

Bab 1

Pengenalan Prosedur dalam Machine Learning menggunakan Scikit Learn

A. KOMPETENSI DASAR

- ◆ Memahami antarmuka Jupyter Notebook.
- ◆ Memahami Prosedur Pemodelan menggunakan Scikit-learn.
- ◆ Memahami Dataset dan Preprocessing Data

B. ALOKASI WAKTU

2 js (2x50 menit)

C. PETUNJUK

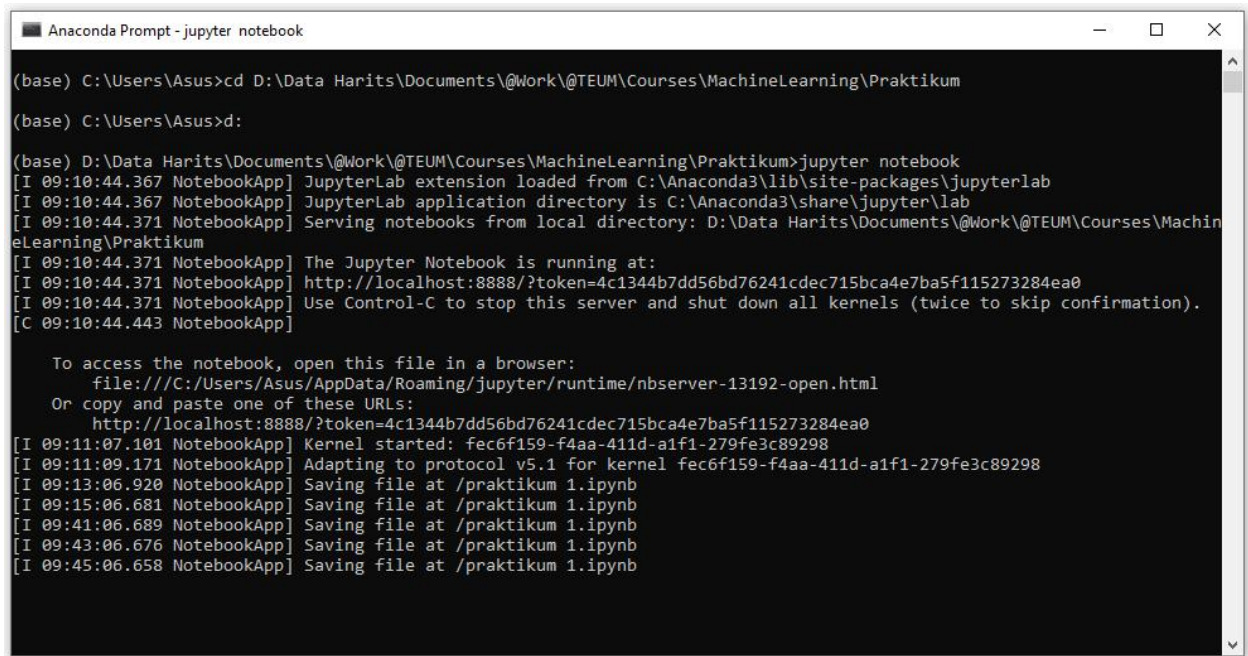
- Awali setiap aktivitas dengan do'a, semoga berkah dan mendapat kemudahan.
- Pahami Tujuan, dasar teori, dan latihan-latihan praktikum dengan baik dan benar.
- Kerjakan tugas-tugas dengan baik, sabar, dan jujur.
- Tanyakan kepada teman lalu asisten/dosen apabila ada hal-hal yang kurang jelas.

D. ANTARMUKA JUPYTER NOTEBOOK

Untuk mengaktifkan jupyter notebook lakukan dengan langkah berikut:

- 1) Windows, ketik keyword: "cmd" dan pilihlah Anaconda Prompt.

