Soal Mentoring 1 - Machine Learning Process

Job Preparation Program, Data Science, Pacmann AI

**Catatan:**

- Make a copy docs ini sebelum menjawab.
- Studi kasus pada soal ini akan **digunakan** untuk mentoring selanjutnya.
- Tulis jawaban Anda dalam word processor dan ekspor ke PDF untuk soal nomor 1 dan 2
- Tulis jawaban Anda dalam jupyter notebook dan python script untuk soal nomor 3 dan 4
- Archive file PDF dan keseluruhan root folder project Anda ke ZIP, **kecuali folder venv**
- Beri nama [NAMA_LENGKAP]_MLPROCESS_1 file archive Anda
- Submit file archive Anda melalui link submission yang telah disediakan

Soal Pengantar

- Posisikan diri Anda sebagai Data Scientist yang bekerja dibidang perbankan.
- Produk pemberian pinjaman (loan) adalah salah satu sumber revenue di industri perbankan.
- Bank akan mengalami kerugian apabila customer gagal mengembalikan pinjaman (gagal bayar – default).
- Anda diminta untuk membuat sistem untuk prediksi customer akan gagal bayar atau tidak.

Soal

1. [25 points] Project Environment Setup

a. [5 points] Buat folder project

- Buat folder dengan nama berformat [NAMA_ANDA]_MLPROCESS

  Gunakan 1 kata pertama dari nama Anda

  Folder ini akan menjadi ROOT FOLDER PROJECT Anda

  Contoh: ADIT_MLPROCESS

- Buat subfolder bernama data, models, dan src dalam folder project Anda

- Buat subfolder bernama raw, interim, dan processed dalam folder data
- **Lampirkan screenshot folder tree yang telah Anda buat**

```
This message is shown once a day. To disable it please create the
/home/chalimasadiah/.hushlogin file.
chalimasadiah@LAPTOP-CH00NPOB:/mnt/c/WINDOWS/system32$ cd
chalimasadiah@LAPTOP-CH00NPOB:~$ mkdir CHALIMA_MLPROCESS
chalimasadiah@LAPTOP-CH00NPOB:~$ cd CHALIMA_MLPROCESS/
chalimasadiah@LAPTOP-CH00NPOB:~/CHALIMA_MLPROCESS$ mkdir data data/raw data/interim data/processed models src
chalimasadiah@LAPTOP-CH00NPOB:~/CHALIMA_MLPROCESS$ tree
.
├── data
│   ├── interim
│   ├── processed
│   └── raw
├── models
└── src

6 directories, 0 files
chalimasadiah@LAPTOP-CH00NPOB:~/CHALIMA_MLPROCESS$
```

b. [5 points] Buat python virtual environment (venv)

- Buatlah venv dengan nama berformat [NAMA_ANDA]_VENV

  Buat venv di dalam folder project Anda

  Gunakan 1 kata pertama dari nama Anda

  Contoh: ADIT_VENV

- **Lampirkan screenshot terminal yang menampilkan command saat Anda membuat venv tersebut**

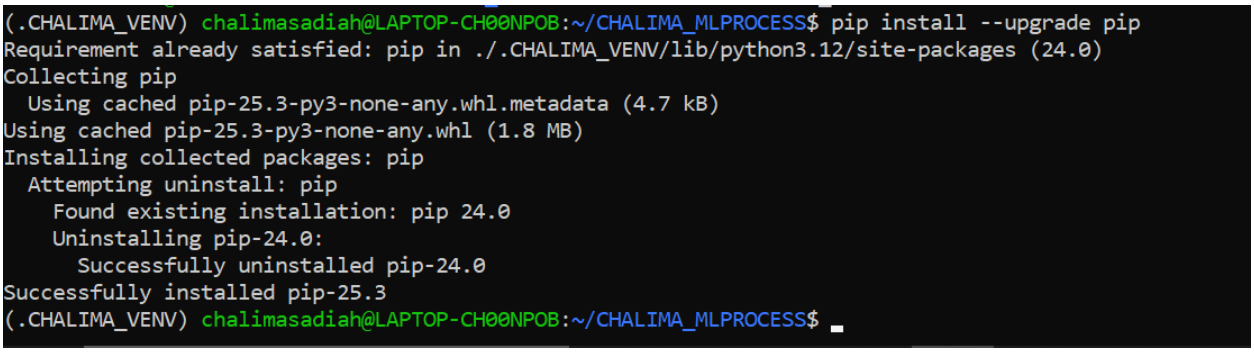
```
chalimasadiah@LAPTOP-CH00NPOB:~/CHALIMA_MLPROCESS$ python3 -m venv .CHALIMA_VENV
chalimasadiah@LAPTOP-CH00NPOB:~/CHALIMA_MLPROCESS$
```

- **Lampirkan screenshot terminal yang menampilkan command saat Anda mengaktifkan venv tersebut**

```
chalimasadiah@LAPTOP-CH00NPOB:~/CHALIMA_MLPROCESS$ source .CHALIMA_VENV/bin/activate
(.CHALIMA_VENV) chalimasadiah@LAPTOP-CH00NPOB:~/CHALIMA_MLPROCESS$
```

c. [5 points] Update PIP

- **Lampirkan screenshot terminal yang menampilkan command saat Anda melakukan update PIP**




d. [5 points] Install dependencies

- Install package berikut:
    - pandas
    - scikit-learn
    - imblearn
    - joblib
    - numpy
    - scipy
    - seaborn
    - fastapi
    - jupyterlab
    - requests
- **Lampirkan screenshot terminal yang menampilkan command saat Anda melakukan pemasangan packages tersebut**

```
(.CHALIMA_VENV) chalimasadiah@LAPTOP-CH00NPOB:~/CHALIMA_MLPROCESS$ touch requirements.txt
(.CHALIMA_VENV) chalimasadiah@LAPTOP-CH00NPOB:~/CHALIMA_MLPROCESS$ nano requirements.txt
(.CHALIMA_VENV) chalimasadiah@LAPTOP-CH00NPOB:~/CHALIMA_MLPROCESS$ nano requirements.txt
(.CHALIMA_VENV) chalimasadiah@LAPTOP-CH00NPOB:~/CHALIMA_MLPROCESS$ pip install -r requirements.txt
Collecting pandas (from -r requirements.txt (line 1))
  Using cached pandas-3.0.0-cp312-cp312-manylinux_2_24_x86_64.manylinux_2_28_x86_64.whl.metadata (79 kB)
Collecting scikit-learn (from -r requirements.txt (line 2))
  Using cached scikit_learn-1.8.0-cp312-cp312-manylinux_2_27_x86_64.manylinux_2_28_x86_64.whl.metadata (11 kB)
Collecting imbalanced-learn (from -r requirements.txt (line 3))
  Downloading imbalanced_learn-0.14.1-py3-none-any.whl.metadata (8.9 kB)
Collecting joblib (from -r requirements.txt (line 4))
  Using cached joblib-1.5.3-py3-none-any.whl.metadata (5.5 kB)
Collecting numpy (from -r requirements.txt (line 5))
  Using cached numpy-2.4.1-cp312-cp312-manylinux_2_27_x86_64.manylinux_2_28_x86_64.whl.metadata (6.6 kB)
Collecting scipy (from -r requirements.txt (line 6))
  Using cached scipy-1.17.0-cp312-cp312-manylinux_2_27_x86_64.manylinux_2_28_x86_64.whl.metadata (62 kB)
Collecting seaborn (from -r requirements.txt (line 7))
  Downloading seaborn-0.13.2-py3-none-any.whl.metadata (5.4 kB)
Collecting fastapi (from -r requirements.txt (line 8))
  Downloading fastapi-0.128.0-py3-none-any.whl.metadata (30 kB)
Collecting jupyterlab (from -r requirements.txt (line 9))
  Using cached jupyterlab-4.5.3-py3-none-any.whl.metadata (16 kB)
Collecting requests (from -r requirements.txt (line 10))
```

e. [5 points] Non aktifkan venv Anda

- **Lampirkan screenshot terminal yang menampilkan command saat Anda menonaktifkan venv**

```
(.CHALIMA_VENV) chalimasadiah@LAPTOP-CH00NPOB:~/CHALIMA_MLPROCESS$ deactivate
chalimasadiah@LAPTOP-CH00NPOB:~/CHALIMA_MLPROCESS$
```

2. [25 points] Rangkum permasalahan bisnis

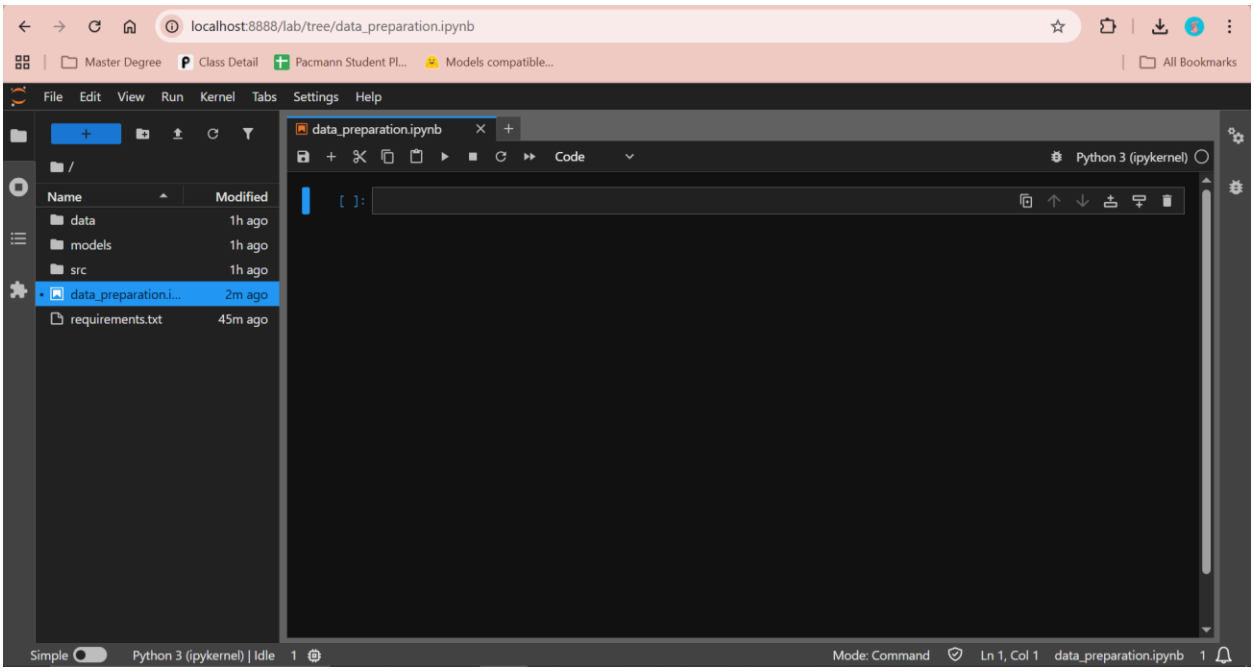
- Rangkum mockup interview antara user dan DS di bagian akhir dari dokumen ini
- **Rangkuman Anda harus bisa menjawab pertanyaan berikut:**

