physical self with its history fragments preserve this identity even after the subject has lost its physical and historical presence by continuing to tell the story However these fragments are meaningless without context much in the way that a meteorite - a fragment of what was once a large object with a long history - is just another stone if we do not already know its history Language fragments are unique in this aspect in that the fragment's history can often be found within the fragment itself What enables Tolkien to find the story within these fragments is the heredity of language Human beings have a "native language" which is stored within our "incarnate beings" By linking modern fragments to the languages that he had been developing since childhood Tolkien was able to provide a context for the fragments - he was able to move back in time Interestingly we can find the motif of fragments within Tolkien's stories themselves and The Lord of the Rings in particular The One Ring was a powerful artifact of the Second Age an object of great power - but when Sauron was cast down and the Ring lost its context was lost with it When Bilbo found it an Age later it was nothing more than a ring with interesting capabilities The Ring was a mere fragment of its dark master a fragment of an Age a fragment without context Nor does it seem that Bilbo was particularly concerned with the history of the Ring for he had made no progress as to its identity by the time he had passed it on to Frodo However it is only when the Ring is passed on to Frodo that the Ring's true nature is revealed and Frodo can begin to move towards truly understanding the Ring and its history Frodo completes the history of the Ring by means of a heredity "native language" of adventure shared between him and Bilbo And when the Ring is finally destroyed it takes with it the remnants of the Second Age In this way I believe that Frodo has a lot in common with Ramer and Tolkien At the conclusion of The Lord of the Rings Merry and Pippin and Sam have successfully reintegrated into hobbit society but not so Frodo As bearer of the One Ring Frodo had pursued its story and completed the tale of the Lord of the Rings but he had also been exposed to its power I suspect this exposure to be similar in effect to Ramer's lucid dreams His dreamwanderings and the pursuit of fragments through time and space have broadened his perspective in a way that other members of the Notion Club seem unable to really identify with even if they have some experience in setting their own dreams free Like Ramer Frodo has experienced the vastness of the world in the sense of time and space and his experience has marked him As a result Frodo is unable in reintegrate into his old life and is ultimately forced to depart In the above I have attempted to explore the importance of fragments not only as tools of creation but as motifs within the Lord of the Rings A more complete analysis should give more attention to those stories in which the

```
fragment in question was not given a context outside of the plot as in The
Hobbit or Smith of Wootton Major I believe it would be interesting to compare
elements of both types of stories - those in which context is given to the
fragment and those where it is not - to determine how the narrative is affected
Author N Malaqai Vasquez
```

[75]:
```python
  # break the text into words
token_word = word_tokenize(clean_text)
print(token_word[23:43])
```

```
['so', 'Tolkien', 'drew', 'from', 'the', 'fragments', 'in', 'the', 'world',
'around', 'him', 'Over', 'the', 'course', 'of', 'the', 'past', 'two', 'weeks',
'we']
```

[76]:
```python
word_freq = FreqDist(token_word)
word_freq
```

[76]:
```
FreqDist({'the': 65, 'of': 54, 'to': 28, 'a': 28, 'in': 24, 'and': 21,
'fragments': 20, 'Tolkien': 18, 'is': 14, 'was': 14, …})
```