- 2) Arahkan ke direktori yang akan dipakai sebagai root directory program praktikum anda.



```
Anaconda Prompt - jupyter notebook

(base) C:\Users\Asus>cd D:\Data Harits\Documents\@Work\@TEUM\Courses\MachineLearning\Praktikum

(base) C:\Users\Asus>

(base) D:\Data Harits\Documents\@Work\@TEUM\Courses\MachineLearning\Praktikum>jupyter notebook
[I 09:10:44.367 NotebookApp] JupyterLab extension loaded from C:\Anaconda3\lib\site-packages\jupyterlab
[I 09:10:44.367 NotebookApp] JupyterLab application directory is C:\Anaconda3\share\jupyter\lab
[I 09:10:44.371 NotebookApp] Serving notebooks from local directory: D:\Data Harits\Documents\@Work\@TEUM\Courses\MachineLearning\Praktikum
[I 09:10:44.371 NotebookApp] The Jupyter Notebook is running at:
[I 09:10:44.371 NotebookApp] http://localhost:8888/?token=4c1344b7dd56bd76241cdec715bca4e7ba5f115273284ea0
[I 09:10:44.371 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[C 09:10:44.443 NotebookApp]

To access the notebook, open this file in a browser:
file:///C:/Users/Asus/AppData/Roaming/jupyter/runtime/nbserver-13192-open.html
Or copy and paste one of these URLs:
http://localhost:8888/?token=4c1344b7dd56bd76241cdec715bca4e7ba5f115273284ea0
[I 09:11:07.101 NotebookApp] Kernel started: fec6f159-f4aa-411d-a1f1-279fe3c89298
[I 09:11:09.171 NotebookApp] Adapting to protocol v5.1 for kernel fec6f159-f4aa-411d-a1f1-279fe3c89298
[I 09:13:06.920 NotebookApp] Saving file at /praktikum 1.ipynb
[I 09:15:06.681 NotebookApp] Saving file at /praktikum 1.ipynb
[I 09:41:06.689 NotebookApp] Saving file at /praktikum 1.ipynb
[I 09:43:06.676 NotebookApp] Saving file at /praktikum 1.ipynb
[I 09:45:06.658 NotebookApp] Saving file at /praktikum 1.ipynb
```

Gambar 1. Command Line Interface Anaconda untuk aktivasi Jupyter Notebook

- 3) Eksekusi perintah “jupyter notebook” untuk menjalankan IDE python pada root directory tersebut. Hasilnya seperti pada Gambar 1 untuk command prompt nya dan Gambar 2 adalah web-based IDE untuk root directory. Daftar project dapat dilihat pada konten. Project yang aktif memiliki icon berwarna hijau. Pembuatan project python dapat dilakukan dengan menekan tombol “New” (lihat Gambar 3).

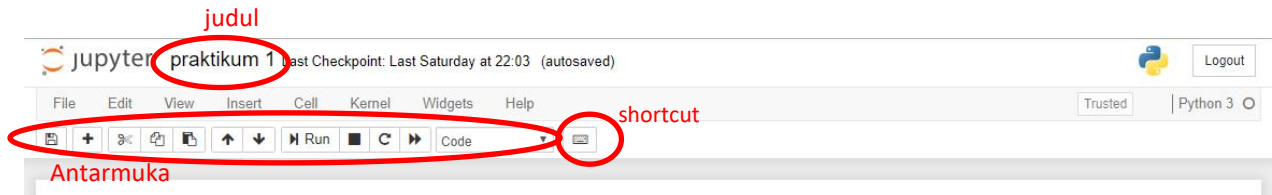


Gambar 2. Web-based IDE untuk project python



Gambar 3. Tambah project baru (Python)

- 4) Gambar 4 menampilkan tab baru pada web browser dimana nama project dapat diubah dengan klik pada judul di sebelah tulisan Jupyter. Biasakan diri Anda dengan antar muka ini dan beberapa shortcut yang mempermudah aktifitas pemrograman.

