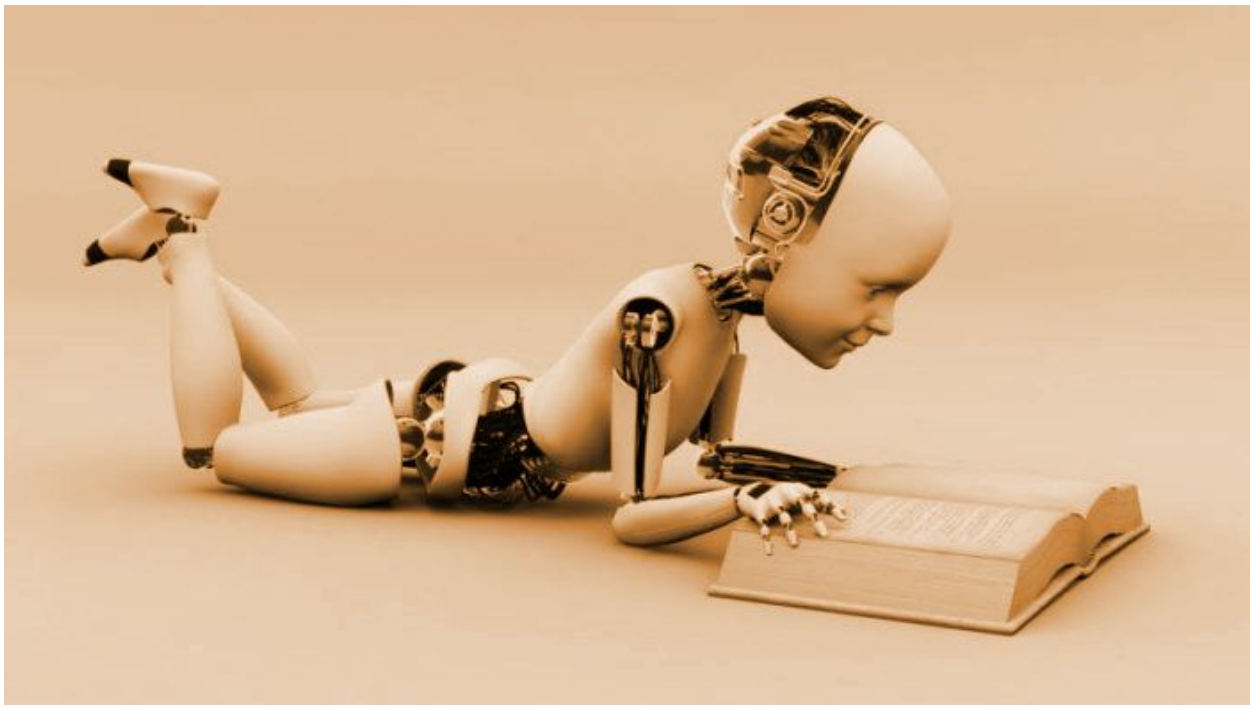
M2/INF-E091.4 Projet
PAUGAM-MOISY
DATE DE DÉBUT : MERCREDI 16-11-2016 | DATE DE FIN : LUNDI 19-12-16

CAHIER D'EXPÉRIENCE DE REMY JAMASA

CLASSIFICATION UTILISANT L'ALGORITHME DES K PLUS PROCHE VOISINS



WEDNESDAY 16-11-2016 - « DÉBUT DU PROJET »

OBJECTIF DU PROJET

Dans le cadre du projet de master de deuxième année de cette année 2016, il nous a été proposé de jouer les apprentis détectives ; le but étant de classifier des morceaux de

verre de sort à déterminer la classe des bouts de verres issue d'une scène de crime pour fournir une aide à la police scientifique.

Réaliser une classification efficace de morceaux de verre recueillis sur des scènes de crime.

RÉCUPÉRATION DU JEU DE DONNÉES

Téléchargement du jeu de données sûr.

