SY 32 (Printemps 2018) - Partie 2
TD 01 : Apprentissage supervisé

1 Classification de sites Internet

On souhaite construire un classifieur permettant de savoir si un site Internet est un site d'hameçonnage (phishing) utilisé par des fraudeurs dans le but d'obtenir des renseignements personnels. Rami et al. [1] proposent de représenter un site Internet par un vecteur de trente caractéristiques (voir Tab. 1). Ces caractéristiques sont représentées par des valeurs -1, 0 ou 1 selon la présence ou la gravité de ces dernières.

Le fichier `phishing_train.data` contient les caractéristiques de 5,000 sites Internet. Chaque ligne du fichier correspond au vecteur des caractéristiques d'un site Internet. Le fichier `phishing_train.label` contient la classe de chacun de ces sites, -1 : non-phishing, +1 : phishing. Le fichier `phishing_test.data` contient les caractéristiques de 6,055 sites Internet dont on souhaiterait connaitre la nature.

TABLE 1 – Attributs d'un site Internet.

#	attributs	valeurs	#	attributs	valeurs
1	having_IP_address	-1, 1	16	SFH	-1, 0, 1
2	URL_length	-1, 0, 1	17	submitting_to_email	-1, 1
3	shortining_service	-1, 1	18	abnormal_URL	-1, 1
4	having_at_symbol	-1, 1	19	redirect	0, 1
5	double_slash_redirecting	-1, 1	20	on_mouseover	-1, 1
6	prefix_suffix	-1, 1	21	right_click	-1, 1
7	having_sub_domain	-1, 0, 1	22	pop_up_window	-1, 1
8	SSL_final_state	-1, 0, 1	23	iframe	-1, 1
9	domain_registration_length	-1, 1	24	age_of_domain	-1, 1
10	favicon	-1, 1	25	DNS_record	-1, 1
11	port	-1, 1	26	web_traffic	-1, 0, 1
12	HTTPS_token	-1, 1	27	page_rank	-1, 1
13	request_URL	-1, 1	28	google_index	-1, 1
14	URL_of_anchor	-1, 0, 1	29	links_pointing_to_page	-1, 0, 1
15	links_in_tags	-1, 0, 1	30	statistical_report	-1, 1

2 Support vector machines (SVM)

On souhaite commencer par construire un premier modèle linéaire basé sur un classifieur SVM. Pour cela, on utilisera la bibliothèque `scikit-learn` :

```
from sklearn import svm
```

Q1. Charger le fichier `phishing_train.data` dans une matrice `xa` et le fichier `phishing_train.label` dans un vecteur `ya`.

Q2. Initialiser un classifieur SVM linéaire :

```
clf = svm.LinearSVC()
```

Q3. Entrainer le classifieur avec les données `xa` et `ya` à l'aide de la fonction `clf.fit`.

Q4. Prédire avec la fonction `clf.predict` les classes des données `xa` et calculer le taux d'erreurs sur les données apprentissages.

Q5. Charger les données de test depuis le fichier `phishing_test.data` et prédire la catégorie de chacun des sites. Calculer à nouveau le taux d'erreurs en utilisant le fichier `phishing_test.label`. Comparer les résultats.

Q6. Évaluer à nouveau le classifieur sur les données d'apprentissage mais avec une validation croisée à cinq couches. Comparer le résultat à ceux calculés précédemment.

Q7. Par défaut, le paramètre C du classifieur SVM vaut 1.0. Utiliser la validation croisée pour tester différentes valeurs de C, on pourra, par exemple, commencer par les valeurs $10^{-5}, 10^{-3}, \ldots, 10^{3}$ puis raffiner la recherche d'une valeur de C optimale. On prendra garde au temps de calcul lorsque C est grand.

Bonus : Tester des SVM à noyaux non-linéaires en utilisant la fonction clf = svm.SVC(kernel=...), on fera attention aux différents paramètres entrant en jeu.

3 Forêt aléatoire et AdaBoost

On souhaite maintenant construire deux autres modèles : forêt aléatoire et AdaBoost.

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import AdaBoostClassifier
```

Q8. Reprendre les mêmes étapes qu'à la section précédente. On testera les classifieurs avec différents paramètres.

4 Pour aller plus loin

Tester d'autres algorithmes : Plus proches voisins (Nearest Neighbors), Analyse discriminante (Discriminant Analysis), Arbre de décision (Decision tree), etc.

Scikit-learn : `http://scikit-learn.org/stable/supervised_learning.html#supervised-learning`.

Références

[1] Rami M. Mohammad, Fadi Thabtah, and Lee McCluskey. Predicting phishing websites based on self-structuring neural network. *Neural Computing and Applications*, 25(2) :443–458, 2014.