| **PERTANYAAN** | **JAWABAN** |
|---|---|
| Latar belakang permasalahan bisnis | *Produk pinjaman (loan) merupakan salah satu sumber pendapatan utama bank. Namun, bank menghadapi risiko kerugian ketika nasabah gagal membayar pinjaman atau mengalami Non-Performing Loan (NPL). Saat ini, bank membutuhkan sistem yang mampu* |

| | |
|---|---|
| | *mendeteksi potensi gagal bayar lebih awal agar dapat mengambil tindakan preventif sebelum pinjaman benar-benar bermasalah.* |
| Objektif bisnis yang ingin dicapai | *Membangun sistem prediksi risiko kredit yang dapat membantu bank mengidentifikasi nasabah dengan potensi gagal bayar sejak dini, sehingga bank dapat menurunkan jumlah NPL dan meminimalkan potensi kerugian.* |
| Metrik pengukuran bisnis yang akan digunakan | *Keberhasilan solusi akan diukur berdasarkan penurunan jumlah atau persentase Non-Performing Loan (NPL) setelah sistem prediksi diterapkan dalam proses pemberian dan monitoring kredit.* |
| Kandidat solusi machine learning yang akan dibangun | *Solusi yang akan dibangun adalah model machine learning berbasis klasifikasi untuk memprediksi apakah seorang nasabah berpotensi mengalami gagal bayar atau tidak. Beberapa algoritma klasifikasi dapat dieksplorasi dan dibandingkan performanya untuk mendapatkan model terbaik sesuai kebutuhan bisnis.* |
| Metrik pengukuran machine learning yang akan digunakan | *Metrik evaluasi utama yang digunakan adalah **Recall**, khususnya untuk kelas gagal bayar (NPL). Hal ini dikarenakan bank lebih memprioritaskan untuk meminimalkan nasabah berisiko yang tidak terdeteksi oleh sistem. False positive masih dapat ditoleransi karena nasabah yang terdeteksi berisiko akan melalui proses verifikasi lanjutan.* |

3. [25 points] Persiapan Dataset

a. [1 points] Buat satu file bernama data_preparation.ipynb dan letakkan pada root folder project Anda




b. [1 points] Letakkan file CSV dataset yang bernama credit_risk_dataset.csv ke folder raw

**Sertakan screenshot hasil dari pemindahan file tersebut.**

```
(.CHALIMA_VENV) chalimasadiah@LAPTOP-CH00NPOB:~/CHALIMA_MLPROCESS$ mv /mnt/c/Users/ASUS/Downloads/credit_risk_dataset.csv data/raw/
(.CHALIMA_VENV) chalimasadiah@LAPTOP-CH00NPOB:~/CHALIMA_MLPROCESS$
```

c. Folder raw terletak di dalam folder data

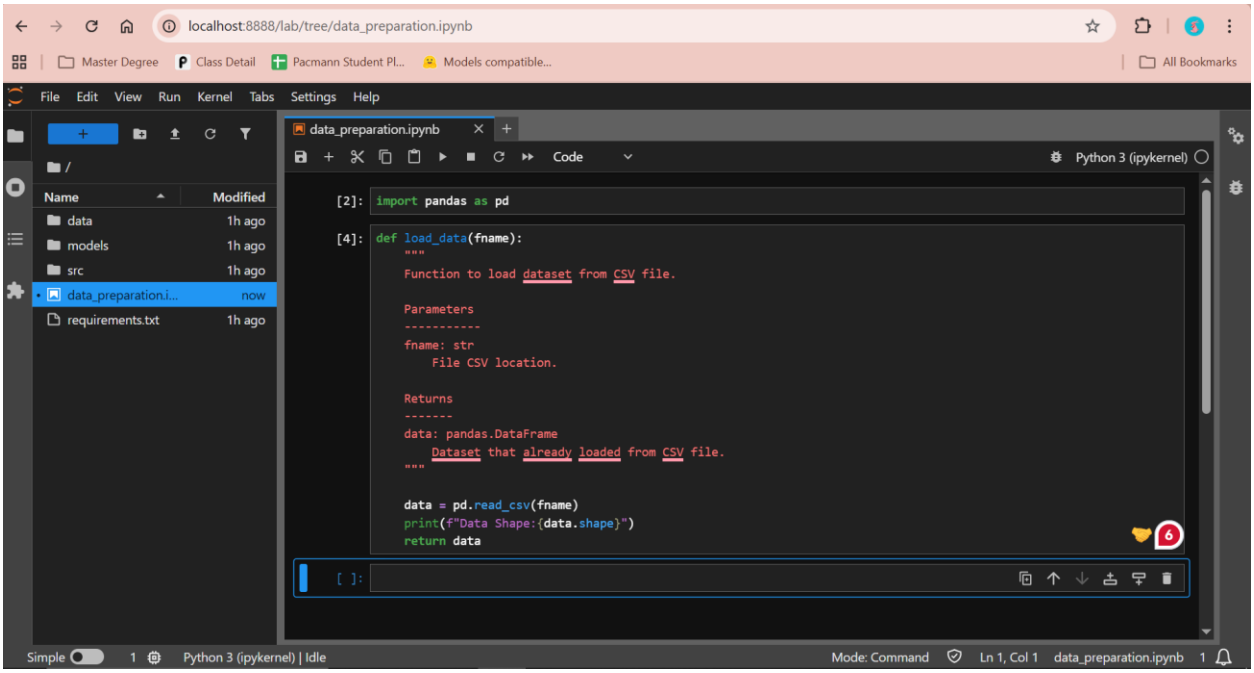
```
(.CHALIMA_VENV) chalimasadiah@LAPTOP-CH00NPOB:~/CHALIMA_MLPROCESS$ mv /mnt/c/Users/ASUS/Downloads/credit_risk_dataset.csv data/raw/
(.CHALIMA_VENV) chalimasadiah@LAPTOP-CH00NPOB:~/CHALIMA_MLPROCESS$ tree
.
├── data
│   ├── interim
│   ├── processed
│   └── raw
│       └── credit_risk_dataset.csv
├── data_preparation.ipynb
├── models
├── requirements.txt
└── src

6 directories, 3 files
(.CHALIMA_VENV) chalimasadiah@LAPTOP-CH00NPOB:~/CHALIMA_MLPROCESS$
```

d. Muat data ke notebook

i.[3 points] Buat fungsi untuk memuat dataset

- Import library pandas sebelum membuat fungsi ini
- Buat fungsi dengan nama load_data
- Fungsi load_data memiliki parameter bernama fname
  - Parameter fname adalah lokasi dimana file CSV dataset berada

- Parameter fname bertipe data string
- Muat data CSV dengan menggunakan fungsi read_csv dari pandas dan simpan dalam variabel bernama data
- Print shape dari data dengan keluaran ”Data Shape: [SHAPE_DATAFRAME]”
- Return data dari fungsi load_data
- Buat docstring untuk fungsi tersebut
- Docstring harus berisi:
  - Penjelasan tentang fungsi tersebut
  - Daftar nama parameter beserta tipe data dan penjelasannya
  - Daftar nama return value beserta tipe data dan penjelasannya

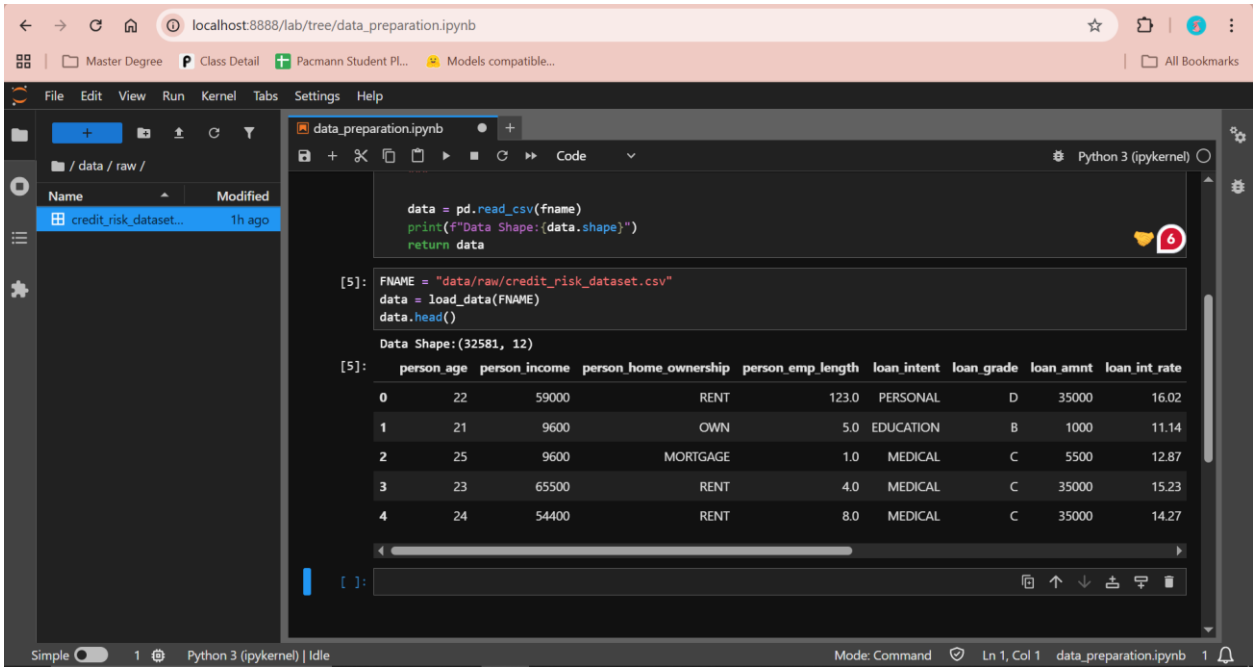



ii. [3 points] Jalankan fungsi

- Buat constant variabel bernama FNAME
- Assign lokasi file dataset ke FNAME
- Panggil fungsi load_data

- Passing FNAME sebagai argumen pada fungsi load_data
- Simpan hasil fungsi load_data ke variabel bernama data
- Cek data yang dimuat dengan menggunakan fungsi head()

**Sertakan screenshot hasil dari fungsi yang dijalankan**




e. Split input dan output dataset

i. [3 points] Buat fungsi untuk split input dan output dataset

- Buat fungsi dengan nama split_input_output
- Fungsi memiliki parameter bernama data dan target_col
  - Parameter data adalah dataset utuh bertipe data DataFrame
  - Parameter target_col adalah nama kolom yang berisi data target, parameter ini bertipe data string
- Buang kolom bernama target_col dan simpan dalam variabel bernama X (upper case)
- Pilih kolom bernama target_col dan simpan dalam variabel bernama y (lower case)
- Print shape dari data dengan keluaran:

