UNIVERSITAS
BINA SARANA INFORMATIKA
Gedung Rektorat Jl. Kramat Raya No. 98, Senen. Jakarta Pusat 10450
Telp. (021) 23231170 Fax (021) 21236158 e-mail : rektorat@bsi.ac.id

SURAT TUGAS

Nomor : 706/3.01/UBSI/III/2024

Yang bertanda tangan di bawah ini, Rektor Universitas Bina Sarana Informatika, menugaskan kepada:

No	NIP	Nama
1	201709188	Nani Purwati, M.Kom
2	201107538	Sri Kiswati, ST, MM
3	200809852	Agung Baitul Hikmah, M.Kom
4	199501152	Pudji Widodo, M.Kom

Sebagai Penulis Buku dengan judul "Algoritma Data Science", dengan masa penugasan:

Masa Penugasan : 4 Maret s/d 30 Agustus 2024

Demikian penugasan ini agar dapat dijalankan sebagaimana mestinya. Atas perhatian dan kerjasamanya kami mengucapkan terima kasih.

Jakarta, 1 Maret 2024
Rektor,

Prof. Dr. Ir. Mochamad Wahyudi, M.Kom, MM, M.Pd, IPU, ASEAN Eng.

Tembusan :
1. Divisi SDM
2. Wakil Rektor I & II

TEKNOSAIN

ALGORITMA DATA SCIENCE

Agung Baitul Hikmah, dkk

ALGORITMA DATA SCIENCE

Agung Baitul Hikmah, dkk

TEKNOSAIN

ALGORITMA DATA SCIENCE

Penulis: Agung Baitul Hikmah; Nani Purwati; Sri Kiswati; Puji Widodo; Hendri Mahmud Nawawi; Vincent Christian

Edisi Pertama: Cetakan I ~ 2024

Data Buku:

Format	:	17 x 24 cm
Halaman	:	xiv + 102 halaman
Isi	:	HVS 70 gram
Cover	:	Ivory 260 gram
Finishing	:	Perfect Binding
ISBN	:	978-623-8075-77-5

Buku ini tersedia sumber elektronisnya

Diterbitkan Oleh:

TEKNOSAIN

Ruko Jambusari No. 7A Yogyakarta 55283
Telp. : 0274-4462373
Web. : www.grahailmu.id
Email : info.teknosain@grahailmu.co.id

Teknosain adalah imprint dari CV. Graha Ilmu dengan nomor Keanggotaan IKAPI 016/DIY/01

KATA PENGANTAR

Puji Syukur kehadirat Alloh SWT, hanya karena hidayah dan ridho-Nya, alm ibunda serta dukungan civitas akademika Universitas Bina Sarana Informatika, maka penulisan buku ajar algoritma *data science* ini dapat selesai dikerjakan oleh penulis.

Setelah membaca dan memahami buku ajar ini, diharapkan pembaca mampu untuk memahami dan melakukan pengembangan model berbasis data mengikuti suatu metodologi algoritma *data science*, menentukan objektif bisnis, teknis dan rencana projek *data science*, mengumpulkan data, menganalisis data, menentukan objek atau memilah data, membersihkan data, mengkonstruksi data, membangun model dan dapat melakukan *deployment* model.

Penulis menyadari banyak kekurangan dalam penulisan buku ajar ini. Penulis mengharapkan kritik dan saran membangun untuk memperbaiki isi buku ajar ini.

Tasikmalaya, April 2024

Penulis,

Agung Baitul Hikmah

DAFTAR ISI

-oo0oo-

DAFTAR GAMBAR

-oo0oo-

DAFTAR TABEL

-oo0oo-

BAB 1
PENGANTAR ALGORITMA DATA SCIENCE

Deskripsi:

Materi Pada BAB 1 ini berisi pembahasan mengenai konsep dasar algoritma *data science*, *artificial intelligence* (*AI*), *machine learning* meliputi konsep algotitma *supervised learning* dan *unsupervised learning*, *data meaning*.

Capaian Pembelajaran:

Setelah melakukan pembelajaran ini, pembaca diharapkan:

1. Mampu memahami konsep dasar algoritma *data science.*
2. Mampu memahami konsep dasar algoritma *artificial intelligence* (*AI*).
3. Mampu memahami konsep dasar *machine learning*.
4. Mampu memahami konsep dasar *data meaning*.

1.1. Konsep Dasar Algortima *Data Science*

Algoritma merupakan serangkaian prosedur yang saling berinteraksi untuk mencapai suatu tujuan dalam memecahkan masalah tertentu. Algoritma dapat digunakan untuk proses penghitungan, pemrosesan data, pencarian, penalaran, optimasi, pembelajaran dan sejenisnya untuk menyelesaikan suatu masalah.

Data science merupakan bidang yang menggabungkan beberapa disiplin ilmu matematika, statistic dan ilmu komputer. *Data science* mampu menganalisa kumpulan data dalam jumlah yang besar dan begitu kompleks, baik data tersebut terstruktur maupun data yang tidak terstruktur yang digunakan untuk membuat keputusan bisnis.

Algoritma *data science* merupakan sebuah instruksi yang digunakan untuk mengolah data dengan tujuan mendapatkan sebuah pengetahuan baru. Algoritma *data science* bermanfaat dalam memberikan urutan data yang terstruktur.

1.2. Konsep Dasar *Artificial intelligence (AI)*

Artificial intelligence merupakan sebuah mesin yang dapat melakukan tugas dengan kemampuan berfikir rasional, bertindak rasional, berfikir seperti manusia dan bertindak seperti manusia. *Artificial intelligence* juga melibatkan penggunaan algoritma, model matematika, dan teknik komputasi lanjutan. *Artificial intelligence* juga memerlukan logika untuk menjalankan berbagai sistem, oleh sebab itu *artificial intelligence* dapat mengambil sebuah keputusan.

Dalam perkembangannya, teknologi *artificial intelligence* diturunkan menjadi beberapa cabang domain, diantaranya adalah *natural language processing*, *speech processing* dan *image processing*.

Cabang domain *artificial intelligence natural language processing* merupakan teknologi *artificial intelligence* pada domain teks bahasa yang mencakup pemahaman dan pembangkitan teks. Input berupa teks dapat berasal dari berbagai sumber seperti dokumen digital, suara yang ditranskripsi menggunakan ASR (*Automatic speech recognition*), bahasa isyarat yang diterjemahkan menggunakan *Sign Language Recognition* atau dokumen cetak yang ditransformasi menjadi dijital menggunakan *OCR*.

Perbedaan antara modul pemahaman dan pembangkitan teks adalah jenis input atau outputnya yang harus berupa teks. Untuk modul pemahaman teks, inputnya harus berupa teks. Sedangkan untuk modul pembangkitan teks, outputnya harus berupa teks. Modul yang kedua input dan outputnya berupa teks biasanya digolongkan ke dalam modul pembangkitan teks, seperti *text summarization, machine translation*, dan *chatbot* contoh aplikasi seperti *chatgpt, gemini, perplexity,copilot*.

Cabang domain artificial intelligence speech processing. Speech processing merupakan teknologi artificial intelligence pada domain

teknologi input atau output berupa gelombang suara, yang mencakup speech recognition, speech synthesis atau text to speech, speaker recognition dan para linguistic.

Cabang domain *artificial intelligence image processing* yang merupakan teknologi *artificial intelligence* pada domain gambar. *Image processing* mencakup analisis gambar dan pembangkitan gambar. *Image processing* juga mencakup pemrosesan video yang dapat diasumsikan sebagai rangkaian gambar.

1.3. Konsep Dasar *Machine Learning*

Machine learning adalah bagian dari ilmu komputer yang bertujuan untuk mengenali pola dan belajar dari data untuk menghasilkan prediksi yang benar. Ini dapat dianggap sebagai bentuk kecerdasan buatan yang mendukung analisis pemerintah, laporan medis, keputusan bisnis, manajemen risiko keuangan dan area lain di mana keputusan dan pengoptimalan didasarkan pada informasi yang disimpan secara digital. Karena peningkatan jumlah data yang disimpan oleh perusahaan di seluruh dunia dan beberapa terobosan dalam perangkat lunak yang bekerja, *machine learning* semakin penting dalam industri. *Machine learning* mampu menggali pengetahuan, memprediksi, maupun mengenal pola dari masalah kompleks yang dihadapi, dapat memodelkan aturan-aturan dengan memodelkan aturan-aturan melalui dengan algoritma-algoritma.

Machine learning dibagi menjadi dua, yakni *unsupervised learning* dan *supervised learning*. *Supervised learning* merupakan pembelajaran menggunakan data berlabel yang mengandung input serta output yang diinginkan. Klasifikasi dan regresi merupakan bentuk dari *supervised learning.* Klasifikasi digunakan untuk mengelompokkan data ke dalam kategori tertentu sedangkan regresi digunakan untuk memprediksi nilai numerik dari data baru. *Unsupervised learning* merupakan pembelajaran menggunakan data yang hanya memiliki *input* tetapi tidak mempunyai *output* spesifik. Fungsi utama dari *unsupervised learning* adalah menemukan fitur atau *pattern* dan akan mengkategorikan hasilnya sebagai prediksi. *Clustering* merupakan bentuk dari *unsupervised learning. Clustering*

digunakan untuk mengelompokkan data ke dalam kelompok-kelompok yang memiliki kesamaan berdasarkan ciri-cirinya.

1.4. Konsep Dasar *Data Meaning*

Istilah *data mining* mulai populer di komunitas pengguna basis data pada tahun 1990-an. Namun, teori dan metode dasar dari *data mining* telah lahir jauh sebelum era 90. *Data mining* berasal dari berbagai disiplin ilmu, dua ilmu yang paling mendasari adalah statistika dan *machine learning*.

Teori-teori statistika yang berakar dari teori matematika yang menitikberatkan pada pembentukan model. Model merupakan asumsi atau pendekatan struktur yang mendekati data sesungguhnya. Sementara itu, *machine learning* lebih mementingkan pengembangan algoritma.

Data Mining adalah proses yang menggunakan satu atau lebih teknik pembelajaran komputer (*machine learning*) untuk menganalisis dan mengekstrak pengetahuan secara otomatis.

Definisi lain adalah pembelajaran berbasis induksi adalah proses pembentukan definisi konsep umum yang dilakukan dengan mengamati contoh dan konsep spesifik yang akan dipelajari. *Data mining* merupakan ilmu yang memanfaatkan data yang sebelumnya kurang terpakai untuk mendapatkan suatu informasi atau pengetahuan baru.

1.5. Tugas dan Proyek Latihan

Berikan contoh model *machine learning unsupervised learning* dan *supervised learning*. Tuliskan deskripsi dari kedua model *unsupervised learning* dan *supervised learning* tersebut meliputi tujuan, data, algoritma dan hasil.

BAB 2
TOOLS PROYEK DATA SCIENCE

Deskripsi:

Materi pada BAB 2 ini berisi pembahasan *tools data science* dengan menjelaskan bahasa pemrograman yang digunakan dan teknik yang berkaitan dengan keterampilan dasar dalam ilmu komputer, matematika, dan statistik untuk melakukan tugas-tugas yang umumnya terkait dengan *data science* menggunakan bahasa pemograman *python* berikut proses *installasi* dan kegunaan *library python*.

Capaian Pembelajaran:

Setelah melakukan pembelajaran ini, pembaca diharapkan:

1. Mampu memahami tools yang digunakan untuk proyek *data science*.
2. Mampu memahami bahasa pemograman *python*.
3. Mampu memahami proses installasi dan kegunaan *library python*.

2.1. *Tools* Proyek *Data Science*

Dalam buku ini, akan dibahas beberapa *tools* yang digunakan untuk proyek *data science*. *tools* adalah salah satu unsur penting seorang *programmer* dalam keberhasilan dan peningkatan kinerja dalam proyek *data science*. Pemilihan *tools* yang tepat dapat menghemat banyak waktu pengerjaan dan memungkinkan seorang *programmer* untuk focus lebih banyak pada analisis data. Hal mendasar yang perlu diputuskan adalah memilih bahasa pemrograman yang akan digunakan.

2.1.1. Bahasa Pemrograman *Python*

Python merupakan bahasa pemrograman bersifat *open-source* dan bagi *programmer* pemula *python* memiliki sejumlah properti yang lengkap kelebihan dari properti yang dimiliki *python* adalah kodenya mudah dibaca, pengetikan dan penggunaan memori yang dinamis, sehingga ideal untuk orang yang belum pernah belajar program.

Python juga bersifat *cross-platform*, *python* dapat digunakan pada sistem operasi Windows, Mac OS, dan Linux. *python* juga merupakan bahasa *interpreter*, dimana kode dieksekusi di konsol *python* tanpa memerlukan langkah *compailer* ke bahasa mesin.

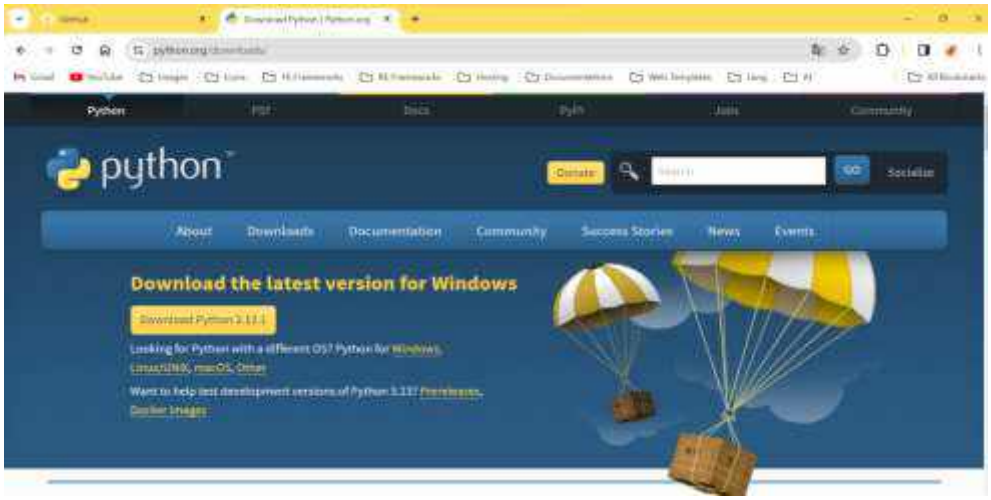
Selain installasi secara offline konsol python juga bisa dijalankan melalui web bebrasis IDE, sejalan dengan perkembangan bahasa pemograman *python*, *Google* menerapkan bahasa pemograman *python yang* diberi nama *Google Interactive Notebook* atau dikenal dengan nama *Google Colaboratory*. Pada umumnya orang mengenal web tersebut disingkat dengan nama *Google Colab*. *Google Colab* merupakan produk berbasis *cloud computing* atau komputasi awan yang dapat digunakan secara gratis bisa diakses di link https://colab.research.google.com/.

2.1.2. *Installasi Python*

Beberapa tools yang digunakan dan compatible dengan aplikasi *python* diantaranya:

1. Installasi Aplikasi *Python*

Langkah pertama, proses penginstallan aplikasi *python*. Untuk mengunduh aplikasi *python* dapat di unduh di link halaman resminya di https://www.python.org/downloads/ seperti yang terlihat pada gambar 2.1.



Gambar 2.1. Download Aplikasi *Python*

Langkah kedua, klik pada file installer yang telah di unduh sebelumnya, kemudian pilih tulisan install now, seperti yang terlihat seperti tampilan di gambar 2.2. Untuk settingannya gunakan secara default.

Gambar 2.2. Installasi Python

Langkah ketiga, jika proses *installasi* berhasil maka akan muncul tampilan seperti gambar 2.3, tidak perlu lagi menginstal pip di konsol,

karena sudah otomatis terinstal. Kemudian klik tombol *close*. Aplikasi *python* sudah berhasil terinstall di laptop.

Gambar 2.3. Installlasi Python Sukses

2. Installasi Aplikasi *Visual Studio Code*

Langkah pertama, proses penginstallan aplikasi pemrograman *visual studio code*. Untuk mengunduh aplikasi *visual studio code* dapat di unduh di link halaman resminya di https://code.visualstudio.com/download seperti yang terlihat pada gambar 2.4.

Gambar 2.4. Download *Visual Studio Code*

Langkah kedua, klik dua kali pada file installer visual studio code yang di unduh. Pastikan kalian klik checkbox "*I accept the agreement*", lalu klik tombol next. Biarkan settingannya default, klik install seperti terlihat pada gambar 2.5.

Gambar 2.5. Installasi Visual Studio Code

Langkah ketiga, jika proses installasi berhasil maka akan muncul tampilan seperti pada gambar 2.6. Klik tombol finish, *visual studio code* berhasil terinstall di laptop.

Gambar 2.6. Installasi Visual Studio Code Sukses

2.1.3. *Library Python*

Python banyak digunakan pada proyek *data science* teruatama untuk tugas yang terkait dengan pengolahan data, seperti eksplorasi data, pemrosesan data, pembersihan data, dan pemodelan data. Library Dasar Python untuk Data Science yang paling populer untuk proyek *data science* adalah *NumPy, SciPy, Pandas, Scikit-Learn, Seaborn*, dan *Matplotlib.*

1. *NumPy and SciPy: Numeric and Scientifc Computation*

NumPy merupakan salah satu *library python* yang digunakan untuk operasi *array* dan operasi matematika dan termasuk *library* yang paling dasar diajarkan di bagian paling awal untuk melakukan pengolahan data di *python* untuk proyek *data science*. *Library NumPy* digunakan untuk mengolah data-data dengan karakteristik big data, yaitu *volume, variety, velocity, dan veracity*.

Untuk bisa menggunakan *NumPy*, langkah pertama adalah mengimpor *library* tersebut dengan menggunakan perintah:

```
import numpy
```

Untuk bisa mengksplor konten help dan dokumentasi dari *library NumPy*, dengan menggunakan perintah:

```
numpy?
```

2. Pandas: Python Data Analysis Library

Pandas merupakan salah satu *library python* yang digunakan untuk menggabungkan data *aggregating, merging, joining*. *Pandas* juga dapat digunakan untuk mengimpor dan mengekspor data dari berbagai format seperti *comma-separated value* (*CSV*), file teks, *Microsoft Excel*, *database SQL*, dan *format HDF5*. *Pandas* dapat digunakan untuk melakukan proses pemisahan data atau *splitting*. *Pandas* juga dapat digunakan untuk menganalisis data yang bersifat *time-series*, yaitu data yang perlu dianalisis secara berkala.

