

Equipe MARINE :

DIALLO MAMADOU ADAMA

Sara Hadj-ali

Abdelhamid KEBIR

Adel Rebbache

Thierno Saidou DIALLO

Glody TAMINA MAKWAYA



L2 Informatique S4

Mini Projet

Rapport de projet

Chargé de cours :

Isabelle GUYON

URL du challenge : https://competitions.codalab.org/competitions/15518?secret_key=3af684e6-15c0-466e-ad18-039e884a719c

URL de la video : <https://www.youtube.com/watch?v=PoViqZ5EJcs&feature=youtu.be>

URL du GitHub : <https://github.com/MarineGroupe/MarineGroupe>

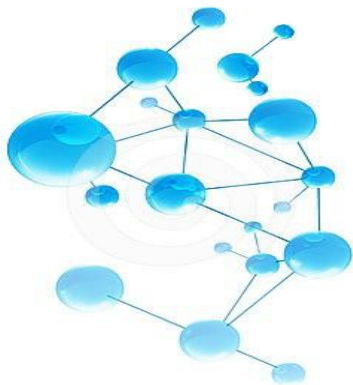
Codalab login : **marine**

ID Codalab (public submission) : **458**

User : **marine**

Score (BAC) : **0,4492**

Activity molecule against HIV infection



Plan du rapport :

-Introduction

I. Objectif du challenge

II. Description des dataset

III. Lecture et représentation graphique

-Variance des features

IV. Construction d'un modèle prédictif

-Pourquoi avoir choisi cette méthode

V. Résultats et discussions

-Problème

-Résultat 1

-Resultatz

-Résultat 3

VI. Problèmes rencontrés lors du projet , et conseils pour les etudians de l'année prochaine

-Conclusion

VII. Algorithmes et références

VIII. Annexes

XI ; References et bibliographie

Introduction :

Le VIH est l'un des principaux problèmes de santé dans le monde. Selon un récent rapport de l'UNSAIDS, on estime que plus de 34 millions de personnes vivent avec une infection de type VIH-1 dans le monde et 2,5 millions de nouvelles infections VIH sont détectées chaque année, dont 1,8 million d'enfants et 1,1 million de personnes mortes de maladies liées au VIH .

Actuellement, 14,8 millions de personnes sont admissibles au traitement contre le VIH, mais seulement 8 millions de personnes sont sous traitement en raison de diverses raisons qui incluent des questions économiques coûteuses , notamment a cause de la méthode HTS .

HTS est une méthode d'expérimentation scientifique utilisée dans la découverte de médicaments, reliant les domaines de la biologie et de la chimie. Cette méthode est coûteuse , C'est pourquoi l'application des méthodes d'apprentissage par machine learning pourrait être une solution , vu que cette dernière , a déjà été appliqué aux problèmes bio-informatiques avec des résultats satisfaisant a la clé .



I) Objectifs du challenge :

L'objectif est de prédire quels composants sont actifs contre le VIH , pour cela , nous allons utiliser un ensemble de données ;

L'ensemble de données est une classification binaire comportant deux classes: active ou inactive

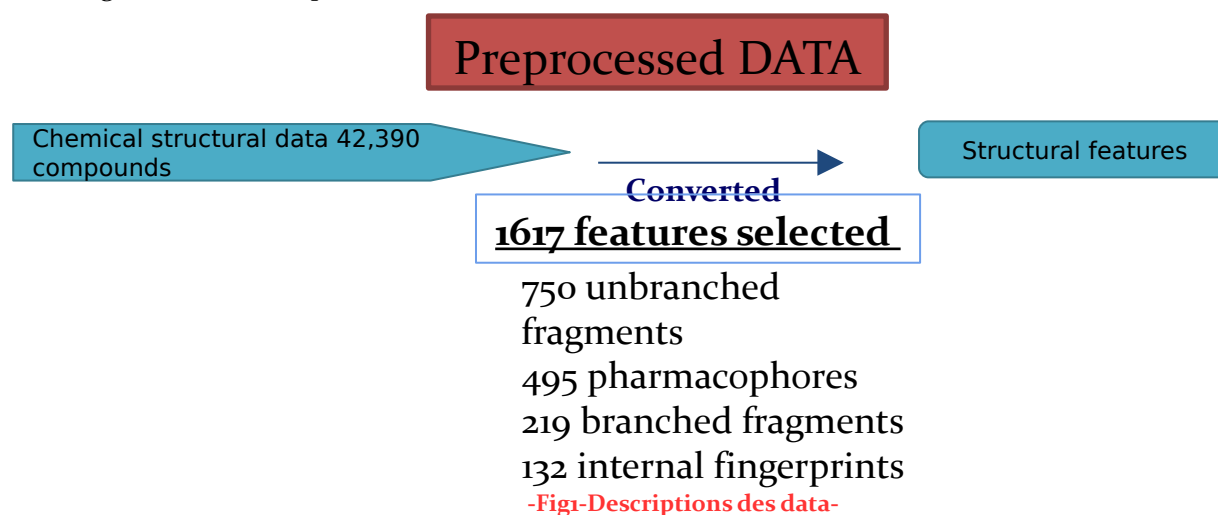
Les variables représentent les propriétés de la molécule déduite de sa structure.