[77]:
```python
token_text_item = word_freq.items()
print(token_text_item)
```

```
dict_items([('Fragments', 1), ('Within', 1), ('the', 65), ('Works', 1), ('of',
54), ('Tolkien', 18), ('When', 2), ('writing', 1), ('The', 13), ('Lord', 7),
('Rings', 6), ('strove', 1), ('to', 28), ('build', 1), ('a', 28), ('mythos', 1),
('England', 1), ('To', 1), ('do', 2), ('so', 3), ('drew', 1), ('from', 2),
('fragments', 20), ('in', 24), ('world', 3), ('around', 1), ('him', 3), ('Over',
1), ('course', 1), ('past', 1), ('two', 1), ('weeks', 1), ('we', 6), ('have',
8), ('used', 2), ('our', 3), ('understanding', 2), ('’', 13), ('s', 13),
('fascination', 1), ('with', 9), ('better', 3), ('understand', 3), ('where', 3),
('stories', 7), ('come', 1), ('however', 1), ('Notion', 5), ('Club', 5),
('Papers', 4), ('demonstrated', 1), ('me', 1), ('that', 13), ('neglected', 1),
('consider', 1), ('how', 3), ('idea', 2), ('“', 5), ('”', 5), ('has', 5),
('affected', 2), ('telling', 1), ('themselves', 3), ('provide', 3), ('glimpse',
1), ('creating', 2), ('process', 1), ('form', 1), ('Ramer', 6), ('ramblings',
2), ('act', 1), ('Creation', 2), ('is', 14), ('one', 1), ('finding', 1), ('and',
21), ('giving', 1), ('them', 3), ('context', 8), ('Looking', 1), ('back', 2),
('now', 1), ('at', 2), ('-The', 1), ('Rings\xad', 1), ('particular', 2), ('-',
8), ('I', 8), ('find', 3), ('lie', 1), ('their', 2), ('very', 1), ('hearts', 1),
('Through', 2), ('this', 6), ('examination', 2), ('hope', 1), ('achieve', 1),
('knowledge', 1), ('use', 1), ('as', 8), ('First', 1), ('will', 1), ('prove',
1), ('brief', 1), ('account', 1), ('was', 14), ('most', 1), ('often', 2),
('interested', 1), ('were', 1), ('those', 4), ('language', 4), ('for', 6),
('example', 2), ('remaining', 1), ('snippets', 1), ('long', 2), ('lost', 4),
('poem', 2), ('or', 3), ('an', 6), ('oddity', 1), ('word', 1), ('modern', 3),
('meaning', 1), ('Being', 1), ('philologist', 1), ('he', 6), ('would', 2),
('take', 1), ('these', 4), ('try', 1), ('history', 9), ('Effectively', 1),
('using', 1), ('work', 1), ('backwards', 1), ('through', 2), ('time', 5), ('by',
4), ('developing', 2), ('fragment', 9), ('etymology', 1), ('sorts', 1),
('class', 1), ('identified', 1), ('clear', 1), ('Book', 1), ('Frodo', 11),
('sings', 1), ('full', 1), ('version', 1), ('Hey', 1), ('diddle', 2),
('establishes', 1), ('it', 12), ('song', 1), ('old', 2), ('even', 3), ('when',
4), ('sang', 1), ('Bree', 1), ('By', 2), ('creates', 1), ('tangible', 1),
('connection', 1), ('between', 2), ('day', 1), ('his', 4), ('can', 4), ('gain',
1), ('insight', 1), ('understood', 1), ('In', 3), ('expresses', 1), ('identity',
3), ('object', 3), ('place', 1), ('union', 1), ('its', 10), ('physical', 2),
('self', 1), ('preserve', 1), ('after', 1), ('subject', 1), ('historical', 1),
('presence', 1), ('continuing', 1), ('tell', 1), ('story', 3), ('However', 2),
('are', 2), ('meaningless', 1), ('without', 2), ('much', 1), ('way', 3),
('meteorite', 1), ('what', 1), ('once', 1), ('large', 1), ('just', 1),
('another', 1), ('stone', 1), ('if', 2), ('not', 5), ('already', 1), ('know',
1), ('Language', 1), ('unique', 1), ('aspect', 1), ('be', 3), ('found', 2),
('within', 5), ('itself', 1), ('What', 1), ('enables', 1), ('heredity', 2),
('Human', 1), ('beings', 2), ('native', 2), ('which', 3), ('stored', 1),
('incarnate', 1), ('linking', 1), ('languages', 1), ('had', 5), ('been', 2),
('since', 1), ('childhood', 1), ('able', 2), ('move', 2), ('Interestingly', 1),
('motif', 1), ('One', 2), ('Ring', 10), ('powerful', 1), ('artifact', 1),
('Second', 2), ('Age', 4), ('great', 1), ('power', 2), ('but', 4), ('Sauron',
1), ('cast', 1), ('down', 1), ('Bilbo', 3), ('later', 1), ('nothing', 1),
('more', 3), ('than', 1), ('ring', 1), ('interesting', 2), ('capabilities', 1),
('mere', 1), ('dark', 1), ('master', 1), ('Nor', 1), ('does', 1), ('seem', 2),
('particularly', 1), ('concerned', 1), ('made', 1), ('no', 1), ('progress', 1),
('passed', 2), ('on', 2), ('only', 2), ('true', 1), ('nature', 1), ('revealed',
1), ('begin', 1), ('towards', 1), ('truly', 1), ('completes', 1), ('means', 1),
('adventure', 1), ('shared', 1), ('And', 1), ('finally', 1), ('destroyed', 1),
('takes', 1), ('remnants', 1), ('believe', 2), ('lot', 1), ('common', 1), ('At',
1), ('conclusion', 1), ('Merry', 1), ('Pippin', 1), ('Sam', 1), ('successfully',
1), ('reintegrated', 1), ('into', 2), ('hobbit', 1), ('society', 1), ('As', 2),
('bearer', 1), ('pursued', 1), ('completed', 1), ('tale', 1), ('also', 1),
('exposed', 1), ('suspect', 1), ('exposure', 1), ('similar', 1), ('effect', 1),
('lucid', 1), ('dreams', 2), ('His', 1), ('dreamwanderings', 1), ('pursuit', 1),
('space', 2), ('broadened', 1), ('perspective', 1), ('other', 1), ('members',
1), ('unable', 2), ('really', 1), ('identify', 1), ('they', 1), ('some', 1),
('experience', 2), ('setting', 1), ('own', 1), ('free', 1), ('Like', 1),
('experienced', 1), ('vastness', 1), ('sense', 1), ('marked', 1), ('result', 1),
('reintegrate', 1), ('life', 1), ('ultimately', 1), ('forced', 1), ('depart',
1), ('above', 1), ('attempted', 1), ('explore', 1), ('importance', 1), ('tools',
1), ('creation', 1), ('motifs', 1), ('A', 1), ('complete', 1), ('analysis', 1),
('should', 1), ('give', 1), ('attention', 1), ('question', 1), ('given', 2),
('outside', 1), ('plot', 1), ('Hobbit', 1), ('Smith', 1), ('Wootton', 1),
('Major', 1), ('compare', 1), ('elements', 1), ('both', 1), ('types', 1),
('determine', 1), ('narrative', 1), ('Author', 1), ('N', 1), ('Malaqai', 1),
('Vasquez', 1)])
```

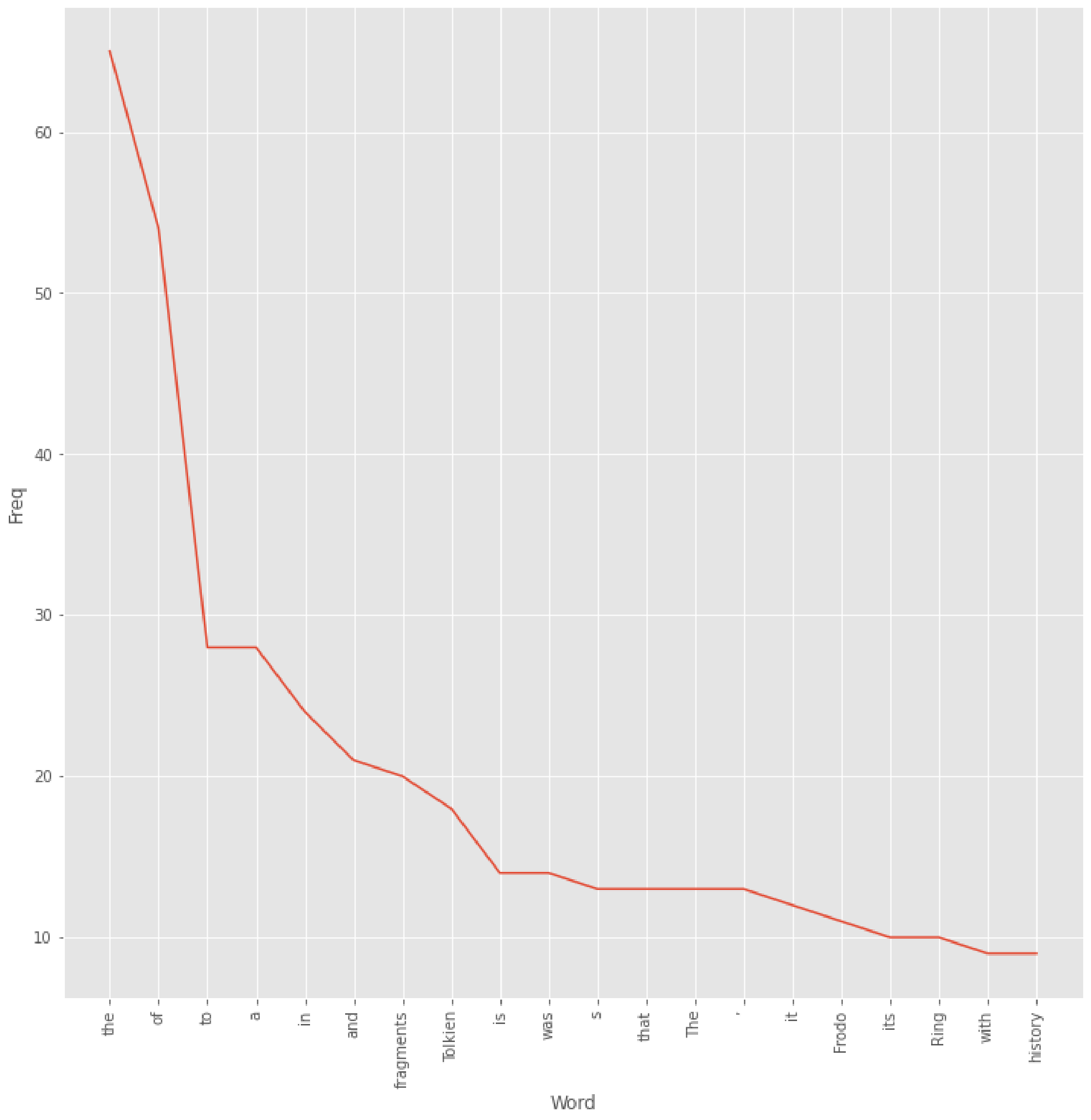
```
[78]: word_freq_df = pd.DataFrame(token_text_item,columns=["Word","Freq"])
      word_freq_df
```

```
[78]:          Word  Freq
      0    Fragments     1
      1       Within     1
      2          the    65
      3        Works     1
      4           of    54
      ..         …   …
      363  narrative     1
      364     Author     1
      365          N     1
      366    Malaqai     1
      367    Vasquez     1

      [368 rows x 2 columns]
```

```
[79]: token_freq_sorted_df = word_freq_df.nlargest(20,"Freq").
       ↪sort_values(by="Freq",ascending=False)
      token_freq_sorted_df
```

```
[79]:          Word  Freq
      2          the    65
      4           of    54
      12          to    28
      14           a    28
      23          in    24
      75         and    21
      22   fragments    20
      5      Tolkien    18
      72          is    14
      106        was    14
      38           s    13
      52        that    13
      8          The    13
      37           ’    13
      156         it    12
      149      Frodo    11
      180        its    10
      235       Ring    10
      40        with     9
      133    history     9
```

```
[80]: fig, ax = plt.subplots(figsize=(12, 12))
      sns.lineplot(x="Word",y="Freq",data=token_freq_sorted_df)
      plt.xticks(rotation=90)
plt.show()
```



