UQAC

Université du Québec
à Chicoutimi

Travail Pratique - Forage de données

Samuel BUSSON
Baptiste ROUPAIN

I / Présentation des Données

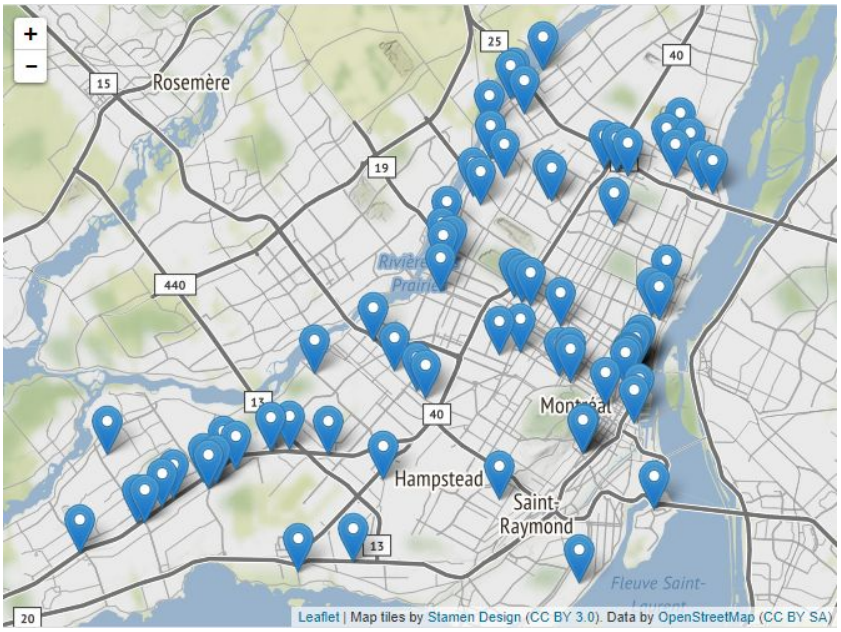
Nos données ont été collectés dans l'objectif de référencer tous les **actes criminels** dans la ville de Montréal afin de pouvoir anticiper les **zones à risques** et de catégoriser les quartiers plus sensibles aux infractions que d'autres. L'ensemble est régulièrement mis à jours et appartient officiellement à la **Division des ressources informationnelles** du Service de Police de Montréal.

Les données ont étés collectés sur le terrain et référencés dans la base de données des casernes de police.

Les **variables** de cet ensemble de données sont :

- **CATÉGORIE** : Nature de l'événement. Liste de valeur :
    - Introduction : introduction par effraction dans un établissement public ou une résidence privée, vol d'arme à feu dans une résidence
    - Vol dans / sur véhicule à moteur : vol du contenu d'un véhicule à moteur (voiture, camion, motocyclette, etc.) ou d'une pièce de véhicule (roue, parechoc, etc.)
    - Vol de véhicule à moteur : vol de voiture, camion, motocyclette, motoneige tracteur avec ou sans remorque, véhicule de construction ou de ferme, tout-terrain
    - Méfait : Graffiti et dommage de biens religieux, de véhicule ou dommage général et tous autres types de méfaits
    - Vol qualifié : Vol accompagné de violence de commerce, institution financière, personne, sac à main, véhicule blindé, véhicule, arme à feu, et tous autres types de vols qualifiés
    - Infraction entraînant la mort : Meurtre au premier degré, meurtre au deuxième degré, homicide involontaire, infanticide, négligence criminelle, et tous autres types d'infractions entraînant la mort
- **DATE** : Date du signalement de l'événement au SPVM au format AAAA-MM-JJ HH:mm:ss (note: la partie de l'heure n'est pas utilisée)
- **QUART** : Moment de la journée du signalement de l'événement au SPVM. Liste de valeur :
    - jour : Entre 8h01 et 16h
    - soir : Entre 16h01 et minuit
    - nuit : Entre 00h01 et 8h

- **PDQ** : Numéro du poste de quartier couvrant le territoire où s'est passé l'événement. Le territoire couvert par chaque poste est disponible dans l'ensemble de données des limites de PDQ
- **X et Y** : Position géospatiale selon la projection MTM8 (SRID 2950)
    - La valeur 0 est utilisée lorsqu'aucune position géographique n'a été fournie lors de la saisie de l'information (moins de 5% des cas).
    - LAT et LONG: position géographique de l'événement après obfuscation à une intersection selon le référentiel géodésique WGS84.
    - La valeur 1 est utilisée lorsqu'aucune position géographique n'a été fournie lors de la saisie de l'information (moins de 5% des cas).