- https://archive.ics.uci.edu/ml/datasets/Glass+Identification

CHOIX DES MÉTHODES DE CLASSIFICATIONS

Pour sélectionner des classifieur nous avons recensé les classifieurs que nous connaissions :

- K-Means (méthode supervisée)
- Reseau de neurones
- **K-Plus Proches Voisins (K-PPV) (méthode non supervisée)**
- MLP

REMARQUES DE L'ENSEIGNANTE

- *Les min et max de chaque composante peut être utile (Summary Statistics)*
- *Le fait qu'il y a des valeurs à 0 ou proche de 0 pourrait impacter la classification avec les réseaux de neurones. Parce que lors d'une somme la plus petite valeur sera négligée. Ce problème n'existe pas dans un classifieur K-PPV.*
- *Ne pas tester tous les classifieurs, car nous n'avons pas le temps ; se focaliser sur deux dans un premier temps.*
- *Si possible éviter les réseaux de neurones, car il y a trop de réglages à faire.*
- *Faire attention à l'étiquette dans le jeu donnés, car il n'est pas un exemple.*

OBJECTIFS DU DONNÉS

1. Centrer et réduire les données
2. Découper la base en deux sous-base *ex: glass-ap et glass-gen*
3. Appliquer la méthode des K-PPV dans deux situations
    a. Sur les données brutes
    b. Sur les données centrées et réduits

    Comparer et expliquer les résultats obtenus.

PRÉPARATION DES DONNÉES

Pour réduire et centrer les données, j'ai opté pour l'utilisation d'un tableur, en l'occurrence, le tableur de Google.

Pour cela, je me suis basé sur les formules suivantes :

Centré	Réduire
$x_i - u$	$\frac{x_i - u}{\gamma}$
u *étant la moyenne et* γ *l'écart type* (*soit* SD)	

https://fr.wikipedia.org/wiki/Variable_centr%C3%A9e_r%C3%A9duite

SUMMARY STATISTICS

	Attribute	Min	Max	Mean	SD	Correlation with class
2	RI	1.5112	1.5339	1.5184	0.003	-0.1642
3	Na	10.73	17.38	13.4079	0.8166	0.503
4	Mg	0	4.49	2.6845	1.4424	-0.7447
5	Al	0.29	3.5	1.4449	0.4993	0.5988
6	Si	69.81	75.41	72.6509	0.7745	0.1515
7	K	0	6.21	0.4971	0.6522	-0.01
8	Ca	5.43	16.19	8.957	1.4232	0.0007
9	Ba	0	3.15	0.175	0.4972	0.5751
10	Fe	0	0.51	0.057	0.0974	-0.1879

BASES DE GÉNÉRALISATION ET APPRENTISSAGE

Après avoir réduit et centré les données, je me suis lancé dans l'implémentation d'un bout de code qui se chargerait de la scission des données en deux bases, une pour la généralisation et une autre pour l'apprentissage.

LES RÉSULTATS DE MA RECHERCHE

J'ai donc délaissé l'étape de la création des sous-bases pour me concentrer sur la recherche d'un algorithme pour me guider dans la mise en place du classifieur KPPV. Ci-après les résultats de ma recherche :

1. https://fr.wikipedia.org/wiki/Recherche_des_plus_proches_voisins
2. http://machinelearningmastery.com/tutorial-to-implement-k-nearest-neighbors-in-python-from-scratch/
3. http://www.saedsayad.com/k_nearest_neighbors.htm
4. https://www.analyticsvidhya.com/blog/2014/10/introduction-k-neighbours-algorithm-clustering/
5. http://scikit-learn.org/stable/modules/neighbors.html
6. 1.6. Nearest Neighbors — scikit-learn 0.18.1 documentation
7. https://www.kaggle.com/c/street-view-getting-started-with-julia/details/knn-tutorial
8. https://datasciencelab.wordpress.com/2013/12/12/clustering-with-k-means-in-python/

FRIDAY 18-11-2016

Parmi les huit résultats que j'ai gardés, je pense utiliser le numéro deux, car il était implémenté en Python et il avait pour objectif d'implémenter le code de zéro.

Aujourd'hui, j'ai continué l'implémentation de la fonction de création des bases de généralisation et d'apprentissage.

