Analisis dades taxis NYC

In [ ]:

```
# Llibreries a fer servir

import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import datetime as dt
# Fa que  els plot apareguin dintre del quadern
%matplotlib inline


# llibreria  pel càlcul de distancies
from geopy.distance import great_circle

# Càlcul distància en km del trajecte fent servir llibreria geopy

def distance_Trip(recull_latitud, recull_longitud, final_latitud, final_longitud):

    coordenades_inicials = (recull_latitud, recull_longitud)
    coordenades_finals = (final_latitud, final_longitud)

    return great_circle(coordenades_inicials, coordenades_finals).km

# Creem una funció per determinar si el trajecte s'ha fet al matí a la tarda o a la r
# amb el següent criteri:
# Entre 6h i 12h: Mati, entre 12 i 16h: Tarda, entre 16h i 22h: tarda-nit, qualsevol

def duration (x) :
    if x in range (6, 12) :
        return 'Matí'
    elif x in range(12, 16):
        return 'Tarda'
    elif x in range(16,22):
        return 'Nit'
    else:
        return 'Nit al tard'

def tractar_verificar():

    data=pd.read_csv("./dat/train.csv")
    data.head( )

# Canviem a català columnes dataframe

    data.rename(columns={'vendor_id': 'prov_id', 'pickup_datetime': 'data_recull',
                         'dropoff_datetime': 'data_final', 'passenger_count': 'nombre_
                        'pickup_longitude': 'recull_longitud', 'pickup_latitude': 'rec
                    'dropoff_longitude': 'final_longitud', 'dropoff_latitude': 'final
                    'store_and_fwd_flag':'guarda_flag', 'trip_duration':'durada_carre


# Files i columnes, fitxer
    print("files i columnes fitxer\n",data.shape)
    print()

#Validació estadística de les variables

    print("Validacio estadística de les variable\n", data.describe())
```

```
    print()

#Verifiquem que no hi han dades amb valors nuls al dataframe
    print("Validacio de nuls\n", data.isnull().sum())
    print()

#Verifiquem que les columnes del nostre dataframe acompleixin amb l'esperat

    print ("Verificacio columnes", data.columns)
    print()

#Verifiquem els tipus de dades de les columnes del dataframe

    print ("Verificacio tipus columnes", data.dtypes)
    print()


#Validem els valor de la columna passenger_count per la sospita de que  hi han valors
#erronis o irrellevants

    print("Valors passenger_count\n", data ['nombre_pas'].value_counts ( ).sort_value
    print()


# Hi ha carreres amb 0 passatgers i amb 7, 8 i 9 en quantitats irrellevants
# Eliminem les files amb dades errònies o no significatives del dataframe

    data = data.drop(data[(data.nombre_pas == 0) | (data.nombre_pas > 6) ].index)

    print("Valors passenger_count despres de filtrar\n", data ['nombre_pas'].value_co
    print()

# Transformem els camps de dates al format que els hi correspon
# Transformem el camp store_and_fwd_flag perquè guardí en format Yes/Not si es guarda

    data['data_recull'] = data['data_recull' ].astype('datetime64[ns]')
    data['data_final'] = data['data_final' ].astype('datetime64[ns]')
    data['guarda_flag'] = data['guarda_flag' ].astype('category')


# Explorem camps de dates i generem columnes, dia de la setmana, mes, hora, minut.
# Aquest detall ens serà beneficiós en el tractament dels properes anàlisis

    data['data_recull']=pd.to_datetime(data['data_recull'])
    data['data_final']=pd.to_datetime(data['data_final'])

#Extracció dia de la setmana(Dilluns-Diumenge)
    data['recull_dia']=data['data_recull'].dt.day_name()
    data ['final_dia']=data['data_final'].dt.day_name( )

#Extracció dia de la setmana(0-6)
    data['recull_dia_no']=data['data_recull'].dt.weekday
    data['final_dia_no']=data['data_final'].dt.weekday

# Extracció del més tant de la recollida com del final de trajecte

    data ['recull_mes' ]=data['data_recull'].dt.month_name()
    data ['final_mes' ]=data['data_final'].dt.month_name()
```

```
# Extracció d'hores i minuts
    data['recull_hora']=data['data_recull'].dt.hour
    data['final_hora']=data['data_final'].dt.hour

    data['recull_min']=data['data_recull'].dt.minute
    data['final_min']=data['data_final'].dt.minute



# Ara afegim al dataframe la columna distància a partir de la funció distance_Trip. L
# el paràmetre axis = 1, indica que la funció és d'aplicació en totes les files

    data['distancia'] =data.apply(lambda x: distance_Trip(x['recull_latitud'],x['recu

    return(data)