Le problème est donc de relier la structure à l'activité (un QSAR = structure quantitative - problème de relation d'activité) à l'écran de nouveaux composés avant de les tester (un HTS = High Throughput Screening problème). A noter que Les modèles QSAR sont des modèles de régression ou de classification ces derniers ,relient les variables prédictives à une valeur catégorielle de la variable réponse .

II) Description des dataSets :

Ce défi repose sur l'ensemble de données HIVA fournis par les étudiants de master sur deux formats différents , un ou Les données sont pré traitées dans une représentation de caractéristiques aussi proche que possible des données brutes , un au autre ou les données sont dans leurs format d'origine . Ainsi , notre première tâche sera

de charger , convertir et pré traiter ces données en format csv .



Les données brutes fournis contiennent : un bloc d'en-tête (nom de la molécule, dimension, nom du programme, date ...) , Une table de connexion ,Un bloc de données (atome et blocs liés) et Une ligne de terminaison. Vous trouverez ci dessous , un exemple :

Dataset

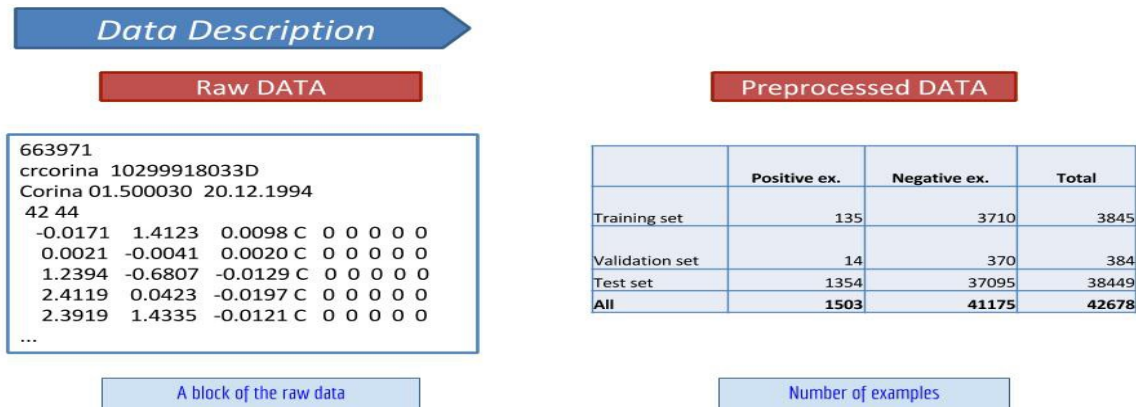
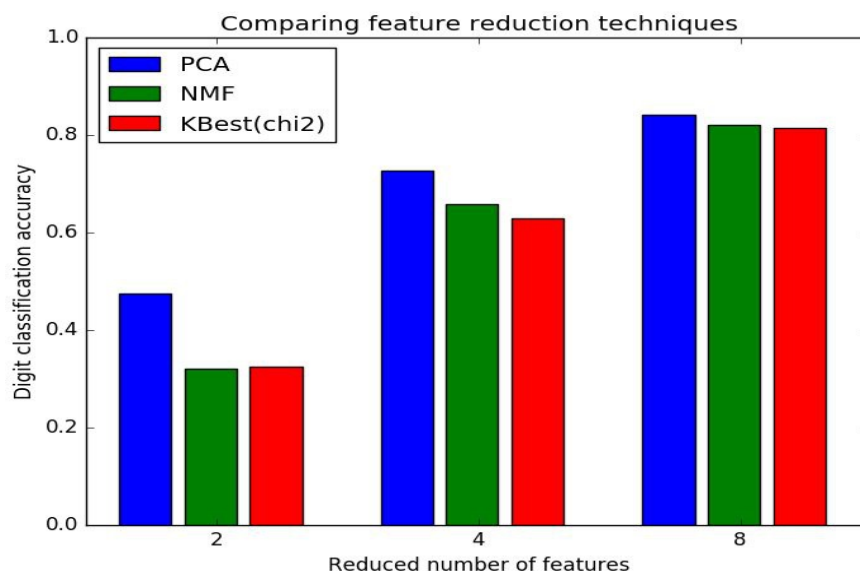


