Université Paris Saclay

Rapport IAS

Guillaume Abadie, Jérôme Coquisart, Mathis Dupont, Martin Vitani

Année 2021

Table des matières

1 Introduction au problème **2**

2 Aperçu du dataset **2**

3 Définition du problème **3**

4 Préprocessing **3**

5 Choix d'algorithme **4**

6 Comparaison des modèles **5**

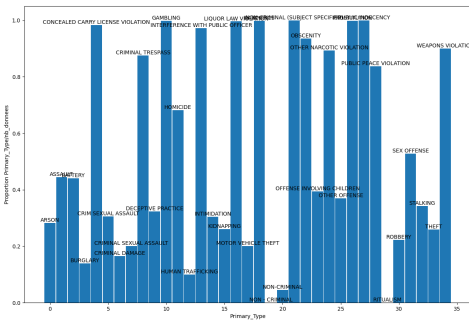
7 Présentation des résultats **6**

8 Conclusion **7**

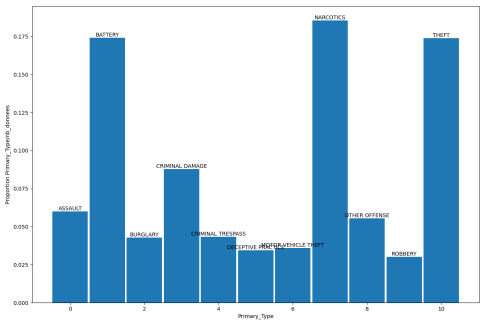
1 Introduction au problème

La ville de Chicago a un ratio de crimes, surtout sur les crimes violents, au dessus de la moyenne nationale des États-Unis. Les crimes dans la ville ont été collectés dès le début du 20ème siècle pour essayer de comprendre pourquoi la ville était sujette à autant de violence. Le dataset correspond aux crimes commis entre 2001 et 2020, et contient environ 7 millions d'entrées. Tous les jours, la police de Chicago alimente la base de donnée avec les nouveaux crimes commis dans la ville. Seuls les meurtres ne sont pas comptabilisé dans la base de donnée.

2 Aperçu du dataset



(a) Types de crimes avec le plus d'arrestation



(b) Pourcentage de crimes en fonction de leurs type

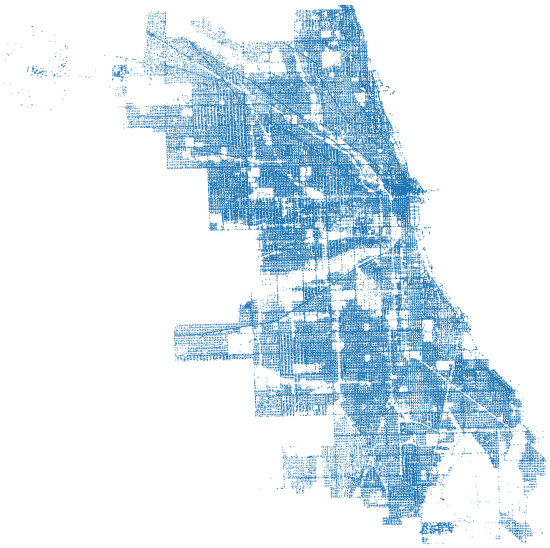


FIGURE 1 – Carte représentant l'ensemble des crimes commis à Chicago

Sur cette carte, chaque point représente un crime.

3 Définition du problème

Notre problème sera le suivant. Il s'agira de déterminer si il y aura oui ou non une arrestation à la suite d'un crime. Pour être plus précis, étant donné le lieu, la description et la date du crime, il faudra dire si cela va mener à l'arrestation d'un suspect. C'est une tâche de classification binaire, en apprentissage supervisé.

4 Préprocessing

Comme le dataset contenait peu de features, nous avons effectué plusieurs modifications sur le dataset avant d'entrainer notre modèle. De plus, certaines features étaient uniques pour chacun de nos crimes.

Nous avons tout d'abord supprimer les données qui contenaient des features vides, et nous avons équilibré le dataset. Cela a été fait avec des commandes shell pour la rapidité et la simplicité. On rappelle que notre dataset contient 7 millions d'entrées à la base. Voici les commandes effectuées.

```
## Nettoyer les données
grep -v -E '(,0,0,2)|(,,)' Crimes2001.csv  > CrimesClean.csv

## Equilibrer les données
# Ici on sépares à l'aide d'expressions régulières les arrested et les non arrested
grep -E '^(([^,]*,)|("[^"]*",)){8}false' Crimes100K.csv > CrimesCleanNonArrested.csv
grep -E '^(([^,]*,)|("[^"]*",)){8}true' Crimes100K.csv > CrimesCleanArrested.csv

# Ici on recombine en n'oubliant pas le header contenant le nom des features
head -n 1 CrimesClean.csv > CrimesEq.csv
n=$(wc -l CrimesCleanArrested.csv | grep -E -o '[0-9]+ ')
cp CrimesCleanArrested.csv tmp.csv
shuf -n $n CrimesCleanNonArrested.csv >> tmp.csv
shuf tmp.csv >> CrimesEq.csv
```