# Iniciem anàlisi univariable sobre les dades de recollida i parada de la setmana.

def analisi_uni(data):

# Fem servir els gràfics de la llibreria seaborn. Fem servir diagrama de barres mostr
# Ordenem els dies a les x

    ordre_dies = ["Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday",
    figure, (ax1, ax2)=plt.subplots(ncols=2, figsize=(30,9))
    ax1.set_title('Recollides')
    ax=sns.countplot(x="recull_dia", data=data, order=ordre_dies, ax=ax1)
    ax2.set_title('Final Ruta')
    ax=sns.countplot (x="final_dia", data=data, order=ordre_dies, ax=ax2)
    plt.show()

# Calcular número trajectes arxivats. En estar el paràmetre normalize a true. Retorna
# Clarament els vehicles tene quasi sempre connexió al servidor y per tant poden envi
# Seaborn no te grafics tipus tarta aixi que ho farem amb un de barres

    print(data['guarda_flag'].value_counts(normalize=True))
    plt.figure(figsize=(10, 10))
    data['guarda_flag'].value_counts(normalize= True).plot (kind='bar' )
    plt.show()

# Mirem els mesos amb més viatges. Posem els mesos ben ordenats

    figure,(ax3, ax4)= plt.subplots(ncols=2, figsize=(30,9))

    ordre_mes = ["January", "February", "March", "April", "May", "June", "July", "Aug

    ax3.set_title('Mes recollida')
    ax=sns.countplot (x="recull_mes", data=data, order=ordre_mes, ax=ax3)
    ax4.set_title('Mes final carrera')
    ax=sns.countplot (x="final_mes", data=data, order=ordre_mes, ax=ax4)
    plt.show()


# Apliquem a les hores d'inici i fi trajecte de totes les files del dataframe la func
# per determinar tram horari en què es van produir

    data['recull_franja_dia']=data['recull_hora'].apply(duration)
    data['final_franja_dia']=data['final_hora'].apply(duration)
```

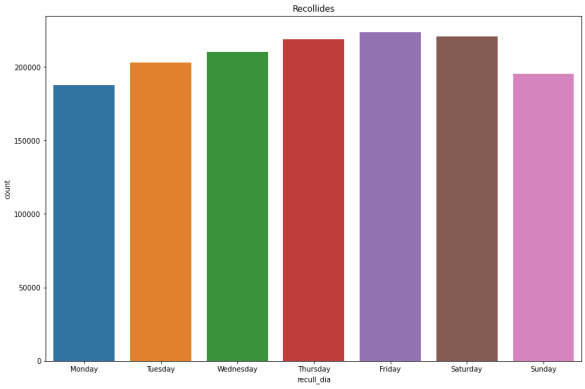
```
#data.head( )

# Ara analitzem la distribució de l'ocupació de passatgers per vehicle.
# Igual sent variables categòriques no té gaire sentit

    plt.figure(figsize=(10, 10))
    sns.distplot (data['nombre_pas'])
    plt.show()

# Analitzem la duració dels viatges amb funció de l'hora del dia

#   data['recull_franja_dia'].value_counts(normalize= True).plot (kind='bar' )

# Analitzem la duració dels viatges amb funció de la franja horària

    plt.figure(figsize=(10, 10))
    data['recull_franja_dia'].value_counts(normalize= True).plot (kind='bar' )
    plt.show()

# Analitzem la durada del viatge

    plt.figure(figsize=(10, 10))
    sns.distplot(data['durada_carrera'],kde=False)
    plt.show()

# Iniciem anàlisis multivariant

def analisi_multi(data):

#  Calculem velocitat mitjana dels taxis i ho afegim al data frame

    data['velo_mitja'] = data['distancia'] / (data['durada_carrera'] / 3600)

# Relacio de la velocitat mitjana i proveidor

    plt.figure(figsize=(10, 10))
    sns.barplot(y='velo_mitja', x='prov_id', data=data)
    plt.show()