Table 1 : Statistics of the data [3]:

III)Lecture et représentation graphique :

Nous nous sommes intéressés à la représentation graphique des données et des résultats en utilisant Pipeline , Rappelons que Pipeline est basé sur des transformations avec un estimateur final. Appliquer séquentiellement une liste de transformées et un estimateur final. Les étapes intermédiaires du pipeline doivent être des «transformations», c'est-à-dire qu'elles doivent mettre en œuvre des méthodes d'ajustement et de transformation. L'estimateur final ne doit mettre en œuvre que l'ajustement.

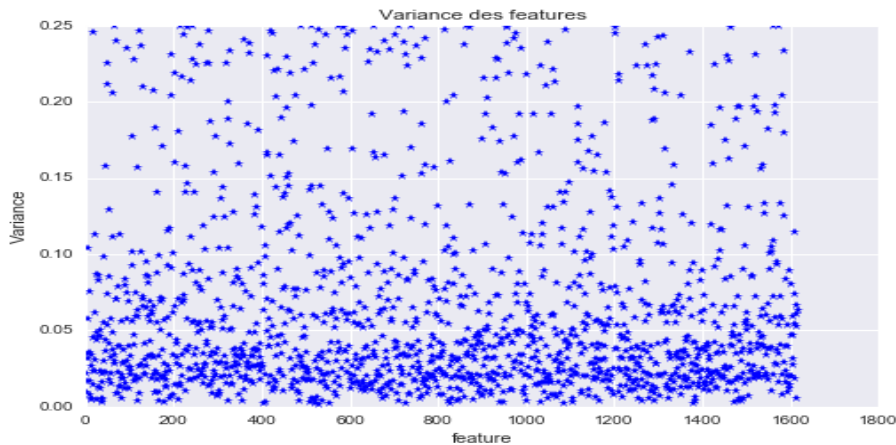
Le but du pipeline est d'assembler plusieurs étapes qui peuvent être validées en croix ensemble en fixant des paramètres différents. Pour cela, il permet de paramétrer les différentes étapes en utilisant leurs noms et le nom du paramètre séparé par un '_'. Cet exemple construit un pipeline qui effectue la réduction de dimensionnalité suivie d'une prédiction avec un classificateur de vecteur de support



-Fig2 : Diagramme de reduction de features[2]-

Variance des features : Grace a l'execution de Readme , nous avons tracer le graphe des variances de features , pour voir si il existe des features a tres basse variance (sans importance) , pour qu'on puisse reduire la dimension de chaque ligne .

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---------|----------|----------|----------|----------|----------|----------|----------|----------|----------|
| Feature | 1353 | 764 | 1049 | 97 | 653 | 1214 | 759 | 216 | 1115 |
| Score | 0.013403 | 0.011273 | 0.006161 | 0.005701 | 0.005559 | 0.005557 | 0.005552 | 0.005233 | 0.005189 |



-Fig3: Variance des features avant et apres la reduction[3]

IV) Construction d'un modele predictif :

Afin de voir à quel point un modèle marche mieux , nous avons décidé de choisir la métrique: BAC (Balanced Précision), qui est utilisé pour les problèmes de classification. Techniquement parlant, la précision Est la précision moyenne de chaque classe.