Untuk bisa menggunakan *Pandas*, langkah pertama adalah mengimpor *library* tersebut dengan menggunakan perintah:

```
import pandas
```

Untuk bisa mengksplor konten help dan dokumentasi dari *library Pandas*, dengan menggunakan perintah:

```
pandas?
```

3. *Matplotlib and Seaborn: Data Visualization*

Matplotlib merupakan salah satu *library python* yang digunakan untuk visualisasi data dua dimensi dengan menggunakan berbagai variasi bagan atau diagram, seperti histogram, diagram batang, diagram garis, diagram lingkaran. *Matplotlib* merupakan *python* 2D *ploting library* yang memiliki banyak *function* untuk melakukan beberapa jenis plot gambar. Salah satu fitur terpenting *Matplotlib* adalah kemampuannya yang baik dengan banyak sistem operasi dan *backend* grafis. *Matplotlib* mendukung beberapa jenis *backend* dan *output*. *Python* juga memiliki *library* sejenis untuk visualisasi data, yaitu *seaborn* yang memiliki tampilan lebih estetis dari *Matplotlib*.

Seaborn merupakan *library python* yang dibangun diatas *library Matplotlib* dan digunakan untuk visualisasi data statistik yang menarik dan informatif. Dengan antarmuka yang mudah digunakan dan plot yang estetis, *Seaborn* menjadi pilihan populer untuk visualisasi data dalam proyek ilmu data dan analisis.

Untuk bisa menggunakan *Matplotlib*, langkah pertama adalah mengimpor *library* tersebut dengan menggunakan perintah:

```
import matplotlib.pyplot as plt
```

Untuk bisa mengksplor konten help dan dokumentasi dari *library Matplotlib*, dengan menggunakan perintah:

```
matplotlib?
```

Untuk bisa menggunakan *Seaborn*, langkah pertama adalah mengimpor *library* tersebut dengan menggunakan perintah:

```
import seaborn as sns
```

Untuk bisa mengksplor konten help dan dokumentasi dari *library Seaborn*, dengan menggunakan perintah:

```
seaborn?
```

4. *SCIKIT-Learn: Machine Learning in Python*

SCIKIT-Learn merupakan salah satu *library python* yang digunakan untuk menerapkan tugas analisis data dan *machine learning*, seperti classification, regression, clustering. Algoritma *machine learning* yang dapat diterapkan menggunakan *SCIKIT-Learn* diantaranya *Support Vector Machine, Decision Tree, Random Forest, K-Means Clutering, dan Neural Network.*

SCIKIT-Learn dibangun dari library *NumPy, SciPy,* dan *Matplolib*. Scikit-Learn berfokus pada tugas pemodelan data daripada tugas manipulasi dan visualisasi data.

Untuk bisa menggunakan *SCIKIT-Learn*, langkah pertama adalah mengimpor *library* tersebut dengan menggunakan perintah:

```
import sklearn
```

Untuk bisa mengksplor konten help dan dokumentasi dari *library SCIKIT-Learn*, dengan menggunakan perintah:

```
sklearn?
```

2.1.4. *Web Integrated Development Environment (WIDE)*

Dengan munculnya aplikasi web, generasi baru IDE untuk bahasa interaktif seperti bahasa pemograman *python*. Pengembangan IDE berbasis web dikembangkan dengan mempertimbangkan agar kode dan seluruh *environment* dapat disimpan di server. Berikut ini aplikasi *Web Integrated Development Environment (WIDE)*:

1. *Jupiter Notebook*

Jupyter Notebook merupakan singkatan dari beberapa bahasa pemrograman seperti Julia (Ju), Python (Py) dan R. Tiga Bahasa pemrograman ini sangat penting bagi seorang *data* science. *Jupyter Notebook*

awalnya dikembangkan oleh tim peneliti yang dipimpin oleh *Fernando Perez* di *University of California, Berkeley*.

Jupyter Notebook pertama kali dirilis pada tahun 2014 sebagai *spin-off* dari proyek *IPython*, yang berfokus pada penyediaan shell interaktif untuk *python*. *Jupyter Notebook* bertujuan untuk menggunakan kembali web, generasi baru IDE yang sama untuk semua bahasa *interpreter* dan tidak hanya menggunakan bahasa pemrograman *python*. Semua kode notebook *python* lama dapat diimpor secara otomatis ke versi baru saat dibuka dengan *platform jupyter*.

Saat ini penelitian jurnal ilmiah mulai menggunakan *Jupyter Notebook* untuk menghasilkan pemodelan matematika, *machine learning*, analisis statistik, dan untuk pengajaran pemrograman, karena *Jupyter Notebook* sudah dilengkapi dengan kode dan sumber datanya. dengan cara ini, eksperimen menjadi lengkap dan dapat digunakan sebagai referensi peneltian.

2. *Google Colaboratory*

Sejalan dengan perkembangan bahasa pemograman *python*, *Google* adalah salah satu perusahaan yang sudah menerapkan bahasa pemograman python. *Google Colaboratory* atau *Google Interactive Notebook* disingkat dengan nama *Google Colab*. *Google Colab* merupakan produk berbasis cloud computing yang dapat digunakan secara *open source* untuk mengakses *Google Colab* di halaman: https://colab.research.google.com/.

Google Colab dibuat khusus untuk programmer atau peneliti yang membutuhkan akses dengan spesifikasi tinggi. *Google Colab* memiliki kode *environment python* dengan format yang mirip dengan *Jupyter notebook*. *Google Colab* dapat berkolaborasi dan berbagi coding dengan lebih dari satu pengguna secara daring sehingga pengguna lebih mudah bereksperimen secara bersamaan dengan fitur yang fleksibel dan dapat menghubungkan dengan *jupyter notebook* di komputer local, *Google Drive*, atau dengan *Github*.

Google Colab memungkinkan kita menggabungkan kode yang dapat dijalankan dalam format *HTML, LaTeX*, dan lainnya. Saat kita membuat notebook di *Google Colab*, *notebook* tersebut akan disimpan di akun Google Drive dan dapat dengan mudah membagikan *notebook Google Colab* dalam proyek *data science*.

2.2. Tugas dan Proyek Latihan

1. Buat panduan langkah demi langkah untuk menginstal Python dan library *NumPy, SciPy, Pandas, Scikit-Learn, Seaborn*, dan *Matplotlib* pada sistem operasi Windows, macOS, dan Linux.
2. Sertakan screenshot dan penjelasan untuk setiap langkah.
3. Buatlah panduan berupa teks, persentasi dan video yang menunjukkan cara menggunakan pada setiap library yang digunakan.
4. Sertakan contoh kode dan visualisasi data pada setiap library yang digunakan.

BAB 3
METODOLOGI DATA SCIENCE

Deskripsi:

Materi Pada BAB 3 ini berisi pembahasan mengenai metodologi data science secara umum untuk mengembangkan suatu aplikasi *artificial intelligence* (*AI*) dengan menjelaskan langkah-langkah utama yang diperlukan untuk menyelesaikan suatu masalah organisasi atau bisnis dengan melakukan tugas-tugas yang umumnya terkait dengan *data science*.

Tujuan Pembelajaran:

Setelah melakukan pembelajaran ini, pembaca diharapkan mampu:

1. Mampu memahami metodologi *data science.*
2. Mampu menjelaskan langkah-langkah utama dalam metodologi *data science*.

3.1. Metodologi *Data Science*

Data dan algoritma *machine learning* tidak cukup sebagai sebuah kegiatan *data science.* Untuk itu diperlukan suatu metodologi untuk membuat suatu sistem *artificial intelligence* yang nantinya dapat berhasil dimanfaatkan, di*deployment* dan dipergunakan.

Secara umum terdapat dua kelompok metodologi yaitu, metodologi teknis dan metodologi bisnis. Metodologi teknis dimulai dari *dataset* kemudian diproses untuk memperoleh pola yang berguna. metodologi yang digunakan untuk metodolgi teknis salah satunya adalah metodologi *knowledge discovery and data mining* (*KDD*). Proses metodologi *knowledge discovery and data mining* (*KDD*) dimulai dengan adanya *dataset* yang mengalami serangkaian proses diantarnya *selection*, *preprocessing data*, *transformation*, *data Mining* dan *evaluation.* Metodologi Bisnis biasanya menggunakan metode *Cross Industry Standard Process For Data Mining*

(CRISP-DM) dengan menempatkan kegiatan *data science* sebagai kegiatan yang berawal dari proses *business understanding*, *data preparation,* proses *modeling, evaluation, deployment, feedback* dan *report*.

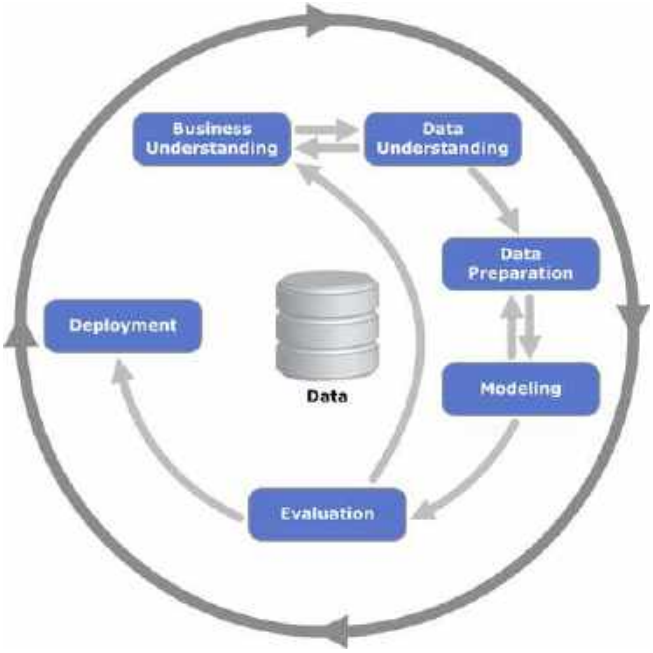
3.2. Langkah Utama Dalam Metodologi *Data Science*

3.2.1. Metode *Cross Industry Standard Process For Data Mining (CRISP-DM)*

Metode Cross Industry Standard Process For Data Mining (CRISP-DM), dalam projek *data science*, sering digunakan sebagai *framework* untuk memulai sebuah proyek *data science*. Metode *CRISP-DM* digunakan sampai dengan menemukan solusi yang diinginkan oleh pengguna.

Tujuan utama dari sebuah proyek *data science*:

1. Bagi penggua: laporan, presentasi, *insights*.
2. Bagi komputer: *deployment*, perangkat lunak.



Gambar 3.1.

Metode *CRISP-DM*

Seperti yang terlihat pada gambar 3.1. Secara umum, terdapat 6 tahapan dalam metodologi *Cross Industry Standard Process For Data Mining (CRISP-DM)*:

1. *Business Understanding*

Pada tahapan *business understanding,* tahapan ini menentukan tujuan bisnis dan menilai situasi dimana organisasi dapat memahami dan memiliki tujuan apa yang ingin dicapai dari perspektif bisnis. Tujuan dari tahapan proses ini adalah untuk mengungkap faktor-faktor penting yang dapat mempengaruhi hasil proyek. Apabila sebuah organisasi mengabaikan langkah ini dapat menghasilkan jawaban yang benar atas pertanyaan yang salah.

2. *Data Understanding*

Pada tahapan *data understanding,* tahapan ini mengharuskan memperoleh data yang tercantum dalam sumber daya proyek. Pengumpulan awal ini meliputi pemuatan data, apabila diperlukan untuk pemahaman data. Jika memperoleh banyak sumber data maka perlu mempertimbangkan bagaimana dan kapan data akan diintegrasikan.

3. *Data Preparation*

Pada tahapan *data preparation*, tahapan ini memutuskan data yang akan di analisa. Kriteria yang mungkin digunakan untuk membuat keputusan, mencakup relevansi data dengan tujuan *data mining*, kualitas data, dan juga batasan teknis seperti batasan volume data atau tipe data. Pemilihan data meliputi pemilihan atribut (kolom) serta pemilihan record (baris) dalam sebuah tabel.

4. *Modeling*

Pada tahapan *modeling*, sebagai langkah pertama dalam tahapan pemodelan, memilih teknik pemodelan sebenarnya yang akan digunakan. Meskipun telah memilih alat selama fase tahapan *business understanding*, pada tahapan ini kita bisa memilih secara spesipik model yang digunakan, misalnya model *decision-tree algortima C.45, neural network, back propagation*.

Jika beberapa model diterapkan, lakukan tugas ini secara terpisah untuk setiap model yang digunakan.

5. *Evaluation*

Pada tahapan *evaluation*, tahapaan ini akan menilai sejauh mana model tersebut memenuhi tujuan bisnis. Selanjutnya menguji model pada aplikasi pengujian. Kemudian akan mengevaluasi hasil dengan melibatkan penilaian hasil yang berupa output *data mining*. Hasil *data mining* melibatkan model-model yang tentu terkait dengan tujuan bisnis awal dan semua temuan lain yang belum tentu terkait dengan tujuan bisnis awal, namun mungkin juga mengungkap tantangan, informasi, atau petunjuk tambahan untuk arah masa depan.

6. *Deployment*

Pada tahapan *deployment*, tahapan ini mengambil hasil evaluasi dan menentukan strategi penerapannya. Jika prosedur umum telah diidentifikasi untuk membuat model yang relevan, prosedur ini didokumentasikan untuk penerapan selanjutnya dengan mempertimbangkan sarana penerapan selama fase tahapan *business understanding*, karena penerapan sangat penting untuk keberhasilan proyek. Dalam hal ini diperlukan analisis prediktif sangat membantu meningkatkan sisi operasional bisnis.

3.3. Tugas dan Proyek Latihan

1. Buat kesimpulan terkait tahapan metodologi metode *Cross Industry Standard Process For Data Mining (CRISP-DM)***.**
2. Buat gambaran metodologi metode *Cross Industry Standard Process For Data Mining (CRISP-DM)* pada suatu masalah bisnis dilingkungan sekitar.

BAB 4
IMPLEMENTASI METODOLOGI CRISP-DM

Deskripsi :

Materi pada BAB 4 ini berisi pembahasan implementasi tahapan yang terdapat pada metodologi *Cross Industry Standard Process For Data Mining (CRISP-DM)* pada proses bisnis beserta contoh penerapan algoritma menggunakan bahasa pemrograman *python*.

Tujuan Pembelajaran :

Setelah membaca dan mempratekkan pada materi ini diharapkan pembaca mampu:

1. Mengimplementasikan metodologi *Cross Industry Standard Process For Data Mining (CRISP-DM)*.
2. Mampu menerapkan algortima kedalam tahapan *Cross Industry Standard Process For Data Mining (CRISP-DM).*

4.1. *Bussiness Understanding*

Dataset yang dipakai untuk implementasi model *Cross Industry Standard Process For Data Mining (CRISP-DM)* diambil dari *dataset* milik https://archive.ics.uci.edu/dataset/222/bank+marketing. bersumber dari penelitian S. Moro, P. Cortez and P. Rita degan judul paper "A Data-Driven Approach to Predict the Success of Bank Telemarketing. Decision Support Systems", Elsevier, 62:22-31, June 2014.

Tujuan analisis ini yaitu untuk meningkatkan efisiensi pemasaran untuk nasabah yang berpotensi membuka dan berinvestasi deposito berjangka. Data yang akan digunakan dibuat model prediksi dalam bentuk classifier berdasarkan kategori di dalam data dengan membandingkan

metode *Support Vector Machine* (SVM), *Decision Tree* dan *Random Forest* dari ke tiga metode tersebut mana yang terbaik akurasi.

4.2. *Data Understanding*

Berdasarkan dataset bank marketing yang diperoleh dari https://archive.ics.uci.edu/dataset/222/bank+marketing. Dataset yang digunakan memiliki **21** variabel yang terdiri dari **20** variabel input dan **1** variabel output**.**

1. **Mengimport *library* yang diperlukan.**

```
import pandas as pd
from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report
from sklearn.metrics import classification_report,
confusion_matrix, accuracy_score
from sklearn import metrics
```

2. **Meload *dataset* bank marketing yang diambil dari https://archive.ics.uci.edu/dataset/222/bank+marketing.**

```
data = pd.read_csv("bank-additional-full.csv", sep=";")
data.info()
```

Hasil ketika di run:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 41188 entries, 0 to 41187
Data columns (total 21 columns):
age                     41188 non-null int64
job                     41188 non-null object
marital                 41188 non-null object
education               41188 non-null object
default                 41188 non-null object
housing                 41188 non-null object
loan                    41188 non-null object
```

```
contact                 41188 non-null object
month                   41188 non-null object
day_of_week             41188 non-null object
duration                41188 non-null int64
campaign                41188 non-null int64
pdays                   41188 non-null int64
previous                41188 non-null int64
poutcome                41188 non-null object
emp.var.rate            41188 non-null float64
cons.price.idx          41188 non-null float64
cons.conf.idx           41188 non-null float64
euribor3m               41188 non-null float64
nr.employed             41188 non-null float64
y                       41188 non-null object
dtypes: float64(5), int64(5), object(11)
memory usage:  6.6+ MB
```

3. Menampilkan statistic deskriptif dari *DataFrame.*

```
data.describe()
```

Hasil ketika di run:

	age	duration	campaign	pdays	previous	emp.var.rate	cons.price.idx	cons.conf.idx	euribor3m	nr.employed
count	41188.00000	41188.000000	41188.000000	41188.000000	41188.000000	41188.000000	41188.000000	41188.000000	41188.000000	41188.000000
mean	40.02406	258.285010	2.567593	962.475454	0.172963	0.081886	93.575664	-40.502600	3.621291	5167.035911
std	10.42125	259.279249	2.770014	186.910907	0.494901	1.570960	0.578840	4.628198	1.734447	72.251528
min	17.00000	0.000000	1.000000	0.000000	0.000000	-3.400000	92.201000	-50.800000	0.634000	4963.000000
25%	32.00000	102.000000	1.000000	999.000000	0.000000	-1.800000	93.075000	-42.700000	1.344000	5099.100000
50%	38.00000	180.000000	2.000000	999.000000	0.000000	1.100000	93.749000	-41.800000	4.857000	5191.000000
75%	47.00000	319.000000	3.000000	999.000000	0.000000	1.400000	93.994000	-36.400000	4.961000	5228.100000
max	98.00000	4918.000000	56.000000	999.000000	7.000000	1.400000	94.767000	-26.900000	5.045000	5228.100000

4. Mengetahui jumlah baris dan kolom dari *DataFrame.*

