Plan du rapport

Partie 1. Prétraitement des données

Remplacement des valeurs nulles par le mode

Vérification des valeurs nulles par colonnes du dataframe en faisant des sommes

Remplacement des valeurs nulles par le mode avec la méthode .fillna()

Vérification des valeurs nulles

Remplacement des valeurs erronées par le mode

Remplacement des valeurs erronées des colonnes Q1 à Q4 avec la méthode where()

Remplacement des valeurs erronées de la colonne Q5 avec la méthode where()

Remplacement des valeurs erronées des colonnes Q6 à Q10 avec la méthode where()

Encodage des variables avec ordinary encoder

Conversion des données en int

Vérification des valeurs (avec value\_counts)

Fractionnement du jeu de données

Partie 2. Développement IA

Modèle de KNN from scratch

Fonction distance

Pour calculer la distance entre 2 points a et b, on réalise une fonction qui prend en  
argument deux listes correspondant aux coordonnées des deux points de même dimension. Pour chaque élément des listes on calcule soit la distance Euclidienne √ ( soit la distance de Manhattan |(.

Une image contenant texte

Description générée automatiquement

Fonction KNN

La fonction KNN prend en argument un jeu de données de test, d’entrainement, de cible, le type de distance calculée (Euclidienne, Manhattan ou Minkowski) et le nombre de voisin. Pour chaque ligne du jeu de données de test, la fonction va calculer la distance avec toutes les lignes du jeu de données d’entrainement, grâce à la méthode .apply() de pandas et en appliquant une fonction lambda. On rajoute une colonne distance. De cette nouvelle colonne créée, on récupère les k index des distances les plus petites dans une variable.

On concatène le jeu d’entrainement avec les cibles. Pour cette liste d’index on récupère la classe la plus fréquente (A, B ou C).

On calcule notre nouvelle cible (ou y) à partir de notre fonction en faisant varier différents paramètres, comme le nombre de voisin et le type de distance.

Une image contenant texte

Description générée automatiquement

Fonction accuracy

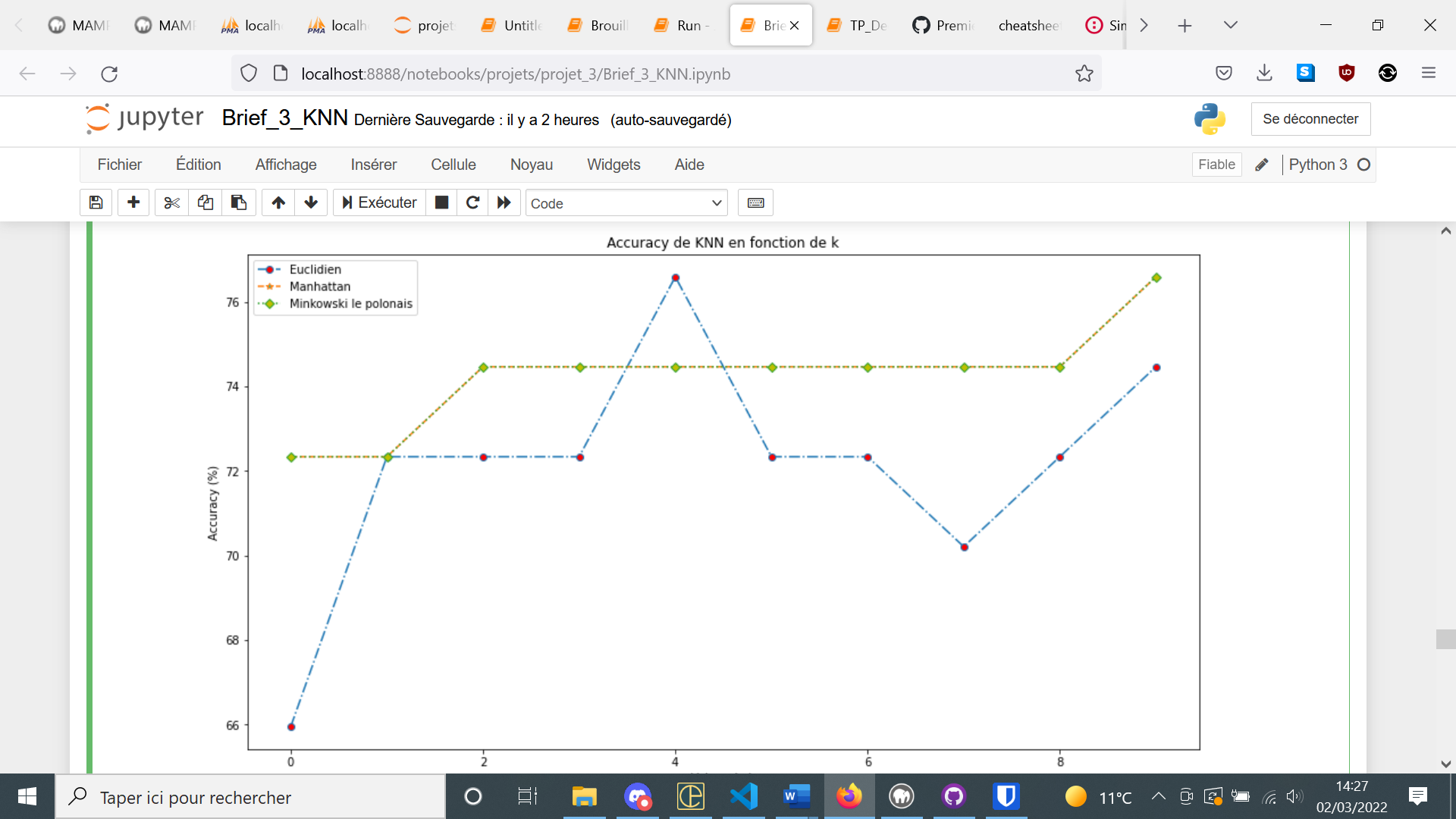
Pour connaître la précision de nos prédictions (c’est à dire, le nombre total de bonnes prédictions) on crée une fonction qui calcule l’accuracy en comparant les valeurs de la cible prédite (y prédit) et la cible réelle (y réelle).