# Analitzem relació entre proveïdor i ocupació dels vehicles.
# Això indicarà les preferències del consumidor

# Sembla que el proveïdor 2 té millors nivells d'ocupació. Té millor servei? El prime
# Per esbrinar-ne les causes farem un catplot que ens permet relacionar una variable
# categoriques. En el nostre cas relació entre duració del servei i proveïdor

    plt.figure(figsize=(10, 10))
    sns.catplot (y='durada_carrera', x='prov_id',data=data, kind='strip')
    plt.show()

# En el nostre cas no és clar quin és millor sembla que el proveïdor 1 fa rutes més l
# per viatge més gran

# Ara procedim a estudiar la correlació. Primer farem un dataframe amb les variables


    numerical = data.select_dtypes(include=['int64', 'float64', 'Int64']) [ : ]
```

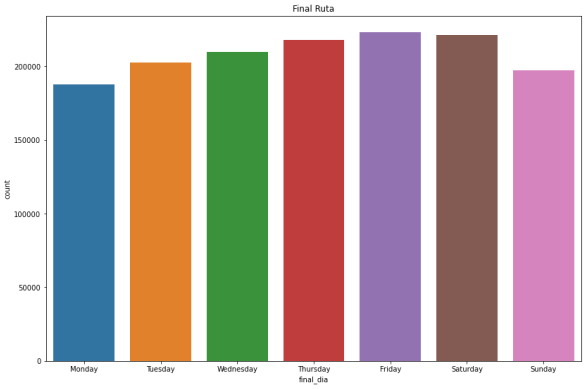
```
    print("Vaiables numériques del dataframe\n", numerical.dtypes)
    print()

# Calculem correlació
    correlation = numerical.dropna( ).corr()

    print("Correlacio entre variables", correlation)
    print()