SATURDAY 19-11-2016

J’ai terminé la fonction qui s’occupe de l'importation du fichier CSV et qui crée les bases d'apprentissage et de généralisation. Et cela en fonction du pourcentage qui lui est passé en passé en paramètre.

TUESDAY 22-11-2016

J’ai finalement dû abandonner ma fonction qui crée les sous-bases, car un étudiant (Volet Djiny) avait bien avancé sur cette partie et ses bases étaient figées. Il nous a été préconisé d’utiliser sa production car cela permettrait à l'équipe de travailler sur les mêmes bases. Nous avons donc utilisé les bases qu’il a produites.

DÉBUT DE L’IMPLÉMENTATION DES K-PPV.

Aujourd’hui j’ai pris lecture de l'implémentation de l'algorithme des K Plus Proches Voisins que j'avais sélectionné précédemment le vendredi 18.

J’ai officiellement commencé l’implémentation des l'algorithme des K Plus Proches Voisins aujourd’hui.

MONDAY 28-11-16

L'implémentation du classifieur K Plus Proche Voisin' est terminé, commenté et l'adapter à mes besoins.

L'importation des données est faite, l'implémentation du calcul de distance Euclidienne est faite.

MES AJOUTS FAITS DANS LE CODE

Les fonctions de calcul de la précision du classifieur est fait ainsi que celui de la précision pour chaque classe.
Ajout de l'option permettant de passer K lors du lancement du programme en ligne de commande. *Si rien n'est passé en paramètre alors par défaut k sera égale à 3*.

Le long de l'implémentation du classifieur, j'ai récupéré les résultats / précisions suivants :

- pour k = 3 une précision de 67.6923076923077%
- pour k = 1 une précision de 66.15384615384615%
- pour k = 2 une précision de 67.6923076923077%
- pour k = 1 une précision de 66.15384615384615%
- pour k = 3 une précision de 67.6923076923077%
- pour k = 5 une précision de 67.6923076923077%
- pour k = 10 une précision de 67.6923076923077%
- pour k = 50 une précision de 53.84615384615385%
- pour k = 70 une précision de 33.84615384615385%
- pour k = 100 une précision de 29.230769230769234%

REMARQUES

- *Dans le jeu de données "train" il n'y a pas d'instance de la classe 4.*
- *J'ai remarqué qu'il n'y avait aucune instance de la classe 4 dans lors de la classification fait par mon algorithme; j'ai donc refait le compte des d'instances pour chaque classe:*
    - *51 éléments de la classe 1*
    - *52 éléments de la classe 2*
    - *12 éléments de la classe 3*
    - *00 éléments de la classe 4*
    - *08 éléments de la classe 5*
    - *06 éléments de la classe 6*
    - *20 éléments de la classe 7*

  pour un total 149 éléments parmi les 149 éléments présents dans la base.

  Tout semble être en order.

MERCREDI 30-11-16

ABSENCE D'INSTANCE DE LA CLASSE 3

Lors de quelques tentatives de classification, l'enseignante m'a fait remarquer que les instances de la classe 3 n'étaient pas représentées; c'est certainement dû à une erreur dans mon programme, elle me fit remarquer.

Je me suis donc lancé dans la recherche de la cause de l'absence d'instance de la classe trois dans les pourcentages de prédiction par classe.

RÉPONSE PLAUSIBLE

Il s'avère que le classifieur ne parvient pas à classer les instances de la classe trois correctement. Et étant donné que mon programme ne tient compte que des prédictions correctes, il en résulte que la classe trois se retrouve avec un pourcentage de 0.