Level 2

Exercici 1

Remove stopwords and stemming your dataset.

[81]:
```
#words that need be removed
stop_words=set(stopwords.words("english"))
print(stop_words)
```

```
{'itself', 'his', 'am', 'nor', 'its', 'our', 'is', 'was', 'o', 'it', 'about',
'and', 'for', 'myself', "she's", 'off', 'so', 'wasn', 'will', 'mustn', 'above',
'after', 'this', 'such', 'with', 'm', 'ain', 'before', 'or', 'are', "didn't",
'all', "aren't", "mustn't", 'there', 'he', 'why', 'to', 'haven', 'doesn', 'we',
'very', 'shouldn', 'few', 'just', 'can', 'but', 'had', 'over', 'him', 'which',
"hasn't", 'each', 'that', 'she', 'once', 'their', "isn't", 'of', 'being', 'in',
'her', "haven't", 'the', 'hadn', 'these', 'if', 'between', 'most', "doesn't",
'll', 'having', 'out', 'shan', 'against', 'mightn', "weren't", 'until', 'own',
'herself', 'below', 'hers', 'whom', 'has', 'doing', 'were', "you'll", 'at',
'here', 'any', 'where', 'while', 'your', 'be', 'from', 'how', 'won',
'themselves', 'needn', 'then', "wasn't", "don't", 'ours', "wouldn't", 'under',
'did', 'both', 'does', 's', 'been', "you're", 'them', 'i', 'isn', "shouldn't",
'on', 're', 'further', 'you', 'weren', 't', 'into', 'same', "won't", 'do',
"should've", "hadn't", 'no', 'ourselves', 'me', 'don', 'yours', 'y', 'because',
'down', "couldn't", 'd', "shan't", 'by', 'some', "mightn't", 've', 'himself',
'theirs', "it's", 'my', 'a', "that'll", 'hasn', 'during', 'who', 'didn',
'should', "needn't", 'not', 'what', 'more', "you'd", 'through', 'they',
'yourself', 'other', 'up', 'when', 'yourselves', 'couldn', 'ma', 'have',
'wouldn', 'those', 'again', 'as', 'only', 'now', 'than', 'aren', 'an', 'too',
"you've"}
```

[82]:
```
#filtered text
filtered_text=[]

for word in token_word:
    if word not in stop_words:
        filtered_text.append(word)
```