Le nombre total d'instances répertoriés à ce jour est de 131 983 et l'ensemble de données ne possède aucune variable de classe

II / Vérification et pré-traitement des données

Notre ensemble de données possède à ce jour 4 valeurs manquantes, dans la champ PDQ.
PDQ correspond au numéro du poste où s'est déroulé l'événement, après quelques recherches sur le site on remarque qu'il s'agit d'un numéro positif, nous avons donc décidé de remplacer cette valeur par **-1** afin de l'identifier plus facilement lors de l'étude des données

Afin de faciliter la clarté des données nous avons également décidé de remplacer :
`'dans / sur véhicule à moteur'` par `'du contenu d\'un véhicule'`.

III / Etude statistique et analyse

Avant tout algorithme, il est nécessaire d'utiliser cette partie du code suivant qui extrait les données du fichier dans un dataframe.

```
file = pd.read_csv("crimeMontreal.csv", encoding = "ISO-8859-1")

dataframe = pd.DataFrame(file) #Transformation en dataframe
```

Pour notre étude statistique des données, nous proposons le code python suivant afin de réaliser un échantillonnage de ses derniers au niveau de leur PDQ :

```
ls_pdq = dataframe["PDQ"].unique() #Liste des PDQ
nb_pdq = len(ls_pdq)
max = 0
PDQ_max = 0
for x in range(0, nb_pdq):
   data = dataframe[(dataframe["PDQ"] == ls_pdq[x])]
   total = data["PDQ"].count()
   if (data["PDQ"].count() > max):
       max = total
       PDQ_max = ls_pdq[x]
   print("Pour le poste de police " + str(ls_pdq[x]) + ", il y a
un total de : " + str(total) + " d'actes criminels enregistrés.")
print("Le poste de police ayant enregistré le plus d'actes
criminels est le " + str(PDQ_max) + " avec un total de " +
str(max) + " enregistrements.")
```

Ce qui peut se raccourcir par :

```
df = dataframe["CATEGORIE"].groupby(dataframe["PDQ"]).count()
```

Ainsi qu'un échantillonnage au niveau de leur catégorie :

```
ls_cat = dataframe["CATEGORIE"].unique() #Liste des CATEGORIES
for x in ls_cat:
   data = dataframe[(dataframe["CATEGORIE"] == x)]
   print("Il y a un nombre total de " + str(data["PDQ"].count())
+ " actes criminels enregistrés dans la catégorie " + str(x))
```

Ce qui nous donne le résultat suivant :

```
Il y a un nombre total de 29810 actes criminels enregistrés dans
la catégorie Méfait
Il y a un nombre total de 17946 actes criminels enregistrés dans
la catégorie Vol de véhicule à moteur
Il y a un nombre total de 7612 actes criminels enregistrés dans
```

```
la catégorie Vols qualifiés
Il y a un nombre total de 39780 actes criminels enregistrés dans
la catégorie Vol dans / sur véhicule à moteur
Il y a un nombre total de 36722 actes criminels enregistrés dans
la catégorie Introduction
Il y a un nombre total de 108 actes criminels enregistrés dans la
catégorie Infractions entrainant la mort
```

Puis le code suivant afin de réaliser un classement au niveau de leur date (du plus récent au plus ancien) :

```
ls_date = dataframe["DATE"].unique() #Liste des DATES
max = 0
date_max = 0
for x in ls_date:
   data = dataframe[(dataframe["DATE"] == x)]
   total = data["DATE"].count()
   if (data["DATE"].count() > max):
       max = total
       date_max = x
   print("Le " + str(x) + ", il y a un total de : " + str(total)
+ " d'actes criminels enregistrés.")
print("Le jour ou il y a eu le plus d'actes criminels est le " +
str(date_max) + " avec un total de " + str(max) + "
enregistrements.")
```

Etant donnée le résultat trop grand, nous choisissons de montrer seulement la date possédant le plus d'actes criminels référencés :

```
Le jour ou il y a eu le plus d'actes criminels est le 2015-12-07
avec un total de 153 enregistrements.
```