ALLER PLUS LOIN

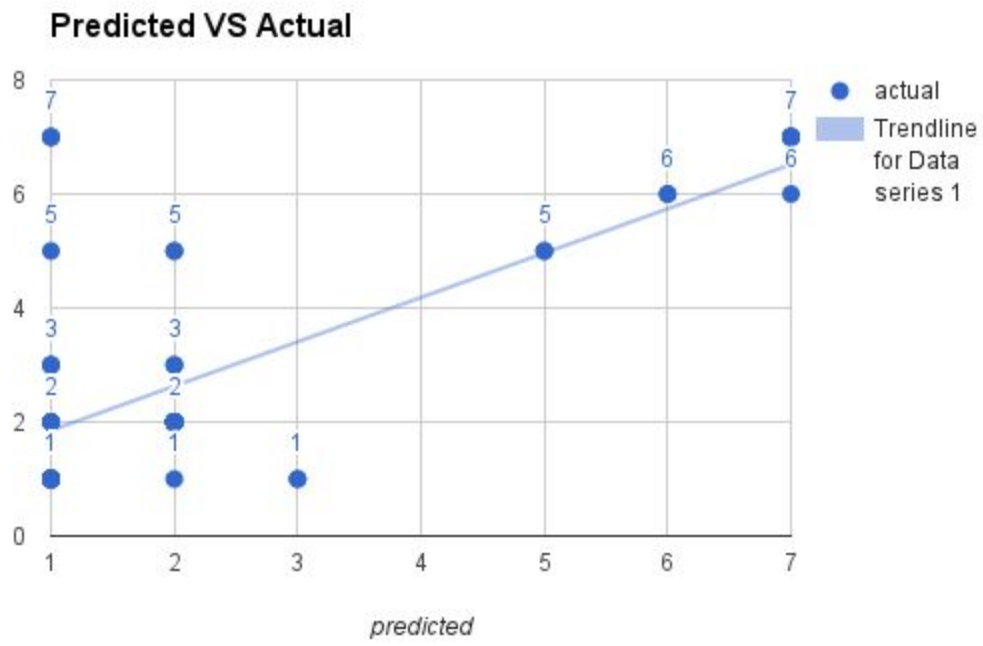
COURBES DES CLASSES PRÉDITES ET DES CLASSES RÉELLES

Pour avoir un meilleur aperçu des prédictions, ma collaboratrice (Johannie Basile) et moi même avons décidé de dessiner des graphes en utilisant les résultats des prédictions obtenus et les classes réelles.

Nos attentent : Les bonnes prédictions devrait se trouver sur une droite affine et linéairement alignées.

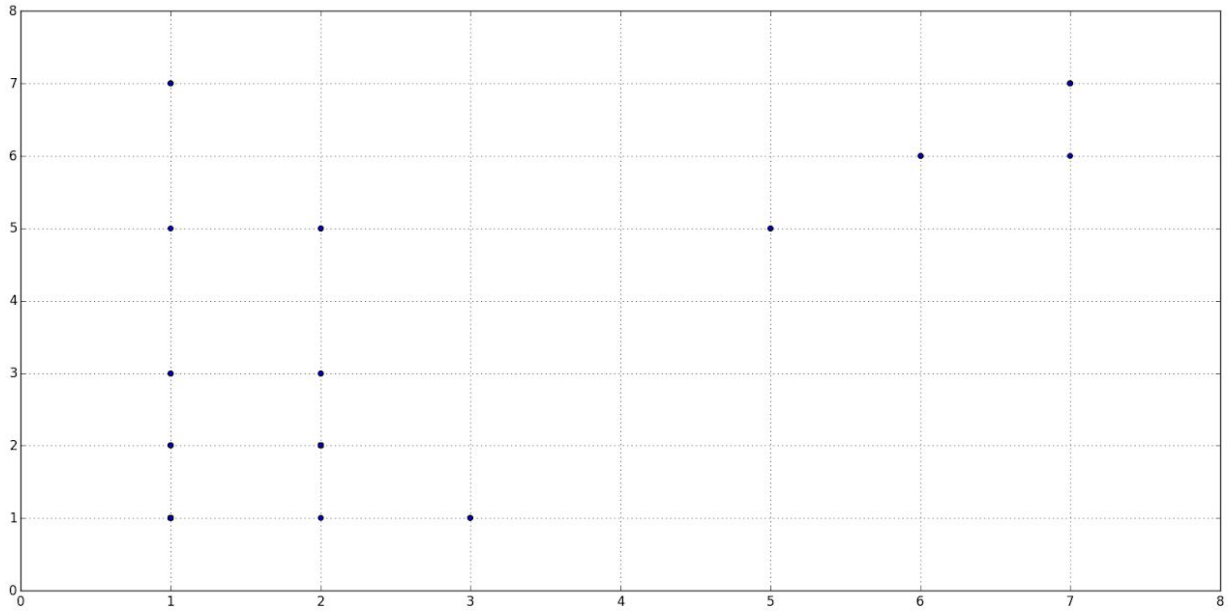
Pour produire nos graphes nous avons voulu utiliser un « tableur » mais nous avons éprouvé des difficultés pour placer les données sur les bons axes; le problème était présent sur le classeur de « LibreOffice » mais aussi celui de « Google » et « MicroSoft » .