[83]:
```
print(filtered_text)
```

```
['Fragments', 'Within', 'Works', 'Tolkien', 'When', 'writing', 'The', 'Lord',
'Rings', 'Tolkien', 'strove', 'build', 'mythos', 'England', 'To', 'Tolkien',
'drew', 'fragments', 'world', 'around', 'Over', 'course', 'past', 'two',
'weeks', 'used', 'understanding', 'Tolkien', ''', 'fascination', 'fragments',
'world', 'better', 'understand', 'Tolkien', ''', 'stories', 'come', 'however',
'The', 'Notion', 'Club', 'Papers', 'demonstrated', 'neglected', 'consider',
'idea', '"', 'fragments', '"', 'affected', 'telling', 'stories', 'The',
'Notion', 'Club', 'Papers', 'provide', 'glimpse', 'Tolkien', ''', 'creating',
'process', 'form', 'Ramer', ''', 'ramblings', 'The', 'act', 'Creation', 'one',
'finding', 'fragments', 'giving', 'context', 'Looking', 'back', 'Tolkien', ''',
'stories', '-The', 'Lord', 'Rings\xad', 'particular', '-', 'I', 'find',
'fragments', 'lie', 'hearts', 'Through', 'examination', 'I', 'hope', 'achieve',
'better', 'knowledge', 'Tolkien', ''', 'use', 'fragments', 'better',
'understand', 'stories', 'First', 'I', 'prove', 'brief', 'account', 'fragments',
'understand', 'The', 'fragments', 'Tolkien', 'often', 'interested', 'language',
'-', 'example', 'remaining', 'snippets', 'long', 'lost', 'poem', 'oddity',
'word', ''', 'modern', 'meaning', 'Being', 'philologist', 'Tolkien', 'would',
'take', 'fragments', 'try', 'provide', 'history', 'context', 'Effectively',
'Tolkien', 'using', 'fragments', 'work', 'backwards', 'time', 'developing',
'fragment', 'etymology', 'sorts', 'The', 'class', 'identified', 'clear',
'example', 'Book', 'I', 'The', 'Lord', 'Rings', 'Frodo', 'sings', 'full',
'version', '"', 'Hey', 'diddle', 'diddle', '"', 'establishes', 'song', 'old',
'even', 'Frodo', 'sang', 'Bree', 'By', 'creating', 'history', 'poem', 'Tolkien',
'creates', 'tangible', 'connection', 'modern', 'day', 'Creation', 'Through',
'examination', 'Ramer', '’', 'ramblings', 'Tolkien', '’', 'Notion', 'Club',
'Papers', 'gain', 'insight', 'Tolkien', 'understood', 'used', 'fragments', 'In',
'The', 'Notion', 'Club', 'Papers', 'Ramer', 'expresses', 'idea', 'identity',
'object', 'place', 'union', 'physical', 'self', 'history', 'fragments',
'preserve', 'identity', 'even', 'subject', 'lost', 'physical', 'historical',
'presence', 'continuing', 'tell', 'story', 'However', 'fragments',
'meaningless', 'without', 'context', 'much', 'way', 'meteorite', '-',
'fragment', 'large', 'object', 'long', 'history', '-', 'another', 'stone',
'already', 'know', 'history', 'Language', 'fragments', 'unique', 'aspect',
'fragment', '’', 'history', 'often', 'found', 'within', 'fragment', 'What',
'enables', 'Tolkien', 'find', 'story', 'within', 'fragments', 'heredity',
'language', 'Human', 'beings', '"', 'native', 'language', '"', 'stored',
'within', '"', 'incarnate', 'beings', '"', 'By', 'linking', 'modern',
'fragments', 'languages', 'developing', 'since', 'childhood', 'Tolkien', 'able',
'provide', 'context', 'fragments', '-', 'able', 'move', 'back', 'time',
'Interestingly', 'find', 'motif', 'fragments', 'within', 'Tolkien', '’',
'stories', 'The', 'Lord', 'Rings', 'particular', 'The', 'One', 'Ring',
'powerful', 'artifact', 'Second', 'Age', 'object', 'great', 'power', '-',
'Sauron', 'cast', 'Ring', 'lost', 'context', 'lost', 'When', 'Bilbo', 'found',
'Age', 'later', 'nothing', 'ring', 'interesting', 'capabilities', 'The', 'Ring',
'mere', 'fragment', 'dark', 'master', 'fragment', 'Age', 'fragment', 'without',
'context', 'Nor', 'seem', 'Bilbo', 'particularly', 'concerned', 'history',
'Ring', 'made', 'progress', 'identity', 'time', 'passed', 'Frodo', 'However',
'Ring', 'passed', 'Frodo', 'Ring', '’', 'true', 'nature', 'revealed', 'Frodo',
'begin', 'move', 'towards', 'truly', 'understanding', 'Ring', 'history',
'Frodo', 'completes', 'history', 'Ring', 'means', 'heredity', '"', 'native',
'language', '"', 'adventure', 'shared', 'Bilbo', 'And', 'Ring', 'finally',
'destroyed', 'takes', 'remnants', 'Second', 'Age', 'In', 'way', 'I', 'believe',
'Frodo', 'lot', 'common', 'Ramer', 'Tolkien', 'At', 'conclusion', 'The', 'Lord',
'Rings', 'Merry', 'Pippin', 'Sam', 'successfully', 'reintegrated', 'hobbit',
'society', 'Frodo', 'As', 'bearer', 'One', 'Ring', 'Frodo', 'pursued', 'story',
'completed', 'tale', 'Lord', 'Rings', 'also', 'exposed', 'power', 'I',
'suspect', 'exposure', 'similar', 'effect', 'Ramer', '’', 'lucid', 'dreams',
'His', 'dreamwanderings', 'pursuit', 'fragments', 'time', 'space', 'broadened',
'perspective', 'way', 'members', 'Notion', 'Club', 'seem', 'unable', 'really',
'identify', 'even', 'experience', 'setting', 'dreams', 'free', 'Like', 'Ramer',
'Frodo', 'experienced', 'vastness', 'world', 'sense', 'time', 'space',
'experience', 'marked', 'As', 'result', 'Frodo', 'unable', 'reintegrate', 'old',
'life', 'ultimately', 'forced', 'depart', 'In', 'I', 'attempted', 'explore',
'importance', 'fragments', 'tools', 'creation', 'motifs', 'within', 'Lord',
'Rings', 'A', 'complete', 'analysis', 'give', 'attention', 'stories',
'fragment', 'question', 'given', 'context', 'outside', 'plot', 'The', 'Hobbit',
'Smith', 'Wootton', 'Major', 'I', 'believe', 'would', 'interesting', 'compare',
'elements', 'types', 'stories', '-', 'context', 'given', 'fragment', '-',
'determine', 'narrative', 'affected', 'Author', 'N', 'Malaqai', 'Vasquez']
```

[84]:
```
#right now we can see the freq of the words without the noisy words

fdist_filtered = FreqDist(filtered_text)
print(fdist_filtered.most_common(20))
```

```
[('fragments', 20), ('Tolkien', 18), ('The', 13), ('’', 13), ('Frodo', 11),
('Ring', 10), ('history', 9), ('fragment', 9), ('context', 8), ('-', 8), ('I',
8), ('Lord', 7), ('stories', 7), ('Rings', 6), ('Ramer', 6), ('Notion', 5),
('Club', 5), ('"', 5), ('"', 5), ('time', 5)]
```

[85]:
```
ps = PorterStemmer()

stemmed_words=[]

for words in filtered_text:
    stemmed_words.append(ps.stem(words))
```