```
data.shape
```

Hasil ketika di run:

```
(41188, 21)
```

5. **Memeriksa nilai yang hilang dalam *DataFrame.***

```
data.isna().sum()
```

Hasil ketika di run:

```
age                             0
job                             0
marital                         0
education                       0
default                         0
housing                         0
loan                            0
contact                         0
month                           0
day_of_week                     0
duration                        0
campaign                        0
pdays                           0
previous                        0
poutcome                        0
emp.var.rate                    0
cons.price.idx                  0
cons.conf.idx                   0
euribor3m                       0
nr.employed                     0
y                               0
dtype: int64
```

6. **Mencari nilai yang hilang (*missing value*) disetiap *variable* dalam *DataFrame.***

```
data['age'].value_counts()
```

Hasil ketika di run:

```
31                      1947
32                      1846
```

```
33                    1833
36                    1780
35                    1759
                      ...
89                       2
91                       2
87                       1
94                       1
95                       1
Name: age, Length: 78, dtype: int64
```

```
data['job'].value_counts()
```

Hasil ketika di run:

```
admin.                10422
blue-collar            9254
technician             6743
services               3969
management             2924
retired                1720
entrepreneur           1456
self-employed          1421
housemaid              1060
unemployed             1014
student                 875
unknown                 330
Name: job, dtype: int64
```

```
data['marital'].value_counts()
```

Hasil ketika di run:

```
married               24928
single                11568
divorced               4612
unknown                  80
Name: marital, dtype: int64
```

```
data['education'].value_counts()
```

Hasil ketika di run:

```
university.degree            12168
high.school                   9515
basic.9y                      6045
professional.course           5243
basic.4y                      4176
basic.6y                      2292
unknown                       1731
illiterate                      18
Name: education, dtype: int64
```

```
data['default'].value_counts()
```

Hasil ketika di run:

```
no                           32588
unknown                       8597
yes                              3
Name: default, dtype: int64
```

```
data['housing'].value_counts()
```

Hasil ketika di run:

```
yes                          21576
no                           18622
unknown                        990
Name: housing, dtype: int64
```

```
data['loan'].value_counts()
```

Hasil ketika di run:

```
no                           33950
yes                           6248
unknown                        990
Name: loan, dtype: int64
```

```
data['contact'].value_counts()
```

Hasil ketika di run:

```
cellular                26144
telephone               15044
Name: contact, dtype: int64
```

```
data['month'].value_counts()
```

Hasil ketika di run:

```
may                     13769
jul                      7174
aug                      6178
jun                      5318
nov                      4101
apr                      2632
oct                       718
sep                       570
mar                       546
dec                       182
Name: month, dtype: int64
```

```
data['day_of_week'].value_counts()
```

Hasil ketika di run:

```
thu                     8623
mon                     8514
wed                     8134
tue                     8090
fri                     7827
Name: day_of_week, dtype: int64
```

```
data['duration'].value_counts()
```

Hasil ketika di run:

```
85                              170
90                              170
136                             168
73                              167
124                             164
                                ...
1108                              1
980                               1
4918                              1
2453                              1
2015                              1
Name: duration, Length: 1544, dtype: int64
```

```
data['campaign'].value_counts()
```

Hasil ketika di run:

```
1                              17642
2                              10570
3                               5341
4                               2651
5                               1599
6                                979
7                                629
8                                400
9                                283
10                               225
11                               177
12                               125
13                                92
14                                69
17                                58
15                                51
16                                51
18                                33
20                                30
19                                26
```

```
21                                  24
22                                  17
23                                  16
24                                  15
27                                  11
29                                  10
25                                   8
26                                   8
28                                   8
30                                   7
31                                   7
35                                   5
33                                   4
32                                   4
34                                   3
40                                   2
42                                   2
43                                   2
37                                   1
39                                   1
41                                   1
56                                   1
Name: campaign, dtype: int64
```

```
data['pdays'].value_counts()
```

Hasil ketika di run:

```
999                         39673
3                             439
6                             412
4                             118
9                              64
2                              61
7                              60
12                             58
10                             52
5                              46
```

```
13                              36
11                              28
1                               26
15                              24
14                              20
8                               18
0                               15
16                              11
17                               8
18                               7
19                               3
22                               3
21                               2
26                               1
20                               1
25                               1
27                               1
Name: pdays, dtype: int64
```

```
data['previous'].value_counts()
```

Hasil ketika di run:

```
0                           35563
1                            4561
2                             754
3                             216
4                              70
5                              18
6                               5
7                               1
Name: previous, dtype: int64
```

```
data['poutcome'].value_counts()
```

Hasil ketika di run:

```
nonexistent                 35563
failure                      4252
```

```
success                     1373
Name: poutcome, dtype: int64
```

```
data['emp.var.rate'].value_counts()
```

Hasil ketika di run:

```
1.4                     16234
-1.8                     9184
 1.1                     7763
-0.1                     3683
-2.9                     1663
-3.4                     1071
-1.7                      773
-1.1                      635
-3.0                      172
-0.2                       10
Name: emp.var.rate, dtype: int64
```

```
data['cons.price.idx'].value_counts()
```

Hasil ketika di run:

```
93.994                  7763
93.918                  6685
92.893                  5794
93.444                  5175
94.465                  4374
93.200                  3616
93.075                  2458
92.201                   770
92.963                   715
92.431                   447
92.649                   357
94.215                   311
94.199                   303
92.843                   282
92.379                   267
93.369                   264
```

```
94.027                    233
94.055                    229
93.876                    212
94.601                    204
92.469                    178
93.749                    174
92.713                    172
94.767                    128
93.798                     67
92.756                     10
Name: cons.price.idx, dtype: int64
```

```
data['cons.conf.idx'].value_counts()
```

Hasil ketika di run:

```
-36.4                     7763
-42.7                     6685
-46.2                     5794
-36.1                     5175
-41.8                     4374
-42.0                     3616
-47.1                     2458
-31.4                      770
-40.8                      715
-26.9                      447
-30.1                      357
-40.3                      311
-37.5                      303
-50.0                      282
-29.8                      267
-34.8                      264
-38.3                      233
-39.8                      229
-40.0                      212
-49.5                      204
-33.6                      178
-34.6                      174
-33.0                      172
```

```
-50.8                    128
-40.4                     67
-45.9                     10
Name: cons.conf.idx, dtype: int64
```

```
data['euribor3m'].value_counts()
```

Hasil ketika di run:

```
4.857                   2868
4.962                   2613
4.963                   2487
4.961                   1902
4.856                   1210
                        ...
1.045                      1
0.956                      1
0.933                      1
3.282                      1
0.996                      1
Name: euribor3m, Length: 316, dtype: int64
```

```
data['nr.employed'].value_counts()
```

Hasil ketika di run:

```
5228.1                 16234
5099.1                  8534
5191.0                  7763
5195.8                  3683
5076.2                  1663
5017.5                  1071
4991.6                   773
5008.7                   650
4963.6                   635
5023.5                   172
5176.3                    10
Name: nr.employed, dtype: int64
```

```
data['y'].value_counts()
```

Hasil ketika di run:

```
no                   36548
yes                   4640
Name: y, dtype: int64
```

7. **Pengecekan nilai yang hilang (*missing value*) disetiap *variable* dalam *DataFrame.***

```
for i in data:
    print('Attribute',[i],':',data[i].dtypes)
```

Hasil ketika di run:

```
Attribute ['age'] : int64
Attribute ['job'] : object
Attribute ['marital'] : object
Attribute ['education'] : object
Attribute ['default'] : object
Attribute ['housing'] : object
Attribute ['loan'] : object
Attribute ['contact'] : object
Attribute ['month'] : object
Attribute ['day_of_week'] : object
Attribute ['duration'] : int64
Attribute ['campaign'] : int64
Attribute ['pdays'] : int64
Attribute ['previous'] : int64
Attribute ['poutcome'] : object
Attribute ['emp.var.rate'] : float64
Attribute ['cons.price.idx'] : float64
Attribute ['cons.conf.idx'] : float64
Attribute ['euribor3m'] : float64
Attribute ['nr.employed'] : float64
Attribute ['y'] : object
```

Kesimpulan:

Dari proses pencarian dan pengecekan nilai yang hilang (*missing value*) disetiap variable dalam DataFrame dapat dilihat bahwa *dataset* tidak memiliki *incorrect data type input* yang artinya semua inputan sesuai dengan tipe data pada *dataset.*

8. Mencari nilai ketidakseimbangan (*imbalance*)

```
data['y'].value_counts()
```

Hasil ketika di run:

```
No                              36548
Yes                              4640
Name: y, dtype: int64
```

Kesimpulan:

Dataset yang dihasilkan termasuk *imbalance* karena terdapat **36548** atau **(88.7%)** yang tidak melakukan investasi untuk deposito berjangka sedangkan yang berinvestasi hanya **4640** atau **(11.3%)**

9. Mencari duplikasi data

```
data[data.duplicated()]
```

Hasil ketika di run:

	age	job	marital	education	default	housing	loan	contact	month	day_of_week	...	campaign	pdays	previous	poutcome	emp.var.rate	con
1266	39	blue-collar	married	basic.9y	no	no	no	telephone	may	thu	...	1	999	0	nonexistent	1.1	
12261	36	retired	married	unknown	no	no	no	telephone	jul	thu	...	1	999	0	nonexistent	1.4	
14234	27	technician	single	professional course	no	no	no	cellular	jul	mon	...	2	999	0	nonexistent	1.4	
16956	47	technician	divorced	high.school	no	yes	no	cellular	jul	thu	...	3	999	0	nonexistent	1.4	
16965	32	technician	single	professional course	no	yes	no	cellular	jul	thu	...	1	999	0	nonexistent	1.4	
20216	55	services	married	high.school	unknown	no	no	cellular	aug	mon	...	1	999	0	nonexistent	1.4	
20834	31	technician	married	professional course	no	yes	no	cellular	aug	tue	...	1	999	0	nonexistent	1.4	
35217	39	admin.	married	university degree	no	no	no	cellular	nov	tue	...	2	999	0	nonexistent	-0.1	
38477	24	services	single	high.school	no	yes	no	cellular	apr	tue	...	1	999	0	nonexistent	-1.8	
32516	33	admin.	married	university degree	no	yes	no	cellular	may	fri	...	4	999	0	nonexistent	1.1	
36551	43	admin.	married	university degree	no	no	no	cellular	jul	thu	...	1	999	0	nonexistent	-2.9	
38591	71	retired	single	university degree	no	no	no	telephone	oct	tue	...	1	999	0	nonexistent	-3.4	

12 rows × 21 columns

Kesimpulan:

Dari hasil pencarian duplikasi data terdapat 12 data yang duplikasi

10. Mencari *unique value* pada variabel dengan tipe data object

```
for i in data.select_dtypes(include='object'):
    print('\nUnique value pada ' + i + ': ' +
str(data[i].unique()))
print ('\nJumlah unique value pada setiap object')
print(data.select_dtypes(include='object').nunique())
```

Hasil ketika di run:

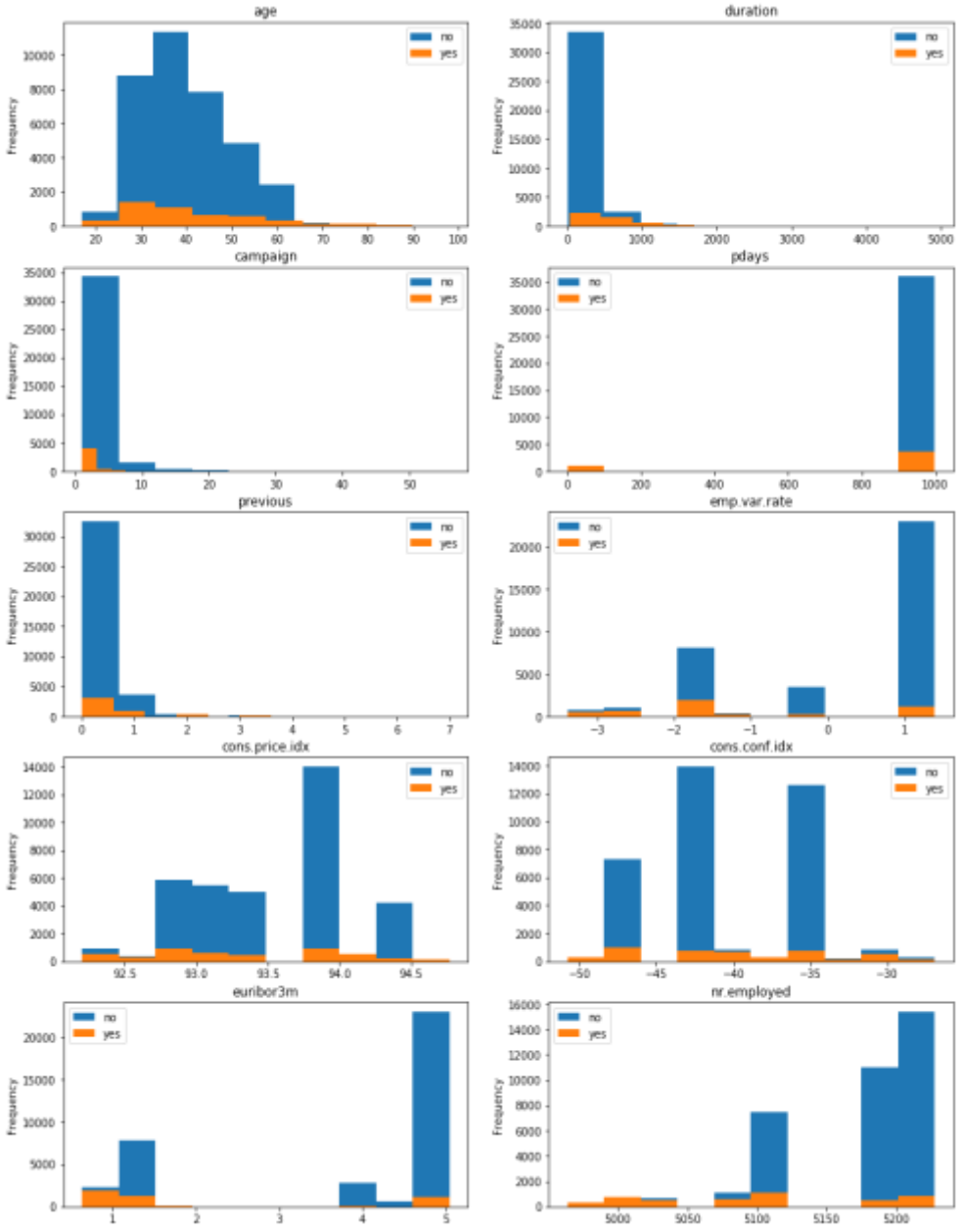
```
Unique value pada job: ['housemaid' 'services' 'admin.'
'blue-collar' 'technician' 'retired'
 'management' 'unemployed' 'self-employed' 'unknown'
'entrepreneur'
 'student']
Unique value pada marital: ['married' 'single' 'divorced'
'unknown']
Unique value pada education: ['basic.4y' 'high.school'
'basic.6y' 'basic.9y' 'professional.course'
 'unknown' 'university.degree' 'illiterate']
Unique value pada default: ['no' 'unknown' 'yes']
Unique value pada housing: ['no' 'yes' 'unknown']
Unique value pada loan: ['no' 'yes' 'unknown']
Unique value pada contact: ['telephone' 'cellular']
Unique value pada month: ['may' 'jun' 'jul' 'aug' 'oct'
'nov' 'dec' 'mar' 'apr' 'sep']
Unique value pada day_of_week: ['mon' 'tue' 'wed' 'thu'
'fri']
Unique value pada poutcome: ['nonexistent' 'failure'
'success']
Unique value pada y: ['no' 'yes']

Jumlah unique value pada setiap object
job                            12
marital                         4
education                       8
```

```
default                         3
housing                         3
loan                            3
contact                         2
month                          10
day_of_week                     5
poutcome                        3
y                               2
dtype: int64
```

11. Menampilkan *histogram* untuk setiap *variabel numeric.*

```
import matplotlib.pyplot as plt
%matplotlib inline
n = len(data.select_dtypes(include='number').columns)
i = 0
for col in
data.select_dtypes(include='number').columns.values:
    plt.subplot(n//2,2,i+1)
    data.groupby('y')[col].plot(kind='hist',stacked=True,f
igsize=(15,20))
    plt.gca().set_title(col)
    plt.legend()
    i += 1
plt.show()
```

Hasil ketika di run:



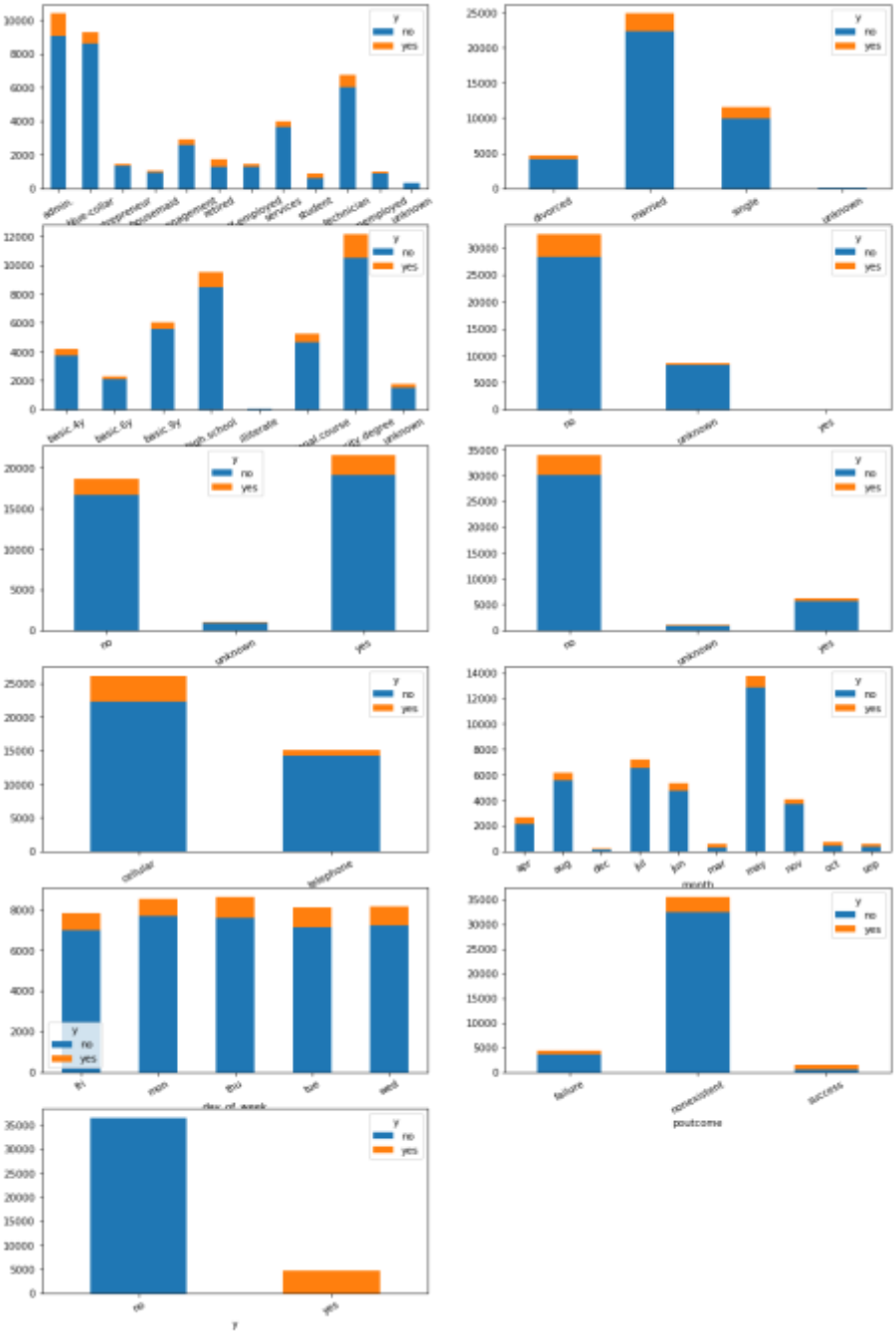
Gambar 4.1. *Histogram Variabel Numerik*

12. Menampilkan *histogram* untuk setiap kolom dengan data kategori.

Histogram merupakan salah satu jenis visualisasi data dalam bentuk grafik batang yang diperoleh dari hasil tabulasi frekuensi. Visualisasi jenis ini digunakan untuk merepresentasikan distribusi frekuensi dari *dataset* yang bersifat numeric. Sehingga didapatkan informasi yang lebih banyak dari data tersebut dan akan memudahkan untuk mendapatkan kesimpulan dari data tersebut.