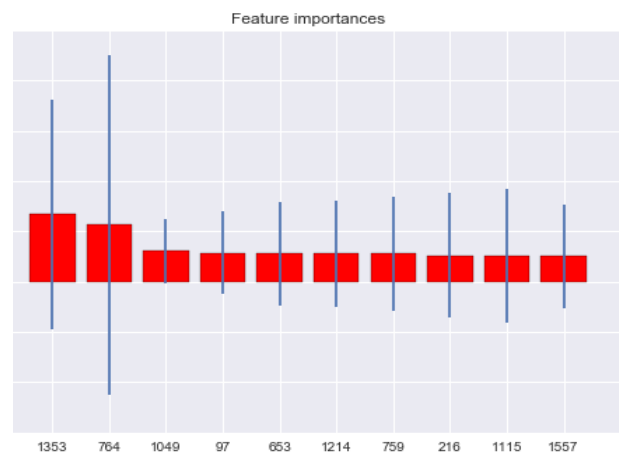
Metric

Balanced accuracy

| | | predicted labels (made by the classifier) | |
|--|-------|--|-------|
| | | face | place |
| true labels (given in the testing data) | face | 9 | 1 |
| | place | 2 | 7 |

$$\text{regular ("overall") accuracy} = \frac{9+7}{9+1+2+7} = 0.842$$

$$\text{balanced accuracy} = \left[\frac{9}{9+1} + \frac{7}{2+7} \right] / 2 = 0.839$$



-Fig4 :Predictions a l'aide de la metrique [3]-

-Fig5:Barres d'erreur features[1]-

Pourquoi avoir choisi cette methode en particulier ? :

Le principal avantage d'utiliser cette métrique au lieu de calculer seulement la précision, c'est qu'il peut traiter des données déséquilibrées.

La figure 3 nous montre que cette métrique Donne des résultats similaires lorsque les données sont équilibrées. Néanmoins, lorsque les données ne le sont pas (telles que Hiva, comme on peut le voir sur le tableau 1), la précision régulière continuera à donner de bonnes notes si L'algorithme se préforme bien sur la plupart des importations des échantillons. Cependant, nous ne savons pas quoi faire.

V) Resultats et discussions :

Probleme : L'un de nos principaux problèmes était de choisir le bon classifieur , pour qu'il soit adéquat avec notre type de dataset , nous avons le choix entre 5 classifieur de la baseline sklearn , et utiliser la Z-normalization . Nous avons donc fais un test automatique pour voir quelle classifieur nous convenais. La table 2 nous illustre les différents classifieurs utilisés , et leur BAC associé (sans normalization) .

Resultat 1 :

Nous avons traverser la matrice de confusion des différentes méthodes sans normalisation.

Quelques algorithmique comme par exemple Logistic régression, ne parvient pas à prédire un exemple positif, alors nous avons obtenu une accadance équilibrée égale à 0,5.

Random forest avec 250 estimateurs nous donne les meilleurs résultats. Cependant , nous pensons que nous pouvons obtenir un meilleur résultat avec SVM en optimisant les paramètres hyper

Results without normalization

Logistic regression

BAC : 0.5

| | | Predicted label | | |
|------------|----|-----------------|---|----------------|
| | | -1 | 1 | |
| True label | -1 | 37094 | 0 | Total 37094 |
| | 1 | 1354 | 0 | |
| Total | | 38448 | 0 | 1354 |

Adaboost

BAC : 0.59

| | | Predicted label | | |
|------------|----|-----------------|-----|----------------|
| | | -1 | 1 | |
| True label | -1 | 36867 | 227 | Total 37094 |
| | 1 | 1113 | 241 | |
| Total | | 37980 | 468 | 1354 |

Random forest

BAC : 0.61

| | | Predicted label | | |
|------------|----|-----------------|-----|----------------|
| | | -1 | 1 | |
| True label | -1 | 36913 | 181 | Total 37094 |
| | 1 | 1042 | 312 | |
| Total | | 37955 | 493 | 1354 |

SVM

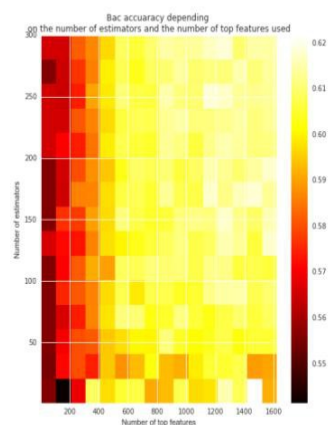
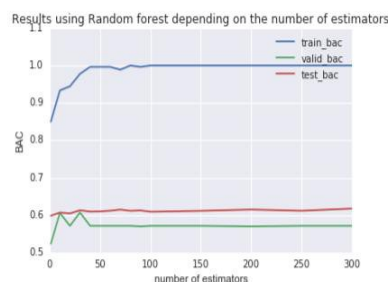
BAC : 0.51

| | | Predicted label | | |
|------------|----|-----------------|----|----------------|
| | | -1 | 1 | |
| True label | -1 | 37087 | 7 | Total 37094 |
| | 1 | 1130 | 24 | |
| Total | | 38217 | 31 | 1154 |