”Original data shape: [SHAPE_DATAFRAME]”

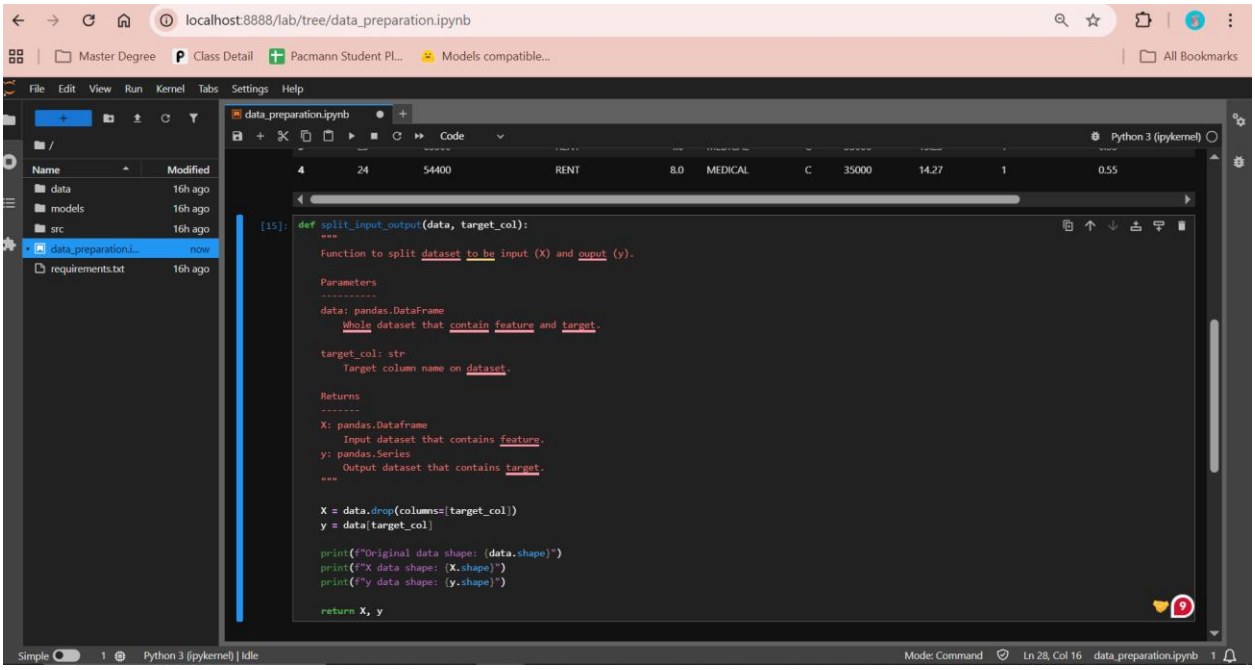
- Print shape dari X dengan keluaran:

  ”X data shape: [SHAPE_DATAFRAME]”

- Print shape dari y dengan keluaran:

  ”y data shape: [SHAPE_DATAFRAME]”

- Kembalikan X dan y
- Buat docstring untuk fungsi tersebut
- Docstring harus berisi:
  - Penjelasan tentang fungsi tersebut
  - Daftar nama parameter beserta tipe data dan penjelasannya
  - Daftar nama return value beserta tipe data dan penjelasannya

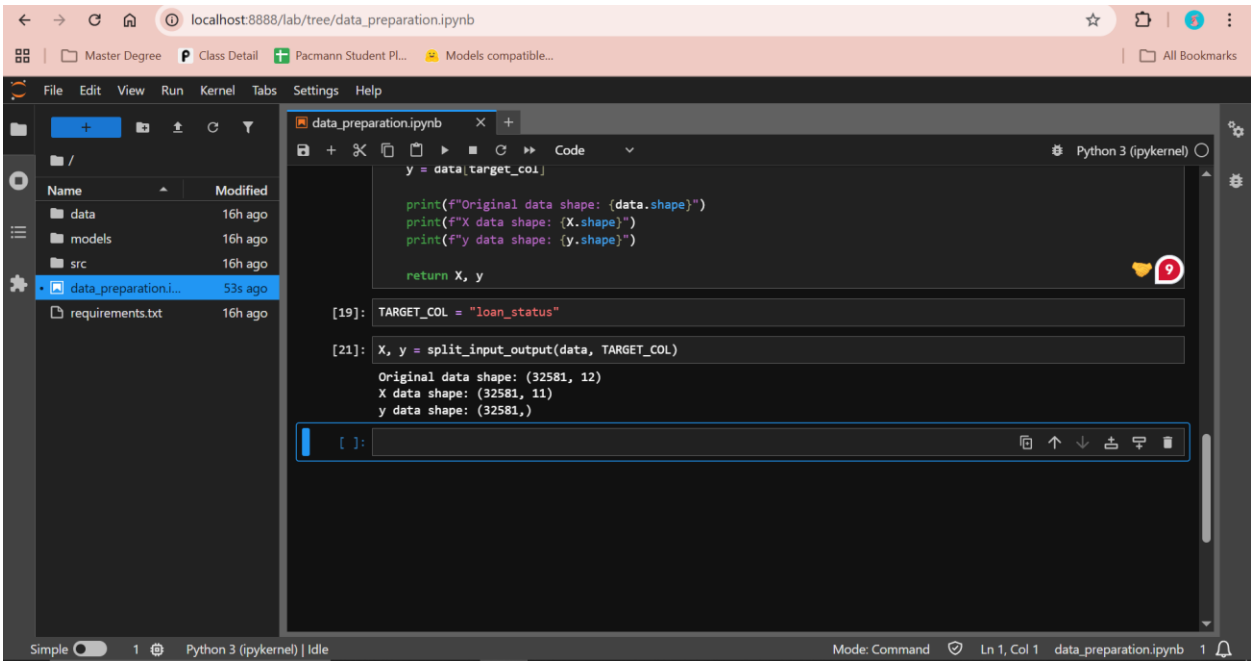



ii. [3 points] Jalankan fungsi

- Siapkan constant variabel bernama TARGET_COL
- Assign nama kolom yang menjadi target pada fungsi TARGET_COL
- Panggil fungsi split_input_output

- Passing dataset utuh dan TARGET_COL sebagai argumen pada fungsi split_input_output
- Simpan kembalian dari fungsi split_input_output ke variabel bernama X dan y