[86]:
```
print(stemmed_words)
```

```
['fragment', 'within', 'work', 'tolkien', 'when', 'write', 'the', 'lord',
'ring', 'tolkien', 'strove', 'build', 'mytho', 'england', 'to', 'tolkien',
'drew', 'fragment', 'world', 'around', 'over', 'cours', 'past', 'two', 'week',
'use', 'understand', 'tolkien', '’', 'fascin', 'fragment', 'world', 'better',
'understand', 'tolkien', '’', 'stori', 'come', 'howev', 'the', 'notion', 'club',
'paper', 'demonstr', 'neglect', 'consid', 'idea', '"', 'fragment', '"',
'affect', 'tell', 'stori', 'the', 'notion', 'club', 'paper', 'provid', 'glimps',
'tolkien', '’', 'creat', 'process', 'form', 'ramer', '’', 'rambl', 'the', 'act',
'creation', 'one', 'find', 'fragment', 'give', 'context', 'look', 'back',
'tolkien', '’', 'stori', '-the', 'lord', 'rings\xad', 'particular', '-', 'i',
'find', 'fragment', 'lie', 'heart', 'through', 'examin', 'i', 'hope', 'achiev',
'better', 'knowledg', 'tolkien', '’', 'use', 'fragment', 'better', 'understand',
'stori', 'first', 'i', 'prove', 'brief', 'account', 'fragment', 'understand',
'the', 'fragment', 'tolkien', 'often', 'interest', 'languag', '-', 'exampl',
'remain', 'snippet', 'long', 'lost', 'poem', 'odditi', 'word', '’', 'modern',
'mean', 'be', 'philologist', 'tolkien', 'would', 'take', 'fragment', 'tri',
'provid', 'histori', 'context', 'effect', 'tolkien', 'use', 'fragment', 'work',
'backward', 'time', 'develop', 'fragment', 'etymolog', 'sort', 'the', 'class',
'identifi', 'clear', 'exampl', 'book', 'i', 'the', 'lord', 'ring', 'frodo',
'sing', 'full', 'version', '"', 'hey', 'diddl', 'diddl', '"', 'establish',
'song', 'old', 'even', 'frodo', 'sang', 'bree', 'by', 'creat', 'histori',
'poem', 'tolkien', 'creat', 'tangibl', 'connect', 'modern', 'day', 'creation',
'through', 'examin', 'ramer', '’', 'rambl', 'tolkien', '’', 'notion', 'club',
'paper', 'gain', 'insight', 'tolkien', 'understood', 'use', 'fragment', 'in',
'the', 'notion', 'club', 'paper', 'ramer', 'express', 'idea', 'ident', 'object',
'place', 'union', 'physic', 'self', 'histori', 'fragment', 'preserv', 'ident',
'even', 'subject', 'lost', 'physic', 'histor', 'presenc', 'continu', 'tell',
'stori', 'howev', 'fragment', 'meaningless', 'without', 'context', 'much',
'way', 'meteorit', '-', 'fragment', 'larg', 'object', 'long', 'histori', '-',
'anoth', 'stone', 'alreadi', 'know', 'histori', 'languag', 'fragment', 'uniqu',
'aspect', 'fragment', '’', 'histori', 'often', 'found', 'within', 'fragment',
'what', 'enabl', 'tolkien', 'find', 'stori', 'within', 'fragment', 'hered',
'languag', 'human', 'be', '"', 'nativ', 'languag', '"', 'store', 'within', '"',
'incarn', 'be', '"', 'by', 'link', 'modern', 'fragment', 'languag', 'develop',
'sinc', 'childhood', 'tolkien', 'abl', 'provid', 'context', 'fragment', '-',
'abl', 'move', 'back', 'time', 'interestingli', 'find', 'motif', 'fragment',
'within', 'tolkien', '’', 'stori', 'the', 'lord', 'ring', 'particular', 'the',
'one', 'ring', 'power', 'artifact', 'second', 'age', 'object', 'great', 'power',
'-', 'sauron', 'cast', 'ring', 'lost', 'context', 'lost', 'when', 'bilbo',
'found', 'age', 'later', 'noth', 'ring', 'interest', 'capabl', 'the', 'ring',
'mere', 'fragment', 'dark', 'master', 'fragment', 'age', 'fragment', 'without',
'context', 'nor', 'seem', 'bilbo', 'particularli', 'concern', 'histori', 'ring',
'made', 'progress', 'ident', 'time', 'pass', 'frodo', 'howev', 'ring', 'pass',
'frodo', 'ring', '’', 'true', 'natur', 'reveal', 'frodo', 'begin', 'move',
'toward', 'truli', 'understand', 'ring', 'histori', 'frodo', 'complet',
'histori', 'ring', 'mean', 'hered', '"', 'nativ', 'languag', '"', 'adventur',
'share', 'bilbo', 'and', 'ring', 'final', 'destroy', 'take', 'remnant',
'second', 'age', 'in', 'way', 'i', 'believ', 'frodo', 'lot', 'common', 'ramer',
'tolkien', 'at', 'conclus', 'the', 'lord', 'ring', 'merri', 'pippin', 'sam',
'success', 'reintegr', 'hobbit', 'societi', 'frodo', 'as', 'bearer', 'one',
'ring', 'frodo', 'pursu', 'stori', 'complet', 'tale', 'lord', 'ring', 'also',
'expos', 'power', 'i', 'suspect', 'exposur', 'similar', 'effect', 'ramer', '’',
'lucid', 'dream', 'hi', 'dreamwand', 'pursuit', 'fragment', 'time', 'space',
'broaden', 'perspect', 'way', 'member', 'notion', 'club', 'seem', 'unabl',
'realli', 'identifi', 'even', 'experi', 'set', 'dream', 'free', 'like', 'ramer',
'frodo', 'experienc', 'vast', 'world', 'sens', 'time', 'space', 'experi',
'mark', 'as', 'result', 'frodo', 'unabl', 'reintegr', 'old', 'life', 'ultim',
'forc', 'depart', 'in', 'i', 'attempt', 'explor', 'import', 'fragment', 'tool',
'creation', 'motif', 'within', 'lord', 'ring', 'a', 'complet', 'analysi',
'give', 'attent', 'stori', 'fragment', 'question', 'given', 'context', 'outsid',
'plot', 'the', 'hobbit', 'smith', 'wootton', 'major', 'i', 'believ', 'would',
'interest', 'compar', 'element', 'type', 'stori', '-', 'context', 'given',
'fragment', '-', 'determin', 'narr', 'affect', 'author', 'n', 'malaqai',
'vasquez']
```

[87]:
```
word_freq_stemmed = FreqDist(stemmed_words)
word_freq_stemmed
```

[87]:
```
FreqDist({'fragment': 30, 'tolkien': 18, 'ring': 17, 'the': 13, '’': 13,
'frodo': 11, 'stori': 10, 'histori': 9, 'context': 8, '-': 8, …})
```