# Presentem gràficament la correlació entre variables
# Forta correlació entre el dia de recollida i el de final de ruta com no podia ser d

    corr =data.corr ()
    sns.heatmap (corr)
    plt.show()


# Analitzem la relació entre el nombre de passatgers per carrera i la duració d'aques

    plt.figure(figsize=(10, 10))
    sns.barplot(x="nombre_pas", y="durada_carrera", data=data)
    plt.show()

# Inici programa principal
# Carreguem i netegem dades

data = tractar_verificar()

# Opcions vàlides:

# uni: anàlisi univariant
# mul: anàlisi multivariant
# fi:  sortir del programa

opc_valides = ["uni", "mul", "fi"]

opcio = input("Tipus d'anàlisi a realitzar(uni/mul/fi): ")

while opcio in opc_valides and opcio != "fi":
    if opcio == "uni":
# Iniciem anàlisis univariant
        analisi_uni(data)
    elif opcio == "mul":
# Iniciem anàlisis multivariant
        analisi_multi(data)

    opcio = input("Tipus d'anàlisi a realitzar(uni/mul/fi): ")

else:
    if opcio == "fi":
        print("Hem sortit del programa")
    else:
        print("No hem selecionat una opcio vàlida")
```

```
files i columnes fitxer
 (1458644, 11)

Validacio estadística de les variable
            prov_id    nombre_pas  recull_longitud  recull_latitud  \
```

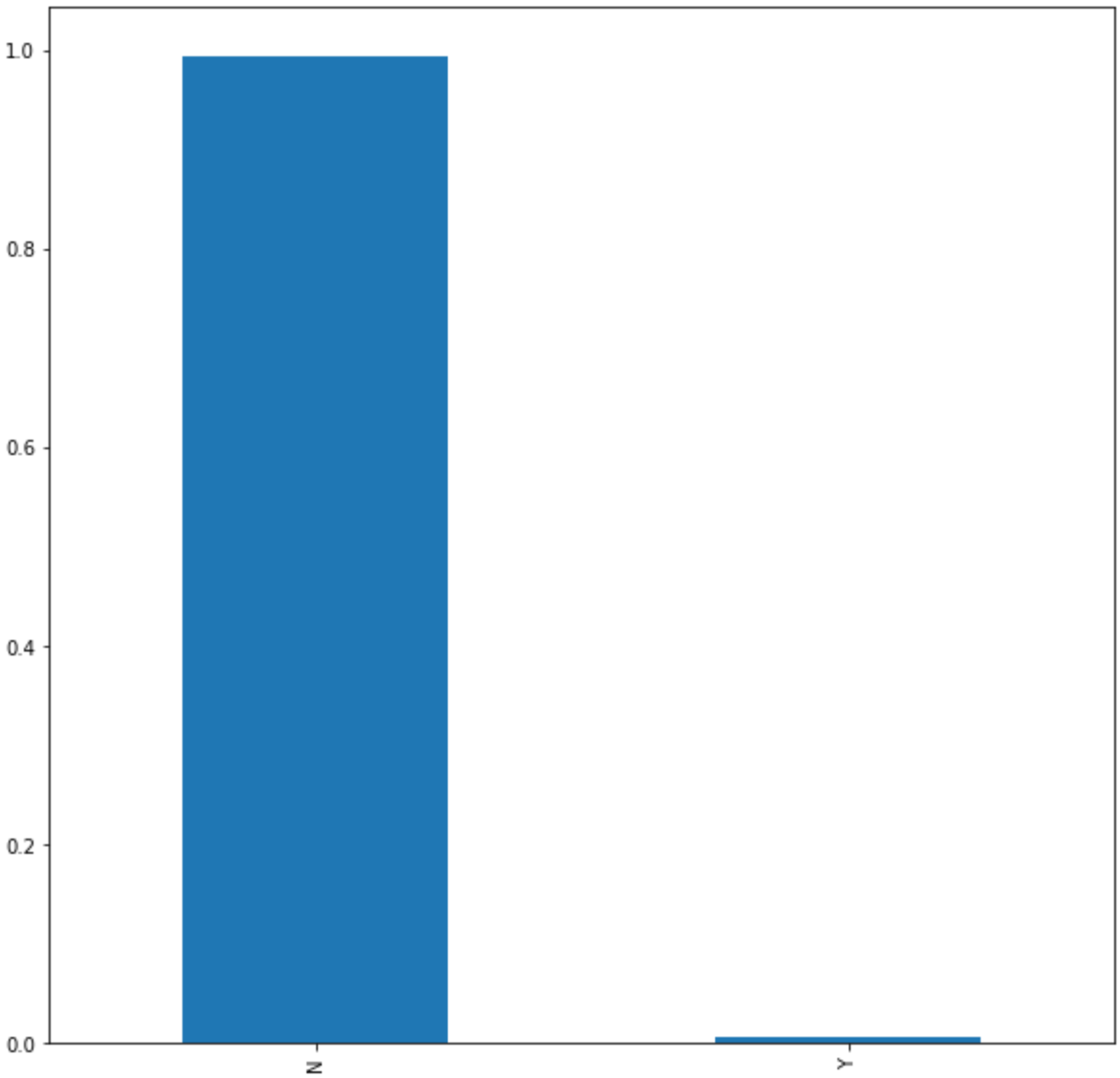
```
count  1.458644e+06  1.458644e+06     1.458644e+06    1.458644e+06
mean   1.534950e+00  1.664530e+00    -7.397349e+01    4.075092e+01
std    4.987772e-01  1.314242e+00     7.090186e-02    3.288119e-02
min    1.000000e+00  0.000000e+00    -1.219333e+02    3.435970e+01
25%    1.000000e+00  1.000000e+00    -7.399187e+01    4.073735e+01
50%    2.000000e+00  1.000000e+00    -7.398174e+01    4.075410e+01
75%    2.000000e+00  2.000000e+00    -7.396733e+01    4.076836e+01
max    2.000000e+00  9.000000e+00    -6.133553e+01    5.188108e+01

       final_longitud  final_latitud  durada_carrera
count    1.458644e+06   1.458644e+06    1.458644e+06
mean    -7.397342e+01   4.075180e+01    9.594923e+02
std      7.064327e-02   3.589056e-02    5.237432e+03