Par ailleurs, nous pouvons continuer nos expériences d'échantillonnage sur les données en délivrant une série possédants la liste de tous les vols qualifiés ayant lieu de nuit, comme nous le démontre le code suivant :

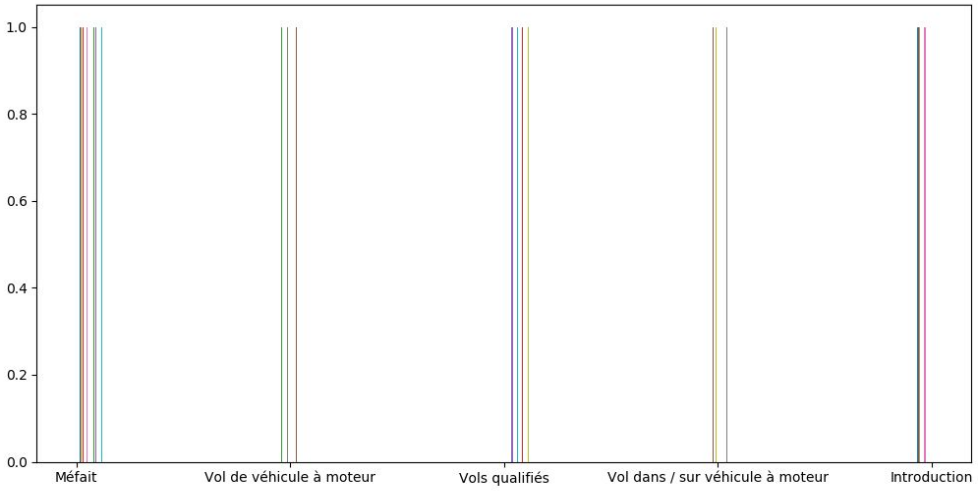
```
>>> df = dataframe[(dataframe["QUART"] == "soir") &
(dataframe["CATEGORIE"] == "Vols qualifiés")]
>>> print("Il y a un nombre total de " + str(len(df)) + "
instances de vols qualifiés le soir.")
>>> Il y a un nombre total de 3740 instances de vols qualifiés le
soir.
```

Statistiques graphiques :

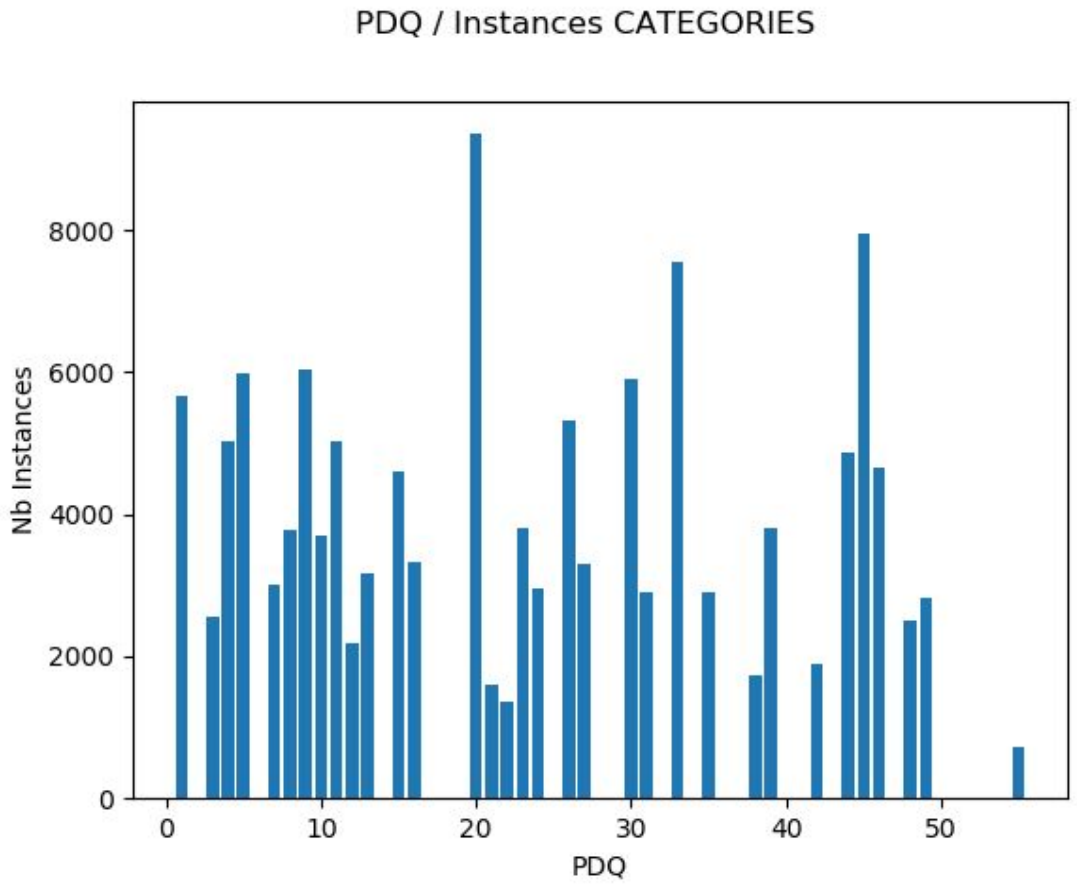
Sur les 20 premières instances de l'ensemble des données, nous avons décidés d'afficher les catégories et leur effectifs avec un diagramme en bâton :