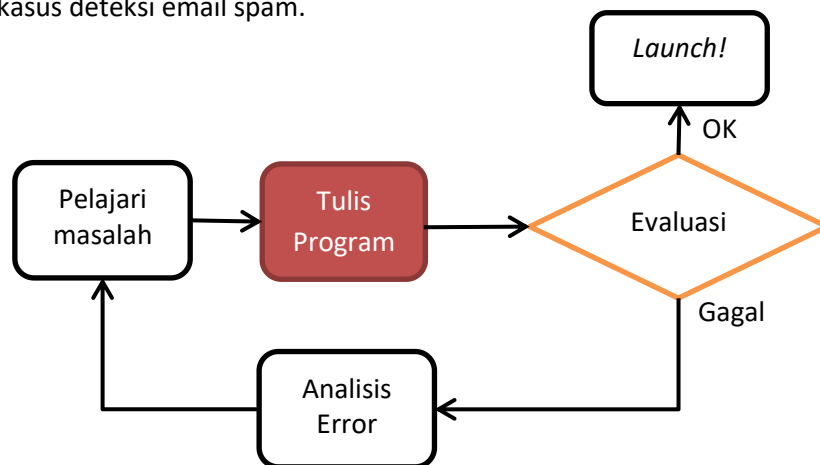


Gambar 4. Project python

E. PENGETAHUAN UMUM PROSEDUR ML DENGAN SCIKIT LEARN

Asumsikan sebuah kasus deteksi email spam.

Tradisional



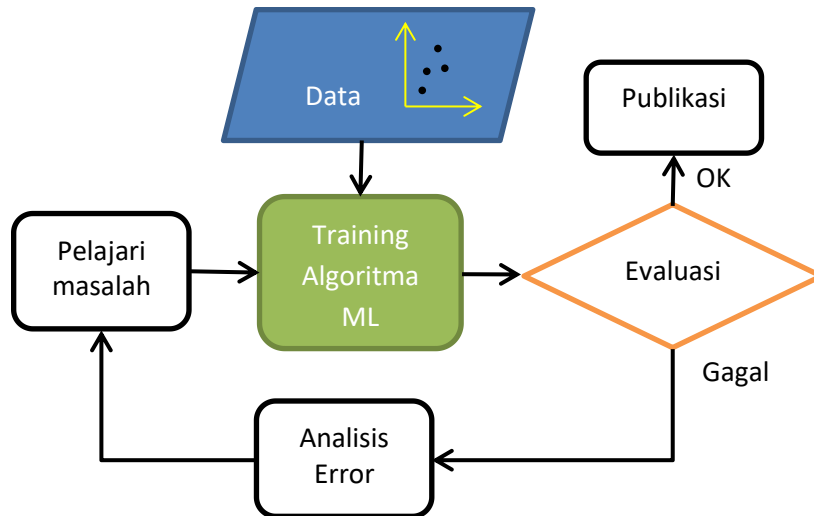
Dalam proses pengembangan aplikasi deteksi spam tersebut maka yang dilakukan oleh peneliti adalah

- 1) mempelajari masalah dengan observasi kata-kata atau frasa khusus yang menjadi ciri spam. Contoh: "winner", "free", "amazing" ,
- 2) menulis program sesuai *rules* (kata-kata spam),
- 3) evaluasi menggunakan beberapa sample email,
- 4) analisis error jika terjadi kegagalan prediksi, dan
- 5) meluncurkan aplikasi jika tidak terjadi kesalahan fungsi.

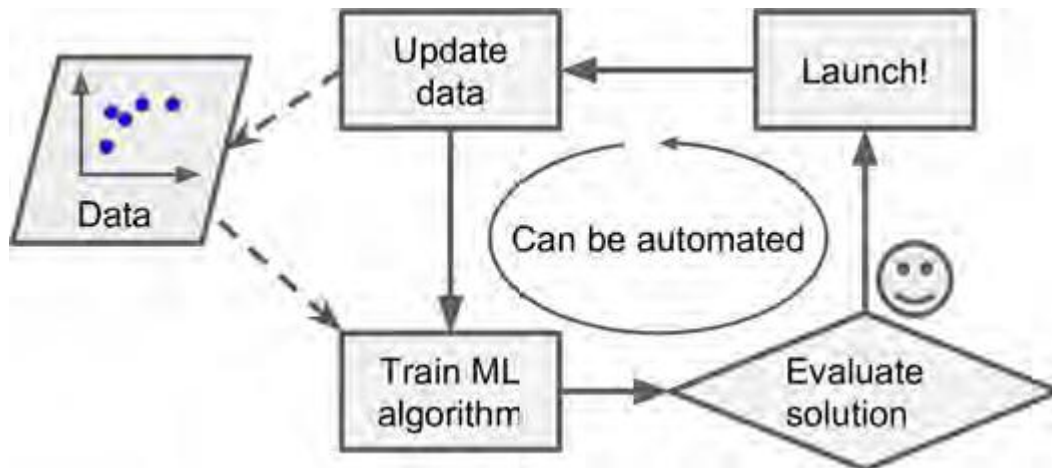
Pendekatan di atas membutuhkan daftar kata yang panjang dan *conditional check* yang bercabang-cabang agar mampu mendeteksi spam secara optimal.