[88]:
```
token_text_item_stemmed = word_freq_stemmed.items()
print(token_text_item_stemmed)
```

```
dict_items([('fragment', 30), ('within', 6), ('work', 2), ('tolkien', 18),
('when', 2), ('write', 1), ('the', 13), ('lord', 7), ('ring', 17), ('strove',
1), ('build', 1), ('mytho', 1), ('england', 1), ('to', 1), ('drew', 1),
('world', 3), ('around', 1), ('over', 1), ('cours', 1), ('past', 1), ('two', 1),
('week', 1), ('use', 4), ('understand', 5), ('’', 13), ('fascin', 1), ('better',
3), ('stori', 10), ('come', 1), ('howev', 3), ('notion', 5), ('club', 5),
('paper', 4), ('demonstr', 1), ('neglect', 1), ('consid', 1), ('idea', 2), ('"',
5), ('"', 5), ('affect', 2), ('tell', 2), ('provid', 3), ('glimps', 1),
('creat', 3), ('process', 1), ('form', 1), ('ramer', 6), ('rambl', 2), ('act',
1), ('creation', 3), ('one', 3), ('find', 4), ('give', 2), ('context', 8),
('look', 1), ('back', 2), ('-the', 1), ('rings\xad', 1), ('particular', 2),
('-', 8), ('i', 8), ('lie', 1), ('heart', 1), ('through', 2), ('examin', 2),
('hope', 1), ('achiev', 1), ('knowledg', 1), ('first', 1), ('prove', 1),
('brief', 1), ('account', 1), ('often', 2), ('interest', 3), ('languag', 6),
('exampl', 2), ('remain', 1), ('snippet', 1), ('long', 2), ('lost', 4), ('poem',
2), ('odditi', 1), ('word', 1), ('modern', 3), ('mean', 2), ('be', 3),
('philologist', 1), ('would', 2), ('take', 2), ('tri', 1), ('histori', 9),
('effect', 2), ('backward', 1), ('time', 5), ('develop', 2), ('etymolog', 1),
('sort', 1), ('class', 1), ('identifi', 2), ('clear', 1), ('book', 1), ('frodo',
11), ('sing', 1), ('full', 1), ('version', 1), ('hey', 1), ('diddl', 2),
('establish', 1), ('song', 1), ('old', 2), ('even', 3), ('sang', 1), ('bree',
1), ('by', 2), ('tangibl', 1), ('connect', 1), ('day', 1), ('gain', 1),
('insight', 1), ('understood', 1), ('in', 3), ('express', 1), ('ident', 3),
('object', 3), ('place', 1), ('union', 1), ('physic', 2), ('self', 1),
('preserv', 1), ('subject', 1), ('histor', 1), ('presenc', 1), ('continu', 1),
('meaningless', 1), ('without', 2), ('much', 1), ('way', 3), ('meteorit', 1),
('larg', 1), ('anoth', 1), ('stone', 1), ('alreadi', 1), ('know', 1), ('uniqu',
1), ('aspect', 1), ('found', 2), ('what', 1), ('enabl', 1), ('hered', 2),
('human', 1), ('nativ', 2), ('store', 1), ('incarn', 1), ('link', 1), ('sinc',
1), ('childhood', 1), ('abl', 2), ('move', 2), ('interestingli', 1), ('motif',
2), ('power', 3), ('artifact', 1), ('second', 2), ('age', 4), ('great', 1),
('sauron', 1), ('cast', 1), ('bilbo', 3), ('later', 1), ('noth', 1), ('capabl',
1), ('mere', 1), ('dark', 1), ('master', 1), ('nor', 1), ('seem', 2),
('particularli', 1), ('concern', 1), ('made', 1), ('progress', 1), ('pass', 2),
('true', 1), ('natur', 1), ('reveal', 1), ('begin', 1), ('toward', 1), ('truli',
1), ('complet', 3), ('adventur', 1), ('share', 1), ('and', 1), ('final', 1),
('destroy', 1), ('remnant', 1), ('believ', 2), ('lot', 1), ('common', 1), ('at',
1), ('conclus', 1), ('merri', 1), ('pippin', 1), ('sam', 1), ('success', 1),
('reintegr', 2), ('hobbit', 2), ('societi', 1), ('as', 2), ('bearer', 1),
('pursu', 1), ('tale', 1), ('also', 1), ('expos', 1), ('suspect', 1),
('exposur', 1), ('similar', 1), ('lucid', 1), ('dream', 2), ('hi', 1),
('dreamwand', 1), ('pursuit', 1), ('space', 2), ('broaden', 1), ('perspect', 1),
('member', 1), ('unabl', 2), ('realli', 1), ('experi', 2), ('set', 1), ('free',
1), ('like', 1), ('experienc', 1), ('vast', 1), ('sens', 1), ('mark', 1),
('result', 1), ('life', 1), ('ultim', 1), ('forc', 1), ('depart', 1),
('attempt', 1), ('explor', 1), ('import', 1), ('tool', 1), ('a', 1), ('analysi',
1), ('attent', 1), ('question', 1), ('given', 2), ('outsid', 1), ('plot', 1),
('smith', 1), ('wootton', 1), ('major', 1), ('compar', 1), ('element', 1),
('type', 1), ('determin', 1), ('narr', 1), ('author', 1), ('n', 1), ('malaqai',
1), ('vasquez', 1)])
```

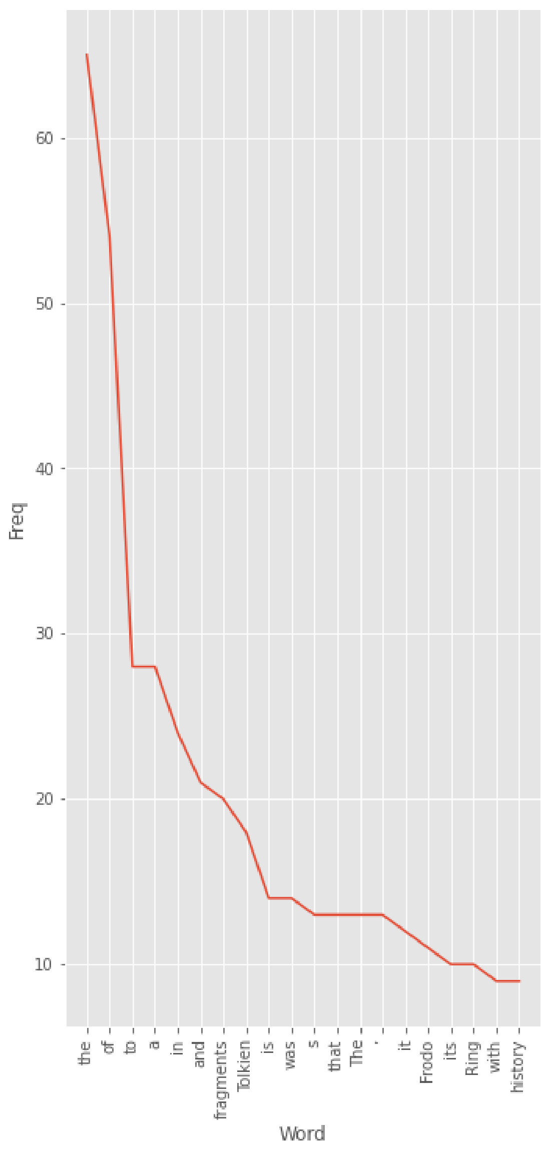
[89]:
```
word_freq_stemmed_df = pd.
 ↪DataFrame(token_text_item_stemmed,columns=["Word","Freq"])
word_freq_stemmed_df.head()
```