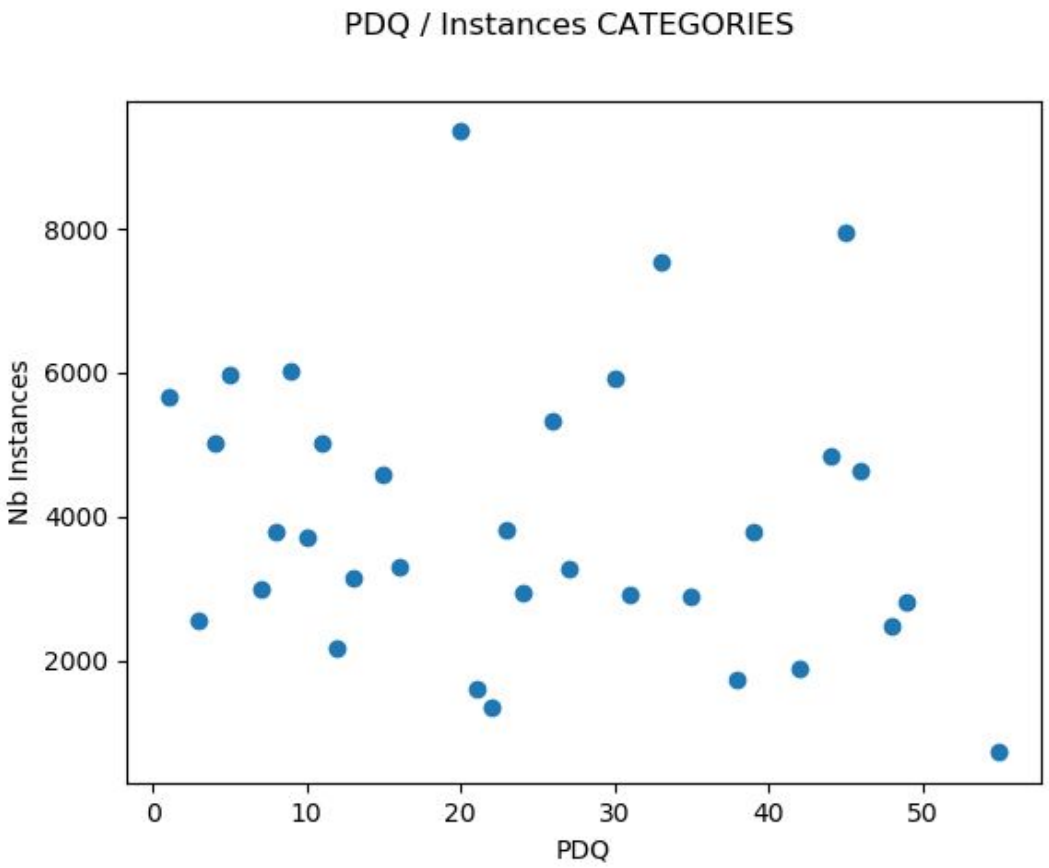
```
plt.hist(dataframe["CATEGORIE"][0:20], 30)
```

```
plt.show()
```



