Travaux Pratiques
Apprentissage Automatique I

TP #5 – KNN & Naïve Bayes Classifications

Professeur

Abdelouahab Moussaoui

Partie 1 — Données Synthétiques

Exercice 1 —

Récupérer les jeux de données synth_train.txt et synth_test.txt. On a $Y \in \{1, 2\}$ et $X \in R^2$. On dispose de 100 données d'apprentissage et 200 données test.

Questions :

1. Charger le jeu de données d'apprentissage dans R (soit avec la commande **read.table**, soit avec l'outil d'importation de **Rstudio : Tools > Import Dataset**). Afficher les données d'apprentissage.
2. Afficher la dimension de l'ensemble d'apprentissage.
3. Afficher les 6 premiers enregistrements.
4. Représenter graphiquement les observations à l'aide de la fonction plot. On pourra colorier les points en fonction de leur classe à l'aide du paramètre col et modifier le symbole avec le paramètre **pch (=point character)** et rajouter une légende à l'aide de la fonction **legend**.
5. Appliquer la fonction **knn** du package class avec **k = 15** voisins pour prédire les points de coordonnées **(0,0)** et **(-2,2)**.
6. Prédire les données de l'ensemble d'apprentissage avec ce classifieur et comparer avec les vraies classes. Calculer le taux d'erreur empirique (nombre de fausses prédictions sur l'échantillon d'apprentissage divisé par taille de cet échantillon). Recommencer avec **k=10**, **k= 5**, **k=3** et **k = 1** voisins respectivement.
7. Représenter graphiquement la frontière de décision pour **k = 15** voisins : commencer par construire une grille de points, prédire ces points, puis ajouter ces points sur le graphique en les coloriant en fonction de leur prédiction. Recommencer avec **k = 1** voisin.
8. Charger le jeu de données de test dans R. Afficher les données test. Prédire les données de l'ensemble test avec **k = 15** voisins puis avec **k = 1** voisin. Calculer le taux d'erreur empirique dans les deux cas. Comparer avec les taux d'erreur des prédictions de l'ensemble d'apprentissage.
9. On choisit à partir de maintenant le classifieur **knn** avec **k = 15** voisins. Calculer le taux de vrai positifs (TVP) et le taux de vrai négatif (TVN) de ce classifieur sur l'échantillon test sachant que $TVP = \frac{VP}{VP + FN}$ et $TVN = \frac{VN}{VN + FP}$ et que ici **1=positif** et **2=négatif**.
10. On souhaite maintenant associer un coût 3 fois plus important aux faux négatifs. On utilise donc la matrice de coût $C = \begin{pmatrix} 0 & 3 \\ 1 & 0 \end{pmatrix}$ dans la règle de classification de Bayes.

    Les probabilités à posteriori **P(Y = 1|X = x)** d'une entrée x sont estimées par la fréquences de voisins de x appartenant à la classe 1. (a) Programmer une fonction **prob_knn** qui estime les probabilités **P(Y = 1|X = x)** d'un ensemble de données d'entrées.

    a. Programmer une fonction **prob_knn** qui estime les probabilités P(Y = 1|X = x) d'un ensemble de données d'entrées.

b. Appliquer cette fonction aux données de l'ensemble test. En déduire leur prédiction avec la règle de Bayes incluant les coûts. Vous pouvez utiliser la fonction I pour construire le code R.
c. Calculer le taux de vrai positifs (**TVP**) et le taux de vrai négatif (**TVN**).
d. Faire un petit bilan méthodologique de cet exercice.

Partie 2 — Données Réelles

Exercice 2 —

Récupérer le jeu de données **real_data.rda**. Les données concernent n = 1260 exploitations agricoles réparties en **K = 2** groupes : le groupe des exploitations saines et le groupe des exploitations défaillantes. On veut construire un score de détection du risque financier applicable aux exploitations agricoles. Pour chaque exploitation agricole on a mesuré une batterie de critères économiques et financiers et finalement **p = 4** ratios financiers ont étés retenus pour construire le score :
— **R2 :** capitaux propres / capitaux permanents,
— **R7 :** dette à long et moyen terme / produit brut,
— **R17 :** frais financiers / dette totale,
— **R32 :** (excédent brut d'exploitation - frais financiers) / produit brut.
La variable qualitative à expliquer est donc la variable difficulté de paiement (0=sain et 1=défaillant).

Questions :

1. Charger le jeu de données dans R avec la commande **load**.
2. Afficher la dimension de l'ensemble d'apprentissage.
3. Afficher les 6 premiers enregistrements.
4. On s'intéresse d'abord à la méthodologie du choix de k
   - **a.** Créer un jeu de données de données d'apprentissage de taille 945 (75% des données) et un jeu de données test de taille 315 (25% des données) avec le code suivant.
   - **b.** Calculer les taux d'erreur sur les données test pour **k** variant de 1 à 100. Avec la fonction plot, représenter ce taux d'erreur test en fonction de **k** (contrôler que l'abscisse du graphique part de 0). Avec la fonction **which.min,** trouver le nombre de voisins qui donne la plus petite erreur test.
   - **c.** Recommencer avec un autre découpage aléatoire apprentissage/test et représenter la courbe d'évolution du taux d'erreur test sur le même graphique qu'à la question précédente.
   - **d.** Exécuter le code suivant et faire un choix pour k.

```
B<- 20
kmax <- 100
err_test <- matrix(NA,kmax,B)

   for (b in 1:B)
     {
          tr <- sample(1:nrow(data),900)
          train <- data[tr,]
          test <- data[-tr,]
          for (k in 1:kmax)
            {
                  pred <- knn(train[,-1],test[,-1],train$DIFF,k)
                  err_test[k,b] <-
   sum(pred!=test$DIFF)/length(test$DIFF)
             }
        }

mean_err_test <- apply(err_test,1,mean)
lim <-c(0,max(err_test))
matplot(err_test,type="l",lty=2,col=2,ylim=lim, xlab="nombre de
voisins",ylab="taux d'erreur")
matpoints(mean_err_test,type="l",col=2,lwd=4)
legend("bottomright", legend=c("Erreur moyenne", "Erreurs
conditionnelles"),
lty=c(1,3),lwd=c(4,2),col=c(2,2))

which.min(mean_err_test)
```

e. Choisir maintenant le nombre **k** de voisin en utilisant par validation croisée (cross validation) **leave-one-out** (**LOO**) avec la fonction **knn.cv**.

