PYTHON FOR DATA ANALYSIS

THOMAS VAQUERO

HAPT DATA SET


CLASSIFICATION SANS DATA CLEANING

LIBRAIRIES UTILISÉES

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import sklearn
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import classification_report, confusion_matrix
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import GradientBoostingClassifier
import seaborn as sn
from matplotlib.cm import ScalarMappable
from sklearn.model_selection import GridSearchCV
from sklearn.calibration import CalibratedClassifierCV
from sklearn.datasets import make_moons
import warnings
from sklearn.datasets import load_digits
from sklearn import svm
```

CRÉATION DES DATA FRAME

- On crée deux data frame train et test en important les fichiers
- On enlève les colonnes « Class » des fichiers train

```
x_train = df_train.drop('Class', axis = 1)
y_train = df_train['Class']
x_test = df_test.drop('Class', axis = 1)
y_test = df_test['Class']
```

MODÈLES

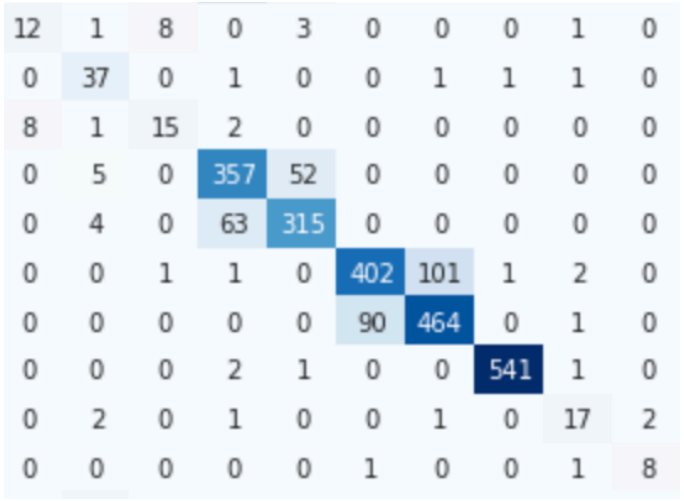

mirror object to mirror
mirror_mod.mirror_object
operation == "MIRROR_X":
irror_mod.use_x = True
irror_mod.use_y = False
irror_mod.use_z = False
operation == "MIRROR_Y":
irror_mod.use_x = False
irror_mod.use_y = True
irror_mod.use_z = False