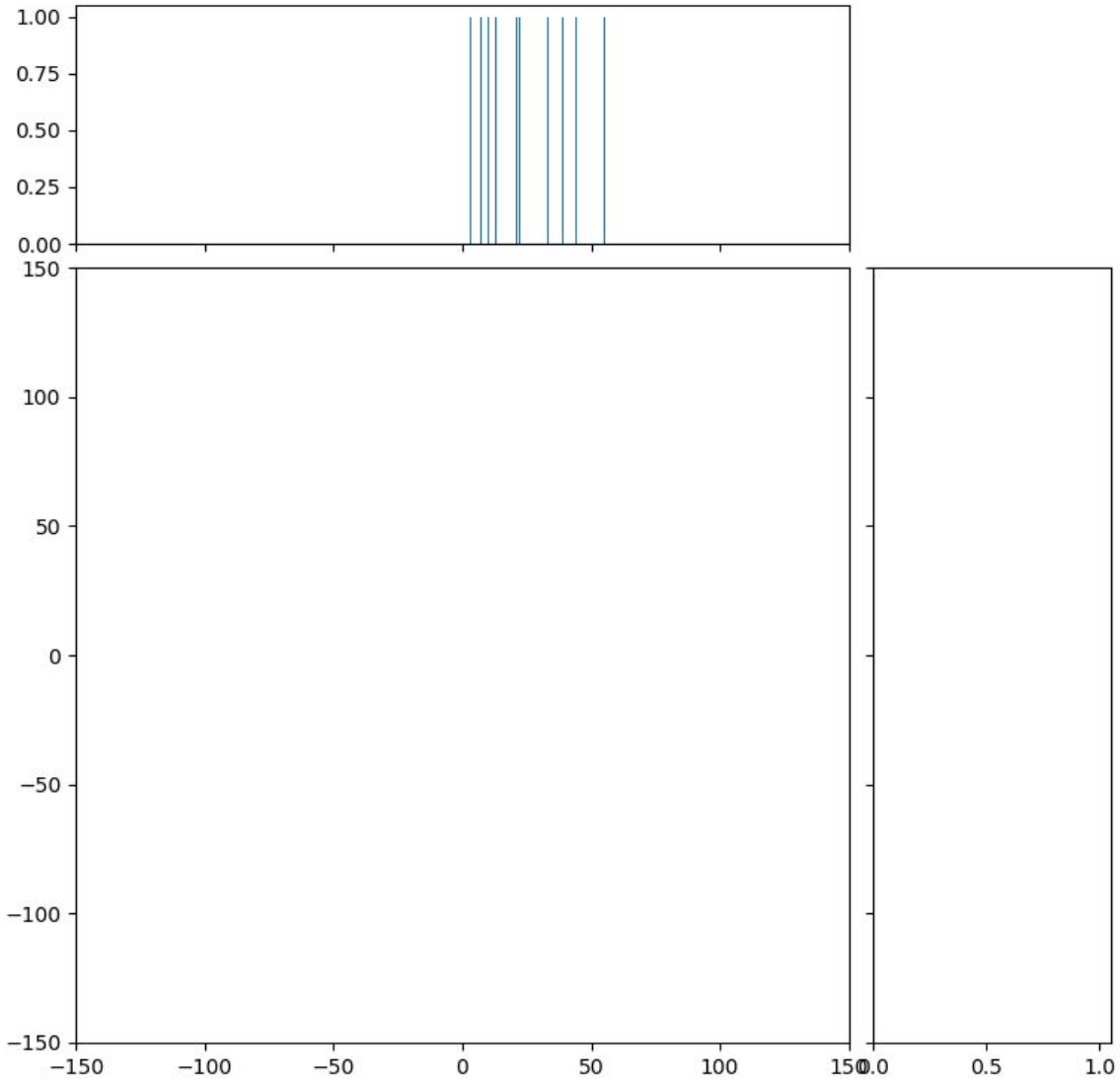
Nous nous plaçons maintenant dans un contexte plus spécialisé. En effet, nous allons, dans le cadre de notre étude statistique, réaliser un graphique avec les données calculées précédemments, notamments celles représentantes du nombre d'acte criminel par poste de police (pdq). Ainsi, nous obtenons le graphe suivant :