**Sertakan screenshot hasil dari fungsi yang dijalankan**

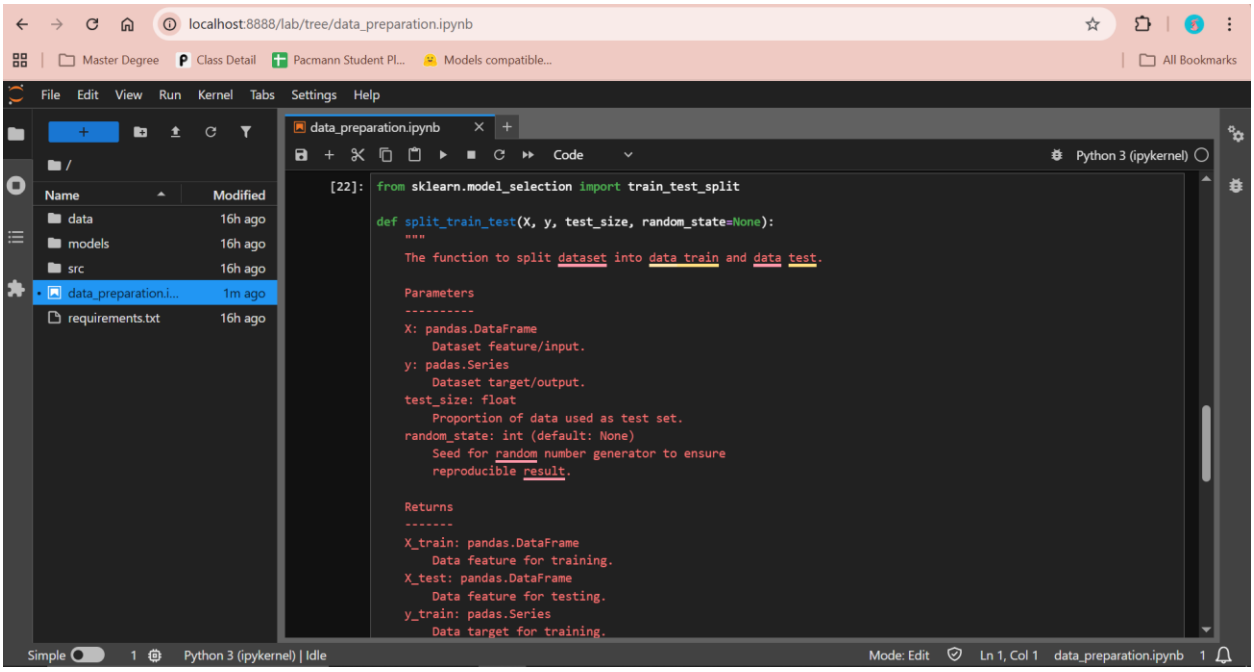


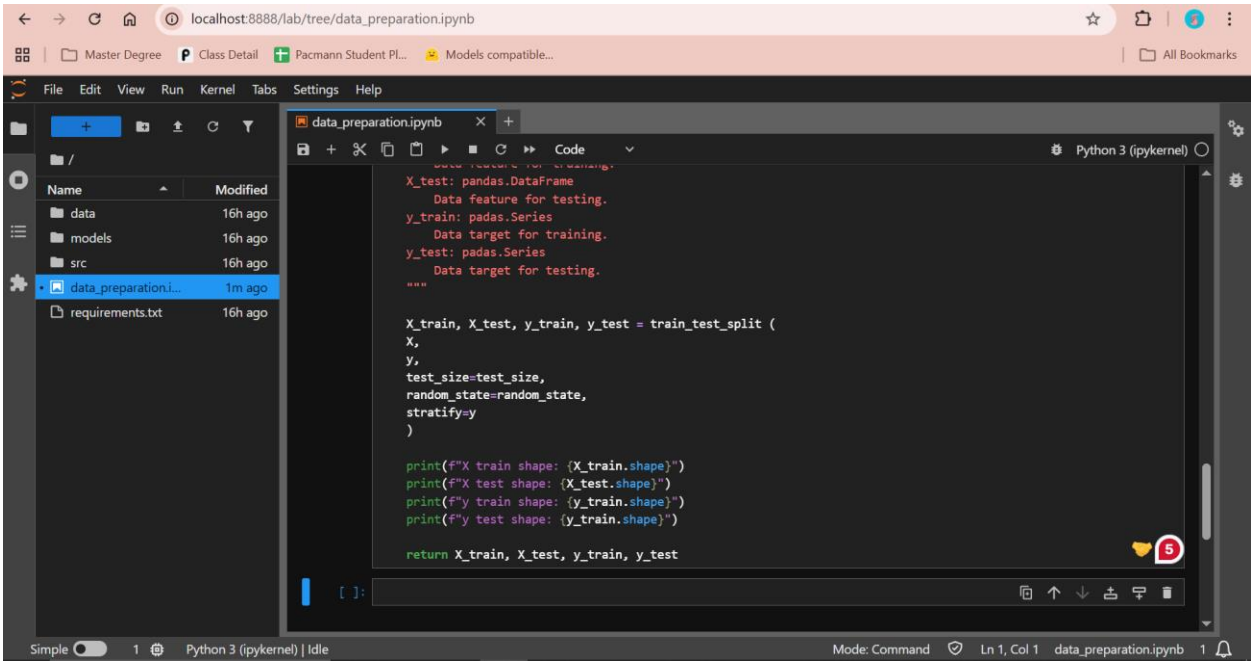

f. Split train dan test dataset

i.[3 points] Buat fungsi untuk split train dan test dataset

- Import fungsi train_test_split dari sklearn.model_selection sebelum membuat fungsi
- Buat fungsi dengan nama split_train_test
- Fungsi memiliki parameter bernama X, y, test_size dan random_state
- Parameter random_state memiliki default value None
- Gunakan fungsi train_test_split untuk memisahkan X dan y menjadi train set dan test set
- Passing nilai test_size dan random_state pada parameter fungsi ke fungsi train_test_split
- Simpan kembalian dari fungsi train_test_split ke variabel X_train, X_test, y_train, dan y_train
- Gunakan stratify ada fungsi train_test_split
- Print shape dari X_train, X_test, y_train, dan y_train dengan keluaran:

    - “X train shape: [SHAPE_X_TRAIN]” untuk X_train
    - “X test shape: [SHAPE_X_TEST]” untuk X_test
    - “y train shape: [SHAPE_Y_TRAIN]” untuk y_train
    - “y test shape: [SHAPE_Y_TEST]” untuk y_test
- Kembalikan  X_train, X_test, y_train, dan y_train
- Buat docstring untuk fungsi tersebut
- Docstring harus berisi:
    - Penjelasan tentang fungsi tersebut
    - Daftar nama parameter beserta tipe data dan penjelasannya
    - Daftar nama return value beserta tipe data dan penjelasannya

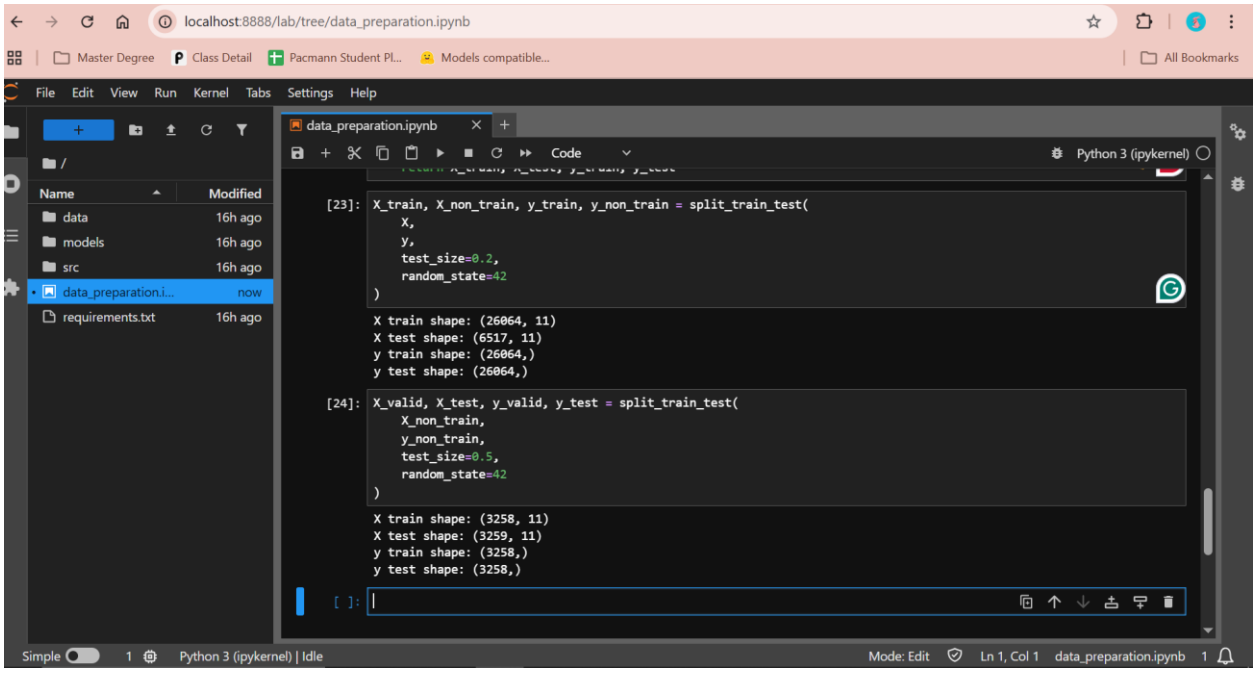





ii. [3 points] Jalankan fungsi

- Panggil fungsi split_train_test
- Passing X dan y ke fungsi split_train_test
- Isi argumen test_size = 0.2 dan random_state = 42 ke fungsi split_train_test
- Simpan keluaran fungsi split_train_test ke variabel bernama X_train, X_non_train, y_train, dan y_not_train
- Panggil lagi fungsi split_train_test
- Passing X_not_train dan y_not_train ke fungsi split_train_test
- Isi argumen test_size = 0.5 dan random_state = 42 ke fungsi split_train_test
- Simpan keluaran fungsi split_train_test ke variabel bernama X_valid, X_test, y_valid, dan y_test

**Sertakan screenshot hasil dari fungsi yang dijalankan**

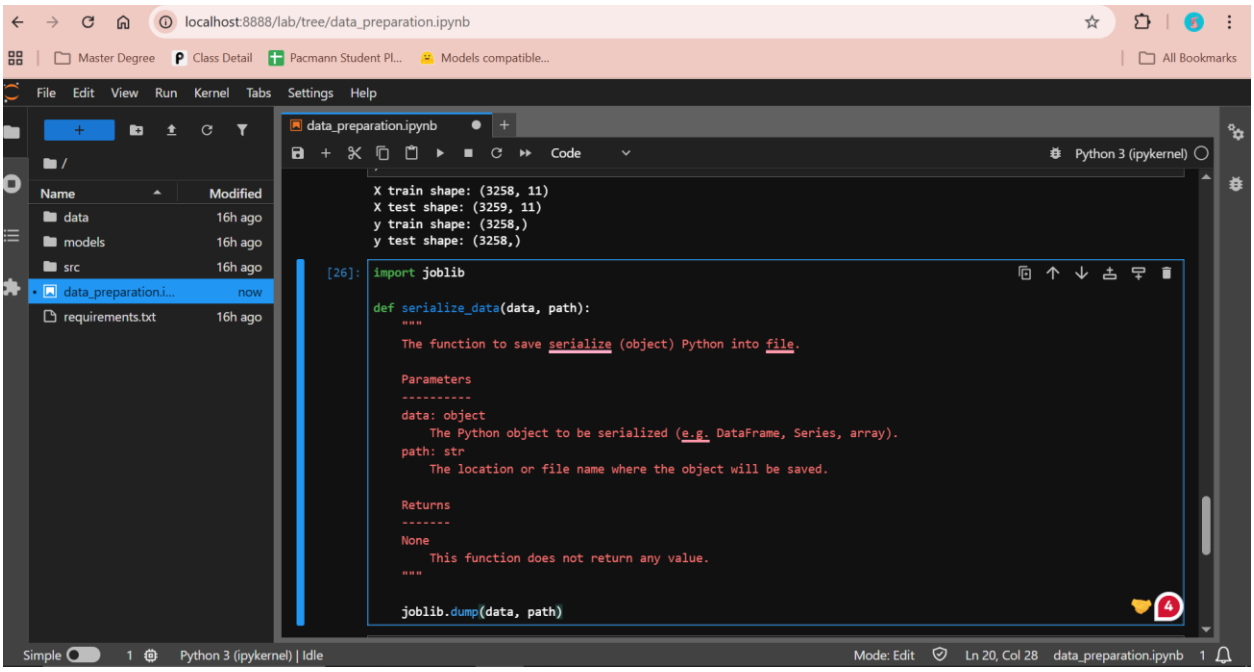



g. Serialize data

i.[2 points] Buat fungsi untuk melakukan serialization

- Import library joblib sebelum membuat fungsi
- Buat fungsi dengan nama serialize_data
- Fungsi memiliki parameter bernama data dan path
    - Parameter data adalah instance yang ingin diserialisasi
    - Parameter path adalah alamat dimana data tersebut ingin diserialisasi
- Gunakan fungsi dump() dari library joblib untuk melakukan serialisasi
- Passing parameter data dan path ke fungsi dump()
- Buat docstring untuk fungsi tersebut
- Docstring harus berisi:
    - Penjelasan tentang fungsi tersebut
    - Daftar nama parameter beserta tipe data dan penjelasannya