Table 2 : Resultats des differents classifieurs testés [3]

Resultat 2: Nous avons voulu comparer les resultat (plusieurs BAC) en fonction du nombre d'estimateur , nous avons donc obtenu 3 courbes , illustrant notre précédent résultat , en comparant les BAC , le but étant de voir avec combien d'estimateurs en utilisant random forest , nous avons un meilleure BAC , et nous remarquons d'après la courbe de la figure 6 qu'avec 250 estimateurs , nous avons de bons résultats. De plus , nous avons aussi voulu voir l'auccurance des BAC en fonction des top features cette fois , pour etre sure du nombre d'estimateur obtenu precedement , et le résultat est le meme , toujours de bons résultat avec environs 250 estimateurs

Results without normalization



-Fig6-: Graphe et diagramme BAC en fonction des nombres d'estimateurs

Resultat 3 :

Nous avons appliqué la normalisation Z, en étant sûre d'avoir considéré la balanced accuracy, en organisant différentes architectures, nous avons déduit qu'en utilisant : **size hidden layers [300,300]**, **activation function = "relu"** nous avons de meilleurs résultats avec une BAC = 0,65

Results after preprocessing

| | | Predicted label | |
|------------|----|-----------------|------|
| | | -1 | 1 |
| True label | -1 | 34964 | 2130 |
| | 1 | 913 | 441 |
| Total | | 35877 | 2571 |

BAC : 0.63

| | | Predicted label | |
|------------|----|-----------------|-----|
| | | -1 | 1 |
| True label | -1 | 36628 | 466 |
| | 1 | 924 | 430 |
| Total | | 37552 | 896 |

BAC : 0.65

Logistic Regression and MultilayerPerceptron (size hidden layers [300,300], activation function = "relu") after preprocessing the data (mean = 0 and variance = 1)

Best result from the last competition : 0.73

-Fig- Resultat apres preprocessing en utilisant la normalisationZ

VI) Problèmes rencontrés lors du projet , et conseils pour les étudiants de l'année prochaine:

Lors de ce projet, nous avons beaucoup appris, notamment sur le côté programmation en python. Cependant, nous avons rencontrés quelques problèmes, le plus gros, n'ayant rien à voir avec le côté programmation ou apprentissage, il est surtout en rapport avec le travail d'équipe, nous avons des difficultés de réunir tout le monde, de faire travailler tout le monde, en sachant que quelques membres de l'équipe ne venaient plus, et ne travaillaient plus, à la fin, nous étions seulement 2 à travailler sur le projet. Je conseille donc aux étudiants de l'année prochaine, de bien choisir leur projet, et surtout de le choisir en même temps que leurs camarades avec qui ils veulent travailler, ou avec qui ils ont des affinités point de vue travail d'équipe.

Le deuxième problème est maintenant lié au temps, notamment pour le choix de classifier, si nous avons eu d'avantage de temps, nous pensons que nous pouvons obtenir un meilleur résultat avec SVM en optimisant les paramètres hyper, nous pensons que nous aurions pu obtenir un meilleur résultat avec SVM en optimisant les hyper paramétrés. Je conseil donc aux étudiants de l'année prochaine sur ce point la de ne pas trop traîner, et de bien choisir les binômes qui travaillent sur la classification

Conclusion :

Ce défi est basé sur la prédiction de composés actifs contre le sida VIH infection. Nous avons adapté ce problème dans un défi de données, le but est de prédire si une molécule

Peut ou non être actif contre le VIH. L'ensemble de données donné a été prétraité, les caractéristiques du jeu de données

Sont clairement décrits et facilement compréhensibles. Un prétraitement de l'ensemble de données combiné à un

Une méthode de classification binaire adéquate a donné un résultat satisfaisant.

Cependant, nous pensons vraiment qu'une autre configuration du réseau neuronal pourrait améliorer les résultats.