F. Machine Learning

Dengan menggunakan pendekatan ML maka prosedur yang dalam pengembangan program deteksi spam di otomasi menjadi:



Akan tetapi, email spam semakin lama semakin kreatif sehingga perlu peng-kini-an data agar aplikasi deteksi spam dapat bekerja dengan baik.



PROSEDUR UMUM DALAM MACHINE LEARNING

1. DATASET (Training set and testing set)

Dataset adalah sekumpulan sampel-sampel data/informasi yang didapat dari uji atau pengukuran. Ilustrasikan dataset sebagai tabel yang terdiri atas input/fitur/independent variabel (x_0, x_1, \dots, x_i) berurutan dari kolom paling kiri. Pada beberapa kasus dataset, tiap vektor X terdapat pula label y yang terletak pada kolom paling kanan. Tiap baris dalam tabel ini disebut sebagai *instance* yang merepresentasikan satu sampel dari dataset. Maka dataset dibagi menjadi dua kelompok besar dalam membangun sebuah model yang menggunakan algoritma ML. Kelompok mayoritas sampel dari dataset disebut sebagai *training set*. Sebuah algoritma akan di-*training* untuk melakukan prediksi, estimasi atau peramalan menggunakan *training set* ini. Hasil training ini adalah sebuah **model** yang dapat memprediksi secara optimal. Kemudian sisa dataset yang tidak digunakan untuk *training set* akan dijadikan sebagai *test set*. Setiap sampel dalam *test-set* memiliki nilai-nilai fitur yang diumpangkan ke model prediksi untuk mengetahui outputnya (y^*). Perbedaan nilai y dengan y^* menentukan performa model prediksi tersebut.

Aplikasi scikit-learn menyediakan dataset-dataset umum untuk berlatih menggunakan machine learning. Cara untuk mendapatkan dataset untuk latihan adalah sebagai berikut (jupyter notebook aktif):

!Cobalah!

```
>>>from sklearn import datasets
>>>iris = datasets.load_iris()
>>>digits = datasets.load_digits()
```

Dataset yang di-*load* adalah 2 jenis dataset tentang iris dan karakter angka. Tiap variable tersebut menyimpan semua data dan metadata terkait. Data tersebut tersimpan dalam `.data`, yang terdiri dari **array** dengan dimensi `n_samples`, `n_features` sebagai kumpulan **nilai fitur/input** dalam dataset. Pada kasus supervised machine learning, **label** disimpan dalam `.target`.

!Cobalah!

```
>>>print(digits.data)
[[ 0.  0.  5. ...  0.  0.  0.]
 [ 0.  0.  0. ... 10.  0.  0.]
 [ 0.  0.  0. ... 16.  9.  0.]
 ...
 [ 0.  0.  1. ...  6.  0.  0.]
 [ 0.  0.  2. ... 12.  0.  0.]
 [ 0.  0. 10. ... 12.  1.  0.]]

>>>digits.target
array([0, 1, 2, ..., 8, 9, 8])
```

Dataset dapat pula didapatkan dari file eksternal seperti `.csv`. Untuk tujuan ini maka dapat dilakukan dengan:

```
>>> import pandas as pd

>>> pd.read_csv('data.csv')
```

2. Preprocessing Dataset

Tahap ini mempersiapkan data agar siap untuk diolah oleh machine learning secara optimal dan sesuai ekspektasi. Preprocessing dapat dilakukan dengan tujuan sebagai berikut:

- a. **Deteksi outlier** atau sampel-sampel yang tidak rasional.
- b. **Missing values** untuk tindakan terhadap sampel yang nilai-nilai fitur atau labelnya tidak lengkap.
- c. **Normalisasi data** untuk menyamakan rentang nilai tiap fitur.
- d. **Konversi data** untuk mengubah tipe data.
- e. **Integrasi data** untuk penggabungan beberapa data menjadi satu.
- f. **Resampling** untuk menyeimbangkan distribusi sampel berdasarkan label yang dimilikinya.
- g. Dan lain lain..