MAJ 19-12-16 : J'ai finalement réussi à produire la courbe avec le tableur de Google.



Prédiction VS Réelles : Google

J'ai donc décidé de générer les graphes directement avec « Python » alors que ma collègue a choisi de l'option « MatLab » même si nous avons pu avoir les courbes que nous recherchons, car recherche a été la plus fructueuse, car elle a réussi à générer la matrice de confusion des prédictions ce qui fut beaucoup plus utile et exploitable. Comme sur les courbes nous nous sommes aperçus que la plupart des points (représentation des prédictions) se chevauchent ce qui rend la lecture plutôt confuse et inexploitable.



Prédiction VS Réelles : Python

Suivant les conseils de l'enseignante, nous envisageons l'utilisation d'outils tel que « Geogebra » et « Wolframalpha » mais ayant passé énormément de temps sur la génération des graphes.

Mais nous avons préféré remettre cela à plus à une autre fois pour de nouveau nous concentrer sur le vrai problème, c'est-à-dire la résolution de prédiction d'instances de la classe 3. Pour pallier ce problème, nous pensons déplacer des instances de la classe trois de la base « d'apprentissage » vers la base de « généralisation ».

MONDAY 5-12-16

Pour faciliter l'exploitation ultérieure des résultats, j'ai ajouté la possibilité de sauvegarder les résultats du classifieur sur le disque dur.

Ajout de la méthode de calcul de la distance Manhattan.

Après quelques tentatives, je me suis rendu compte qu'avec la distance de Manhattan la précision du classifieur a baissée.

Il faudrait que je fasse un peu plus de tests en utilisant cette méthode de calcul.

Ajout de l'acceptation de flages permettant la customisation du classifieur; lancer le programme avec l'option « -h » pour accéder à la liste des options disponibles.

FRIDAY 9-12-16

Aujourd'hui j'ai commencé la rédaction du rapport.

L'introduction est terminé, présentation du jeu de donné faite et explication de la procédure de pré traitements de données commencé.

WEDNESDAY 14-12-16

Ajout du nombre d'instances rencontrées lors du calcul des précisions par classe ; maintenant, nous avons une idée précise des nombres d'instance de chaque classe qui a été traitée.

Récupération des résultats pour tirer la rédaction du rapport:

- Courbes des résultats pour différentes valeurs de K pour les deux types de mesures.
    - Pour K allant de 1 à 20 on obtient les meilleurs résultats en utilisant la mesures de « Euclidienne ».

MONDAY 19-11-2016 « RETOUR SUR EXPÉRIENCE ET FIN »

S'il fallait refaire cette expérience, j'aurais voulu

- regarder les courbes de pertes pour chaque classe celle, je pense permettrait d'avoir un meilleur aperçu des classes qui se classent mal.
- je pense que le classifieur apprend mal ; je regrette de ne pas avoir continué sur ma lancée et utilisé un jeu d'apprentissage plus conséquent. Cela aurait certainement amélioré la phase d'apprentissage du classifieur.
- explorer la version Euclidienne avec un K plus grand pour confirmer ou infirmer la tendance observée.
- essayer d'autre type de mesure de distance tel que :
    - Chebyshev
    - Minkowski
    - Canberra
    - cordes carrées
    - Khi-carré
    - Mahalanobis
- essayer d'autre classifieur le « K-moyennes » par exemple.
- j’aurais du appliquer la méthode des K-PPV sur les données brutes comme sur les données centrées et réduits puis comparer les résultats obtenus.