Ensuite, en python, nous avons supprimé les features uniques et celles qui se répétaient

- ID
- Case Number
- Block
- Updated On
- Longitude
- Latitude
- Location

À partir de la feature *date*, nous avons extrait les features suivantes

- Part of the day
- Weekday
- Weekend
- Month
- Hour

Enfin, à partir des coordonnées géographiques, nous avons appliqué un algorithme de k-moyennes pour séparer les zones de Chicago et en déduire une nouvelle feature *Cluster*. Voici ce que l'on obtiens lorsque l'on affiche la localisation des crimes en fonction de leurs clusters.

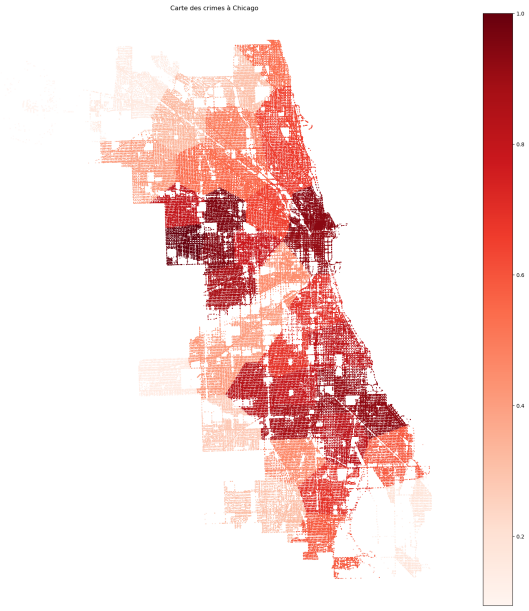


FIGURE 2 – Clusters des crimes de Chicago

Sur cette dernière carte, les clusters plus rouges concentrent le plus de crimes.

5 Choix d'algorithme

Comme notre dataset est assez conséquent, nous avons utilisé les algorithmes de la bibliothèque *scikit-learn*. Pour classifier nos crimes, nous utilisons le DecisionTree et le modèle Gaussien Naïf. Différentes fonctions dans le fichier main.py nous ont permis de déterminer les meilleurs hyper-paramètres à choisir pour le DecisionTree.