operation == "MIRROR_Z":
irror_mod.use_x = False
irror_mod.use_y = False
irror_mod.use_z = True
selection at the end -add
_ob.select= 1
er_ob.select=1
ntext.scene.objects.activ
("Selected" + str(modifier
irror_ob.select = 0
bpy.context.selected_ob
ata.objects[one.name].se
int("please select exactly
OPERATOR CLASSES
types.Operator):
X mirror to the selected
ject.mirror_mirror_x"
ror X"

TROIS MODÈLES UTILISÉS

Classifier Tree → classifier = DecisionTreeClassifier()

Random Forest → RFC = RandomForestClassifier()

LSVM → lsvm = sklearn.svm.SVC(kernel = 'linear')



12	1	8	0	3	0	0	0	1	0
0	37	0	1	0	0	1	1	1	0
8	1	15	2	0	0	0	0	0	0
0	5	0	357	52	0	0	0	0	0
0	4	0	63	315	0	0	0	0	0
0	0	1	1	0	402	101	1	2	0
0	0	0	0	0	90	464	0	1	0
0	0	0	2	1	0	0	541	1	0
0	2	0	1	0	0	1	0	17	2
0	0	0	0	0	1	0	0	1	8

	precision	recall	f1-score	support
1	0.96	0.99	0.98	496
10	0.72	0.72	0.72	25
11	0.69	0.67	0.68	49
12	0.72	0.67	0.69	27
2	0.98	0.97	0.97	471
3	0.99	0.98	0.98	420
4	0.96	0.89	0.92	508
5	0.91	0.97	0.94	556
6	1.00	1.00	1.00	545
7	1.00	0.78	0.88	23
8	0.91	1.00	0.95	10
9	0.58	0.66	0.62	32
accuracy			0.95	3162
macro avg	0.87	0.86	0.86	3162
weighted avg	0.95	0.95	0.95	3162

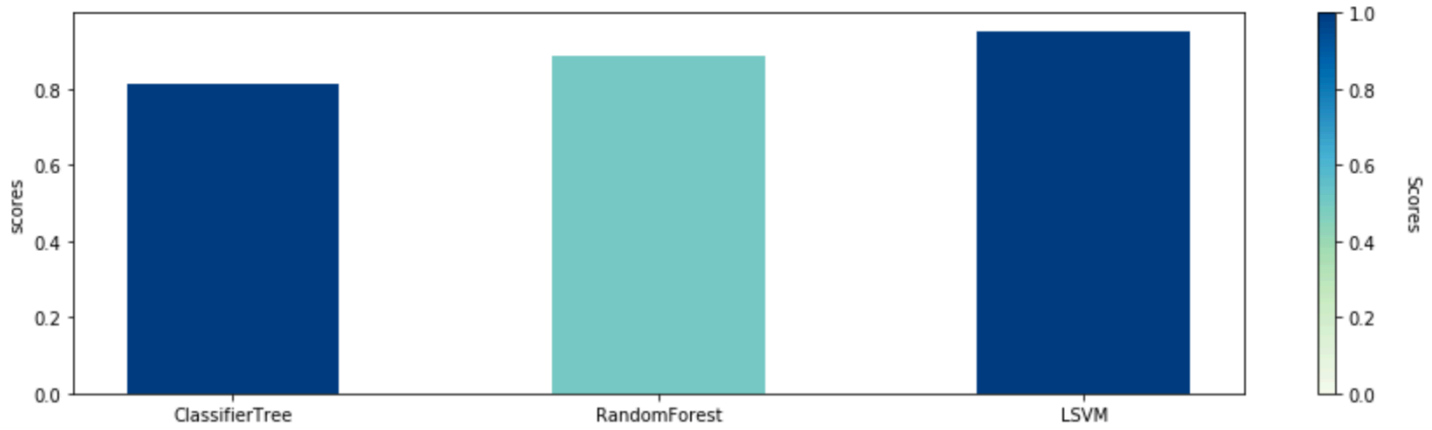
```
0.9522454142947502
```

SCORE DES MODÈLES