VII)Algotirhmes et references :

Classe Classifieur avec RandomForestClassifier[4] :

Cette classe s'enchainera avec la classe de preprocessing dans un pipeline , et effectuera les predictions
Le classifieur sera remplacé par un pipeline comme decris ci dessous :

```
myclassifier = Pipeline([
('feature_selection', SelectFromModel(LinearSVC(penalty="l1"))),
('classification', RandomForestClassifier())
])
```

```
from sklearn.base import BaseEstimator
from sklearn.ensemble import RandomForestClassifier
from sklearn.pipeline import Pipeline
from myPreprocessing import Preprocessor
import pickle
class Classifieur(BaseEstimator):
    def __init__(self):
        classif = RandomForestClassifier(n_estimators=200, max_features="auto")
        self.clf = Pipeline([('preproc',Preprocessor(classif)),
('class',classif)])
        self.clf = classif

    def fit(self, X, y):
        return self.clf.fit(X, y)

    def predict(self, X):
        return self.clf.predict(X)

    def predict_proba(self, X):
        return self.clf.predict_proba(X)

    def save(self, path="."):
        pickle.dump(self, open(path + '_model.pickle', "w"))

    def load(self, path="."):
        self = pickle.load(open(path + '_model.pickle'))
        return self
```

Exemples de tests Unitaires [5]:

```
Myclassifier=
Classifieur()
```

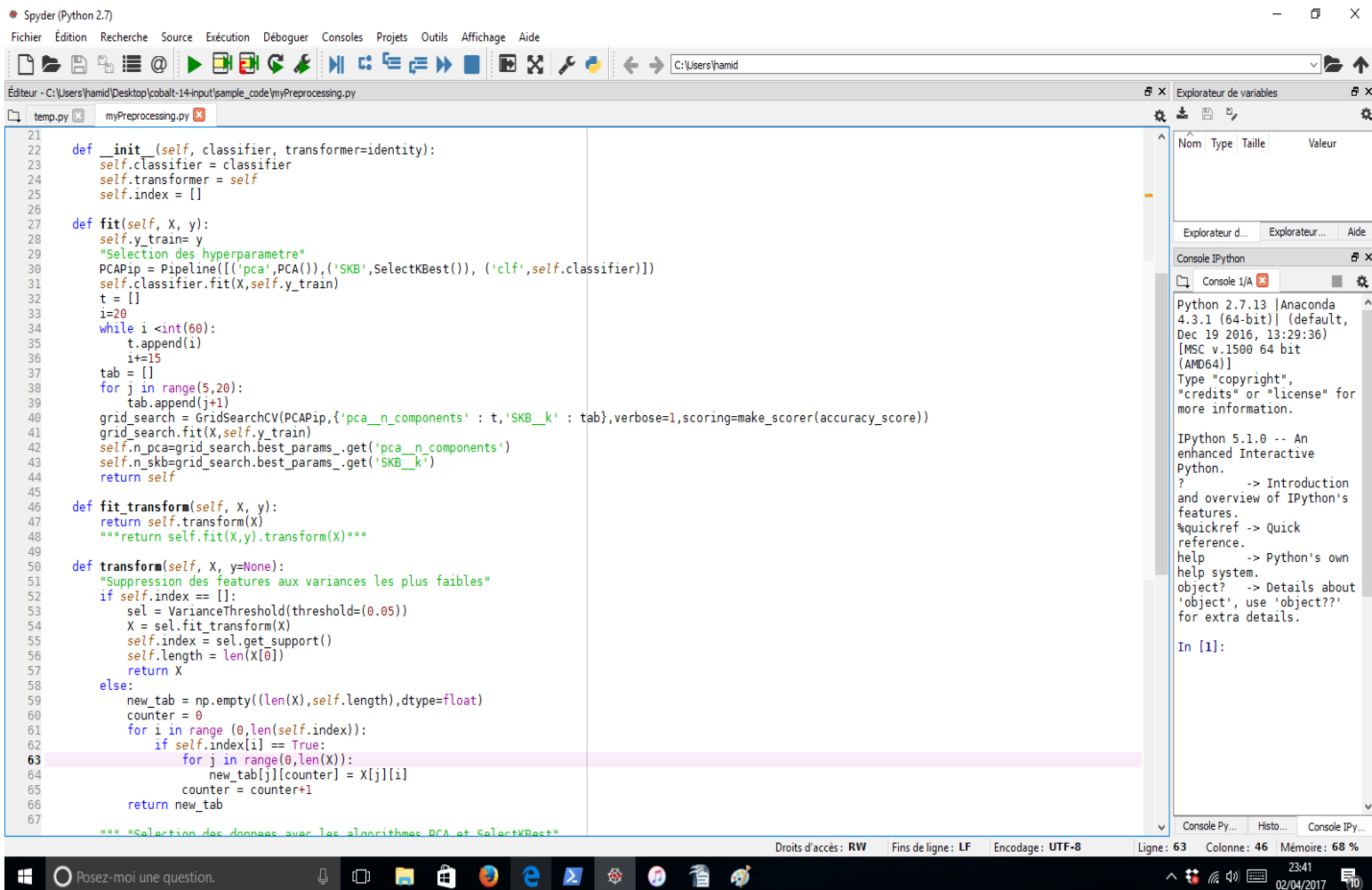
Train

```
Ytrue_tr = D.data['Y_train']
myclassifier.fit(D.data['X_train'], Ytrue_tr)
```