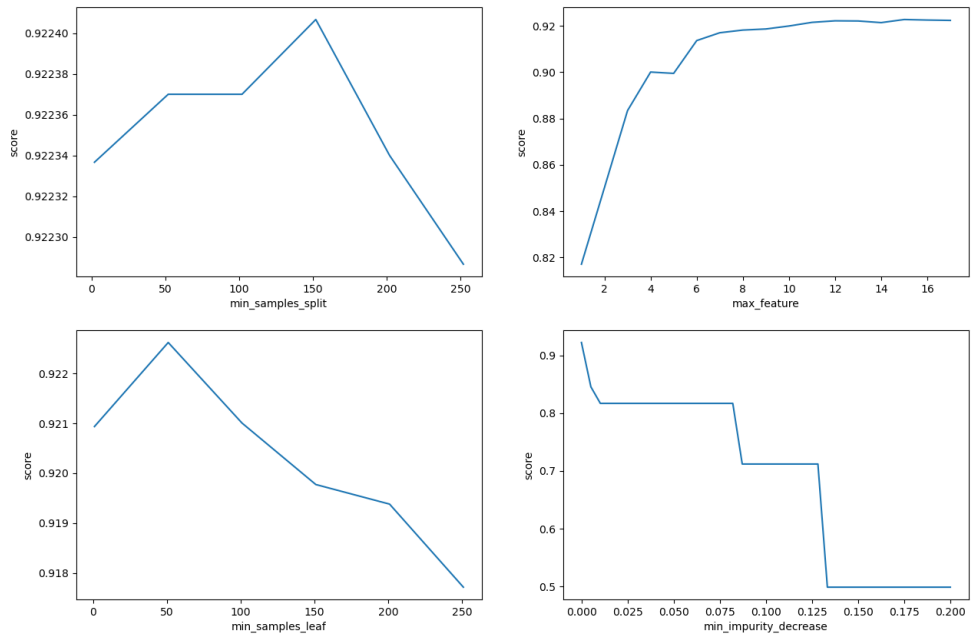


FIGURE 3 – Cross-validation pour le DecisionTree

6 Comparaison des modèles

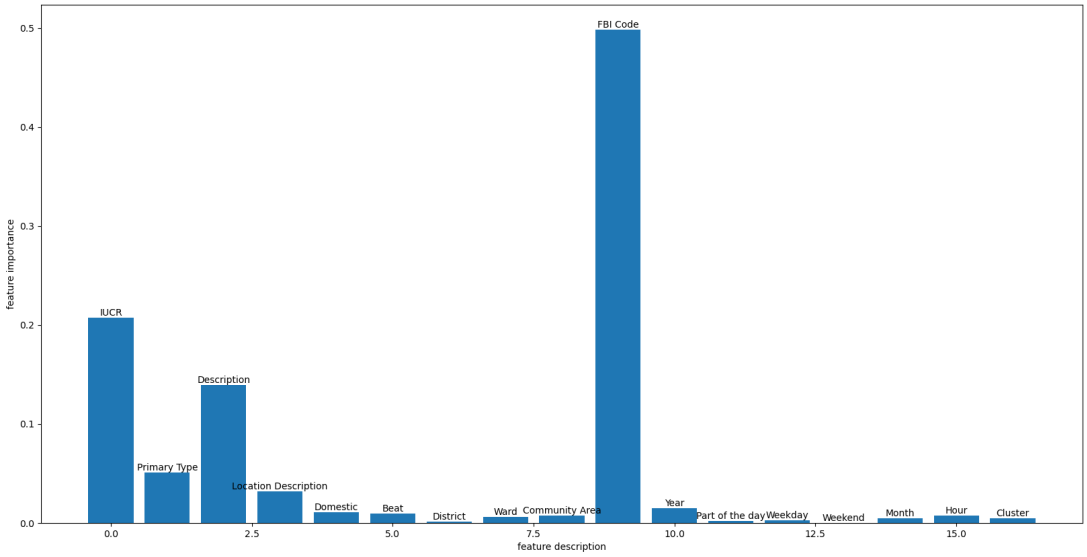


FIGURE 4 – Importance des features pour le DecisionTree

L'arbre de décision est un algorithme de classification qui nous aide à prendre des décision : selon le type de crime, selon le lieu, le moment etc., y a-t-il eu arrestation ou pas ? Pour chaque features, il essaye de répondre à des questions sur celles-ci afin de les séparer en différentes catégorie : Le crime a-t-il eu lieu sur la place publique ou dans un appartement ? (on peut voir sur la figure 4 que *location description* est une feature importante à la décision, car la question posée sur cette feature est pertinente). L'arbre de décision que nous avons obtenue est vraiment grand, car beaucoup de features, "mais nous avons crée un DecisionTree avec une partie restreinte de nos features afin de pouvoir le regarder : ? ?"

Le modèle bayésien naïf est un modèle basé sur l'EMV (Estimateur du Maximum de Vraisemblance). Le but est simple, maximiser la vraisemblance. Cependant une hypothèse importante de ce modèle est l'indépendance de chaque données, ce qui n'est pas du tout notre cas dans ce projet. Ainsi, on peut prévoir une score moyen pour ce modèle avec nos données.

Comme nous avons beaucoup de données (1M pour les calculs des scores), le bayésien naïf va avoir du mal, alors que cela ne pas changer grand chose pour le DecisionTree.

7 Présentation des résultats

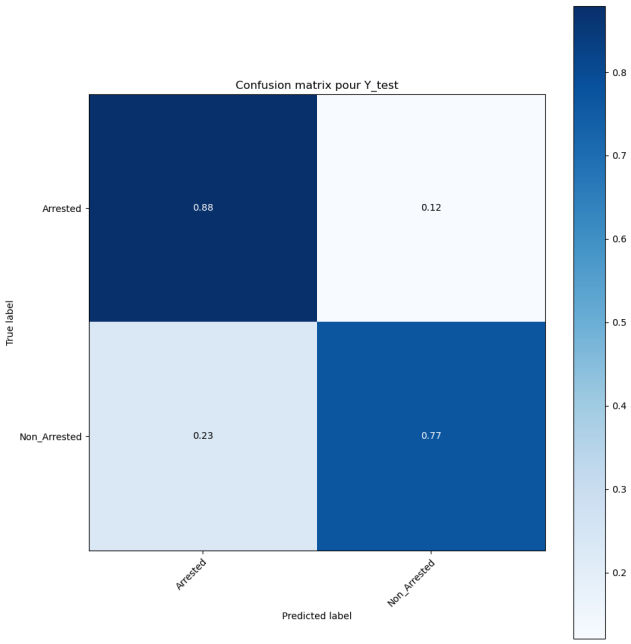


FIGURE 5 – Matrice de confusion du DecisionTree

Voici la matrice de confusion générée à partir de l'arbre décision, entrainé sur nos 1 millions de données. On peut voir que notre modèle arrive mieux à valider le fait qu'il y ai arrestation que le fait qu'il n'y ai pas d'arrestation.

Les scores obtenues pour nos deux modèles sont :

```
Score GaussNB :  0.703753
Score DecisionTree :  0.82345
```

La comparaison des modèles nous a donné raison, l'arbre de décision nous donne un meilleur score que le bayésien naïf.

8 Conclusion