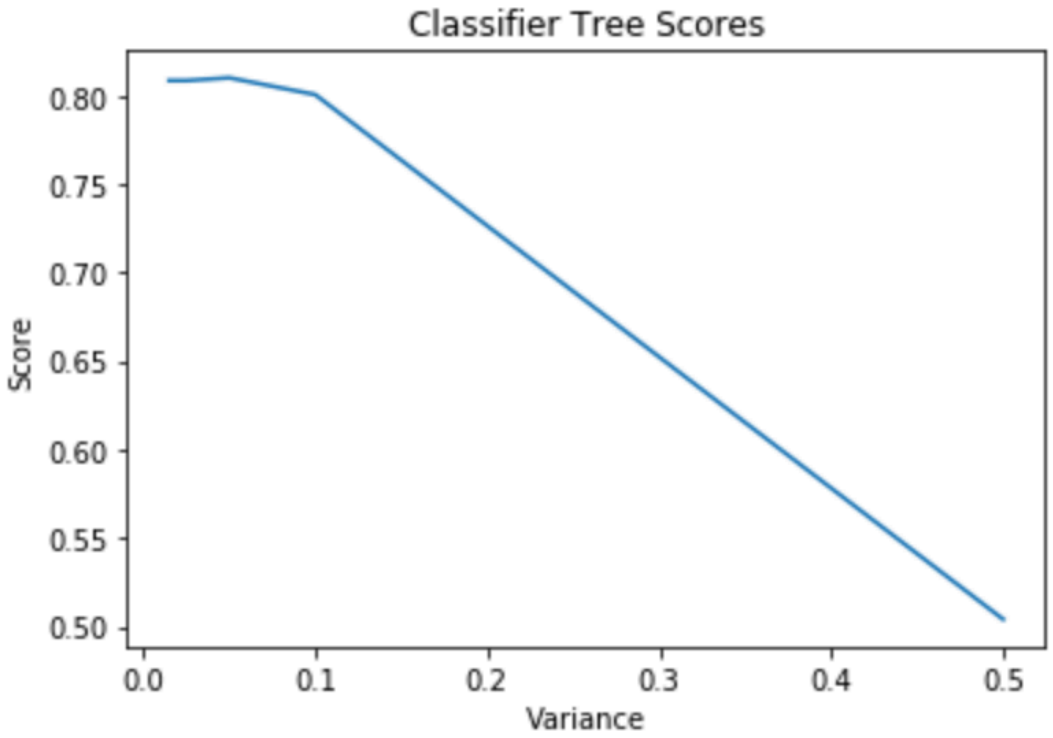
- On entraîne les modèles (avec la méthode fit) et on affiche :
  - Le rapport de classification
  - La matrice de confusion
- Scores obtenus :
  - Classifier Tree → 0.8118279569892473
  - Random Forest → 0.8870967741935484
  - LSVM → 0.9522454142947502

CONCLUSION SUR LES SCORES

- Meilleure prédiction obtenue avec le **LSVM**

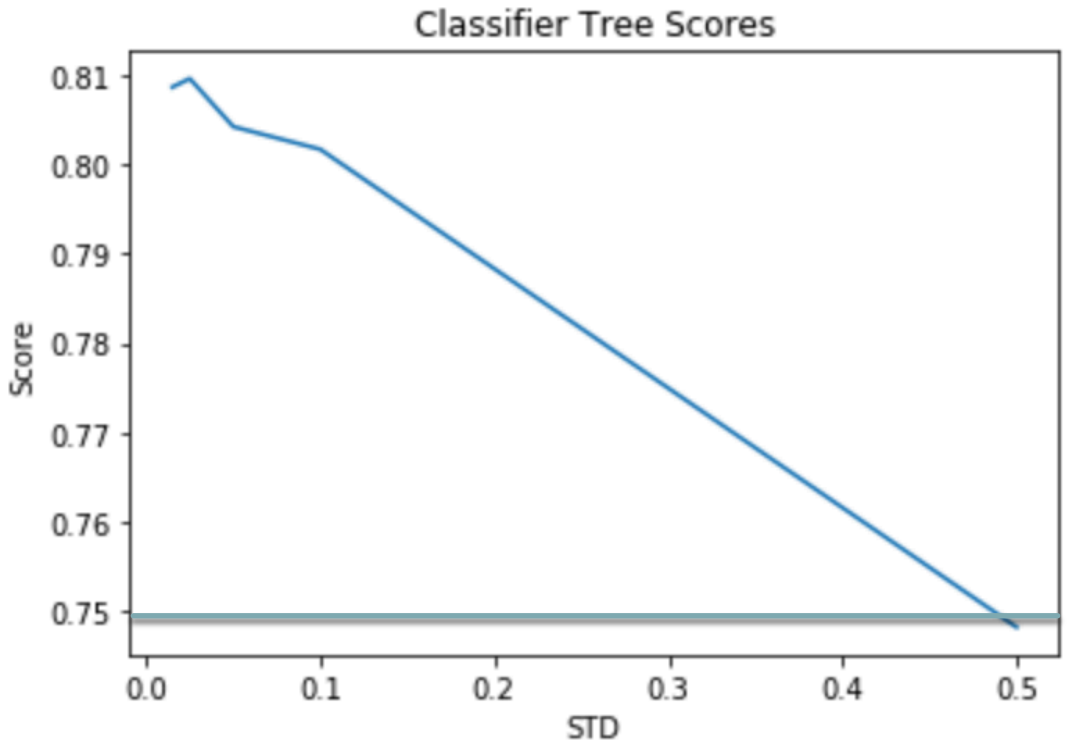



DATA CLEANING



FILTRAGE SUR LES COLONNES

- Objectif : retirer des colonnes pouvant nuire à la prédiction
- Méthode : On fixe plusieurs valeurs de variance puis on lance chaque modèle. On retire a chaque fois les colonnes dont la variance est supérieure au seuil et on les supprime
- Sur le schéma on peut voir que pour une variance supérieure à 0.1, le score diminue de manière significative



VARIATION DU STD (ÉCART TYPE)

- On effectue le même filtrage avec la valeur STD
- On note une diminution du score mais moins importante
- Un filtrage sur l'écart type permet de limiter le score à une limite de 0.75


VARIATION DES HYPERPARAMÈTRES

GRIDSEARCH

- On applique un 1er gridsearch avec le meilleur modèle (SVM)
- On utilise les paramètres par défaut pour commencer (C et Gamma)
- On applique la méthode GridSearch avec la méthode fit