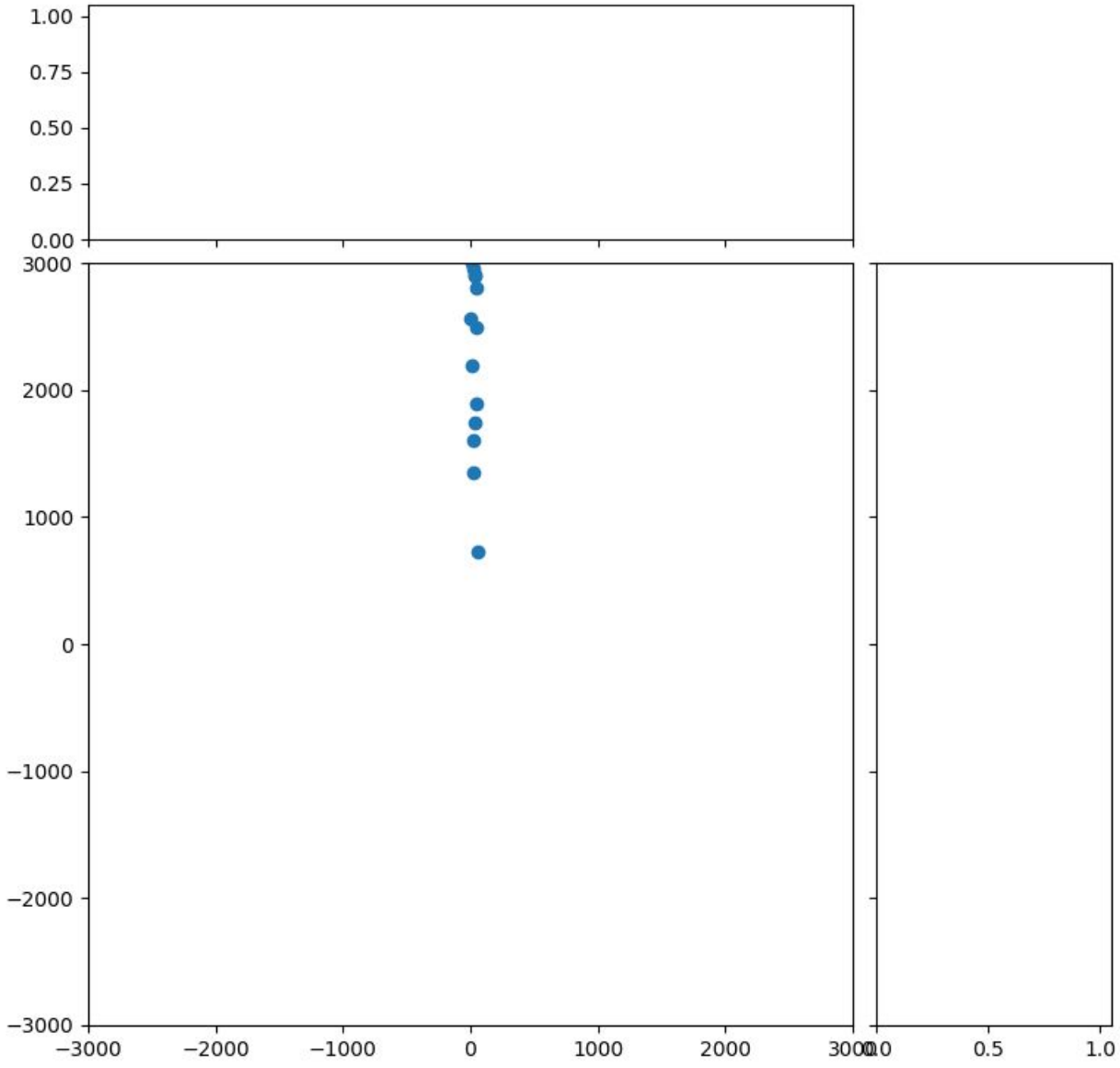





```
df1 = dataframe["PDQ"].unique().tolist() #Liste des PDQ
df2 = dataframe["CATEGORIE"].groupby(dataframe["PDQ"]).count().tolist()
#par défaut, ils sont rangés dans le même ordre
print(df1, df2)
df1.pop(33)
print(len(df1), len(df2))
plt.bar(df1, df2)
plt.suptitle('PDQ / Instances CATEGORIES')
plt.ylabel('Nb Instances')
plt.xlabel('PDQ')
plt.show()
```