```
%matplotlib inline
import seaborn as sns
import matplotlib.pyplot as plt
n = len(data.select_dtypes(include='object').columns)
i = 0
for obj in data.select_dtypes(include='object'):
    ax = plt.subplot(6,2,i+1)
    data.groupby([obj,
'y']).size().unstack().plot(kind='bar', stacked=True,
ax=ax,figsize=(15,25))
    plt.xticks(rotation=30)
    i += 1
plt.show()
```

Hasil ketika di run:

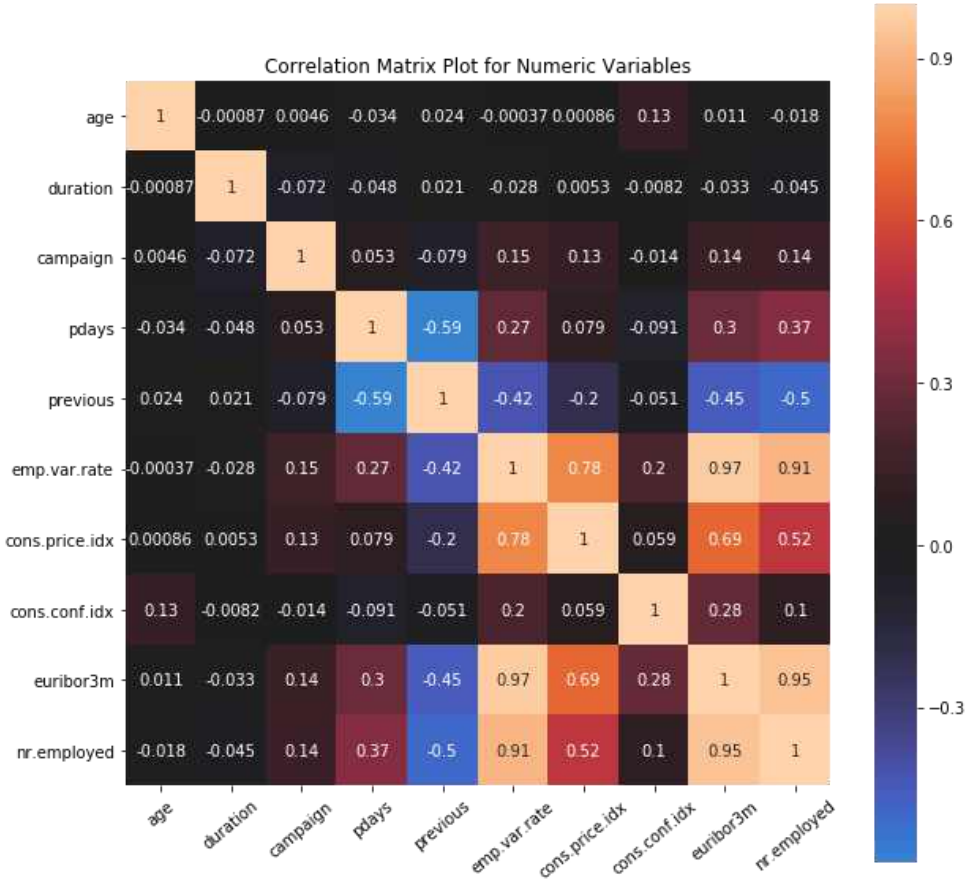


Gambar 4.2. *Histogram* Kolom

13. Menampilkan Correlation Matrix.

```
%matplotlib inline
plt.figure(figsize=(10, 10))
sns.heatmap(data.select_dtypes(include='number').corr(),
cbar=True, square=True, annot=True, center=0)
plt.xticks(rotation=40)
plt.title('Correlation Matrix Plot for Numeric Variables')
plt.show()
```

Hasil ketika di run:



Gambar 4.3. *Correlation Matrix Data Understanding*

14. Mencari nilai *Outliers.*

```
for i in ['age','job','marital','education','default',
          'housing','loan','contact','month','day_of_week'
,
          'duration','campaign','pdays','previous','poutco
me',
          'emp.var.rate','cons.price.idx','cons.conf.idx',
          'euribor3m','nr.employed','y']:
    if (data[i].dtypes in ['int64','float64']):
        print('\nAttribute-',[i],':',data[i].dtypes)
        Q1=data[i].quantile(0.25)
        print('Q1',Q1)
        Q3=data[i].quantile(0.75)
        print('Q3',Q3)
        IQR=Q3-Q1
        print('IQR',IQR)
        min=data[i].min()
        max=data[i].max()
        min_IQR=Q1-1.5*IQR
        max_IQR=Q3+1.5*IQR
        if (min<min_IQR):
            print('Low outlier is found',min_IQR)
        if (max>max_IQR):
            print('High outlier is found',max_IQR)
```

Hasil ketika di run:

```
Attribute- ['age'] : int64
Q1 32.0
Q3 47.0
IQR 15.0
High outlier is found 69.5

Attribute- ['duration'] : int64
Q1 102.0
Q3 319.0
```

```
IQR 217.0
High outlier is found 644.5

Attribute- ['campaign'] : int64
Q1 1.0
Q3 3.0
IQR 2.0
High outlier is found 6.0

Attribute- ['pdays'] : int64
Q1 999.0
Q3 999.0
IQR 0.0
Low outlier is found 999.0

Attribute- ['previous'] : int64
Q1 0.0
Q3 0.0
IQR 0.0
High outlier is found 0.0

Attribute- ['emp.var.rate'] : float64
Q1 -1.8
Q3 1.4
IQR 3.2

Attribute- ['cons.price.idx'] : float64
Q1 93.075
Q3 93.994
IQR 0.9189999999999969

Attribute- ['cons.conf.idx'] : float64
Q1 -42.7
Q3 -36.4
IQR 6.300000000000004
High outlier is found -26.949999999999992

Attribute- ['euribor3m'] : float64
Q1 1.344
```

```
Q3 4.961
IQR 3.617

Attribute- ['nr.employed'] : float64
Q1 5099.1
Q3 5228.1
IQR 129.0
```

Kesimpulan hasil data understanding yang di dapatkan berdasarkan pengujian diatas:

1. Memiliki 21 variabel (20 variabel input dan 1 variabel output) dan 41188 entri.
2. Terdapat 12 duplikasi data.
3. Dataset pada variabel prediksi (y) termasuk *imbalance* karena terdapat 88.7% yang tidak melakukan investasi untuk deposito berjangka sedangkan yang berinvestasi hanya 11.3%
4. Variabel euribor3m dengan variabel emp.var.rate memiliki correlation value yang tinggi (0.97)

4.3. *Data Preparation*

1. Menghapus Duplikasi Data

```
data_prep = data.drop_duplicates()
data_prep.shape
```

Hasil ketika di run:

```
(41176, 21)
```

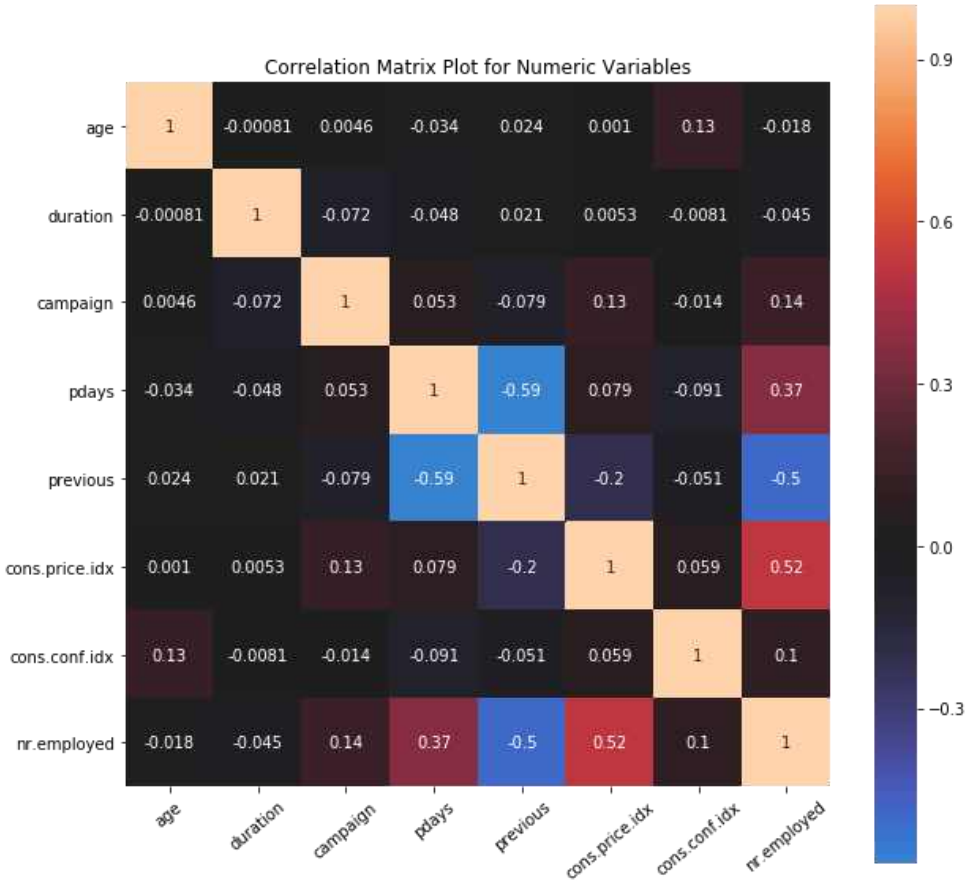
2. Menghapus variable euribor3m dan variabel emp.var.rate memiliki correlation value yang tinggi (0.97)

```
pd.options.mode.chained_assignment = None
data_prep.drop(['emp.var.rate','euribor3m'], axis=1,
inplace=True)
```

3. **Menampilkan *Correlation Matrix* setalah variable euribor3m dan variabel emp.var.rate di hapus**

```
plt.figure(figsize=(10, 10))
sns.heatmap(data_prep.select_dtypes(include='number').corr
(), cbar=True, square=True, annot=True, center=0)
plt.xticks(rotation=40)
plt.title('Correlation Matrix Plot for Numeric Variables')
plt.show()
data_prep.info()
```

Hasil ketika di run:



Gambar 4.4. *Correlation Matrix Data Preparation*

4. Mengubah data numeric ke dalam sekala 0-1

```
from sklearn.preprocessing import MinMaxScaler
sc = MinMaxScaler()
col =
data_prep.select_dtypes(include='number').columns.tolist()
data_prep[col] = sc.fit_transform(data_prep[col])
data_prep.hist(figsize=(10,10))
data_prep.describe()
```

	age	duration	campaign	pdays	previous	cons.price.idx	cons.conf.idx	nr.employed
count	41176.000000	41176.000000	41176.000000	41176.000000	41176.000000	41176.000000	41176.000000	41176.000000
mean	0.284244	0.052525	0.028507	0.963426	0.024716	0.535744	0.430843	0.769130
std	0.128650	0.052726	0.050369	0.187124	0.070709	0.225560	0.193634	0.273162
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.185185	0.020740	0.000000	1.000000	0.000000	0.340608	0.338912	0.512287
50%	0.259259	0.036600	0.018182	1.000000	0.000000	0.603274	0.376569	0.859735
75%	0.370370	0.064864	0.036364	1.000000	0.000000	0.696753	0.602510	1.000000
max	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000

5. Mengubah variabel dengan tipe data object menjadi numeric

```
from sklearn.preprocessing import LabelEncoder,
OrdinalEncoder
import numpy as np
oe = OrdinalEncoder(dtype=np.uint8)
col =
data_prep.drop(['y'],axis=1).select_dtypes(include='object
').columns.tolist()
data_prep[col] = oe.fit_transform(data_prep[col])
le = LabelEncoder()
data_prep['y'] = le.fit_transform(data_prep['y'])
data_prep.head()
```

	age	job	marital	education	default	housing	loan	contact	month	day_of_week	duration	campaign	pdays	previous	poutcome	cons.price.idx	cons.conf.
0	0.481481	3	1	0	0	0	0	1	6	1	0.053070	0.0	1.0	0.0	1	0.696753	0.60
1	0.493827	7	1	3	1	0	0	1	6	1	0.030257	0.0	1.0	0.0	1	0.696753	0.60
2	0.246914	7	1	3	0	2	0	1	6	1	0.045954	0.0	1.0	0.0	1	0.696753	0.60
3	0.283951	0	1	1	0	0	0	1	6	1	0.030704	0.0	1.0	0.0	1	0.696753	0.60
4	0.481481	7	1	3	0	0	2	1	6	1	0.062424	0.0	1.0	0.0	1	0.696753	0.60

4.4. *Modeling*

1. **Membuat model dengan 3 metode algortima yaitu *SVM, Decision Tree* dan *Random Forest***

```
models = { 'svm':SVC(kernel="rbf", gamma="auto"),
'decision_tree':DecisionTreeClassifier(),
'random_forest':RandomForestClassifier(n_estimators=100),
}
x_data = data_prep.iloc[:, :-1]
y_data = data_prep.iloc[:, -1]
```

2. **Membagi *dataset* menjadi *data training* dan *data testing***

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test =
train_test_split(x_data, y_data, test_size=0.3,
random_state=123, stratify=y_data)
```

4.5. *Evaluation*

1. **Evaluasi Algoritma Model *SVM***

```
print("[INFO] using '{}' model".format("svm"))
model = models['svm']
model.fit(x_train, y_train)
y_pred_SVM = model.predict(x_test)
akurasi_SVM = metrics.accuracy_score(y_test, y_pred_SVM) *
100
cm_SVM = confusion_matrix(y_test, y_pred_SVM)
print("[INFO] evaluating...")
predictions = model.predict(x_test)
print(classification_report(y_test, predictions, y_data))
print(akurasi_SVM)
```

Hasil ketika di run:

```
89.80004857119728
```

2. **Evaluasi Algoritma Model *Decision Tree***

```
print("[INFO] using '{}' model".format("decision_tree"))
model = models['decision_tree']
model.fit(x_train, y_train)
y_pred_DT = model.predict(x_test)
akurasi_DT = metrics.accuracy_score(y_test, y_pred_DT) *
100
cm_DT = confusion_matrix(y_test, y_pred_DT)
print("[INFO] evaluating...")
predictions = model.predict(x_test)
print(classification_report(y_test, predictions, y_data))
print(akurasi_DT)
```

Hasil ketika di run:

```
88.31053185461022
```

3. **Evaluasi Algoritma Model *Random Forest***

```
print("[INFO] using '{}' model".format("random_forest"))
model = models['random_forest']
model.fit(x_train, y_train)
y_pred_RF = model.predict(x_test)
akurasi_RF = metrics.accuracy_score(y_test, y_pred_RF) *
100
cm_RF = confusion_matrix(y_test, y_pred_RF)
print("[INFO] evaluating...")
predictions = model.predict(x_test)
print(classification_report(y_test, predictions, y_data))
print(akurasi_RF)
```

Hasil ketika di run:

```
91.34623168461103
```

4. **Perbandingan akurasi dari setiap algoritma**

```
print("Support Vector Machines (SVMs) : ", akurasi_SVM)
```

```
print("Decision Trees : ", akurasi_DT)
print("Random Forests : ", akurasi_RF)
```

Hasil ketika di run:

```
Support Vector Machines (SVMs) :  89.80004857119728
Decision Trees :  88.31053185461022
Random Forests :  91.34623168461103
```

Kesimpulan:

1. Dengan algortima model SVM didapatkan hasil akurasi = 89.8
2. Dengan algortima model *Decision Tree* didapatkan hasil akurasi = 88.3
3. Dengan algortima model *Random Forest* didapatkan hasil akurasi = 91.3

Didapatkan bahwa algortima model *Random Forest* memiliki akurasi yang paling tinggi yaitu 91.3

