Carlos Cavalcante

Collège de Bois de Boulogne

420-A52-BB

Approche supervisée

TP - Algorithme KNN

**Intoduction:**

En intelligence artificielle, la méthode des k plus proches voisins est une méthode d’apprentissage supervisé. En abrégé k-NN ou KNN, de l'anglais k-nearest neighbors.

Dans ce cadre, on dispose d’une base de données d'apprentissage constituée de N couples « entrée-sortie ». Pour estimer la sortie associée à une nouvelle entrée x, la méthode des k plus proches voisins consiste à prendre en compte (de façon identique) les k échantillons d'apprentissage dont l’entrée est la plus proche de la nouvelle entrée x, selon une distance à définir.

On importe les libraries necéssaires et le dataset.

**Vérifier si manque de donnée**

Pour faire le pre-traitement, on a trouvé les valeurs manquantes et lui rempli avec la moyenne de la colonne en utilisant la méthode update().

En fait, dans la colonne des Bare\_Nuclei, nous avons 16 valeurs manquantes, qui ont été renseignées avec la moyenne de cette colonne, qui dans ce cas correspond à la valeur arrondie 3.

**Matrice de Corrélation e Heatmap**

Après, on a dévélopé une Matrice de Correlation, pour savoir quelles sont l’intensité et aussi la relation avec les valeurs de X, laquelle montre le degré de relation entre une paire de variables. Elle est comprise entre -1 et +1. Ainsi, nous pouvons évaluer la force et la direction de ces variables.

J'ai utilisé la carte thermique en plus de la matrice numérique, ce qui facilite grandement la visualisation des relations et de leur force.

**Training and Test**

En suite, on fair la séparation train/test. On prends 80% du dataset pour procéder avec l’apprentissage et 20% restant, on utilize pour faire du test.

Si tel est le cas, cela indique clairement que le modèle choisi est le bon. À l'avenir, nous généraliserons ce qui confirmera le choix du modèle.

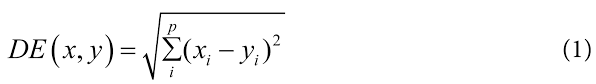
Pour la Seed, il s’agit de générer des nombres aléatoires dans un logiciel à l’aide d’un générateur de nombres aléatoires. C'est une simple fonction mathématique qui génère une suite de nombres suffisamment aléatoires pour la plupart des applications.

**Classifiation avec Euclidean distance**

Mesure de distance pour identifier la distance entre le point P1 et les autres points de mon jeu de données. De cette façon, nous pourrons savoir à quels points P1 est le plus proche, de cette manière nous aurons quelle classe elle est la plus similaire. Le "vote" sera donc terminé et nous saurons comment classer P1.

La distance euclidienne:

distance qui calcule la racine carrée de la somme des différences carrées entre les coordonnées de deux points:



Pour développer le modèle en laissant la valeur de K, j’ai préféré faire le calcul et j’ai touvé la valeur 22.

Pour l’exactitude (Accuracy) sur la base des valeurs test, j’ai trouvé: 0.962

Cela viens dire que la metrique est trés bonne et pourtant le modèle est fiable.

**Matrice de Confusion**

La matrice de confusion est une matrice qui mesure la qualité d'un système de classification.

Un des intérêts de la matrice de confusion est qu'elle montre rapidement si un système de classification parvient à classifier correctement.

Confusion Matrix:

array([

[90, 1],

[ 1, 48]

])

True Positive = 90

True Negative = 48

**Out-of-Sample**

Et la dernière étape pour évaluer le modèle et vérifier si l'algorithme fonctionne, nous utiliserons la méthode avec échantillon.

  Les données hors échantillon ne sont pas visibles et vous ne produisez que les prévisions / prévisions

J’ai aussi utlilisé pour prédire la cible les données X suivantes ([4,2,1,1,1,2,3,2,1]), et j’ai obtenu la valeur:

Les résultas sont:

(2,)

[2 2]

Que confirme la fiabilité du modèle.

Conclusion:

Le modèle perform bien en se basant en toutes les étapes suivies et une accuracy de 0.96 en training.