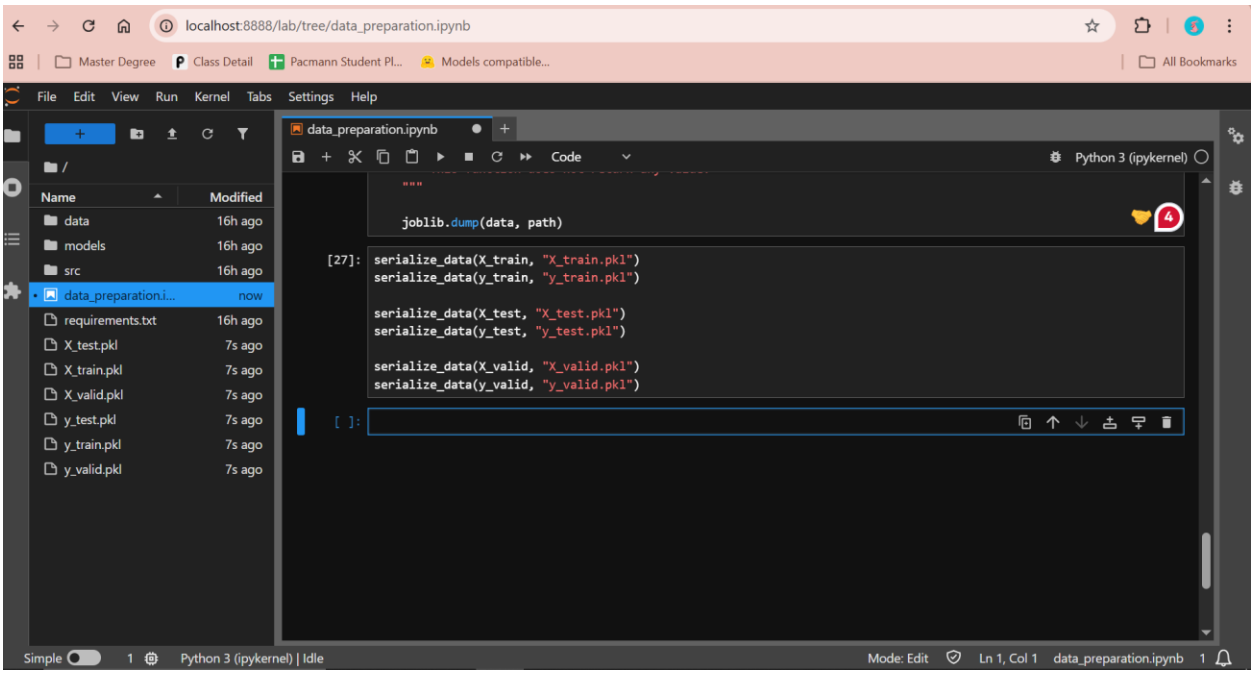
- Daftar nama return value beserta tipe data dan penjelasannya




ii. [2 points] Jalankan fungsi

- Panggil fungsi serialize_data()
    - Untuk serialize data X_train, passing X_train untuk parameter data dan “X_train.pkl” untuk parameter path
    - Untuk serialize data y_train, passing y_train untuk parameter data dan “y_train.pkl” untuk parameter path
    - Untuk serialize data x_test, passing x_test untuk parameter data dan “x_test.pkl” untuk parameter path
    - Untuk serialize data y_test, passing y_test untuk parameter data dan “y_test.pkl” untuk parameter path
    - Untuk serialize data x_valid, passing x_valid untuk parameter data dan “x_valid.pkl” untuk parameter path
    - Untuk serialize data y_valid, passing y_valid untuk parameter data dan “y_valid.pkl” untuk parameter path

**Sertakan screenshot hasil dari fungsi yang dijalankan**

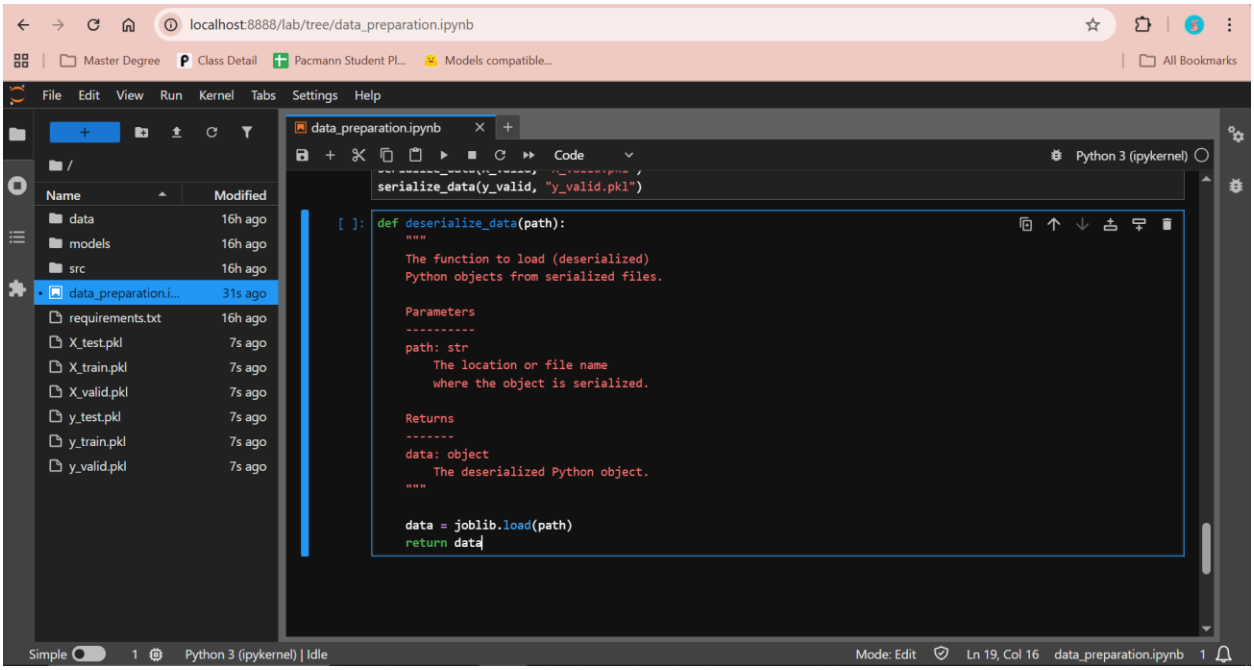



h. Deserialize data

i.[1 points] Buat fungsi untuk melakukan deserialization

- Import library joblib sebelum membuat fungsi jika belum (dapat dijadikan satu dengan fungsi serialize_data())
- Buat fungsi dengan nama deserialize_data
- Fungsi memiliki parameter bernama path
    - Parameter path adalah alamat dimana data tersebut ingin dideserialisasi
- Gunakan fungsi load() dari library joblib untuk melakukan serialisasi
- Passing parameter path ke fungsi dump()
- Simpan kembalian dari fungsi load() ke variabel bernama data
- Kembalikan data
- Buat docstring untuk fungsi tersebut
- Docstring harus berisi:
    - Penjelasan tentang fungsi tersebut

- Daftar nama parameter beserta tipe data dan penjelasannya
- Daftar nama return value beserta tipe data dan penjelasannya




4. [25 points] Buat python utility script

a. [5 points] Buat satu file baru bernama utils.py pada folder src

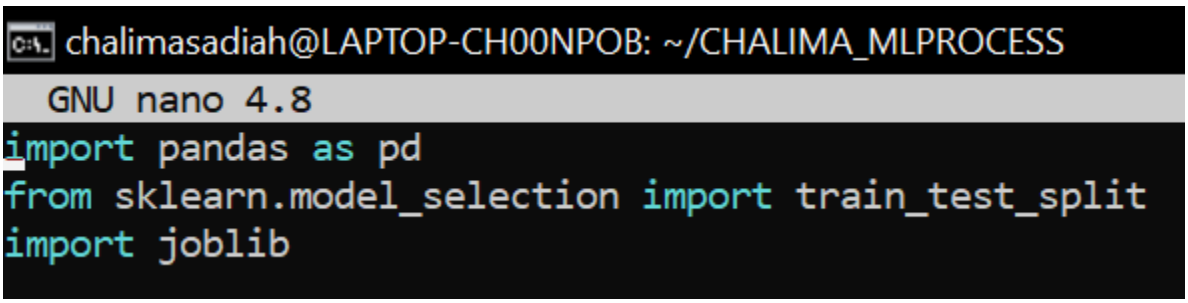
```
(.CHALIMA_VENV) chalimasadiah@LAPTOP-CH00NPOB:~/CHALIMA_MLPROCESS$ touch src/utils.py
(.CHALIMA_VENV) chalimasadiah@LAPTOP-CH00NPOB:~/CHALIMA_MLPROCESS$ tree
.
├── X_test.pkl
├── X_train.pkl
├── X_valid.pkl
├── data
│   ├── interim
│   ├── processed
│   └── raw
│       └── credit_risk_dataset.csv
├── data_preparation.ipynb
├── models
├── requirements.txt
├── src
│   └── utils.py
├── y_test.pkl
├── y_train.pkl
└── y_valid.pkl

6 directories, 10 files
(.CHALIMA_VENV) chalimasadiah@LAPTOP-CH00NPOB:~/CHALIMA_MLPROCESS$
```

b. [5 points] Copy dan paste ketiga library yang digunakan pada data_preparation.ipynb ke utils.py

i. Fungsi DataFrame dari library pandas

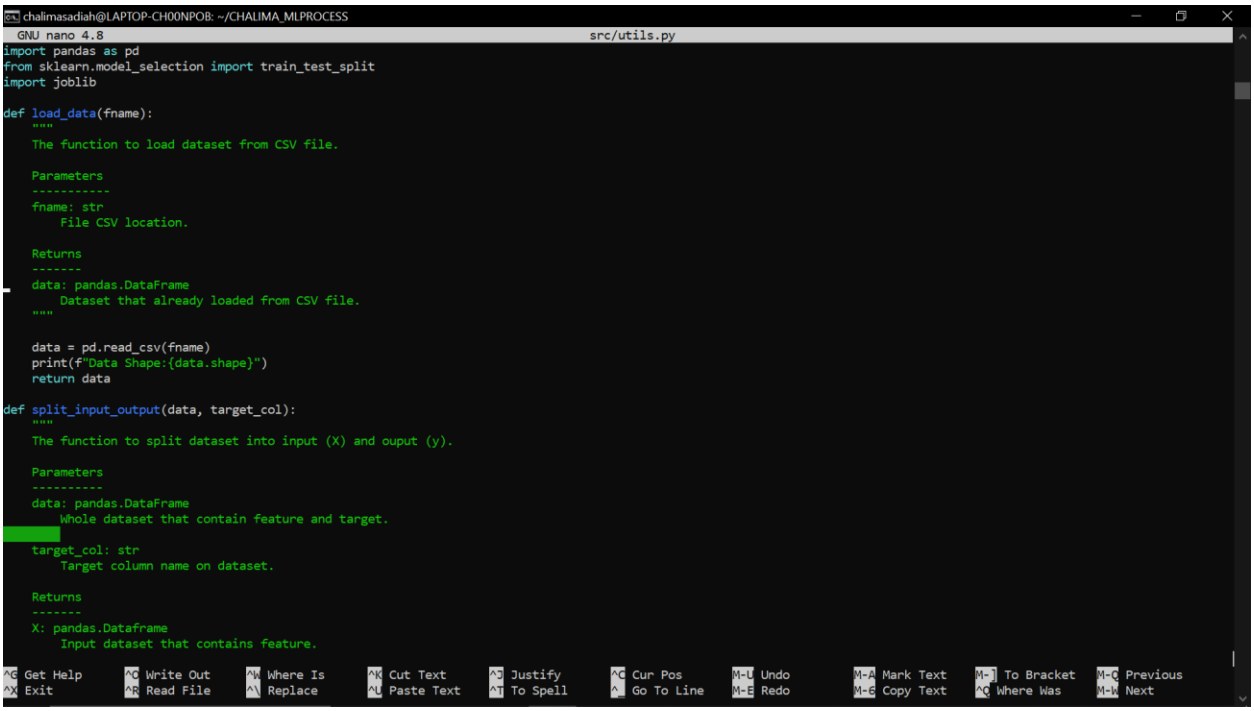
ii. Fungsi train_test_split dari library sklearn.model_selection