min     -1.219333e+02   3.218114e+01    1.000000e+00
25%     -7.399133e+01   4.073588e+01    3.970000e+02
50%     -7.397975e+01   4.075452e+01    6.620000e+02
75%     -7.396301e+01   4.076981e+01    1.075000e+03
max     -6.133553e+01   4.392103e+01    3.526282e+06

Validacio de nuls
 id                  0
prov_id             0
data_recull         0
data_final          0
nombre_pas          0
recull_longitud     0
recull_latitud      0
final_longitud      0
final_latitud       0
guarda_flag         0
durada_carrera      0
dtype: int64

Verificacio columnes Index(['id', 'prov_id', 'data_recull', 'data_final', 'nombre_pas
',
       'recull_longitud', 'recull_latitud', 'final_longitud', 'final_latitud',
       'guarda_flag', 'durada_carrera'],
      dtype='object')

Verificacio tipus columnes id                   object
prov_id              int64
data_recull         object
data_final          object
nombre_pas           int64
recull_longitud    float64
recull_latitud     float64
final_longitud     float64
final_latitud      float64
guarda_flag         object
durada_carrera       int64
dtype: object

Valors passenger_count
 9          1
8           1
7           3
0          60
4       28404
6       48333
3       59896
5       78088
2      210318
```

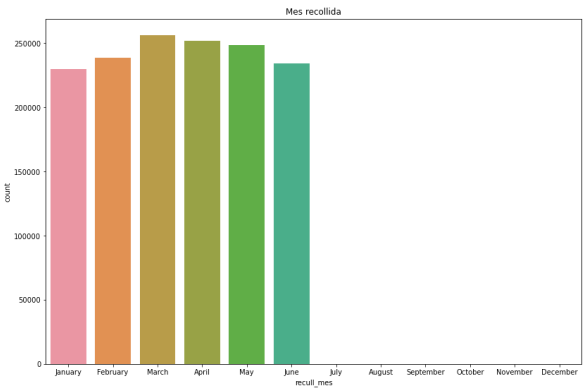
```
1    1033540
Name: nombre_pas, dtype: int64

Valors passenger_count despres de filtrar
 4      28404
6       48333
3       59896
5       78088
2      210318
1     1033540
Name: nombre_pas, dtype: int64
```

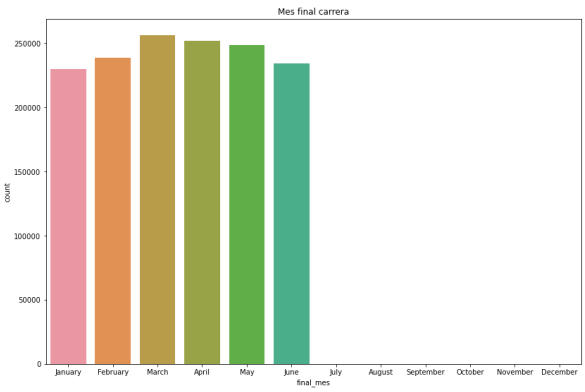




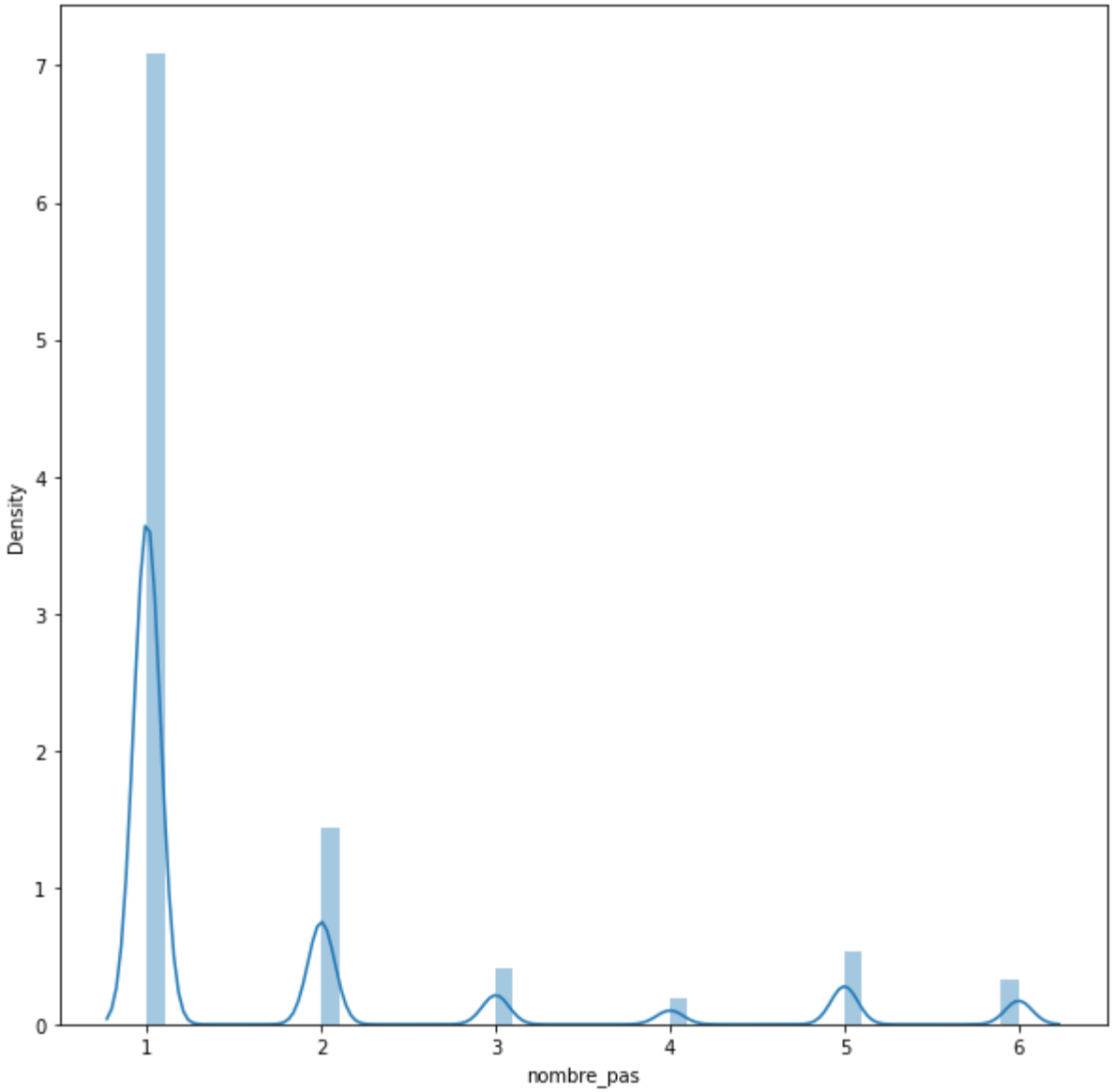

```
N    0.994486
Y    0.005514
Name: guarda_flag, dtype: float64
```

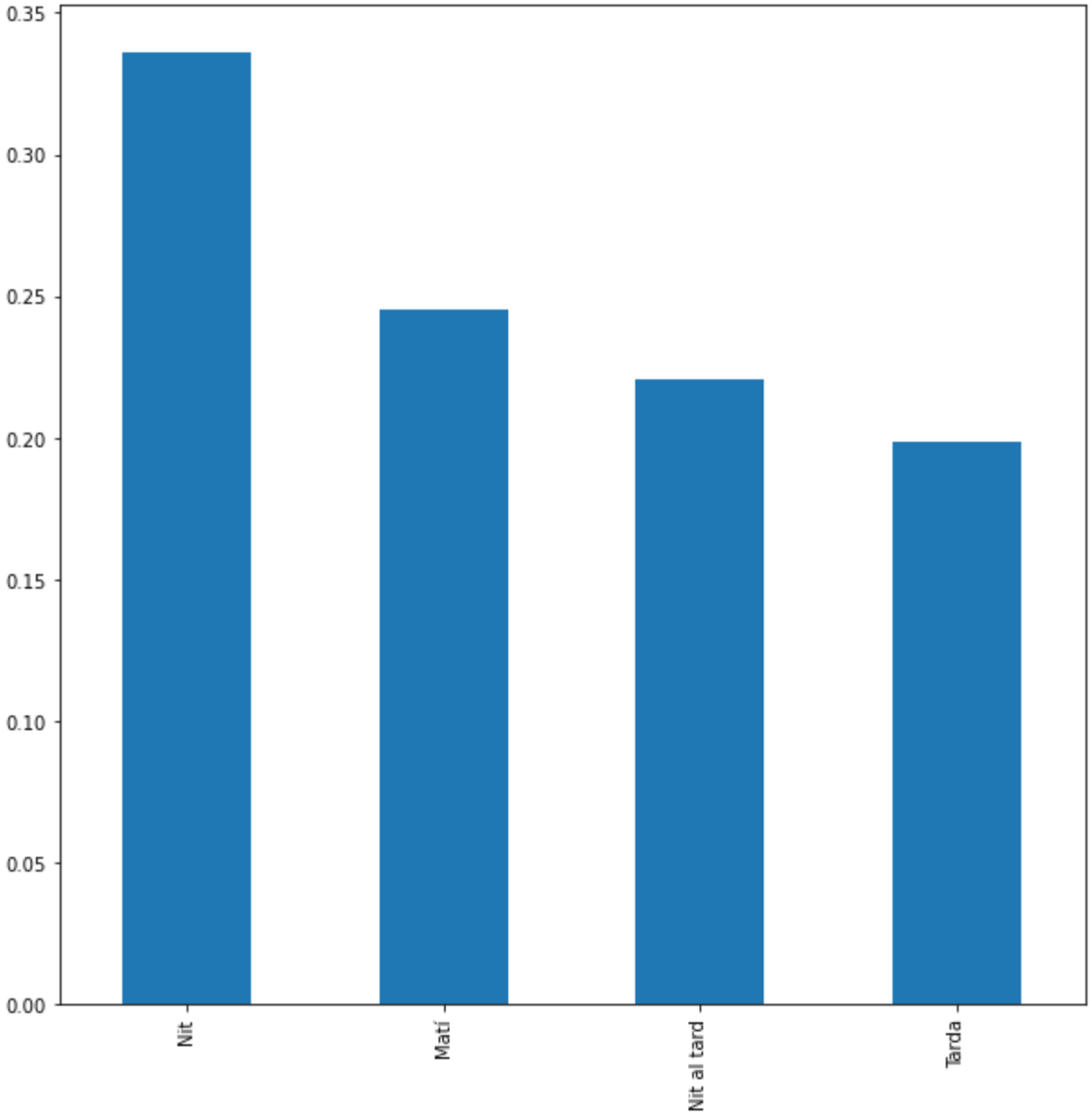




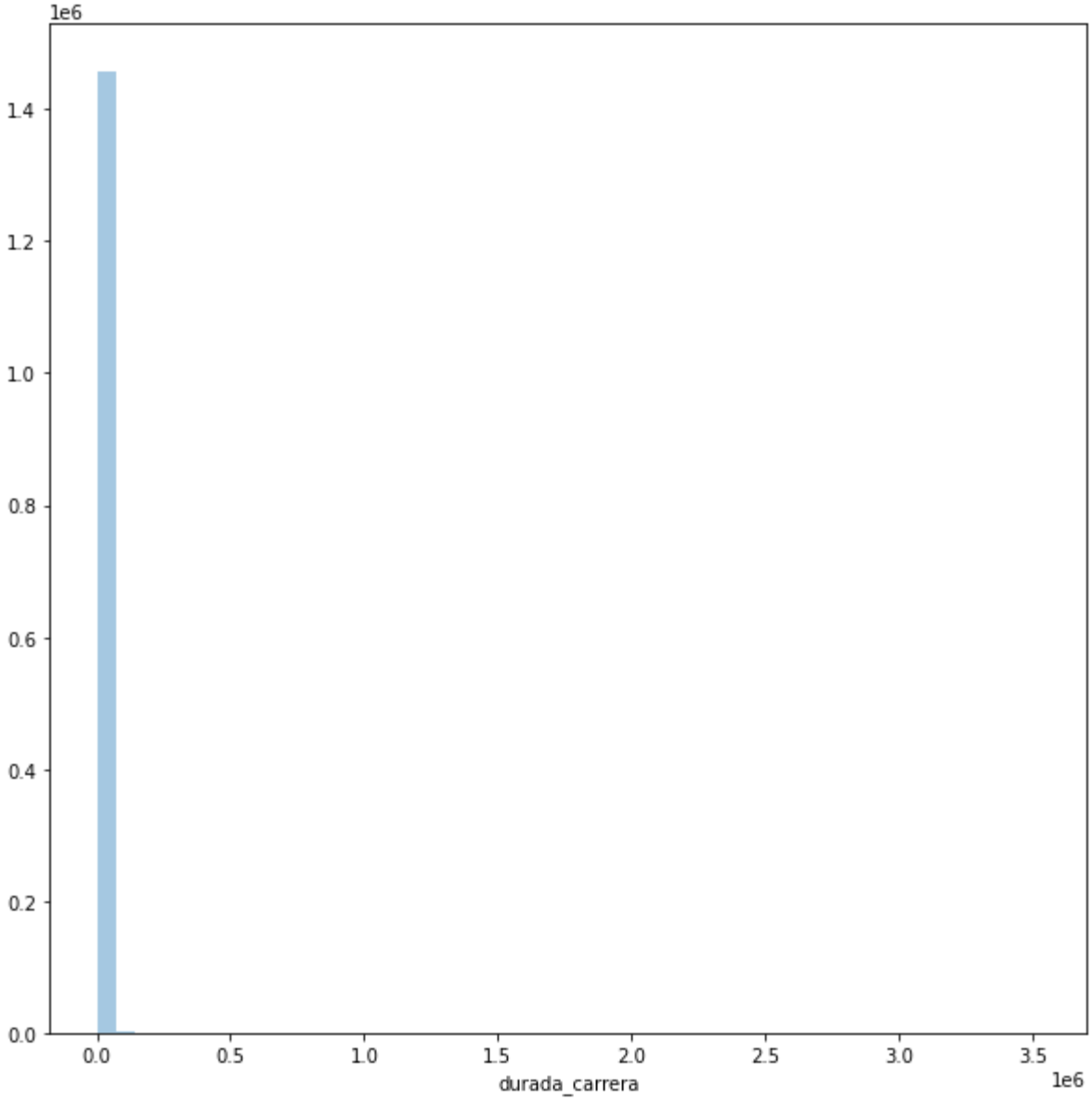



```
C:\Users\Alumne_mati1\anaconda3\lib\site-packages\seaborn\distributions.py:2619: Futu
reWarning: `distplot` is a deprecated function and will be removed in a future versio
n. Please adapt your code to use either `displot` (a figure-level function with simil
ar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