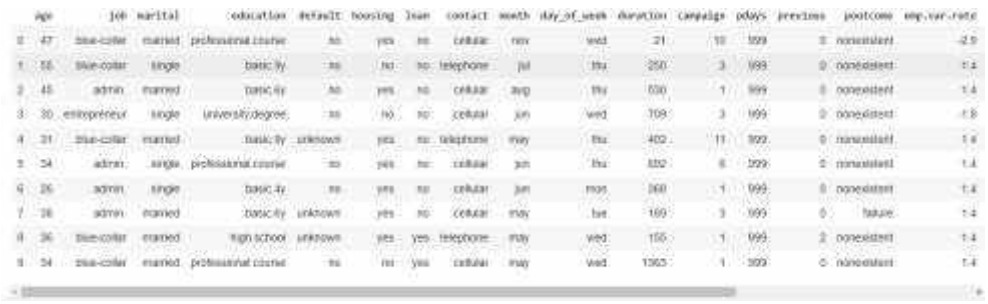
4.6. *Deployment*

Program sederhana untuk memprediksi apakah nasabah bank akan membuka deposito dan berinvestasi atau tidak sesuai dengan ketentuan variabel yang ada pada *dataset*.

```
from sklearn.preprocessing import MinMaxScaler
from sklearn.preprocessing import OrdinalEncoder
import numpy as np

def prediksi_berinvestasi_atau_tidak(data):
    data = data.drop_duplicates()
    data.drop(['emp.var.rate','euribor3m'], axis=1,
inplace=True)
    numeric =
data.select_dtypes(include='number').columns.tolist()
    data[numeric] = sc.transform(data[numeric])
    obj =
data.select_dtypes(include='object').columns.tolist()
    data[obj] = oe.transform(data[obj])
```

```
    # Prediksi
    idx = 0
    for result in base_model.predict(data):
        if result == 0:
            print('Klien ',data.index[idx],' diprediksi
tidak akan berinvestasi.')
        else:
            print('Klien ',data.index[idx],' diprediksi
akan berinvestasi')
        idx += 1
new_data = data.drop(['y'],axis=1)
n_ppl = 10
data_sync = pd.DataFrame()
for col in new_data.columns:
    data_sync = pd.concat([data_sync,
new_data[col].sample(n=n_ppl).to_frame().reset_index().dro
p(['index'],axis=1)],axis=1)
data_sync
```

Hasil ketika di run:



	age	job	marital	education	default	housing	loan	contact	month	day_of_week	duration	campaign	pdays	previous	poutcome	emp.var.rate
0	47	blue-collar	married	professional.course	no	yes	no	cellular	nov	wed	21	10	999	0	nonexistent	-2.9
1	55	blue-collar	single	basic.9y	no	no	no	telephone	jul	thu	250	3	999	0	nonexistent	1.4
2	45	admin	married	basic.6y	no	yes	no	cellular	aug	thu	630	1	999	0	nonexistent	1.4
3	30	entrepreneur	single	university.degree	no	no	no	cellular	jun	wed	709	3	999	0	nonexistent	1.4
4	31	blue-collar	married	basic.9y	unknown	yes	no	telephone	may	thu	402	11	999	0	nonexistent	1.1
5	54	admin	single	professional.course	no	yes	no	cellular	jun	thu	832	6	999	0	nonexistent	1.4
6	26	admin	single	basic.6y	no	yes	no	cellular	jun	mon	368	1	999	0	nonexistent	1.4
7	38	admin	married	basic.6y	unknown	yes	no	cellular	may	tue	169	3	999	0	failure	1.1
8	36	blue-collar	married	high.school	unknown	yes	yes	telephone	may	wed	155	1	999	2	nonexistent	1.1
9	34	blue-collar	married	professional.course	no	no	yes	cellular	may	wed	1363	1	999	0	nonexistent	1.1

```
prediksi_berinvestasi_atau_tidak(data_sync)
```

```
Klien  0  diprediksi tidak akan berinvestasi.
Klien  1  diprediksi akan berinvestasi
Klien  2  diprediksi akan berinvestasi
Klien  3  diprediksi akan berinvestasi
Klien  4  diprediksi akan berinvestasi
Klien  5  diprediksi akan berinvestasi
Klien  6  diprediksi tidak akan berinvestasi.
```

```
Klien  7  diprediksi tidak akan berinvestasi.
Klien  8  diprediksi tidak akan berinvestasi.
Klien  9  diprediksi akan berinvestasi
```

4.7 Tugas dan Proyek Latihan

Buatlah analisa dengan model *Cross Industry Standard Process For Data Mining (CRISP-DM)* untuk memprediksi tingkat penjualan barang dengan mengambil data dari https://www.kaggle.com/c/rossmann-store-sales.

BAB 5
STUDI KASUS MODEL REGRESI LINIER

Deskripsi :

Materi pada BAB 5 ini berisi pembahasan mengenai studi kasus dengan menggunakan model *regresi linier* dengan fungsi *standardscaler* untuk melakukan permodelan pada data dan memprediksi sebuah data dengan akurat sehingga mudah dimengerti oleh pembaca.

Tujuan Pembelajaran :

Setelah membaca dan mempratekkan pada materi ini diharapkan pembaca mampu :

1. Mampu menerapkan model *regresi linier* dengan fungsi *standardscaler.*
2. Mampu menjelaskan alur pembuatan data dan menerapkannya pada bisnis.

5.1. *Domain Project*

Domain project yang dipilih dalam projek ini adalah mengenai korban titanic. Titanic adalah kapal super Inggris yang tenggelam pada tanggal 15 April 1912, saat dalam perjalanan dari Southampton, Inggris, menuju New York City setelah bertabrakan dengan gunung es di Samudra Atlantik Utara. Tragedi tersebut mengakibatkan tewasnya 1.514 penumpang, menjadikannya bencana maritim paling mematikan sepanjang sejarah kapal ini dibangun antara tahun 1909 dan 1911 di galangan kapal Harland and Wolff di Belfast, Irlandia. Kapal ini mampu mengangkut 2.224 penumpang.

Dari pembahasan di atas, dapat disimpulkan bahwa data informasi waktu, lokasi, dan penumpang merupakan kunci untuk memahami keadaan dan skala kecelakaan tragis ini. "*Machine Learning Dataset Titanic*"

adalah aplikasi berbasis machine learning yang memahami data tentang korban penumpang Titanic, dimulai dengan sebaran usia berdasarkan kelangsungan hidup, jenis kelamin, dan pengaruh jumlah saudara laki-laki dan perempuan. Pengaruh jumlah orang tua atau keturunan, kelas tiket, dan biaya penjualan tiket terhadap kelangsungan hidup dan gender.

Karena Titanic mempunyai jumlah penumpang yang besar, rumus matematika yang rumit diperlukan untuk menentukan statistic korban Titanic, dan buku besar diperlukan untuk mencatat informasi penumpang. Hal ini tidak menutup kemungkinan adanya kesalahan dalam penggunaan rumus dan waktu yang digunakan untuk mengumpulkan informasi tentang korban Titanic mungkin tidak efektif dan efisien.

Berdasarkan hal tersebut, dikembangkanlah aplikasi yang memfasilitasi analisis statistik tentang korban *Titanic* berupa *statistik biola plot* dan *pointplot* menggunakan pustaka *python* seperti *pandas, seaborn, numpy, matplotlib, tensorflow,* dan *sckit.* Tujuan dari percobaan dengan model regresi linier dengan fungsi *standardscaler* ini adalah untuk menganalisis, memvisualisasikan, dan memodelkan dataset tersebut untuk mengetahui berapa banyak penumpang yang masih hidup, dari mana mereka berasal dari mana, dan berapa harga tiket mereka.

5.2. *Bussiness Understanding*

1. *Problem Statements*

Berdasarkan latar belakang diatas, berikut ini rumusan masalah yang dapat diselesaikan pada proyek ini:

a) Bagaimana cara menganalisis korban Titanic dengan skor akurasi 80%.
b) Bagaimana cara menggunakan *machine learning* model regresi linier dengan fungsi standardscaler guna memprediksi korban korban Titanic?.

2. *Goals*

a) Melakukan analisis dengan baik agar dapat digunakan dalam pembuatan model regresi linier dengan fungsi *standardscale.*
b) Mengetahui cara membuat model model regresi linier dengan fungsi *standardscaler* untuk memprediksi korban korban *Titanic*.

2. *Solution Statements*

Solusi yang dapat dilakukan untuk memenuhi tujuan dari projek ini diantaranya:

a) Untuk melakukan analisis dapat dilakukan beberapa teknik diantaranya:
   1) Menggunakan eksplorasi data untuk memahami hubungan antara distribusi variable dan keberlangsungan hidup korban *Titanic*.
   2) Memperbaiki nilai data yang hilang dan lakukan pembersihan data.
   3) Melakukan analisis lebih lanjut seperti hubungan antara kelas penumpang, gender dan keberlangsungan hidup.
   4) Membagi *dataset* menjadi dua bagian dengan perbandingan 80% untuk data latih dan 20% untuk data uji.
b) Untuk pembuatan model dipilih penggunaan model regresi linier dengan fungsi *standardscaler*. Model dan fungsi tersebut dipilih karena mudah digunakan dan cocok untuk kasus ini.

Berikut cara kerja, kelebihan dan kekuarangan algoritma *standardscaler* dan *regresi linier*:

Cara kerja fungsi *StandardScaler* :

1) Pertama dengan menghitung rata-rata (*mean*) dan standar deviasi (*standard deviation*) dari setiap fitur dalam kumpulan data (*dataset*).
2) Untuk setiap fitur, algoritma mengurangkan rata-rata nilai fitur dan membagi hasilnya dengan standar deviasi.

Kelebihan dan kekurangan fungsi *StandardScaler*:

1) Kelebihannya adalah fungsi *StandardScaler* menghilangkan *numerik* yang dapat terjadi karena perbedaan skala antar fitur.
2) Kekuranganya adalah sulitnya menemukan nilai nilai yang hilang.

5.3. *Data Understanding*

Data pada project ini menggunakan dataset Titanic yang bersumber pada halaman website https://www.kaggle.com/. *Dataset* tersebut yang menjelaskan berbagai faktor yang akan mempengaruhi tingkat prediksi pada para korban *Titanic*. Informasi *dataset* dapat dilihat pada table 1:

Tabel 5.1. *Dataset Titanic*

Jenis	Keterangan
Sumber	Kaggle Dataset : Titanic Dataset https://www.kaggle.com/datasets/yasserh/titanic-dataset
Lisensi	CC0: Public Domain
Kategori	Penumpang, umur, gender
Jenis dan ukuran berkas	CSV (61.19 kB)

Pada berkas *dataset titanic*.csv berisi **1309** ***rows*** dan **14 *columns***. Kolom-kolom tersebut terdiri dari **5** ***string*****, 4** ***integer*****, 2** ***float*****, 1** ***Object***. Untuk penjelasan mengenai variabel-variabel pada dataset titanic ini dapat dilihat sebagai berikut:

a) Survival merupakan jumlah orang yang selamat dari kecelakaan *titanic*

b) Pclass merupakan kelas penumpang seperti kelas 1 untuk orang kaya kelas 2 untuk yang menengah dan kelas ke 3 untuk orang kelas bawah.

c) Name merupakan nama nama penumpang.

d) Sex merupakan jenis kelamin para penumpang di titanic.

e) Age merupakan umur-umur yang berada di kapal titanic.

f) Sibsp merupakan jumlah saudara/pasangan di kapal titanic.

g) Parch merupakan jumlah orangtua dan anak anak di kapal titanic.

h) Ticket merupakan nomor tiket setiap penumpang.

i) Fare merupakan tarif setiap penumpang baik kelas 1 maupun kelas 3

j) Cabin merupakan kabin di dalam titanic.

k) Embarked merupakan tempat yang ingin dituju setiap penumpang (C = Cherbourg, Q = Queenstown, S = Southampton).

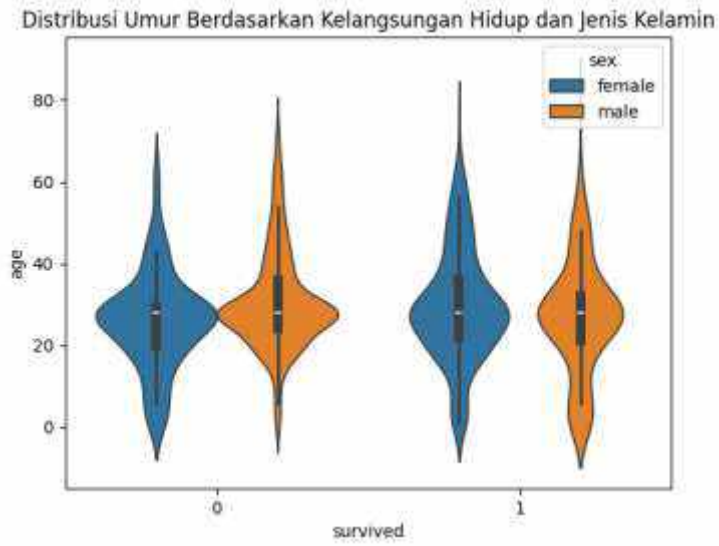
Berikut beberapa tahapan sebelum visualisasi data pada preparation sebagai berikut :

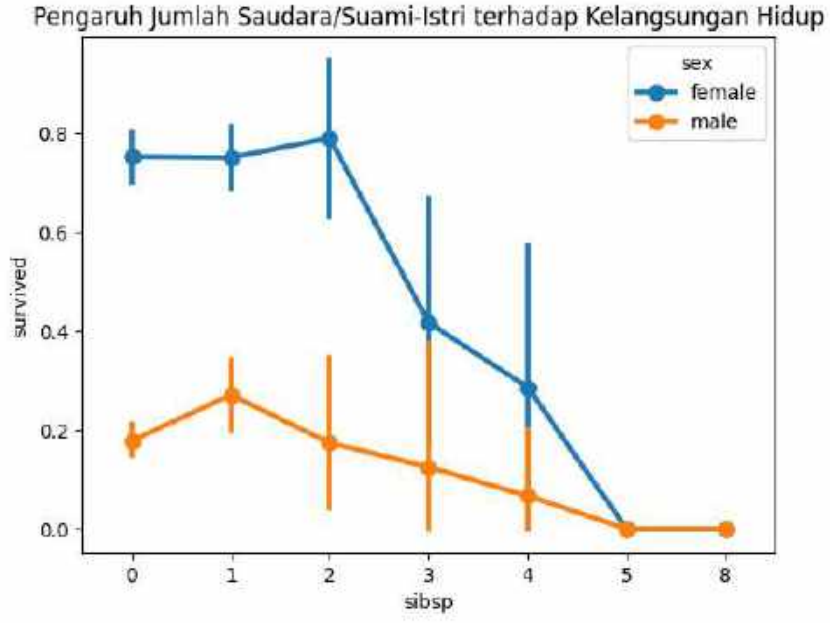
1. Meload *Dataset* ke dalam sebuah *Dataframe* menggunakan *pandas*.
2. df.info() digunakan untuk mengecek tipe kolom pada dataset.
3. df.isna().sum() digunakan untuk mengecek apakah ada kolom yang kosong.
4. df.describe() digunakan untuk mendapatkan info mengenai dataset terhadap nilai rata-rata, median, banyaknya data niali A1 hingga A3.

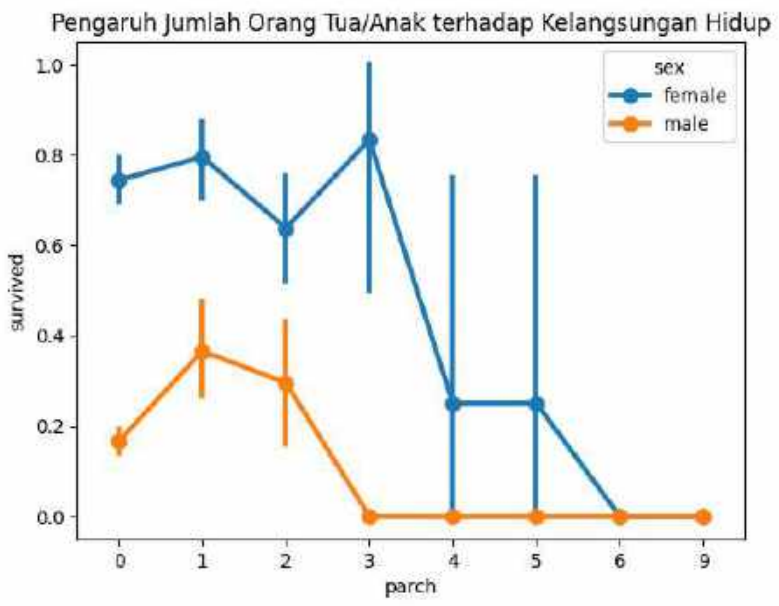
5.4. *Data Preparation*

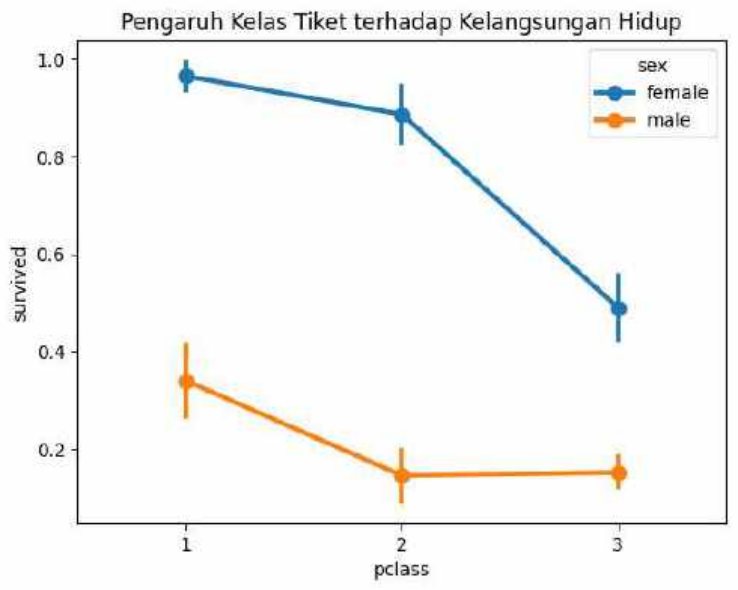
Berikut beberapa tahapan visualisasi data pada data preparation :