Making predictions

```
Ypred_tr =
myclassifier.predict(D.data['X_train'])
Ypred_va =
myclassifier.predict(D.data['X_valid'])
Ypred_te =
myclassifier.predict(D.data['X_test'])
```

Squelette de la classe de preprocessing SimpleTransform[4] :



```
21
22 def __init__(self, classifier, transformer=identity):
23     self.classifier = classifier
24     self.transformer = self
25     self.index = []
26
27 def fit(self, X, y):
28     self.y_train = y
29     "Selection des hyperparametre"
30     PCAPip = Pipeline([['pca', PCA()], ['SKB', SelectKBest()], ['clf', self.classifier]])
31     self.classifier.fit(X, self.y_train)
32     t = []
33     i = 20
34     while i < int(60):
35         t.append(i)
36         i += 15
37     tab = []
38     for j in range(5, 20):
39         tab.append(j+1)
40     grid_search = GridSearchCV(PCAPip, {'pca_n_components' : t, 'SKB_k' : tab}, verbose=1, scoring=make_scorer(accuracy_score))
41     grid_search.fit(X, self.y_train)
42     self.n_pca = grid_search.best_params_.get('pca_n_components')
43     self.n_skb = grid_search.best_params_.get('SKB_k')
44     return self
45
46 def fit_transform(self, X, y):
47     return self.transform(X)
48     """return self.fit(X, y).transform(X)"""
49
50 def transform(self, X, y=None):
51     "Suppression des features aux variances les plus faibles"
52     if self.index == []:
53         sel = VarianceThreshold(threshold=(0.05))
54         X = sel.fit_transform(X)
55         self.index = sel.get_support()
56         self.length = len(X[0])
57         return X
58     else:
59         new_tab = np.empty((len(X), self.length), dtype=float)
60         counter = 0
61         for i in range(0, len(self.index)):
62             if self.index[i] == True:
63                 for j in range(0, len(X)):
64                     new_tab[j][counter] = X[j][i]
65                     counter = counter + 1
66         return new_tab
67
68 """ "Selection des donnees avec les algorithmes PCA et SelectKBest"
```

Cette classe transforme la matrice de données initiale en une nouvelle matrice avec de meilleurs variables (features).

Exemples de tests unitaires[5] :

```
Prepro =
Preprocessor()
```

```
# Preprocess on the data and load it back into D
D.data['X_train'] = Prepro.fit_transform(D.data['X_train'],
D.data['Y_train'])
D.data['X_valid'] = Prepro.transform(D.data['X_valid'])
D.data['X_test'] = Prepro.transform(D.data['X_test'])
```

```
# Scatter-plots of the 2 first principal components
# Scatter plots of pairs of features that are most relevant
import matplotlib.pyplot as plt
X = D.data['X_train']
Y = D.data['Y_train']
plt.scatter(X[:, 0], X[:, 1], c=Y)
plt.xlabel('PC1')
```



```
plt.ylabel('PC2')
plt.colorbar()
plt.show()
```

Squelette de la classe de visualisation DataManager :

Python 2.7.13 [Anaconda 4.3.1 (64-bit)] (default, Dec 19 2016, 13:29:36) [MSC v.1500 64 bit (AMD64)]
Type "copyright", "credits" or "license()" for more information.