iii. Library joblib

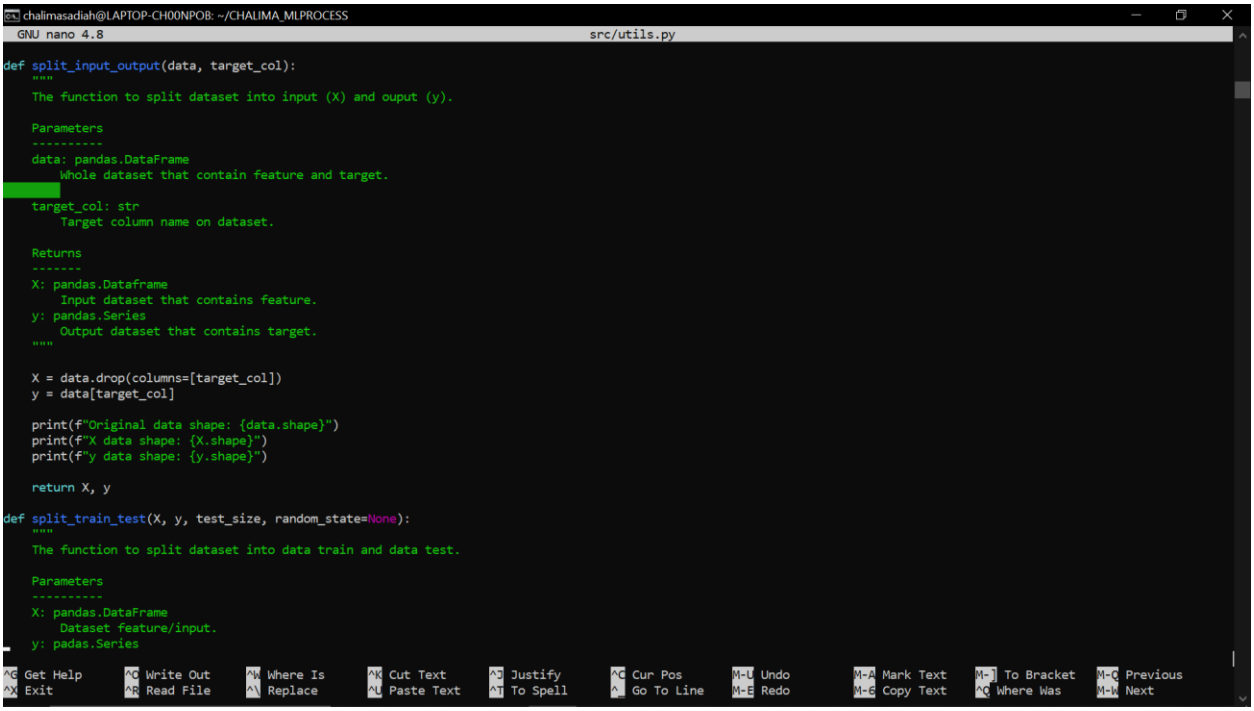



c. [5 points] Copy dan paste ke lima fungsi yang telah dibuat di data_preparation.ipynb ke utils.py

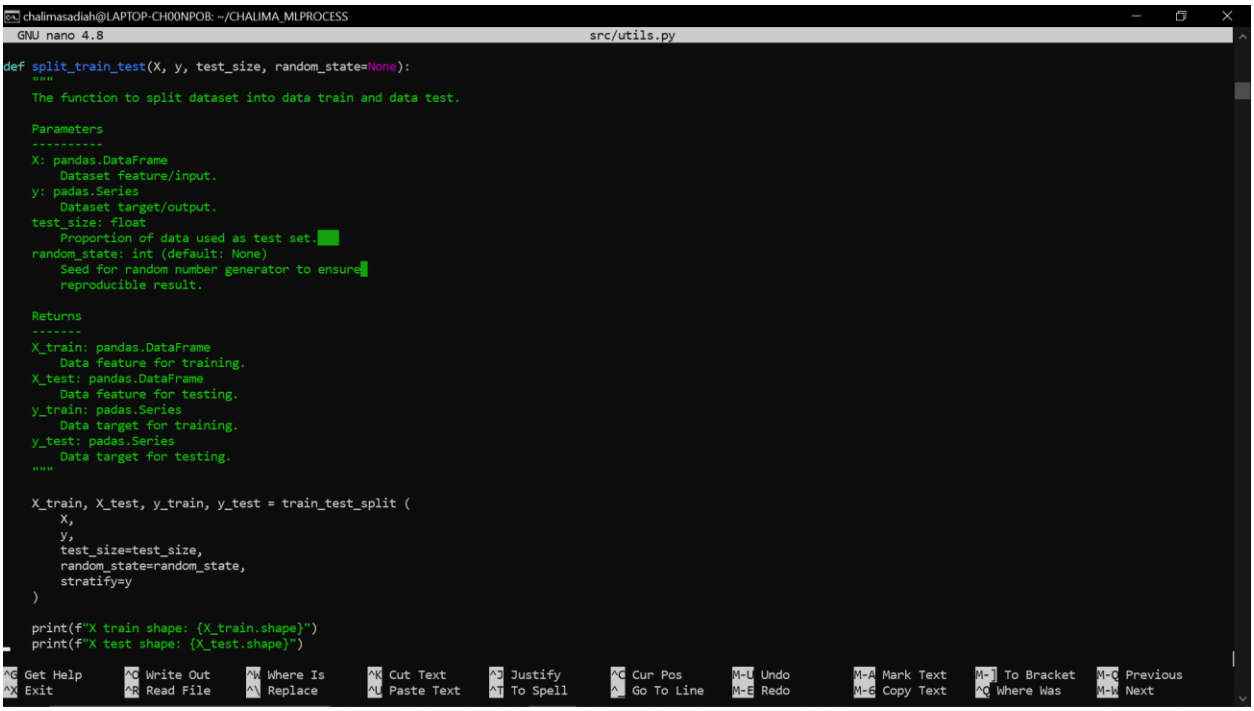
i. load_data()



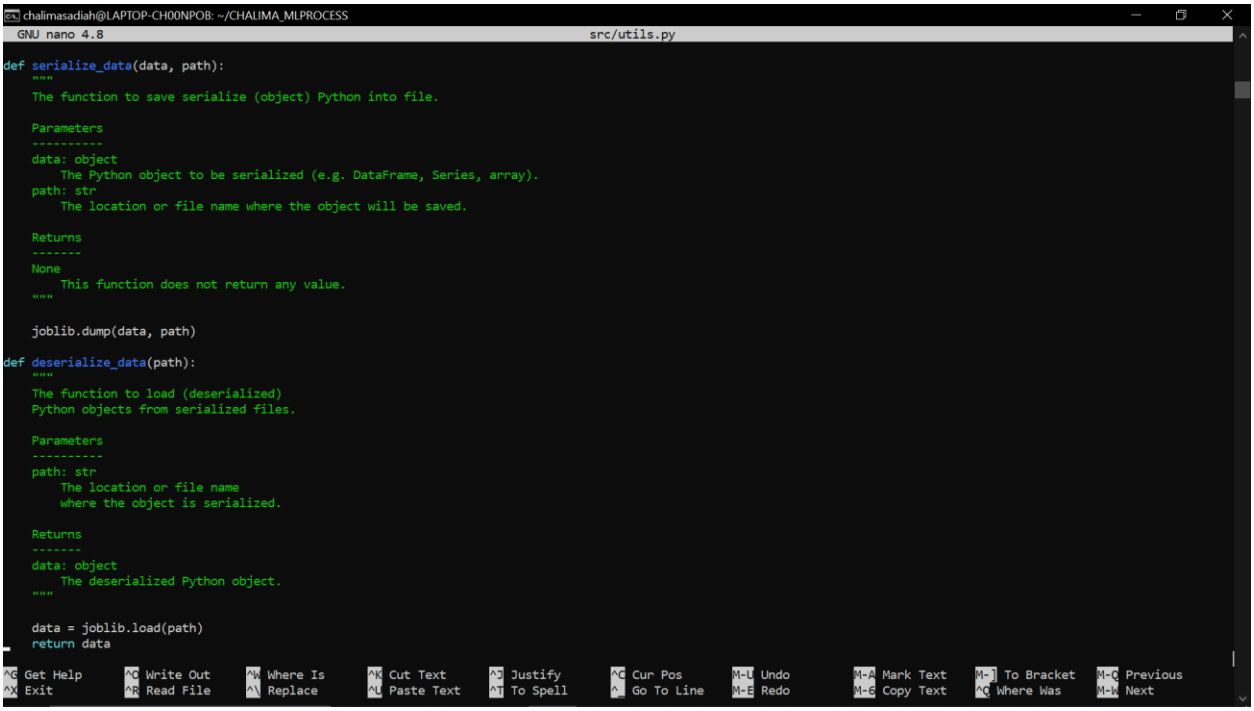

ii.split_input_output()



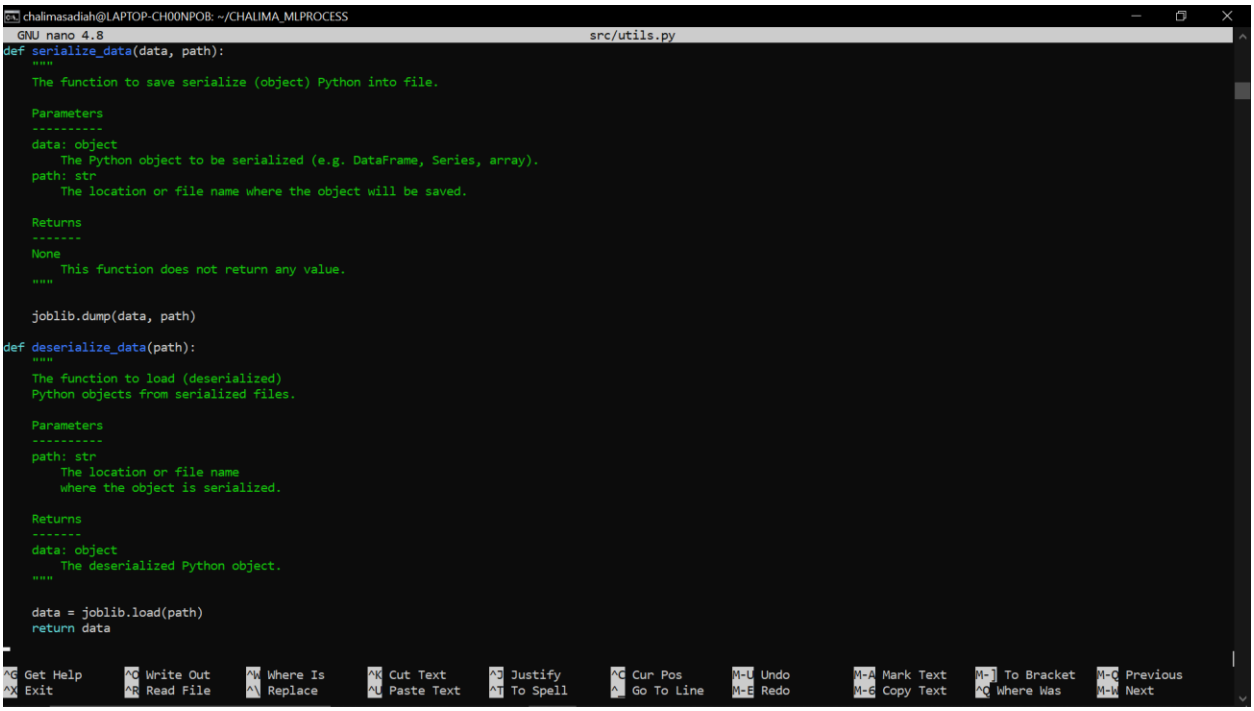

iii.split_train_test()




iv.serialize_data()




v.deserialize_data()