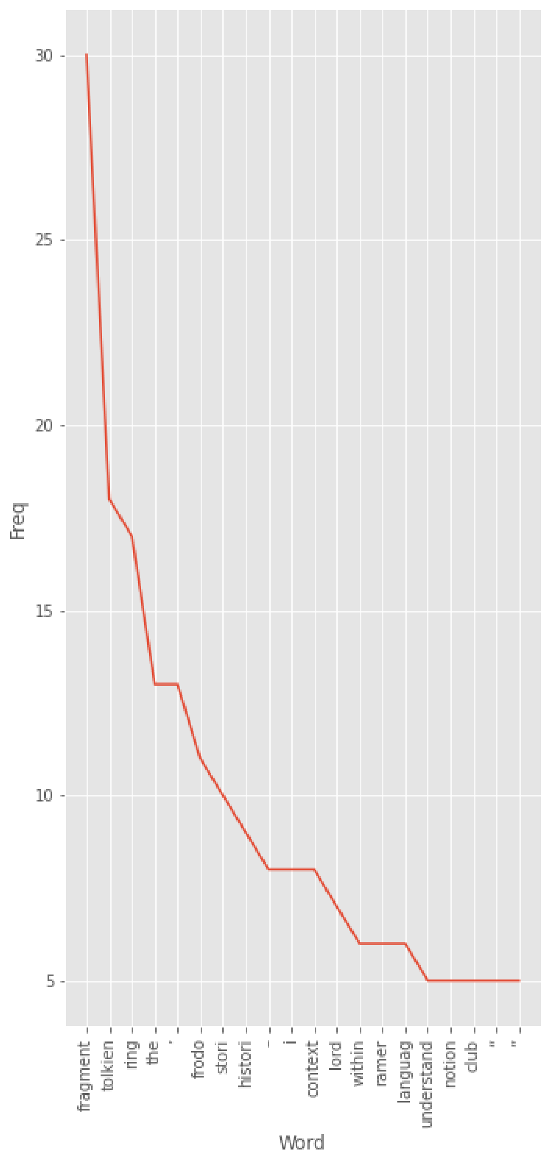
[89]:
```
       Word  Freq
0  fragment    30
1    within     6
2      work     2
3   tolkien    18
4      when     2
```

[90]:
```
stemmed_freq_sorted_df = word_freq_stemmed_df.nlargest(20,"Freq").
 ↪sort_values(by="Freq",ascending=False)
stemmed_freq_sorted_df.head()
```

[90]:
```
        Word  Freq
0   fragment    30
3    tolkien    18
8       ring    17
6        the    13
24         ’    13
```

[91]:
```
fig, axes = plt.subplots(1,2,figsize=(12, 12))
sns.lineplot(x="Word",y="Freq",data=token_freq_sorted_df,ax=axes[0])
sns.lineplot(x="Word",y="Freq",data=stemmed_freq_sorted_df,ax=axes[1])
axes[0].tick_params(axis='x', rotation=90)
axes[1].tick_params(axis='x', rotation=90)
plt.show()
```





The stemmed text shows better the repeated words, the tokenized text have a lot of prepositions and articles.

Level 3

Exercici 1

Perform sentiment analysis on your dataset.

[92]:
```python
tokenized_text =sent_tokenize(raw_text)
print(tokenized_text)
```

```
['Fragments Within the Works of Tolkien When writing The Lord of the Rings,
Tolkien strove to build a mythos of England.', 'To do so, Tolkien drew from the
```

fragments in the world around him.', 'Over the course of the past two weeks, we have used our understanding of Tolkien's fascination with the fragments in our world to better understand where Tolkien's stories come from; however, The Notion Club Papers have demonstrated to me that we have neglected to consider how the idea of "fragments" has affected the telling of the stories themselves.', 'The Notion Club Papers provide a glimpse of Tolkien's creating process in the form of Ramer's ramblings: The act of Creation is one of finding fragments and giving them context.', 'Looking back now at Tolkien's stories -The Lord of the Rings\xad in particular - I find that fragments lie at their very hearts.', 'Through this examination, I hope to achieve better knowledge of Tolkien's use of fragments so as to better understand the stories themselves.', 'First, I will prove a brief account of fragments as we understand them.', 'The fragments that Tolkien was most often interested in were those of language - for example, the remaining snippets of a long lost poem or an oddity in a word's modern meaning.', 'Being the philologist that he was, Tolkien would take these fragments and try to provide a history, a context, for them.', 'Effectively, Tolkien was using these fragments to work backwards through time by developing for the fragment an etymology of sorts.', 'The class identified a clear example of this in Book I of The Lord of the Rings where Frodo sings a full version of "Hey diddle diddle" and establishes it as a song that was old even when Frodo sang it in Bree.', 'By creating a history for the poem, Tolkien creates a tangible connection between the modern day and his Creation.', 'Through examination of Ramer's ramblings in Tolkien's Notion Club Papers we can gain insight as to how Tolkien understood and used fragments.', 'In The Notion Club Papers, Ramer expresses the idea that the identity of an object or place is the union of its physical self with its history; fragments preserve this identity even after the subject has lost its physical and historical presence by continuing to tell the story.', 'However, these fragments are meaningless without context, much in the way that a meteorite - a fragment of what was once a large object with a long history - is just another stone if we do not already know its history.', 'Language fragments are unique in this aspect in that the fragment's history can often be found within the fragment itself.', 'What enables Tolkien to find the story within these fragments is the heredity of language: Human beings have a "native language" which is stored within our "incarnate beings." By linking modern fragments to the languages that he had been developing since childhood, Tolkien was able to provide a context for the fragments - he was able to move back in time.', 'Interestingly, we can find the motif of fragments within Tolkien's stories themselves and The Lord of the Rings in particular.', 'The One Ring was a powerful artifact of the Second Age, an object of great power - but when Sauron was cast down and the Ring lost, its context was lost with it.', 'When Bilbo found it an Age later, it was nothing more than a ring with interesting capabilities.', 'The Ring was a mere fragment of its dark master, a fragment of an Age, a fragment without context.', 'Nor does it seem that Bilbo was particularly concerned with the history of the Ring, for he had made no progress as to its identity by the time he had passed it on to Frodo.', 'However, it is only when the Ring is passed on to Frodo that the Ring's true nature is revealed and Frodo can begin to move towards truly understanding the Ring and its history.', 'Frodo completes the history of the

```
Ring by means of a heredity "native language" of adventure shared between him
and Bilbo.', 'And when the Ring is finally destroyed, it takes with it the
remnants of the Second Age.', 'In this way I believe that Frodo has a lot in
common with Ramer and Tolkien.', 'At the conclusion of The Lord of the Rings,
Merry and Pippin and Sam have successfully reintegrated into hobbit society, but
not so Frodo.', 'As bearer of the One Ring, Frodo had pursued its story and
completed the tale of the Lord of the Rings, but he had also been exposed to its
power.', 'I suspect this exposure to be similar in effect to Ramer's lucid
dreams: His dream-wanderings and the pursuit of fragments through time and space
have broadened his perspective in a way that other members of the Notion Club
seem unable to really identify with, even if they have some experience in
setting their own dreams free.', 'Like Ramer, Frodo has experienced the vastness
of the world in the sense of time and space, and his experience has marked
him.', 'As a result, Frodo is unable in reintegrate into his old life and is
ultimately forced to depart.', 'In the above I have attempted to explore the
importance of fragments not only as tools of creation but as motifs within the
Lord of the Rings.', 'A more complete analysis should give more attention to
those stories in which the fragment in question was not given a context outside
of the plot, as in The Hobbit or Smith of Wootton Major.', 'I believe it would
be interesting to compare elements of both types of stories - those in which
context is given to the fragment and those where it is not - to determine how
the narrative is affected.', 'Author: N. Malaqai Vasquez']
```