IPython 5.1.0 -- An enhanced Interactive Python.
? -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help -> Python's own help system.
help system. -> Details about 'object?', use 'object??' for extra details.

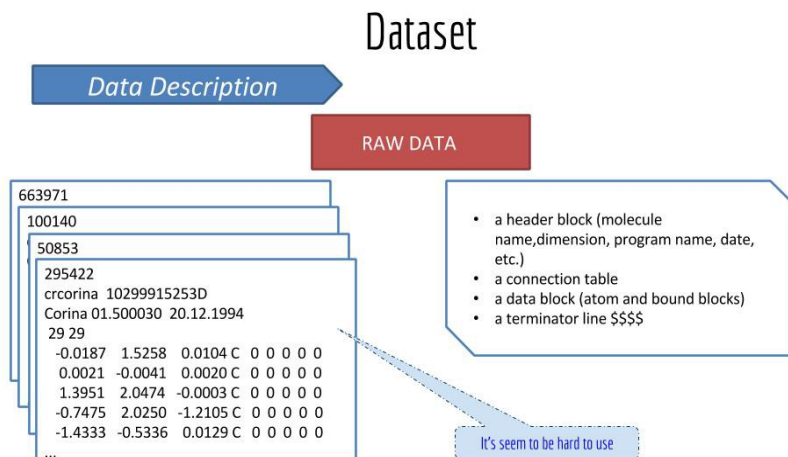
In [1]:

Python 2.7.13 [Anaconda 4.3.1 (64-bit)] (default, Dec 19 2016, 13:29:36) [MSC v.1500 64 bit (AMD64)]
Type "copyright", "credits" or "license()" for more information.

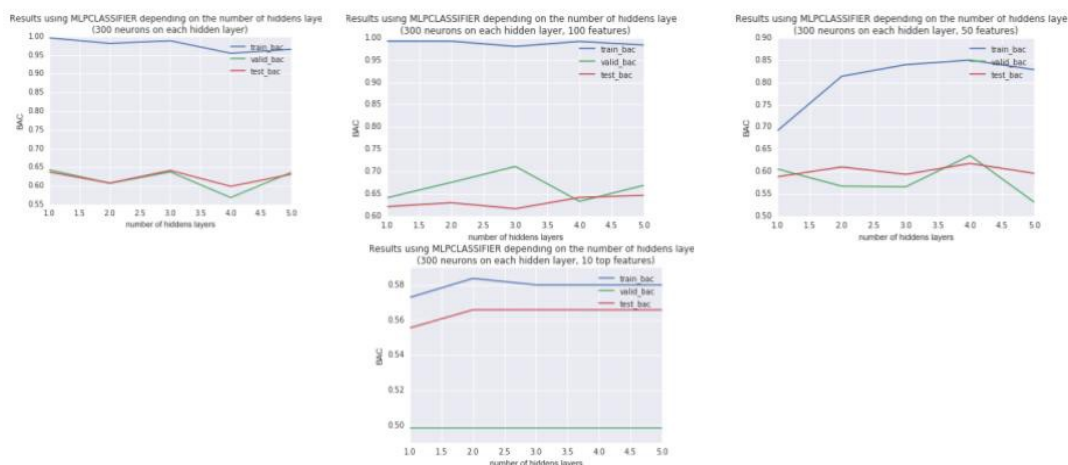
IPython 5.1.0 -- An enhanced Interactive Python.
? -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help -> Python's own help system.
help system. -> Details about 'object?', use 'object??' for extra details.

In [1]:

VIII) Annexes :



Results after preprocessing



autre figure de description des dataset

IX) References et bibliographie :

- [1] Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.
- [2] AutoSKLearn: Efficient and Robust Automated Machine Learning, Feurer et al., Advances in Neural Information Processing Systems 28 (NIPS 2015).
- [3] Foundations and applications. I. Guyon et al. Eds. Springer, 2006.
- [4] StartingKit(CodeLab)
- [5] https://github.com/madclam/L2_iris/blob/master/my_code

Original owners



Made by **National Cancer Institute (NCI)**
via the DTP AIDS Antiviral Screen program

Donor of database



Isabelle Guyon, 955 Creston Road, Berkeley,
CA 94708, USA isabelle@clopinet.com

http://dtp.nci.nih.gov/docs/aids/aids_data.html

<http://www.clopinet.com/isabelle/>