f. Faire un petit bilan méthodologique concernant le choix du paramètre k.

5. On veut maintenant non seulement choisir k mais également avoir une idée de l'erreur de prédiction de ce classifieur. Pour cela, il faut utiliser des données n'ayant jamais été utilisées. Les données doivent donc être découpées en trois parties : apprentissage/validation/test

a. Couper aléatoirement les données deux parties : un ensemble "apprentissage-validation" de taille 945 (75 % des données) et un ensemble test de taille 315 (25% des données).

b. Utiliser la première approche pour choisir k sur l'ensemble "apprentissage-validation" :

i. Choisir k en découpant les 945 données de l'ensemble "apprentissage-validation" en deux parties : une partie "apprentissage" de taille 630 (50% des données) et une partie "validation" de taille 315 (25 % des données). Choisir k qui minimise le taux d'erreur moyen sur les ensembles de validations de B = 25 découpages.

ii. Construire le classifieur avec ce nombre de voisins sur l'ensemble "apprentissage-validation" et calculer le taux d'erreur des données test.

c. Utiliser la seconde approche pour choisir k par validation croisée LOO sur l'ensemble "apprentissage-validation". Calculer ensuite le taux d'erreur des données test.

Partie 3 — Mini-Projets

Exercice 1 — MNIST Multi-classes classification with KNN

Ici on veut classer les chiffres décimaux de la base MNIST en utilisant l'algorithme des KNN.

Questions :

1. Charger les données du *MNIST.*
2. Afficher les informations concernant les données.
3. Proposer un modèle de classification basé sur les KNN avec différentes valeurs de K.
4. Comparer ces deux modèles avec la régression logistique & les Arbres de décision.
5. Visualiser les résultats comparatifs graphiquement.
6. Conclusion

Exercice 2 — KNN Naïve Bayes and Logistic Regression

Ici on veut faire une classification en utilisant les données *"BankData.csv"* représentant les différents clients et leurs activités dans cette Bank.

Questions :

1. Charger les données du *"BankData.csv".*
2. Obtenir et afficher des informations à partir des données et nettoyer les données.
3. Visualisez les données.
4. Créer un modèle :
   - *KNN.*
   - *Bayésien Naïve.*
   - *Régression Logistique*
5. Comparer ces trois modèles.
6. Visualiser les résultats comparatifs graphiquement.
7. Conclusion

Exercice 3 — Binary classification with KNN Naïve Bayes and Logistic Regression

Fichier *"Social_Network_Ads.csv"* nous donne des informations sur les clients qui ont acheté / n'a pas acheté un produit particulier.

Questions :

2. Charger les données *"Social_Network_Ads.csv"*.
3. Obtenir et afficher des informations à partir des données et nettoyer les données.
4. Visualisez les données.
5. Créer un modèle :
   - KNN.
   - Bayésien Naïve.
   - Régression Logistique
6. Comparer ces trois modèles.
7. Visualiser les résultats comparatifs graphiquement.
8. Conclusion

Exercice 4 — Network Intrusion Detection with KNN Naïve Bayes and Logistic Regression

Dans ce projet, nous proposons de développer des modèles de classification à base de KNN, Naïve Bayes et la Régression Logistique pour la détection d'intrusion. Pour cela vous devez utiliser les données d'entrainement et de test dans le dossier nommé *"Data Intrusion"* où vous disposer de deux fichiers nommés respectivement "Train_data .csv" et "Test_data.csv" pour l'entrainement et le test de vos modèles respectivement.

- ***À propos de l'ensemble de données***

L'ensemble de données à auditer a été fourni et consiste en une grande variété d'intrusions simulées dans un environnement de réseau militaire. Il a créé un environnement pour acquérir des données de vidage TCP/IP brutes pour un réseau en simulant un LAN typique de l'US Air Force. Le réseau local était focalisé comme un environnement réel et soumis à de multiples attaques. Une connexion est une séquence de paquets TCP commençant et se terminant à une certaine durée entre laquelle les données circulent vers et depuis une adresse IP source vers une adresse IP cible sous un protocole bien défini. En outre, chaque connexion est étiquetée comme normale ou comme une attaque avec exactement un type d'attaque spécifique. Chaque enregistrement de connexion comprend environ 100 octets.

Pour chaque connexion TCP/IP, 41 caractéristiques quantitatives et qualitatives sont obtenues à partir des données normales et d'attaque (3 caractéristiques qualitatives et 38 caractéristiques quantitatives). La variable de **classe** a deux catégories :

- Normal
- Anormal

Questions :

9. Charger les données.
10. Obtenir et afficher des informations à partir des données et nettoyer les données.
11. Visualisez les données.
12. Créer un modèle :
  - *KNN.*
  - *Bayésien Naïve.*
  - *Régression Logistique*

13. Comparer ces trois modèles.
14. Visualiser les résultats comparatifs graphiquement.
15. Conclusion