[94]:
```
df_text = pd.DataFrame({'lines': tokenized_text})
df_text.head()
```

[94]:
```
                                               lines
0  Fragments Within the Works of Tolkien When wri…
1  To do so, Tolkien drew from the fragments in t…
2  Over the course of the past two weeks, we have…
3  The Notion Club Papers provide a glimpse of To…
4  Looking back now at Tolkien's stories -The Lor…
```

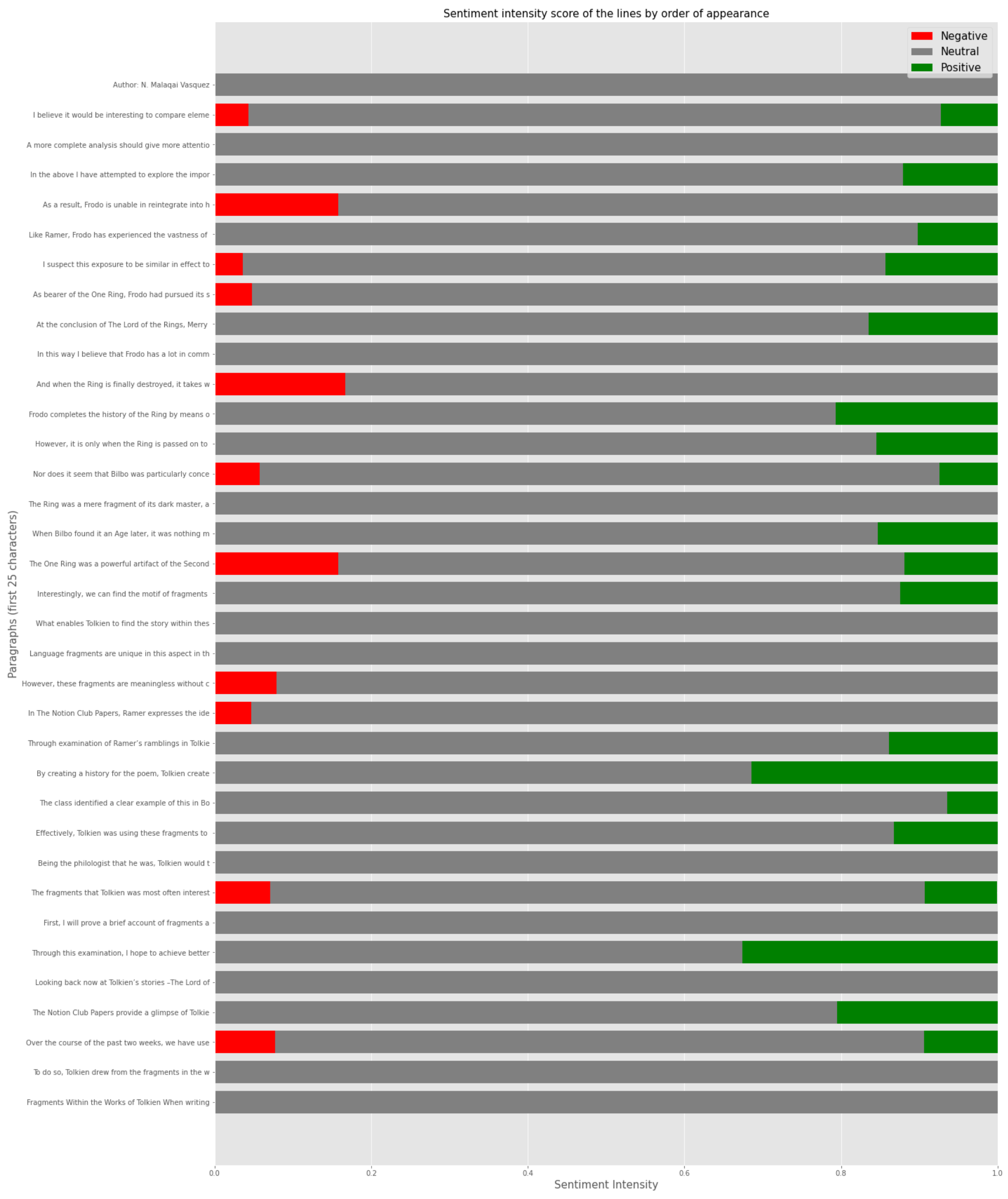
[96]:
```
sentiment_analyzer = SentimentIntensityAnalyzer()

sentiments = df_text['lines'].apply(lambda x: sentiment_analyzer.
 ↪polarity_scores(x))

df_text['compound'] = sentiments.apply(lambda x : x['compound'])
df_text['negative'] = sentiments.apply(lambda x : x['neg'])
df_text['neutral'] = sentiments.apply(lambda x : x['neu'])
df_text['positive'] = sentiments.apply(lambda x : x['pos'])
```

[106]:
```
df_text.head()
```

[106]:
```
                                               lines  compound  negative  \
0  Fragments Within the Works of Tolkien When wri…    0.0000     0.000
```

```
1  To do so, Tolkien drew from the fragments in t…    0.0000     0.000
2  Over the course of the past two weeks, we have…    0.2732     0.077
3  The Notion Club Papers provide a glimpse of To…    0.6908     0.000
4  Looking back now at Tolkien's stories -The Lor…    0.0000     0.000

   neutral  positive
0    1.000     0.000
1    1.000     0.000
2    0.829     0.094
3    0.795     0.205
4    1.000     0.000
```

[109]:
```
plt.figure(figsize=(20,30))

names = df_text['lines'].str[:50]

barWidth = 0.75

# Create negative Bars
plt.barh(y=names,
        width=df_text.negative,
        height=barWidth, color='red', label='Negative')
# Create neutral Bars
plt.barh(y=names,
        width=df_text.neutral,
        height=barWidth,  left=df_text.negative,  color='gray', label='Neutral')
# Create positive Bars
plt.barh(y=names,
        width=df_text.positive,
        height=barWidth,  left=[i+j for i,j in zip(df_text.negative, df_text.
 ↪neutral)],color='green', label='Positive')

plt.yticks(names)
plt.ylabel("Paragraphs (first 25 characters)", fontsize=15)
plt.xlabel("Sentiment Intensity", fontsize=15)
plt.legend(fontsize=15)
plt.title("Sentiment intensity score of the lines by order of appearance",␣
 ↪fontsize=15)

plt.show()
```



The sentiment analysis shows that the text is almost neutral, obviously the author is a Tolkien fan and the article is about the universe of the LOTR.