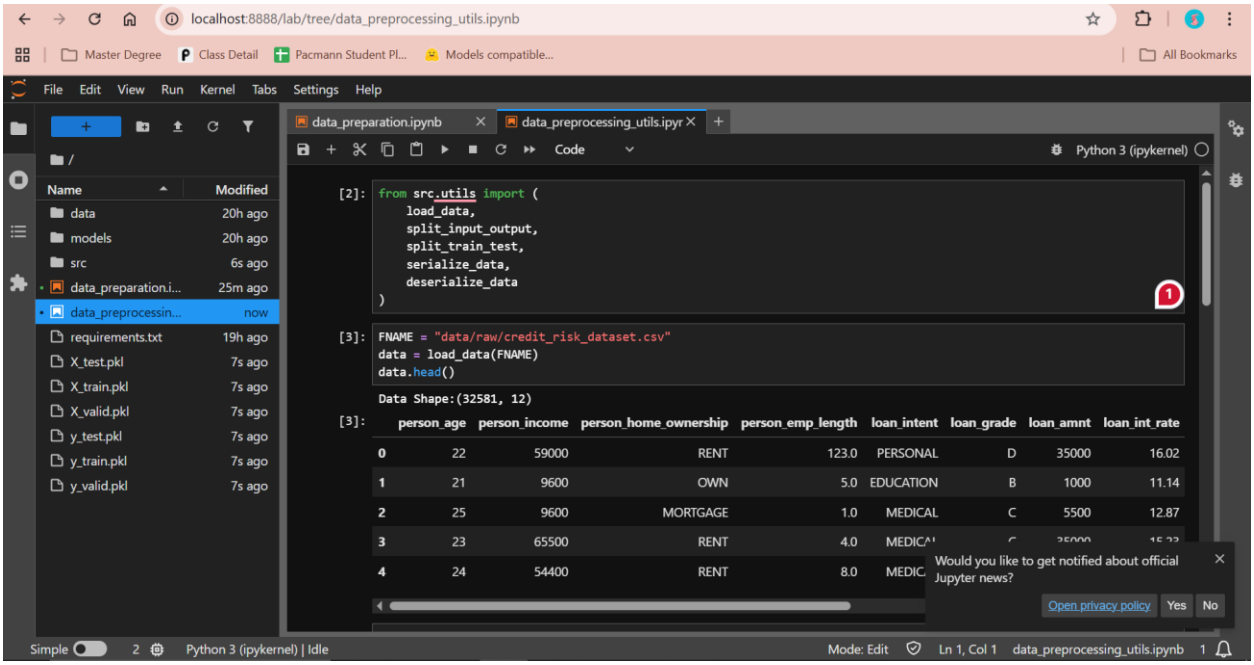


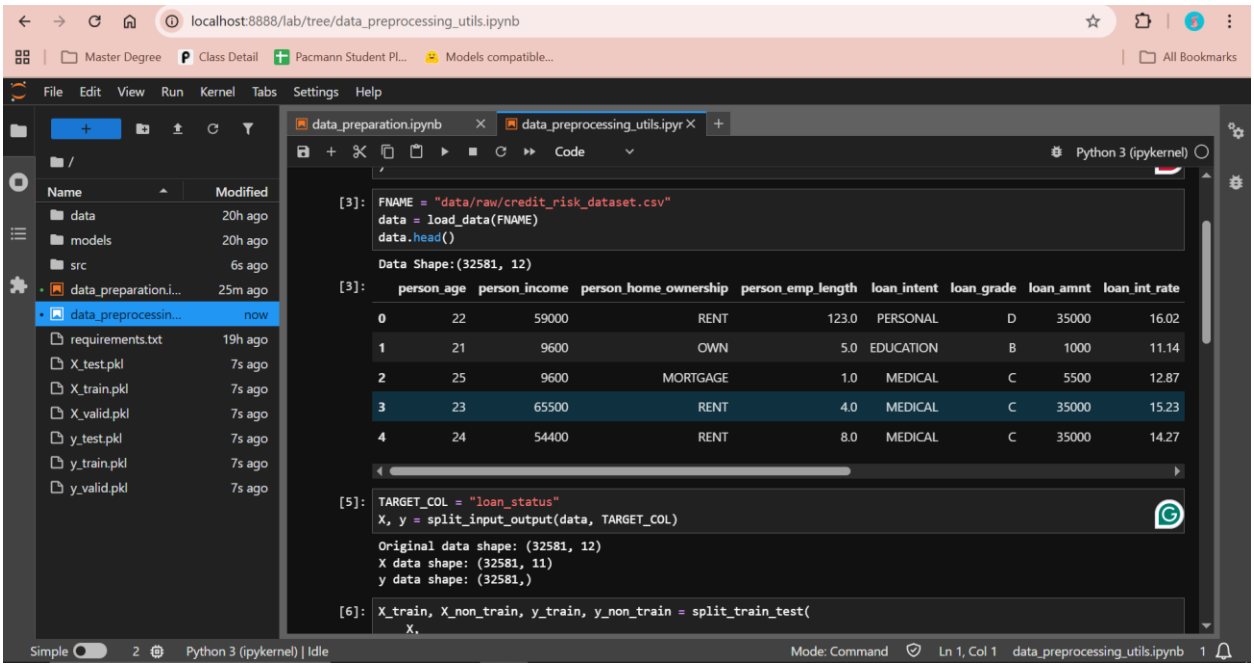

d. [10 points] Guna pengecekan apakah utils.py bisa digunakan, silahkan lakukan hal berikut:

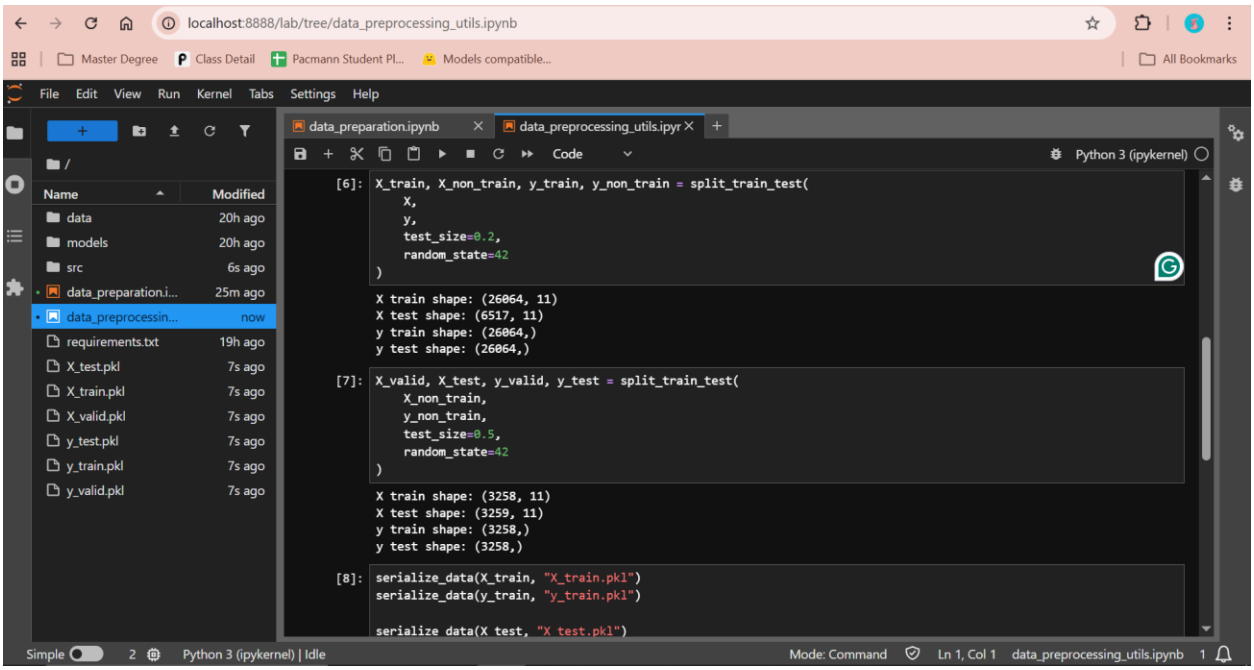
i. Buat satu file baru bernama notebook data_preprocessing_utils.ipynb di root folder project Anda