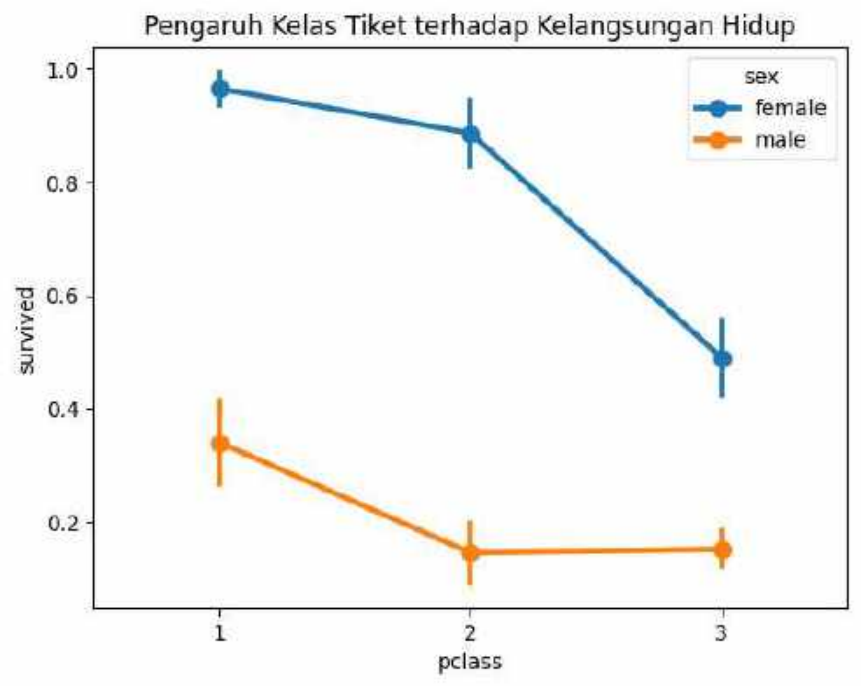
1. Pertama, bagi kumpulan data (dataset) Anda menjadi dua format variabel: variabel untuk kolom numerik dan variabel kolom, seperti tipe string.
2. Lalu visualisasikan datanya untuk melihat berapa banyak orang yang selamat dan siapa yang tidak


Distribusi Umur Berdasarkan Kelangsungan Hidup dan Jenis Kelamin
sex
female
male
80
60
40
20
0
age
0
1
survived


Pengaruh Jumlah Saudara/Suami-Istri terhadap Kelangsungan Hidup
sex
female
male
0.8
0.6
0.4
0.2
0.0
survived
0
1
2
3
4
5
8
sibsp


Pengaruh Jumlah Orang Tua/Anak terhadap Kelangsungan Hidup
sex
female
male
survived
parch
1.0
0.8
0.6
0.4
0.2
0.0
0
1
2
3
4
5
6
9


Pengaruh Kelas Tiket terhadap Kelangsungan Hidup
sex
female
male
survived
pclass
1.0
0.8
0.6
0.4
0.2
1
2
3



Gambar 5.1. Visualisasi Grafik Dataset *Titanic*

Berikut adalah tahapan tahapan dalam melakukan pra-pemrosesan data :

1. Muat Kumpulan data (*dataset*) Titanic dari file *CSV* ke dalam struktur bingkai data (*dataframe*) menggunakan library Pandas. Dengan Pandas anda akan dengan mudah mengakses dan mengedit data.
2. Memperbaiki nilai yang hilang pada fungsi usia dengan menggantinya dengan nilai usia rata-rata (*mean*) keseluruhan penumpang. Hal ini dilakukan untuk memastikan data lebih lengkap dan tersedia untuk dianalisis

```
mean_age = titanic_data["age"].mean()
titanic_data["age"].fillna(mean_age, inplace=True)
```

3. Kolom "kabin" dan "ticket" dihapus karena dianggap tidak relevan dengan ananlisis. Mengapa harus dihapus? Karena membantu

menyederhanakan struktur data dan mengurangi dimensi yang tidak diperlukan.

```
titanic_data.drop(['cabin', 'ticket'], axis=1, inplace=True)
```

4. Selanjutnya dalam proyek ini akan digunakan algoritma StandardScaler dari library Sckit-Learn untuk menstandarkan fitur numerik "age" (usia) dan "fare" (tarif). StandardScaler membantu menghilangkan perbedaan skala antar fitur dan menghindari fitur mendominasi dengan rentang nilai yang luas

```
scaler = StandardScaler()
numeric_features = ['age', 'fare']
titanic_data[numeric_features] = scaler.fit_transform(titanic_data[numeric_features])
```

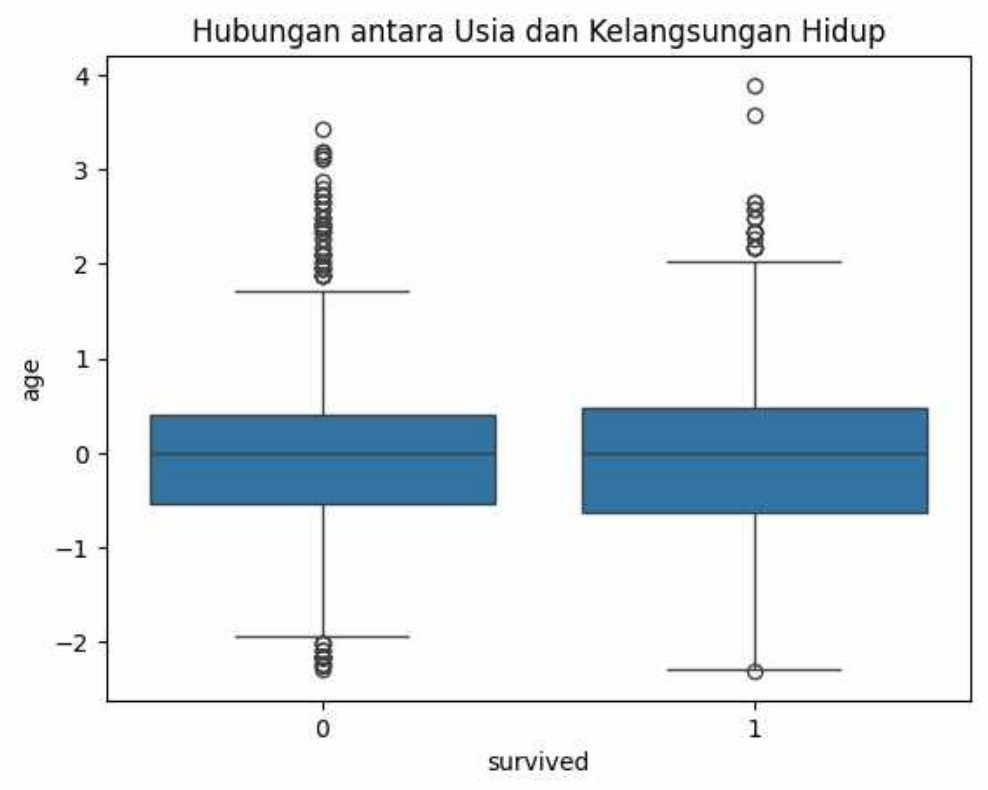
5. Pisahkan data menjadi fitur (x) dan label (y) dan gunakan train_test_split dari *Library* Sckit-Learn untuk membagi Kumpulan data menjadi data pelatihan dan pengujian. Data pelatihan digunakan untuk melatih model dan data pengujian digunakan untuk menguji performa model

```
X = titanic_data.drop('survived', axis=1)
y = titanic_data['survived']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

X	DataFrame	(1309, 11)	pclass		name	sex
X_test	DataFrame	(262, 11)	pclass		name	sex
X_train	DataFrame	(1047, 11)	pclass	name	sex	age
y	Series	(1309,)	0 1			
y_test	Series	(262,)	1148 0			
y_train	Series	(1047,)	772 0			

6. Pada Langkah terakhir jika ingin menvisualisasikan data, anda bisa membuat visualisasi data menggunakan *library Seaborn* dan *Matplotlib*.

```
sns.boxplot(x='survived', y='age', data=titanic_data)
plt.title('Hubungan antara Usia dan Kelangsungan Hidup')
plt.show()
```



Gambar 5.2. Visualisasi Grafik Dataset *Titanic*

5.6. *Modelling*

Setelah melakukan *preprocessing dataset,* langkah selanjutnya adalah memodelkan data. Fase ini menggunakan algoritma yang disebut StandardScaler. Pertama-tama pilih fitur yang akan digunakan untuk pemodelan. ini disimpan dalam variabel "features" (fitur). Fitur-fitur tersebut kemudian diubah menjadi representasi encode one-hot menggunakan "*pd_get_dummies()*".

```
features = ['pclass', 'sex', 'age', 'sibsp', 'parch', 'fare']
x = pd.get_dummies(data[features])
y = data['survived']
```

Selanjutnya, kita menggunakan "train_test_split()" untuk membagi data menjadi data pelatihan ("x_train", "y_train") dan data pengujian ("x_test", "y_test"). Proporsi data uji adalah 20% dari total Kumpulan data.

```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=42)
```

Data pelatihan yang berisi "NaN" atau nilai yang hilang dihapus menggunakan ".dropna()" untuk fitur "x_train" dan label "y_train". Untuk ciri "age" dan "fare", isi nilai yang hilang dengan rata-rata menggunakan ".fillna()"

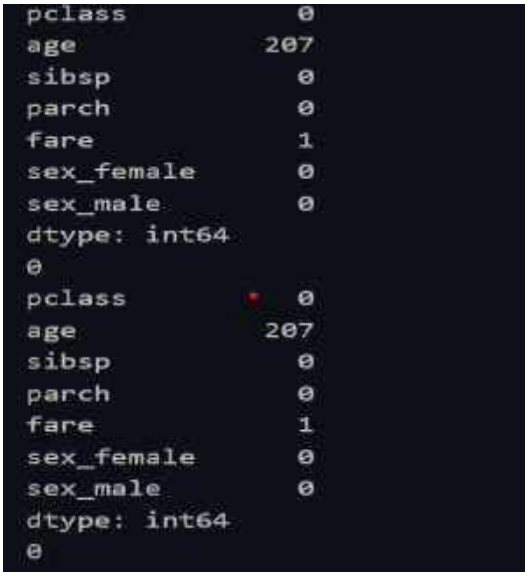
```
print(x_train.isnull().sum())
print(y_train.isnull().sum())

print(np.isnan(x_train).sum())
print(np.isnan(y_train).sum())

x = pd.get_dummies(data[features], dummy_na=False)

x_train = x_train.dropna()
y_train = y_train.dropna()

x_train['age'].fillna(x_train['age'].mean(), inplace=True)
x_train['fare'].fillna(x_train['fare'].mean(), inplace=True)