```
hyperparametres_possibles = {
    'C'     : [0.5, 1, 1.5],
    'gamma' :[0.5, 0.1, 0.15]
}
grid = GridSearchCV(estimator=svm.SVR(),
                                param_grid=hyperparametres_possibles,
                                n_jobs=-1, cv=10, verbose=2)
```

RÉSULTATS DES GRIDSEARCH ET VARIATION DES PARAMÈTRES

- On augmente le paramètres C
- On lance 3 fois le GridSearch sur le SVM et on observe les changements
- On note une bonne amélioration du score (le résultat ci-dessous montre le score obtenu après 3 variations)

Entrée [30]:

```
hyperparametres_possibles2 = {
    'C'     : [ 10, 15, 20],
    'gamma' :[0.01, 0.05, 1]
}
grid = GridSearchCV(estimator=lsvm,
                                param_grid=hyperparametres_possibles2,
                                n_jobs=-1)
grid.fit(x_train, y_train)
s3 = grid.score(x_train, y_train)
print(s3)
```

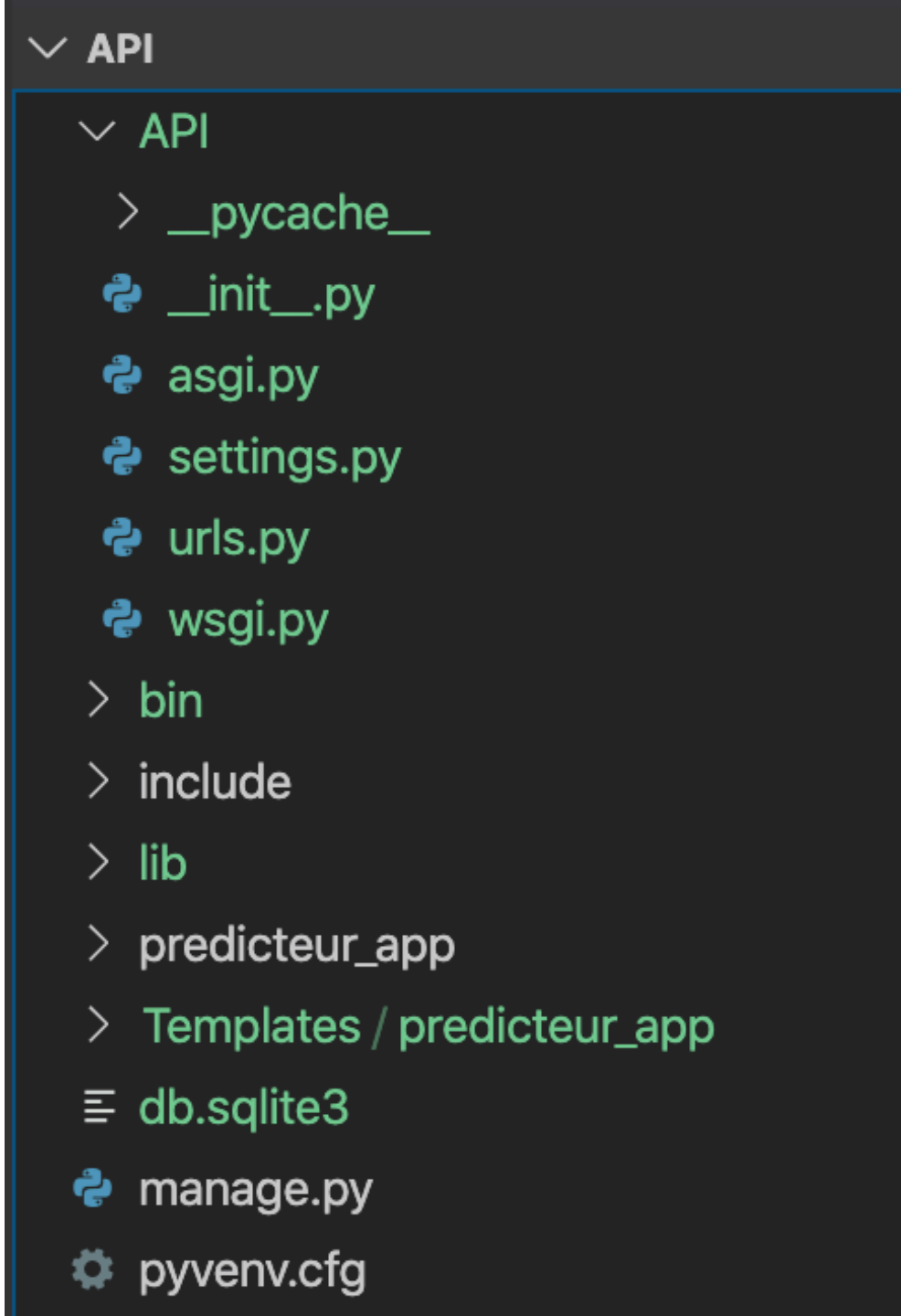
```
0.996523754345307
```

API DJANGO


mirror_mod.mirror_object
operation == "MIRROR_X":
mirror_mod.use_x = True
mirror_mod.use_y = False
mirror_mod.use_z = False
operation == "MIRROR_Y":
mirror_mod.use_x = False
mirror_mod.use_y = True
mirror_mod.use_z = False
operation == "MIRROR_Z":
mirror_mod.use_x = False
mirror_mod.use_y = False
mirror_mod.use_z = True
selection at the end -add
ob.select= 1
er_ob.select=1
ntext.scene.objects.activ
("Selected" + str(modifier
irror_ob.select = 0
bpy.context.selected_ob
ata.objects[one.name].se
int("please select exactly
OPERATOR CLASSES
types.Operator):
X mirror to the selected
ject.mirror_mirror_x"
ror X"

CRÉATION D'UNE APPLICATION SUR DJANGO

- Utilisation de Django
- Création d'une application
- → API non aboutie (seulement la création d'une view)



MERCI