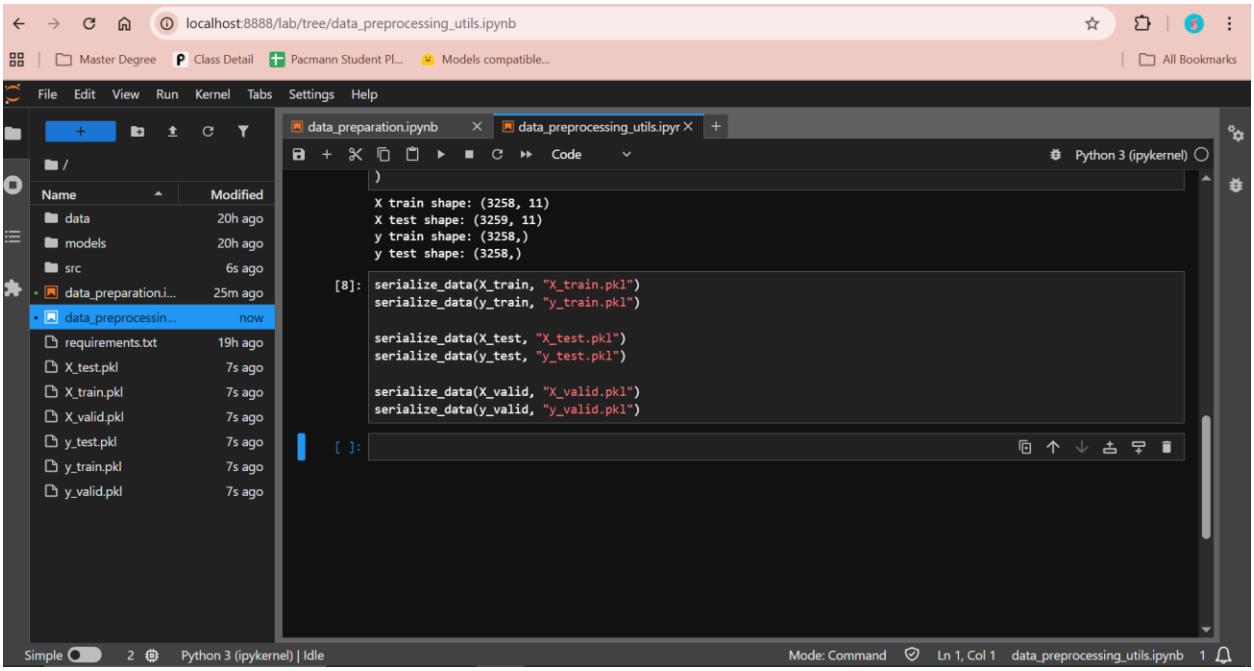
ii. Lakukan ulang apa yang Anda lakukan di data_preprocessing.ipynb namun dengan menggunakan utils.py dibanding Anda menggunakan fungsinya langsung

**Sertakan screenshot hasil dari fungsi yang dijalankan (sertakan screenshot terminal)**


localhost:8888/lab/tree/data_preprocessing_utils.ipynb
Master Degree
Class Detail
Pacmann Student Pl...
Models compatible...
All Bookmarks
File Edit View Run Kernel Tabs Settings Help
data_preparation.ipynb
data_preprocessing_utils.ipyr
Code
Python 3 (ipykernel)
Name
Modified
data 20h ago
models 20h ago
src 6s ago
data_preparation.i... 25m ago
data_preprocessin... now
requirements.txt 19h ago
X_test.pkl 7s ago
X_train.pkl 7s ago
X_valid.pkl 7s ago
y_test.pkl 7s ago
y_train.pkl 7s ago
y_valid.pkl 7s ago
[2]:
from src.utils import (
load_data,
split_input_output,
split_train_test,
serialize_data,
deserialize_data
)
[3]:
FNAME = "data/raw/credit_risk_dataset.csv"
data = load_data(FNAME)
data.head()
Data Shape:(32581, 12)
[3]:
person_age person_income person_home_ownership person_emp_length loan_intent loan_grade loan_amnt loan_int_rate
0 22 59000 RENT 123.0 PERSONAL D 35000 16.02
1 21 9600 OWN 5.0 EDUCATION B 1000 11.14
2 25 9600 MORTGAGE 1.0 MEDICAL C 5500 12.87
3 23 65500 RENT 4.0 MEDICAL C 35000 15.23
4 24 54400 RENT 8.0 MEDICAL C 35000 14.27
Would you like to get notified about official Jupyter news?
Open privacy policy
Yes
No
Simple
Python 3 (ipykernel) | Idle
Mode: Edit
Ln 1, Col 1
data_preprocessing_utils.ipynb


localhost:8888/lab/tree/data_preprocessing_utils.ipynb
Master Degree
Class Detail
Pacmann Student Pl...
Models compatible...
All Bookmarks
File Edit View Run Kernel Tabs Settings Help
data_preparation.ipynb
data_preprocessing_utils.ipyr
Code
Python 3 (ipykernel)
Name
Modified
data 20h ago
models 20h ago
src 6s ago
data_preparation.i... 25m ago
data_preprocessin... now
requirements.txt 19h ago
X_test.pkl 7s ago
X_train.pkl 7s ago
X_valid.pkl 7s ago
y_test.pkl 7s ago
y_train.pkl 7s ago
y_valid.pkl 7s ago
[3]:
FNAME = "data/raw/credit_risk_dataset.csv"
data = load_data(FNAME)
data.head()
Data Shape:(32581, 12)
[3]:
person_age person_income person_home_ownership person_emp_length loan_intent loan_grade loan_amnt loan_int_rate
0 22 59000 RENT 123.0 PERSONAL D 35000 16.02
1 21 9600 OWN 5.0 EDUCATION B 1000 11.14
2 25 9600 MORTGAGE 1.0 MEDICAL C 5500 12.87
3 23 65500 RENT 4.0 MEDICAL C 35000 15.23
4 24 54400 RENT 8.0 MEDICAL C 35000 14.27
[5]:
TARGET_COL = "loan_status"
X, y = split_input_output(data, TARGET_COL)
Original data shape: (32581, 12)
X data shape: (32581, 11)
y data shape: (32581,)
[6]:
X_train, X_non_train, y_train, y_non_train = split_train_test(
X,
Simple
Python 3 (ipykernel) | Idle
Mode: Command
Ln 1, Col 1
data_preprocessing_utils.ipynb


localhost:8888/lab/tree/data_preprocessing_utils.ipynb
Master Degree
Class Detail
Pacmann Student Pl...
Models compatible...
All Bookmarks
File Edit View Run Kernel Tabs Settings Help
data_preparation.ipynb
data_preprocessing_utils.ipyr
Code
Python 3 (ipykernel)
Name
Modified
data 20h ago
models 20h ago
src 6s ago
data_preparation.i... 25m ago
data_preprocessin... now
requirements.txt 19h ago
X_test.pkl 7s ago
X_train.pkl 7s ago
X_valid.pkl 7s ago
y_test.pkl 7s ago
y_train.pkl 7s ago
y_valid.pkl 7s ago
[6]: X_train, X_non_train, y_train, y_non_train = split_train_test(
X,
y,
test_size=0.2,
random_state=42
)
X train shape: (26064, 11)
X test shape: (6517, 11)
y train shape: (26064,)
y test shape: (26064,)
[7]: X_valid, X_test, y_valid, y_test = split_train_test(
X_non_train,
y_non_train,
test_size=0.5,
random_state=42
)
X train shape: (3258, 11)
X test shape: (3259, 11)
y train shape: (3258,)
y test shape: (3258,)
[8]: serialize_data(X_train, "X_train.pkl")
serialize_data(y_train, "y_train.pkl")
serialize_data(X_test, "X_test.pkl")
Simple 2 Python 3 (ipykernel) | Idle
Mode: Command Ln 1, Col 1 data_preprocessing_utils.ipynb 1


localhost:8888/lab/tree/data_preprocessing_utils.ipynb
Master Degree
Class Detail
Pacmann Student Pl...
Models compatible...
All Bookmarks
File Edit View Run Kernel Tabs Settings Help
data_preparation.ipynb
data_preprocessing_utils.ipyr
Code
Python 3 (ipykernel)
Name
Modified
data 20h ago
models 20h ago
src 6s ago
data_preparation.i... 25m ago
data_preprocessin... now
requirements.txt 19h ago
X_test.pkl 7s ago
X_train.pkl 7s ago
X_valid.pkl 7s ago
y_test.pkl 7s ago
y_train.pkl 7s ago
y_valid.pkl 7s ago
)
X train shape: (3258, 11)
X test shape: (3259, 11)
y train shape: (3258,)
y test shape: (3258,)
[8]: serialize_data(X_train, "X_train.pkl")
serialize_data(y_train, "y_train.pkl")
serialize_data(X_test, "X_test.pkl")
serialize_data(y_test, "y_test.pkl")
serialize_data(X_valid, "X_valid.pkl")
serialize_data(y_valid, "y_valid.pkl")
[ ]:
Simple 2 Python 3 (ipykernel) | Idle
Mode: Command Ln 1, Col 1 data_preprocessing_utils.ipynb 1

Mockup Percakapan

| | | |
|---|---|---|
| Pengguna | : | Hai, Pak David! Terima kasih sudah menyempatkan waktu untuk bertemu. |
| David | : | Hai! Tentu saja, tidak masalah. Saya senang bisa membantu. Ada yang bisa saya bantu? |
| Pengguna | | Jadi begini, Pak David, kami di bank sedang memiliki masalah dengan risiko kredit, nih. Kami ingin bikin sistem prediksi yang bisa bantu kita lebih cepet nangkep pinjaman yang berpotensi jadi masalah, gitu. |
| David | | Ah, mengerti. Jadi tujuannya adalah untuk lebih akurat dalam memprediksi pinjaman bermasalah. Keren, tujuan bisnisnya apa nih? |
| Pengguna | | Ya, tentu aja. Kami pengen turunin jumlah NPL (Non-Performing Loan) dan lebih cepet deteksi pinjaman yang berisiko biar bisa langkah preventif lebih awal. |
| David | | Oke, paham. Jadi pengukuran keberhasilannya nanti gimana? |
| Pengguna | | Gampang, nih. Yang jelas pengen liat jumlah NPL yang beneran turun. |
| David | | Nah, kalo soal solusi machine learning, ada preferensi khusus gak? |
| Pengguna | | Hmm, nggak terlalu sih. Yang penting bisa handle data dalam jumlah besar dan kita bisa coba beberapa model yang beda-beda. |
| David | | Bagus. Terus kalo pengen evaluasi kinerjanya gimana? |
| Pengguna | | Ya, kita memperbolehkan adanya false alarm. Ini karena nantinya kita akan saring lagi yang diprediksi sebagai NPL, apakah emang benar akan NPL? Dan kita juga akan beri treatment agar meminimalisir nasabah yang akan NPL. |
| David | | Terakhir, ada insight apa aja dari eksplorasi data awalnya? |

| | | |
|---|---|---|
| Pengguna | | Udah ada beberapa pola menarik terkait perilaku pembayaran pelanggan dan faktor risiko lainnya. Mudah-mudahan nanti model bisa nangkep pola-pola itu dengan baik. |
| David | | Keren, terima kasih udah sharing. Dengan info yang udah kamu kasih, saya bakal buat solusi yang sesuai dengan kebutuhan kamu. |
| Pengguna | | Mantap, Pak David! Kami bener-bener berharap bisa kerja sama dengan Anda. |
| David | | Sama-sama, semoga bisa ngasih hasil yang memuaskan. Kalo ada pertanyaan atau hal lain yang perlu dibahas, jangan ragu buat hubungi gue lagi ya. |
| Pengguna | | Deal! Makasih banyak, Pak David. Sampai ketemu lagi! |
| David | | Sama-sama! Sampai jumpa dan semoga proyeknya lancar ya! |

Dataset

Dataset yang digunakan adalah Credit Risk Dataset.

Hyperlink:

https://www.kaggle.com/datasets/laotse/credit-risk-dataset

Mirror hyperlink:
https://drive.google.com/file/d/1DrYNdjQkPiDxeZzUsaaIoWNxWs71DEO0/view?usp=sharing