Tahapan ini akan dibahas pada Bab praktikum berikutnya secara khusus. Pada bab praktikum ini kita gunakan dataset yang telah '*bersih*' dan siap untuk tahap *training* algoritma ML.

3. Pemodelan: Learning & Predicting

Tahap ini memegang peranan penting dimana sebuah algoritma ML dilatih untuk melakukan prediksi. Dalam kasus dataset **digits**, sebuah algoritma ML harus memprediksi angka (output) yang ditunjukkan oleh gambar (input).

Dalam scikit-learn, sebuah algoritma ML adalah objek Python yang mengimplementasikan fungsi `fit(X,y)` untuk proses training dan fungsi `predict(T)` untuk testing. Dalam latihan pada praktikum ini digunakan algoritma Support Vector Machine (SVM).

!Cobalah!

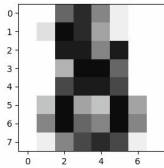
```
>>> from sklearn import svm
>>> clf = svm.SVC(gamma=0.001, C=100.)
```

Dari program di atas maka `clf` adalah objek klasifikasi yang harus di-*training* menggunakan fungsi `fit` pada dataset `digits` yang telah di load sebelumnya. Di bawah ini `clf` di-*training* menggunakan semua dataset kecuali sampel yang terakhir untuk testing. Pemilihan semua array kecuali baris terakhir dapat dilakukan dengan sintaks `[:-1]`.

!Cobalah!

```
>>> clf.fit(digits.data[:-1], digits.target[:-1])
SVC(C=100.0, cache_size=200, class_weight=None, coef0=0.0, decision_function_shape='ovr',
    degree=3, gamma=0.001, kernel='rbf', max_iter=-1, probability=False, random_state=None,
    shrinking=True, tol=0.001, verbose=False)
```

Berikutnya, clf dapat diuji kemampuan prediksinya menggunakan sampel yang terakhir (gambar di bawah). Gunakan sintaks `[-1:]` untuk array terakhir dari `digits.data`. Hasil dari prediksi gambarnya adalah angka 8.



Cobalah!

```
>>> clf.predict(digits.data[-1:])  
array([8])
```

4. Validasi

Tahapan ini dilakukan untuk mengetahui bagaimana sebuah model yang telah di-*training* dapat memprediksi sampel yang baru. Dalam machine learning tahapan ini disebut sebagai *Cross Validation* (CV). Terdapat dua jenis CV yang sering dilakukan antara lain:

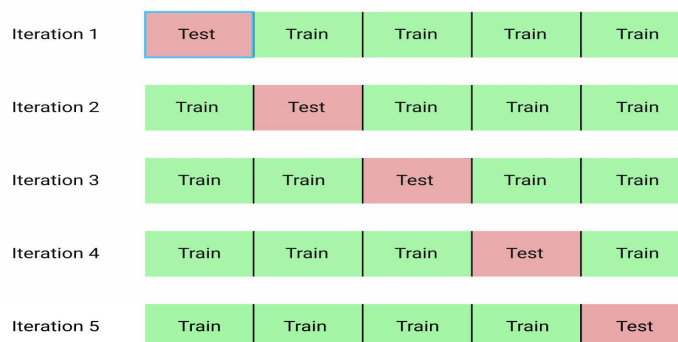
i. Hold-out CV

HOCV membagi dataset menjadi 2 bagian: training set dan test set. Pada umumnya training set memiliki porsi yang cukup besar (contoh 80%) sedangkan test-set mengambil sisa sampel (contoh 20%). contoh di bawah adalah HOCV pada dataset iris dengan test set sebesar 40%.

```
>>> from sklearn.model_selection import train_test_split  
...  
>>> X_train, X_test, y_train, y_test = train_test_split(  
... iris.data, iris.target, test_size=0.4, random_state=0)
```

ii. K-fold CV

KFCV pada dasarnya adalah HOCV yang iteratif sebanyak **K**. Pada tiap iterasi memiliki K partisi dimana 1 partisi akan menjadi test set dan yang lain sebagai training set. Contoh di bawah ini adalah ilustrasi 5-fold CV.