En ce qui concerne maintenant la réalisation d'un histogramme, nous n'avons pas pu aboutir jusqu'à la fin de notre projet étant donné qu'un problème au niveau des limites est rapidement survenu. Cependant, voici les résultats obtenus lorsque l'on définit respectivement une limite de 3000 (de -3000 à 3000) puis de 150 (de -150 à 150) :


1.00
0.75
0.50
0.25
0.00
150
100
50
0
−50
−100
−150
−150
−100
−50
0
50
100
150
0.0
0.5
1.0

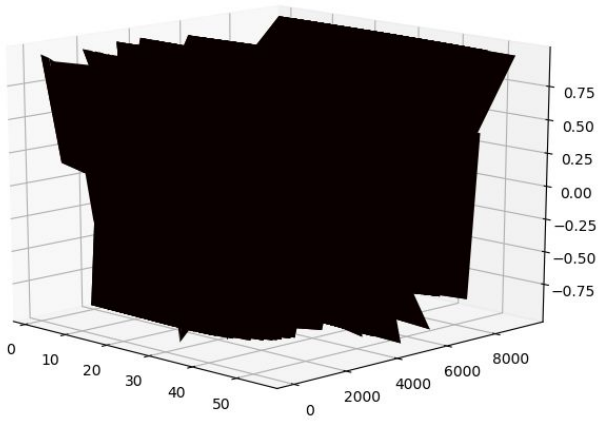


Nous utilisons toujours les mêmes liste de données **`df1`** et **`df2`** de l'exemple précédent et avons importé la librairie **`NullFormatter`** de **`matplotlib.ticker`**.

```
nullfmt = NullFormatter()
#definitions des axes :
left, width = 0.1, 0.65
bottom, height = 0.1, 0.65
bottom_h = left_h = left + width + 0.02
rect_scatter = [left, bottom, width, height]
rect_histx = [left, bottom_h, width, 0.2]
rect_histy = [left_h, bottom, 0.2, height]
plt.figure(1, figsize=(8, 8))
axScatter = plt.axes(rect_scatter)
axHistx = plt.axes(rect_histx)
axHisty = plt.axes(rect_histy)
```

```
axHistx.xaxis.set_major_formatter(nullfmt)
axHisty.yaxis.set_major_formatter(nullfmt)
axScatter.scatter(df1, df2)
bindwidth = 0.25
xymax = 150 # On définit notre limite à 150
lim = (int(xymax/bindwidth) + 1) * bindwidth
axScatter.set_xlim((-lim, lim))
axScatter.set_ylim((-lim, lim))
bins = np.arange(-lim, lim + bindwidth, bindwidth)
axHistx.hist(df1, bins=bins)
axHisty.hist(df2, bins=bins, orientation="horizontal")
axHistx.set_xlim(axScatter.get_xlim())
axHisty.set_ylim(axScatter.get_ylim())
plt.show() #Tracement du graphe
```

Enfin, notre graphique de surface permettant de visualiser sur une échelle tridimensionnel la quantité d'actes criminels référencés par poste de police a pour résultat le suivant :



Ce qui n'est pas à la hauteur de nos attentes mais l'erreur s'explique surement à cause de la répartition inexactes et non croissante des actes criminels vis à vis des postes de police dans les listes `df1` et `df2`. Le code que nous avons utilisé est le suivant :

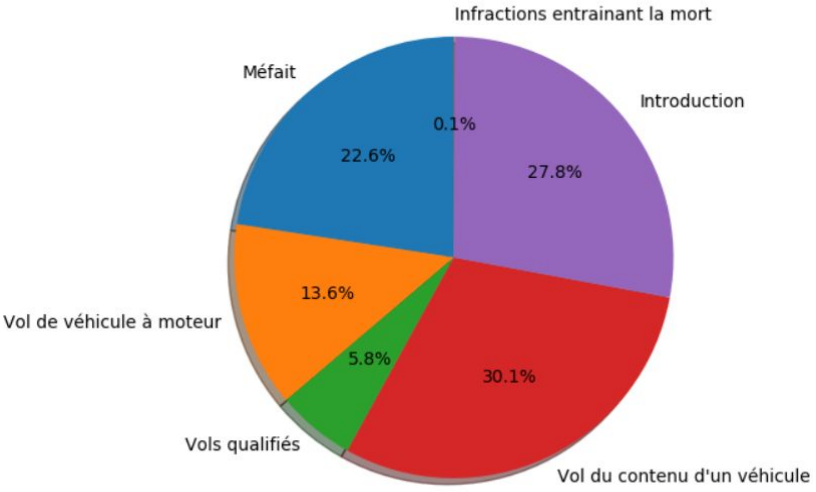
```
df1, df2 = np.meshgrid(df1, df2)
R = np.sqrt(df1**2 + df2**2)
Z = np.sin(R)
ax.plot_surface(df1, df2, Z, rstride=1, cstride=1, cmap='hot')
plt.show()
```