Une image contenant texte

Description générée automatiquement

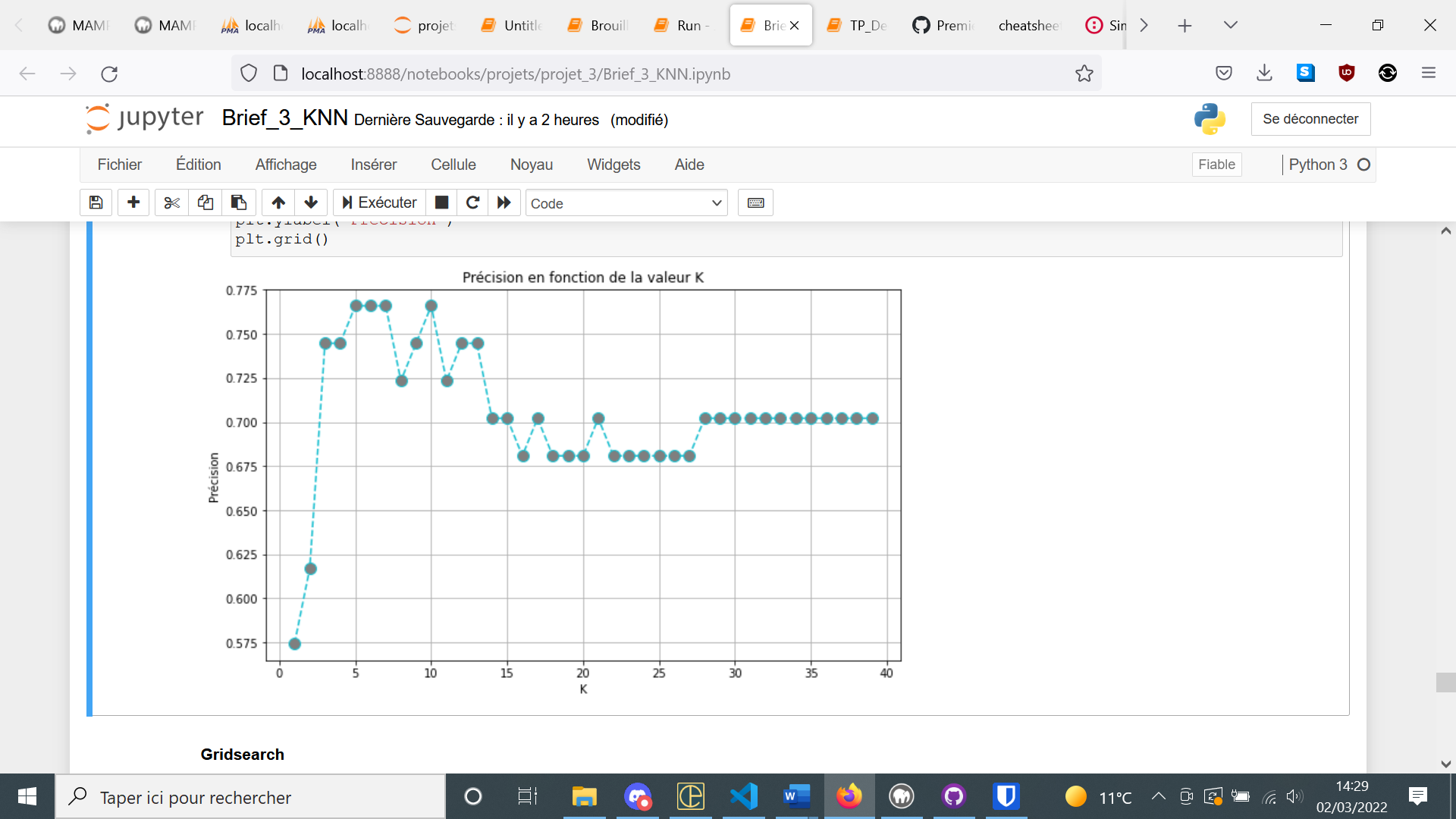
Courbe de précision du KNN en fonction de k

On réalise un graphique afin de visualiser les différents scores de précision en fonction du nombre de voisin k pour chaque type de distance.



KNN avec Scikit Learn

On utilise le modèle KNeighborsClassifier() de Scikit Learn puis on réalise une courbe de précision en fonction de k pour la méthode Minkowski



Réalisation d’une Gridsearch

On réalise une comparaison des hyperparamètres en faisant varier le nombre de k et les distances pour trouver les meilleurs paramètres qui prédisent les données avec une cross validation de 5 kfolds.

Matrice de confusion

Conclusion

Notre modèle est meilleur que celui de SK Learn.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Distance | K | Score |
| KNN from scratch | Euclidienne | 4 | 76,59% |
| Sk Learn | Manhattan | 10 | 70,21% |