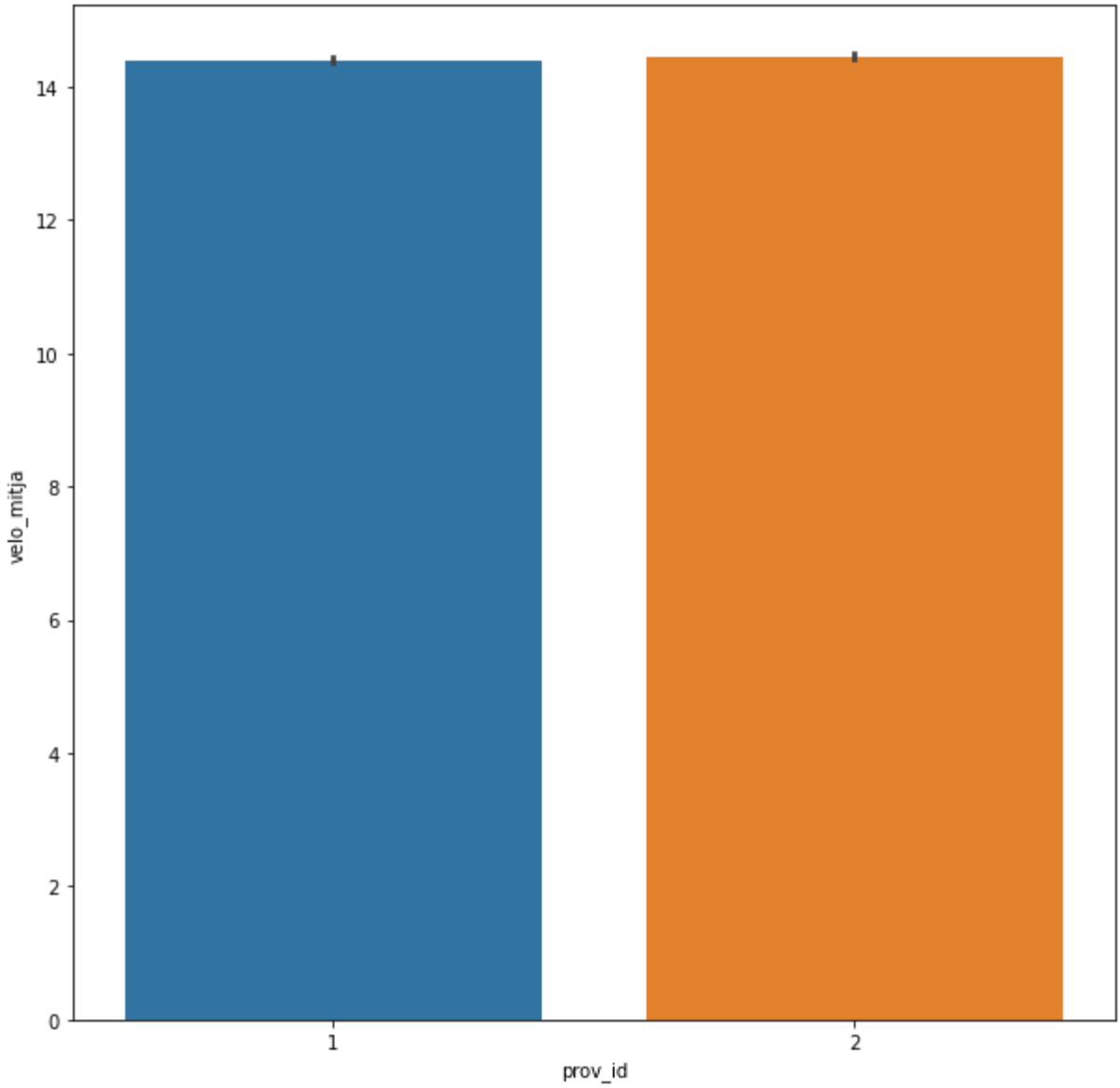

Density
nombre_pas
0
1
2
3
4
5
6
7
1
2
3
4
5
6



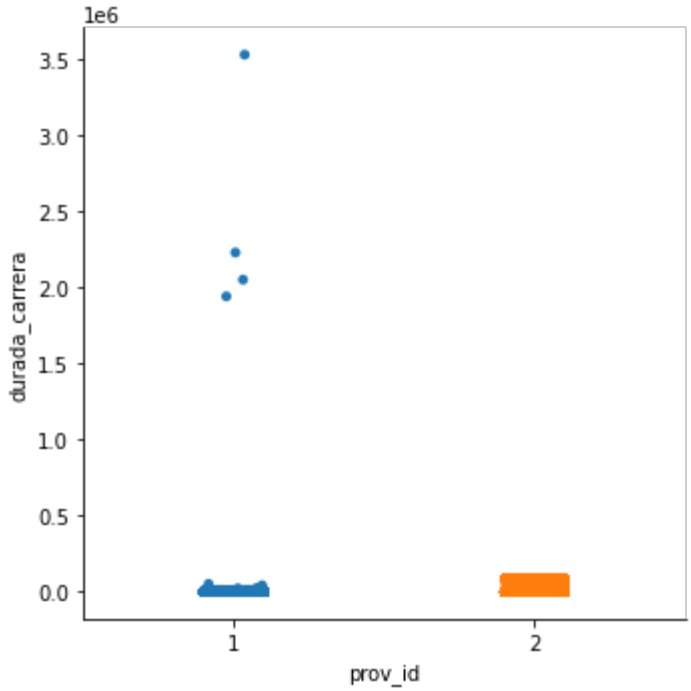

```
C:\Users\Alumne_mati1\anaconda3\lib\site-packages\seaborn\distributions.py:2619: Futu
reWarning: `distplot` is a deprecated function and will be removed in a future versio
n. Please adapt your code to use either `displot` (a figure-level function with simil
ar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```




```
Tipus d'anàlisi a realitzar(uni/mul/fi): mul
```




```
<Figure size 720x720 with 0 Axes>
```




```
Vaiables numériques del dataframe
 prov_id              int64
nombre_pas            int64
recull_longitud     float64
recull_latitud      float64
final_longitud      float64
final_latitud       float64
durada_carrera        int64
recull_dia_no         int64
final_dia_no          int64
recull_hora           int64
final_hora            int64
recull_min            int64
final_min             int64
distancia           float64
velo_mitja          float64
dtype: object
```