IV / Méthodes unique de traitement des données pour notre ensemble :

A) Les catégories d'actes criminels, que nous disent les statistiques ?

Grâce à la méthode ***unique***, nous avons pu récupérer la liste de toutes les différentes catégories existantes (soit celles énoncées ci-dessus):

```
labels = file["CATEGORIE"].unique()
sizes = []
for lab in labels:
   curVal = file[file["CATEGORIE"] == lab].shape[0]
   sizes.append(curVal)
```



Nous avons pu en tirer ce diagramme, par conclusions peu d'actes entraîne la mort cependant presque la moitié des infractions sont du vols (Ici en **orange** **vert** et **rouge**). Mais il y a également beaucoup d'introduction et de méfait sur les 131 983 observations.

B) Échantillonnage par coordonnées X et Y

Nous proposons aussi de classer les données de notre ensemble selon les coordonnées géographique ou a eu lieu les actes. En effet, ce code vas permettre à l'utilisateur de savoir combien et quels sont les actes criminels référencés à proximité selon les coordonnées et le rayon saisis. De ce fait, il en retournera une série contenant un nombre d'actes criminels variés. Ce code permet ainsi de savoir quels sont les zones avec un fort / bas taux de criminalité à Montréal.

```
import numpy as np
import pandas as pd
file = pd.read_csv("crimeMontreal.csv", encoding = "ISO-8859-1")
dataframe = pd.DataFrame(file) #Transformation en dataframe
print(dataframe["X"].mean()) #nous calcule la moyenne des
```

```
coordonnées X
print(dataframe["Y"].mean()) #nous calcule la moyenne des
coordonnées Y
print("Insérez les coordonnées de X : ")
X = input()
print("Insérez les coordonnées de Y : ")
Y = input()
print("Maintenant, insérez le rayon : ")
range = input()
#Conversion des floats en datatypes
dt1 = np.float32(X)
dt2 = np.float32(Y)
dt3 = np.float32(range)
data = dataframe[((dataframe["X"] <= (dt1 + dt3)) &
(dataframe["X"] >= (dt1 -dt3))) & ((dataframe["Y"] <= (dt2 +
dt3)) & (dataframe["Y"] >= (dt2 - dt3)))]
print(data)
print("Le nombre d'instance est de " + str(data["X"].count()))
```

Résultat pour `X = 297654` , `Y = 5041877` et `range = 2`

```
                    CATEGORIE       DATE  ...  LONGITUDE   LATITUDE
3                       Méfait  2017-07-30  ... -73.591457  45.516776
70696              Introduction  2016-09-26  ... -73.591457  45.516776
72115              Introduction  2017-05-03  ... -73.591457  45.516776
84026   Vol dans / sur véhicule à moteur  2015-10-08  ... -73.591457  45.516776
84349                   Méfait  2016-05-20  ... -73.591457  45.516776
84417   Vol dans / sur véhicule à moteur  2015-07-02  ... -73.591457  45.516776
84465              Introduction  2015-08-13  ... -73.591457  45.516776
85741              Introduction  2016-06-02  ... -73.591457  45.516776
86221                   Méfait  2015-05-30  ... -73.591457  45.516776
117217        Vol de véhicule à moteur  2018-01-29  ... -73.591457  45.516776
117218  Vol dans / sur véhicule à moteur  2018-11-05  ... -73.591457  45.516776
117219  Vol dans / sur véhicule à moteur  2018-09-30  ... -73.591457  45.516776
117220                  Méfait  2018-04-13  ... -73.591457  45.516776

[13 rows x 8 columns]
Le nombre d'instance est de 13
```