scaler = StandardScaler()
x_train_scaled = scaler.fit_transform(x_train)
```

Oleh karena itu, fitur yang dipilih digunakan dalam model yang dinormalisasi (semua fitur yang sudah mewakili representasi one-hot encoding) dengan StandardScaler di Sckit-Learn. Hal ini dilakukan untuk

memastikan bahwa rentang nilai setiap fitur adalah seragam, sehingga menghindari masalah numerik dan meningkatkan konvergensi model.

5.7. *Evaluation*

Model yang dikembangkan pada proyek ini adalah model klasifikasi yang menggunakan algoritma *Random Forest Classifier*. Hal ini, proyek ini menggunakan algoritma *Random Forest Classifier* karena dapat menangani baik masalah klasifikasi *biner (survived atau tidak)* dan bekerja dengan baik pada berbagai data.



Algoritma Fungsi *StandardScaler* digunakan untuk menormalkan fitur numerik dalam dataset. Dalam kasus Titanic, karakteristik seperti "age" dan "fare" merupakan karakteristik numerik yang dapat mempunyai rentang nilai. *StandardScaler* memungkinkan Anda menyesuaikan atau menormalkan nilai fitur-fitur ini sehingga tidak mendominasi fitur lain dalam proses *machine learning*.

BAB 6
STUDI KASUS MODEL ANN

Deksripsi :

Materi pada BAB 6 ini berisi pembahasan mengenai studi kasus dengan menggunakan model algoritma *ANN* (*Artificial Neural Networks*) untuk melakukan permodelan pada data dan memprediksi sebuah data dengan akurat sehingga mudah dimengerti oleh pembaca.

Tujuan Pembelajaran :

Setelah membaca dan mempratekkan pada materi ini diharapkan pembaca mampu:

1. Mampu menerapkan algoritma *ANN* (*Artificial Neural Networks*).
2. Mampu menjelaskan alur pembuatan data dan menerapkannya pada bisnis.

6.1. Domain Project

Domain project yang dipilih dalam projek machine learning ini adalah mengenai korban titanic dengan judul Proyek "Prediksi Risiko Diabetes pada pasien berdasarkan serangkaian factor seperti usia, tekanan darah dan sebagainya".

Diabetes adalah suatu kondisi Kesehatan yang menjadi kekhawartiran umum bagi masyarakat umum. Saat ini, masyarakat perlu mewaspadai penyakit kencing manis (hiperglikemia). Penderita diabetes tingkat tinggi berisiko mengalami kematian atau komplikasi. Selain itu, diabetes juga dapat menyebabkan penyakit jantung, stroke, penyakit ginjal, kebutaan dan kerusakan saraf pada kaki. Diagnosis diabetes dapat ditegakkan melalui pengujian rutin dan penilaian resiko pribadi.

Dalam penelitian medis, pengumpulan data pasien termasuk berbagai variable seperti usia, jenis kelamin, riwayat keluarga, BMI (indeks masa tubuh), tekanan darah, dan kadar gula darah membantu dokter dan peneliti memahami factor resiko yang menyebabkan perkembangan diabetes. Dataset tersebut kemungkinan besar merupakan model pelatihan machine learning yang dapat memprediksi apakah anda beresiko terkena diabetes berdasarkan profil kesehatan anda. Dengan menggunakan teknik seperti machine learning dan penambangan data, kami menghasilkan model yang membantu mendeteksi diabetes sejak dini dan memungkinkan intervesi medis yang tepat serta perubahan gaya hidup untuk mengurangi resiko diabetes atau mengobati penyakit tersebut.

6.2. *Bussiness Understanding*

1. ***Problem Statements***

Berdasarkan latar belakang diatas, berikut ini rumusan masalah yang dapat diselesaikan pada proyek ini:

a) Bagaimana kita dapat membangun model prediktif yang dapat mengidentifikasi resiko diabetes seseorang berdasarkan factor Kesehatan spesifik yang disertakan dalam dataset ini?
b) Seberapa akurat model prediktif ini dalam mendeteksi potensi kasus diabetes?
c) Berdasarkan hasil prediksi model ini bagaimana caranya? Intervensi medis dan perubahan gaya hidup mempengaruhi resiko diabetes?

2. ***Goals***

a) Memahami factor kecehatan yang memiliki dampak terbesar dalam memprediksi kemungkinan terkena diabetes.
b) Membangun model prediktif yang akurat dan andal untuk memprediksi resiko diabetes pada populasi yang lebih luas berdasarkan data Kesehatan yang ada.
c) Pengetahuan tentang cara membuat model machine learning untuk memprediksi resiko diabetes.

3. ***Solution Statements***

Solusi yang dapat dilakukan untuk memenuhi tujuan dari proyek ini diantaranya dengan berbagai teknik dapat digunakan untuk prapemrosesan data seperti:

1. Identifkasi dan penanganan nilai yang hilang (missing values) Anda dapat menggantinya dengan mean/median atau menghapus baris/kolom yang berisi nilai yang hilang.
2. Pisahkan kolom target (label) yang ingin anda prediksi (misalnya, kolom yang akan diprediksi menunjukkan apakah seseorang menderita diabetes).
3. Menormalkan atau membakukan fitur numerik sehingga semua fitur numerik skala yang sama.
4. Pisahkan dataset menjadi subset data pelatihan dan subset data pengujian, lalu uji performa model pada data.

Untuk membuat model, kami memutuskan untuk menggunakan model dengan model algoritma *ANN (Artificial Neural Network)*. Algoritma ini dipilih karena kemampuannya dalam menangani masalah kompleks dan memodelkan hubungan nonlinier antara fitur dan label.

Cara kerja Algoritma *ANN (Artificial Neural Network)*:

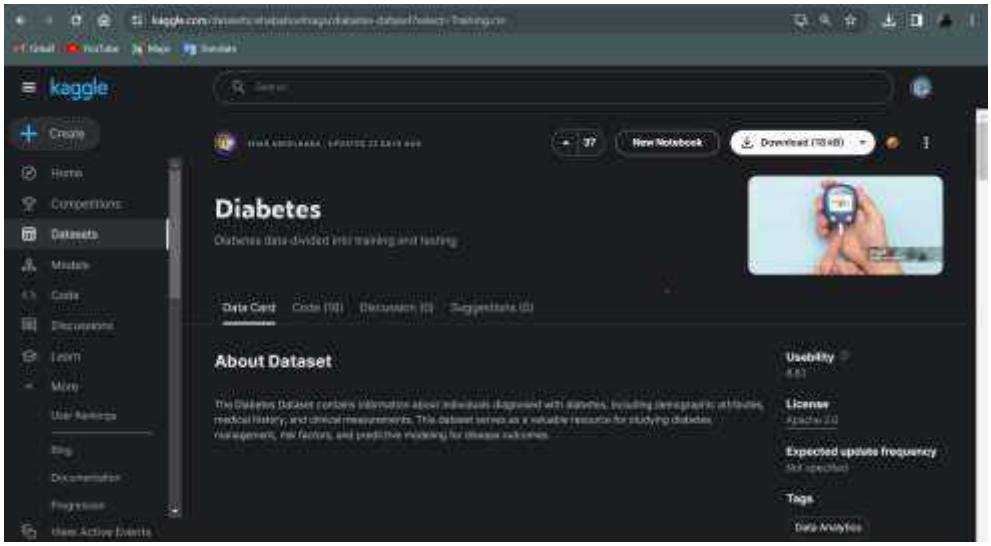
1. Membagi data menjadi fitur (x) dan label (y) dan menggunakannya sebagai variable target.
2. Menerapkan preprocessing pada data.
3. Membagi data menjadi subset data pelatihan dan data uji.

Kelebihan dan kekurangan Algoritma *ANN (Artificial Neural Network)*:

1. Kelebihannya adalah *ANN (Artificial Neural Network)* dapat memodelkan hubungan non-linear yang kompleks antara fitur dan label, sehingga cocok untuk berbagai masalah prediksi yang kompleks.

2. Kekurangan alasan utamanya adalah *ANN (Artificial Neural Network)* memerlukan jumlah yang besar data untuk dilatih secara efektif. Jika dataset terlalu kecil, Model ANN cenderung overfit.

6.3. *Data Understanding*



Gambar 6.1. *Dataset Diabetes*

Data pada project ini menggunakan dataset kaggle, dimana fokus pada data tersebut menjelaskan factor-faktor yang akan mempengaruhi sebuah resiko diabetes.

Tabel 6.1. *Dataset* Diabetes

Jenis	Keterangan
Sumber	Kaggle Dataset : Diabetes Dataset *https://www.kaggle.com/datasets/ehababoelnaga/diabetes-dataset?select=Training.csv*
Lisensi	Apache 2.0
Kategori	Glucose, insulin, age
Jenis dan Ukuran Berkas	CSV (18 Kb)

Pada berkas yang diunduh yakni diabetes.csv berisi 403 rows x 19 columns. Kolom kolom tersebut teridi dari 13 buah kolom bertipe float (numerik), 3 buah kolom bertipe integer dan 3 buah kolom bertipe objek. Untuk penjelasan mengenai variabel-variabel pada dataset diabetes ini dapat dilihat sebagai berikut:

1. id merupakan parameter bernilai unique.
2. chol merupakan jumlah kolesterol.
3. stab.glu merupakan glukosa yang stabil.
4. hdl merupakan lipoprotein kepadatan tinggi.
5. ratio merupakan rasio kolesterol/HDL.
6. glyhb merupakan Hemoglobin Glikosilasi.
7. location merupakan lokasi pasien.
8. age merupakan umur setiap pasien.
9. gender merupakan jenis kelamin setiap pasien.
10. height merupakan tinggi badan setiap pasien.
11. weight merupakan berat badan setiap pasien.
12. frame merupakan 3 level untuk kecil, sedang, besar.
13. bp. 1s merupakan Tekanan Darah Sistolik Pertama.
14. bp. 1d merupakan Tekanan Darah Diastolik Pertama.
15. bp. 2s merupakan Tekanan Darah Sistolik Kedua.
16. bp. 2d merupakan Tekanan Darah Diastolik Kedua.
17. waist merupakan dalam inci.
18. hip merupakan inci.
19. time.ppn merupakan Waktu Postprandial saat Lab Digambar dalam hitungan menit.

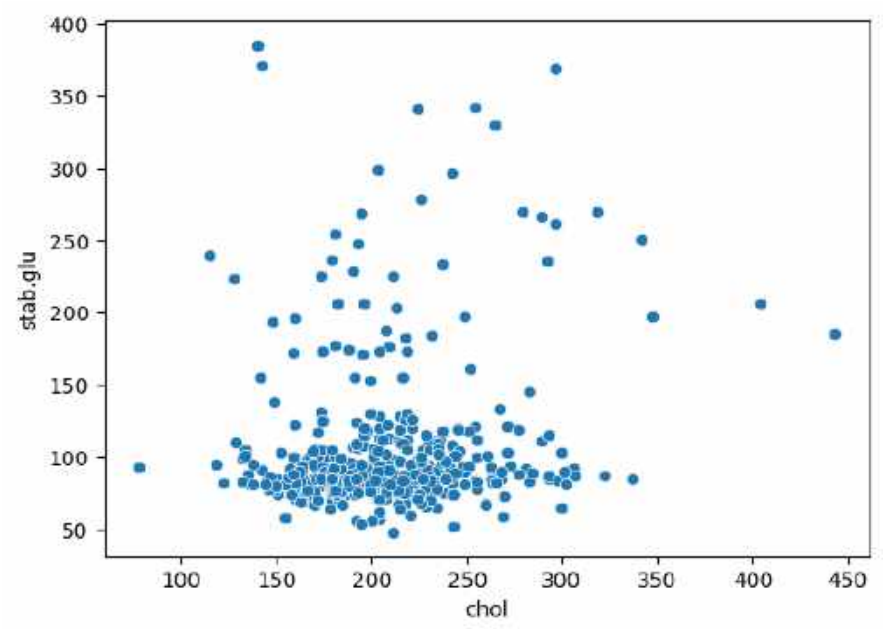
Berikut beberapa tahapan sebelum visualisasi data pada data preparation sebagai berikut:

1. Meload Dataset ke dalam sebuah Dataframe menggunakan *Pandas*.
2. df.info() digunakan untuk mengecek tipe kolom pada dataset.
3. df.isna().sum() digunakan untuk mengecek apakah ada kolom yang kosong.
4. df.describe() digunakan untuk mendapatkan info mengenai dataset terhadap nilai rata-rata, median, banyaknya data.

6.4. *Data Preparation*

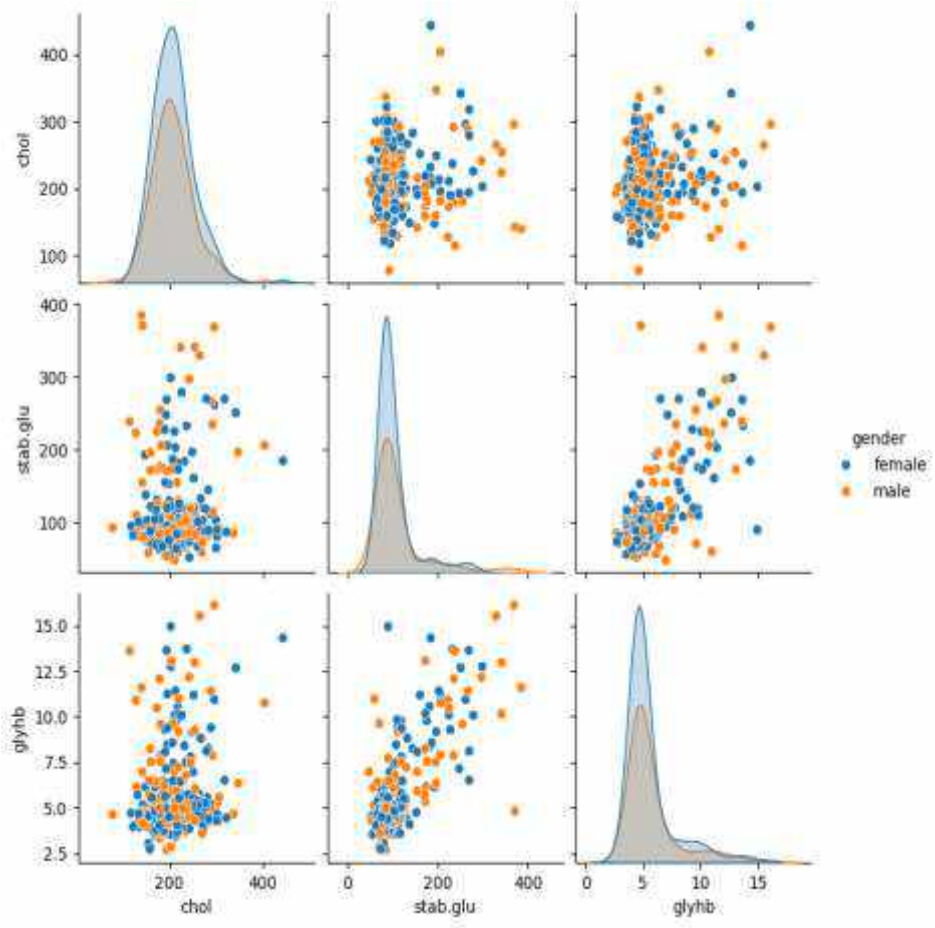
Berikut beberapa tahapan visualisasi data pada *data preparation* :

1. Pertama kita membagi dataset menjadi dua format variable yaitu variable kolom numerik dan variable kolom tipe objek.
2. Selanjutnya, visualisasikan scatter plot yang digunakan untuk memprediksi jumlah kolesterol pada manusia.



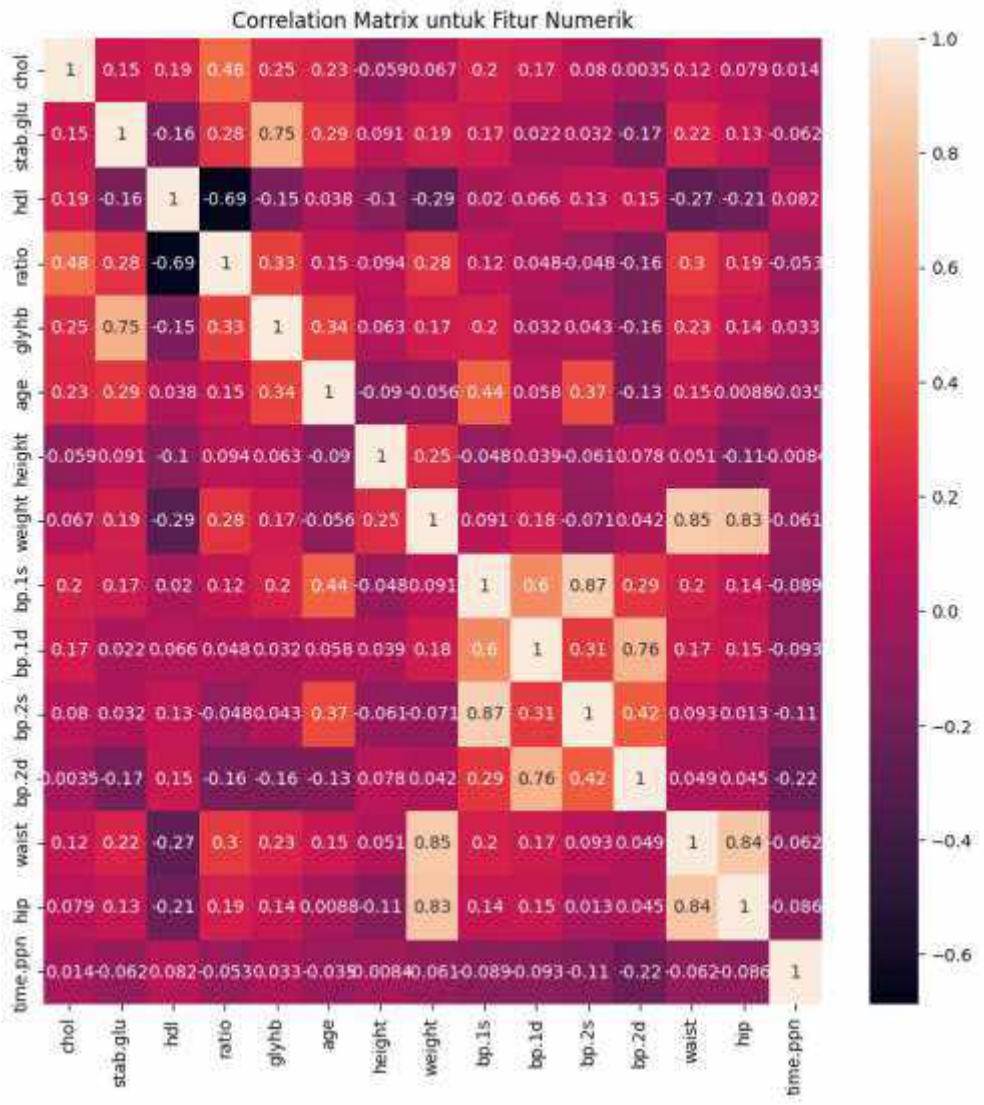
Gambar 6.2. *Visualisasi Data Preparation* Diabetes

3. Selanjutnya, lakukan visualisasi data numerik. Secara rinci dapat dilihat sebagai berikut :



Gambar 6.3. *Visualisasi Data Numerik* Diabetes

4. Melakukan visualisasi untuk melihat sepenuhnya korelasi antar fitur dalam *dataset* anda.



Gambar 6.4. *Corellation Matrix* Diabetes

Berikut adalah tahapan tahapan dalam melakukan pra-pemrosesan data:

1. Memuat *dataset* diabetes dari file *csv* ke dalam struktur dataframe menggunakan Pandas. Ini memungkinkan kita untuk dengan mudah mengakses dan memanipulasi data.

2. Contoh data dibuat menggunakan nilai acak dalam rentang tertentu untuk fitur tertentu menggunakan fungsi np.random. Data tersebut mencakup berbagai karakteristik seperti kolesterol, kadar gula darah dan HDL (high-density lipoprotein) yang stabil, serta rasio kolesterol

```
np.random.seed(42)
data = {
    'id': np.arange(1001, 1021),
    'chol': np.random.uniform(150, 250, 20),
    'stab.glu': np.random.randint(70, 120, 20),
    'hdl': np.random.uniform(30, 70, 20),
    'ratio': np.random.uniform(3, 10, 20),
    'glyhb': np.random.uniform(4, 10, 20),
    'location': np.random.choice(['A', 'B', 'C'], 20),
    'age': np.random.randint(25, 65, 20),
    'gender': np.random.choice(['Male', 'Female'], 20),
    'height': np.random.uniform(150, 190, 20),
    'weight': np.random.uniform(50, 100, 20),
    'frame': np.random.choice(['Small', 'Medium', 'Large'], 20),
    'bp.1s': np.random.randint(90, 160, 20),
    'bp.1d': np.random.randint(60, 100, 20),
    'bp.2s': np.random.randint(90, 160, 20),
    'bp.2d': np.random.randint(60, 100, 20),
    'waist': np.random.uniform(28, 40, 20),
    'hip': np.random.uniform(35, 50, 20),
    'time.ppn': np.random.randint(120, 720, 20)
}
```

3. Selanjutnya, buat kolom kategorikal yang diidentifikasi dalam dataset. Dalam hal ini, lokasi, gender dan frame diidentifikasi sebagai kolom kategorikal

```
categorical_cols = ['location', 'gender', 'frame']
```

4. Selanjutnya, buat ColumnTransformer yang akan dibuat untuk melakukan praproses data. Terdiri dari dua transformer diantaranya *StandardScaler* untuk fitur numerik dan *OneHotEncoder* untuk fitur kategorikal

```
preprocessor = ColumnTransformer(
    transformers=[
        ('num', StandardScaler(), X.select_dtypes(include=['number']).columns),
        ('cat', OneHotEncoder(), categorical_cols)
    ])
```

5. Setelah ColumTransformer akan digunakan untuk menerapkan pra-pemrosesan yang ditentukan sebelumnya pada fitur-fitur dalam dataset.

```
X_processed = preprocessor.fit_transform(X)
```

6.5. *Modelling*

1. Setelah melakukan *preprocessing dataset*, Langkah selanjutnya adalah memodelkan data. Fase ini menggunakan algoritma yang disebut *ANN (Artificial Neural Networks)*. Pertama, data dibagi menjadi subset pelatihan dan pengujian menggunakan train_test_split dari *library Sckit-Learn*.

```
X_train, X_test, y_train, y_test = train_test_split(X_processed, y, test_size=0.2, random_state=42)
```

2. Setelah memisahkan subset pelatihan dan pengujian, buat model neural network yang dibuat menggunakan Sequential API dari TensorFlow. Kedua sistem mencakup beberapa lapisan Dense (padat) dengan fungsi aktivasi ReLu dan lapisan output dengan aktivasi linear (mengamsumsikan "glyhb" adalah target regresi).

```
model = Sequential()
model.add(Dense(32, activation='relu', input_dim=X_train.shape[1]))
model.add(Dense(16, activation='relu'))
model.add(Dense(1, activation='linear'))
```

3. Model disusun dengan mengoptimalkan fungsi kerugian mean squared error (MSE) untuk Adam dan masalah regresi. Model dilatih menggunakan beberapa periode data pelatihan dengan ukuran batch tertentu, dan model dievaluasi pada data pengujian dan mean squared error (MSE) digunakan untuk mengukur performa.

```
model.compile(optimizer='adam', loss='mean_squared_error')
model.fit(X_train, y_train, epochs=50, batch_size=32, validation_split=0.2)

y_pred = model.predict(X_test)
loss = mean_squared_error(y_test, y_pred)
print(f'Test Mean Squared Error: {loss:.4f}')
```

4. Beberapa prediksi model ditampilkan dengan membandingkan nilai actual dengan nilai prediksi dari data pengujian.

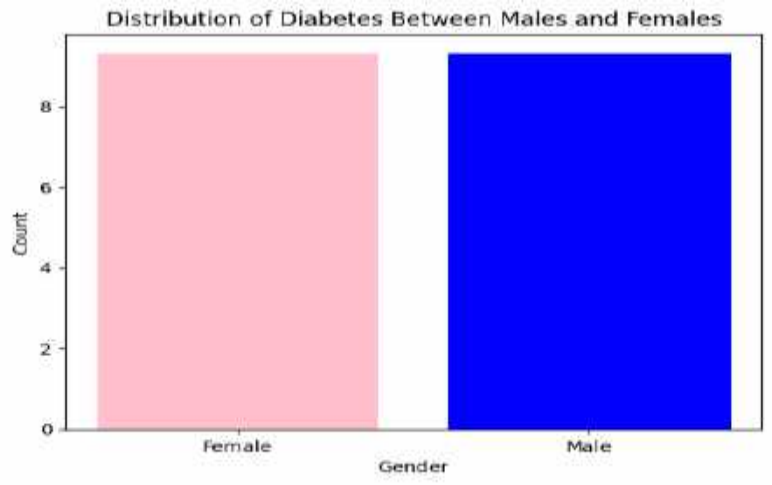
```
print("\nSample Predictions:")
print(pd.DataFrame({'Actual': y_test, 'Predicted': y_pred.flatten()}).head())
```

```
Epoch 45/50
1/1 [==============================] - 0s 42ms/step - loss: 13.1727 - val_loss: 25.6175
Epoch 46/50
1/1 [==============================] - 0s 42ms/step - loss: 12.5689 - val_loss: 25.1177
Epoch 47/50
1/1 [==============================] - 0s 43ms/step - loss: 11.9809 - val_loss: 24.6248
Epoch 48/50
1/1 [==============================] - 0s 43ms/step - loss: 11.4107 - val_loss: 24.1410
Epoch 49/50
1/1 [==============================] - 0s 42ms/step - loss: 10.8573 - val_loss: 23.6632
Epoch 50/50
1/1 [==============================] - 0s 39ms/step - loss: 10.3219 - val_loss: 23.1862
1/1 [==============================] - 0s 59ms/step
Test Mean Squared Error: 17.6297