Dari perbedaan y dan y^* dalam validasi di atas dapat dilihat seberapa baik performa algoritma ML dalam melakukan prediksi. Perbedaan label dan hasil prediksi ini di akumulasi dalam bentuk tabel *confusion matrix*.

n=165		Predicted: NO	Predicted: YES	
Actual: NO		TN = 50	FP = 10	60
Actual: YES		FN = 5	TP = 100	105
		55	110	

Image source: Data school

Tabel di atas menunjukkan akumulasi dari prediksi biner yang menghasilkan True Negative (TN) dan True Positive (TP) jika antara label asli (true label) dan prediksi memiliki nilai yang sama. Sedangkan, False Negative (FN) jika label asli bernilai positive namun diprediksi negative. Begitu pula sebaliknya, False Positive (FP) jika label asli bernilai negative namun diprediksi positive. Dari tabel confusion matrix ini maka dapat dihitung performa algoritma berdasarkan rumus-rumus di-bawah ini:

Classification

1. accuracy = $TP+TN / \text{Total}$
2. error / Misclassification rate = $1 - \text{Accuracy}$
3. false positive rate = $FP / \text{Actual No}(60)$
4. false negative rate = $FN / \text{Actual Yes}(105)$
5. true positive rate/Sensitivity = $TP / \text{Actual Yes}$

Pada regression dan unsupervised machine learning memiliki metode evaluasi yang khusus pula, contoh:

Regression Problem:

1. Mean Absolute Error
2. Root Mean squared Error

Clustering analysis:

1. Silhouette score
2. Cophenet correlation coefficient

Script berikut adalah contoh program untuk cross validation dan penghitungan performa klasifikasi iris.

```
>>> from sklearn.model_selection import cross_val_score
>>> clf = svm.SVC(kernel='linear', C=1)
>>> scores = cross_val_score(clf, iris.data, iris.target, cv=5)
>>> scores
array([0.96..., 1. ..., 0.96..., 0.96..., 1. ...])
```


5. Model Persistence

Tahap ini adalah finalisasi model menjadi sebuah objek untuk implementasi di lapangan. Model yang telah dilatih dan di validasi dapat di simpan menggunakan cara berikut ini:

!Cobalah!

```
>>> import pickle
>>> s = pickle.dumps(clf) // menyimpan model
>>> clf2 = pickle.loads(s) // load model
>>> clf2.predict(X[0:1])array([0])
>>> y[0]
0
```

Jika ML beroperasi dalam big data maka cara di bawah ini lebih optimal karena model dapat disimpan dalam bentuk file.

```
>>> from joblib import dump, load
>>> dump(clf, 'filename.joblib') //menyimpan model
>>> clf = load('filename.joblib') //load model
```

G. TUGAS

1. Lakukan langkah 4 di atas dengan melakukan training pada 80% dataset digits dan testing pada 20% (praktekkan langkah 4 untuk HOCV).
2. Lakukan langkah 4 untuk menghitung akurasi, presisi dan recall dari digits data.
3. Carilah dataset sederhana dengan label biner dengan jumlah sampel minimum 100 + (2 digit terakhir NIM anda). Tiap kelompok tidak boleh menggunakan dataset yang sama. Topik dataset adalah klasifikasi 2 jenis tumbuhan (contoh: apel vs jeruk), hewan (contoh: anjing vs kucing), suku (contoh: jawa vs batak), genre game (contoh: FPS vs Platformer), genre musik (contoh: dangdut vs keroncong), dll.
4. Buatlah diagram prosedur (pengambilan data hingga validasi) yang dilakukan untuk membangun model prediksi menggunakan algoritma SVM.
5. Buatlah program, tunjukkan dan jelaskan program dalam jupyter notebook anda.