Sample Predictions:
      Actual  Predicted
0   6.150794   2.072956
17  6.962774   1.863704
15  7.367663   2.354322
1   4.695214   3.035129
```

5. Hasil di atas menunjukkan output yang mewakili contoh data pengujian. Setiap baris memiliki dua kolom : kolom “actual” berisi nilai sebenarnya dari variable target (dalam hal ini “glyhb”) dan kolom “diprediksi (predicted)” berisi nilai yang diprediksi oleh model untuk variable target tersebut. Misalnya pada baris pertama nilai sebenarnya adalah 6,150794 namun nilai prediksi model adalah 2,072956. Hal ini menunjukkan bahwa model memperkirakan nilai “glyhb” untuk sampel ini adalah sekitar 2,072956 cukup jauh dari nilai sebenarnya

6.6. *Evaluation*

Model yang dikembangkan pada proyek ini adalah *Artificial Neural Networks (ANN)*. Dalam hal ini *Artificial Neural Networks (ANN)* yang dibuat adalah model regresi yang berarti bahwa model tersebut bertujuan untuk memprediksi nilai kontinu, yaitu nilai "*glyhb*" (kadar gula darah hemoglobin) berdasarkan fitur lain dalam dataset.



Gambar 6.5. Model *Regresi* Diabetes

Model *Artificial Neural Networks (ANN)* ini memiliki beberapa lapisan padat (dense) yang merupakan lapisan yang terhubung sepenuhnya. Lapisan ini menggunakan fungsi aktivasi ReLu (Rectifed Linear Activation) untuk memasukkan nonlinier ke dalam model. Model ini dibuat menggunakan library TensorFlow dan Keras, yang merupakan framework popular untuk mengembangkan model machine learning. Setelah membangun model, model tersebut dikompilasi menggunakan fungsi kerugian pengoptimal adam dan mean squared error (MSE) karena ini merupakan masalah regresi. Model tersebut kemudian dilatih dan dievaluasi menggunakan data pelatihan.

BAB 7
STUDI KASUS MODEL GRADIENT BOOSTING REGRESSOR

Deksripsi :

Materi pada BAB 7 ini berisi pembahasan mengenai studi kasus dengan menggunakan model algoritma *gradient boosting regressor* untuk melakukan permodelan pada data dan memprediksi sebuah data dengan akurat sehingga mudah dimengerti oleh pembaca.

Tujuan Pembelajaran :

Setelah membaca dan mempratekkan pada materi ini diharapkan pembaca mampu:

1. Mampu menerapkan algoritma *gradient boosting regressor.*
2. Mampu menjelaskan alur pembuatan data dan menerapkannya pada bisnis.

7.1. *Bussiness Understanding*

Contoh studi kasus yang penulis ambil dari penelitian yang dilakukan Hendri Mahmud Nawawi pada tahun 2020 terkait prediksi harga emas berdasarkan pengaruh dari komoditas lain terhadap emas. Emas adalah salah satu komoditas tambang paling berharga di dunia. Keberadaan Emas itu sendiri memiliki banyak fungsi dan peran mulai dari perhiasan, komponen dalam barang elektronik, instrumen pertukaran, kendaraan investasi, hingga cadangan devisa di suatu negara. Harga emas di pasar itu sendiri relatif stabil bahkan cenderung selalu naik meski ada penurunan harga yang signifikan dalam 6 tahun terakhir pasca krisis ekonomi dunia.

Harga emas terus meningkat di pasaran dunia sehingga membuat para investor banyak tertarik untuk berinvestasi pada logam mulia ini.

Emas adalah komoditas utama di pasar ekonomi dan moneter. Setiap hari, nilai emas meningkat dan tidak bisa dikendalikan.

Untuk melakukan prediksi harga emas berdasarkan pengaruh dari komoditas lain terhadap emas dari dataset penelitian yang diperoleh dari data sekunder (*public*) yang diambil dari berbagai sumber data di internet. untuk mengetahui perkiraan harga emas dimasa yang akan datang dengan harapan bisa menjadi tolak ukur untuk meminimalisir resiko kerugian dari berinvestasi emas khususnya harga emas di Indonesia.

Salah satu pengetahuan penting berinvestasi emas adalah prediksi harga emas. Sebagaimana kita ketahui bahwa peramalan merupakan dugaan atau prediksi terhadap kejadian yang akan datang setelah mempelajari kejadian dimasa lalu atau kejadian yang pernah dialami sebelumnya.

Model yang diusulkan adalah metode algortima *Gradient Boosting Regressor*. *Gradient Boosting Regressor* merupaka algoritma pembelajaran mesin yang dapat digunakan untuk masalah regresi dan klasifikasi. Ini menghasilkan model prediksi yang terdiri dari *ensemble* model prediksi lemah pada pohon keputusan.

Supaya masalah yang dibahas fokus terhadap materi yang akan bahas maka dibatasi dalam ruang lingkup pengujian model algoritma *Decission Tree Regressor, Random Forest Regressor, AdaBoost Regressor* dan *Gradient Boosting Regressor* untuk melakukan prediksi terhadap harga emas di Indonesia berdasarkan komoditas lain diantaranya harga saham IHSG, harga minyak mentah, harga perak dan kurs mata uang rupiah terhadap dollar dengan menggunakan parameter *max_leaf_nodes* untuk model *Decission Tree* dan parameter *n_estimator* untuk model *Random Forest Regressor, AdaBoost Regressor* dan *Gradient Boosting Regressor*.

7.2. *Data Understanding*

Dataset yang di ambil merupakan data sekunder yang diperoleh dari berbagai sumber diantaranya dataset harga emas dan harga perak diperoleh dari https://www.bullion-rates.com, histori harga minyak

mentah, IHSG dan Kurs nilai mata uang rupiah terhadap dollar amerika diperoleh dari https://id.investing.com histori harga minyak mentah mengacu pada data histori yang dikeluarkan oleh *West Texas Intermediate* (WTI) dan histori IHSG mengacu pada histori data dari *Jakarta Stock Exchange Composite* (JKSE). Jumlah data yang dikumpulkan sebanyak 1.333 data dari histori 5 tahun terakhir yaitu bulan Januari 2015 sampai dengan bulan Juni 2020.

Untuk mengetahui data secara statistik dijelaskan pada tabel 7.1, pada tabel 7.1 dataset yang dijadikan penelitian tidak dinormalisasi terlebih dahulu hal ini bertujuan untuk melihat data sesuai dengan angka aslinya.

Tabel 7.1. Statistika *Dataset*

	IHSG (Level)	Minyak Mentah (IDR)	Perak/Gram	Kurs IDR/USD	Emas/Gram
ount	1333.000000	1.333000e+03	1333.000000	1333.000000	1333.000000
ean	5572.268777	7.118154e+05	7239.293226	13768.545386	576675.238560
td	644.557975	1.590955e+05	514.357135	634.455873	80472.437025
in	3937.630000	1.547045e+05	5826.110000	12472.500000	456243.000000
5%	5042.870000	6.168268e+05	6878.800000	13320.000000	522808.000000
0%	5702.820000	6.989822e+05	7224.300000	13644.500000	560760.000000
5%	6117.360000	8.049369e+05	7516.020000	14137.500000	593864.000000
ax	6689.290000	1.151881e+06	8907.930000	16575.000000	873369.000000

Salah satu hal penting dalam melakukan penelitian terhadap dataset adalah mengetahui statitik dari dataset itu sendiri, table 7.1 memberikan informasi pada kolom *count* dapat dilihat jumlah data yang digunakan pada penelitian ini, jumlah total data yang digunakan pada penelitian ini adalah sebanyak 1333 data, *mean* adalah nilai tengah dari masing-masing atribut penelitian, std adalah standard deviasi atau simpangan dataset penelitian, *min* adalah nilai paling kecil dari dataset, 25%, 50%, 75% menginformasikan bahwa jumlah dataset pada quartil 1, quartil 2 dan quartil 3 dan nilai *max* adalah nilai paling tinggi dari dataset penelitian.

Informasi atau dataset yang digunakan bisa dilihat pada table 7.2, tabel ini menginformasikan atribut penelitian yang digunakan.

Tabel 7.2. *Dataset* Penelitian

Tanggal	IHSG (Level)	Minyak Mentah (IDR)	Perak/ Gram	Kurs IDR/USD	Emas/Gram
1/2/2015	5242.77	660864.33	6347.01	12542.5	478114
1/5/2015	5219.99	631880.10	6576.78	12627.5	489501
1/6/2015	5169.06	606673.98	6743.27	12657.5	496694
1/7/2015	5207.12	619728.03	6767.45	12738.5	495950
1/8/2015	5211.83	618657.20	6656.78	12680.0	491648
1/9/2015	5216.66	611826.54	6678.14	12651.5	495142
1/12/2015	5187.93	580366.83	6707.57	12597.5	498140
1/13/2015	5214.36	578099.28	6911.58	12597.5	498905
1/14/2015	5159.67	611478.24	6819.82	12613.0	497554
1/15/2015	5188.71	580900.00	6814.24	12560.0	508426
1/16/2015	5148.38	612885.38	7172.66	12587.5	516521
1/19/2015	5152.09	602911.93	7197.70	12618.5	519322
1/20/2015	5166.09	583771.76	7271.54	12584.0	523109
1/21/2015	5215.27	596294.40	7275.96	12480.0	518961
1/22/2015	5253.18	578296.13	7325.29	12487.5	520445
...	...	...	...	...	...
6/23/2020	4879.13	569515.20	8164.37	14160.0	803923
6/24/2020	4964.73	537081.30	7957.72	14130.0	801128

6/25/2020	4896.73	548856.00	8086.58	14175.0	801409
6/26/2020	4904.09	547327.80	8110.56	14220.0	807641
6/29/2020	4901.82	565526.50	8145.73	14245.0	808483
6/30/2020	4905.39	559793.85	8368.50	14255.0	818574

Tabel 7.2 menjelaskan informasi tentang dataset penelitian yang digunakan pada penelitian ini jumlah total data sebanyak 1333 *rows*/baris dan 6 kolom yaitu tanggal, IHSG (Level), Minyak Mentah (Rupiah), Perak (Gram), Kurs (IDR/USD) dan Emas/Gram.

Secara rinci informasi atau nilai dataset pada ke enam kolom yang dijadikan atribut penelitian ini dijelaskan pada tabel 7.3.

Tabel 7.3. Informasi Nilai *Atribut*

Attribute	Deskripsi	Satuan
Tanggal	Merupakan tanggal transaksi yang terjadi	Date
IHSG	Merupakan histori harga Index Harga Saham Gabungan (IHSG) berdasarkan data yang di peroleh dari Jakarta Exchange (JKSE)	Level
Minyak Mentah	Merupakan histori harga minyak mentah saat itu yang telah dikonversi dari nilai dollar ke nilai rupiah per barrelnya	Rupiah/ Barel
Perak	Merupakan histori harga perak dalam satuan gram	Rupiah/Gram
Kurs	Merupakan harga nilai tukar rupiah terhadap dollar amerika serikat	Rupiah
Emas	Merupakan harga histori emas dalam satuan gram	Rupiah/Gram

7.3. *Data Preparation*

Langkah pertama adalah dengan melakukan *preprocessing* data dari dataset penelitian yang digunakan, hal ini bertujuan untuk melihat dataset yang digunakan apakah terdapat *missing values* atau tidak. Pada penelitian ini *preprocessing* data yang digunakan adalah dengan menghapus data yang dianggap *missing* misalnya tidak ada datanya atau kosong nilainya. Hasilnya dijelaskan pada tabel 7.4

Tabel 7.4. *Preprocessing Dataset*

	Sebelum *Preprocessing*	Setelah *Preprocessing*
Jumlah Kolom	6	6
Jumlah Baris	1333	1333

Jika diperhatikan pada dataset penelitian yang digunakan antara data sebelum preprocessing dan setelah preprocessing tidak ada perbedaan antara jumlah kolom dan jumlah baris khususnya. Dari tabel 7.4. dapat ditarik kesimpulan bahwa tidak ada missing values pada dataset penelitian ini atau missing values = 0%.

Dari dataset yang dimiliki yaitu atribut Tanggal, IHSG (level), Minyak Mentah, Perak/Gram, Kurs IDR/USD dan Emas/Gram maka salah satu atribut dipakai untuk dijadikan target, karena penelitian ini akan melakukan prediksi terhadap harga emas maka yang dijadikan variabel target adalah Emas/Gram. Maka variable selain emas dijadikan variabel independen (yang mempengaruhi) dan emas/gram sebagai variable dependen (yang dipengaruhi).

Tabel 7.5. *Atribut* Target

Data Ke	Emas/Gram
0	478114
1	489501
2	496694
3	495950
4	491648
5	495142
6	498140
7	498905
8	497554
9	508426
10	516521
11	519322
12	523109
13	518961
14	520445
15	519025
	...
1315	774830
1316	756996
1317	764356
1318	780955
1319	776430
1320	785833
1321	777571
1322	779886
1323	783989
1324	780746
1325	788543
1326	797558
1327	803923
1328	801128

1329	801409
1330	807641
1331	808483
1332	818574

Tabel 7.5 memberikan informasi harga emas yang akan dijadikan variable target atau label.

Dataset yang digunakan terdiri dari enam atribut, satu diantaranya dijadikan sebagai variabel target, maka sisanya ada lima atribut yaitu Tanggal, IHSG, Minyak Mentah, Harga Perak dan Kurs Rupiah, karena kasus yang digunakan adalah regresi maka fitur yang digunakan pada pemilihan atribut ini hanya nilai dengan tipe numerik saja yang dipakai, proses pemilihan atribut ini dinamakan *feature selection* sehingga dari ke lima atribut yang dipakai adalah IHSG, Minyak Mentah, Harga Perak dan Kurs Rupiah yang digunakan sisanya atribut tanggal ditinggalkan karena tidak termasuk tipe data numerik.

```
data = pd.read_csv("emas.csv", sep=";")
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 41188 entries, 0 to 1332
Data columns (total 6 columns):
Tanggal                      1333 non-null object
ISHG (Level)                 1333 non-null float64
Minyak Mentah (IDR)          1333 non-null float64
Perak/Gram                    1333 non-null float64
Kurs IDR/USD                 1333 non-null float64
Emas/Gram                    1333 non-null int64
dtypes: float64(4), int64(1), object(11)
memory usage:  62.6+ MB
```

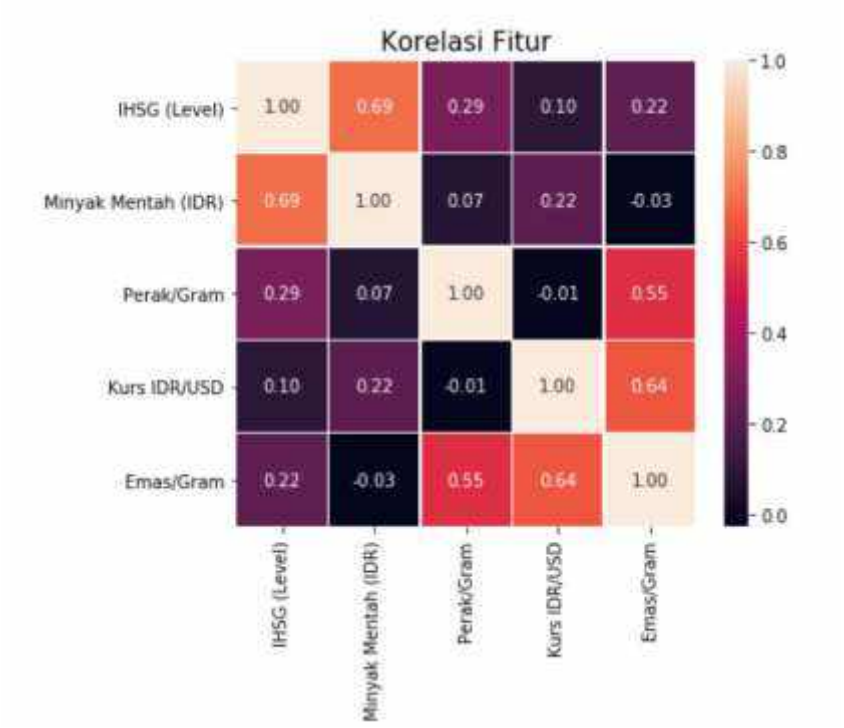
Tipe data untuk atribut tanggal adalah object bukan salah satu dari tipe data numerik dan sisanya termasuk tipe data numerik. Ke empat fitur yang dipilih ini dinamakan prediktor atau variabel independen.

Tabel 7.6. *Atribut Prediktor*

Data Ke	IHSG (Level)	Minyak Mentah (IDR)	Perak/Gram	Kurs IDR/USD
0	5242.77	660864.33	6347.01	12542.5
1	5219.99	631880.10	6576.78	12627.5
2	5169.06	606673.98	6743.27	12657.5
3	5207.12	619728.03	6767.45	12738.5
4	5211.83	618657.20	6656.78	12680.0
5	5216.66	611826.54	6678.14	12651.5
6	5187.93	580366.83	6707.57	12597.5
7	5214.36	578099.28	6911.58	12597.5
8	5159.67	611478.24	6819.82	12613.0
9	5188.71	580900.00	6814.24	12560.0
10	5148.38	612885.38	7172.66	12587.5
11	5152.09	602911.93	7197.70	12618.5
12	5166.09	583771.76	7271.54	12584.0
13	5215.27	596294.40	7275.96	12480.0
14	5253.18	578296.13	7325.29	12487.5
15	5323.89	568621.28	7338.00	12472.5
…	…	…	…	…
1317	5035.06	540876.60	7823.62	13890.0
1318	4920.68	553608.00	8140.98	13980.0
1319	4854.75	509305.10	7929.46	14015.0
1320	4880.36	512172.50	7940.22	14125.0
1321	4816.34	523948.80	7829.27	14115.0
1322	4986.46	540774.20	7886.52	14090.0
1323	4987.78	534571.70	7945.60	14082.5
1324	4925.25	547061.40	7879.38	14085.0
1325	4942.27	560475.00	7961.71	14100.0
1326	4918.83	573925.10	8054.58	14185.0
1327	4879.13	569515.20	8164.37	14160.0
1328	4964.73	537081.30	7957.72	14130.0
1329	4896.73	548856.00	8086.58	14175.0

1330	4904.09	547327.80	8110.56	14220.0
1331	4901.82	565526.50	8145.73	14245.0
1332	4905.39	559793.85	8368.50	14255.0

Model analisis multivariat regresi dipilih karena pengolahan variabel penelitian ini melibatkan jumlah variabel independen sebanyak n dan lebih dari satu. Tujuannya adalah mencari pengaruh variabel-variabel tersebut terhadap suatu *output* (variabel dependen) secara simultan atau serentak. Pada hasil prediksi harga emas nilai koefisien masing-masing variabel dicatat untuk melihat seberapa besar pengaruh harga saham IHSG, Harga minyak mentah, Harga perak dan Kurs nilai rupiah terhadap dollar mempengaruhi harga emas. Secara lebih jelas divisualisasikan dengan gambar 7.2



Gambar 7.1. Korelasi Fitur

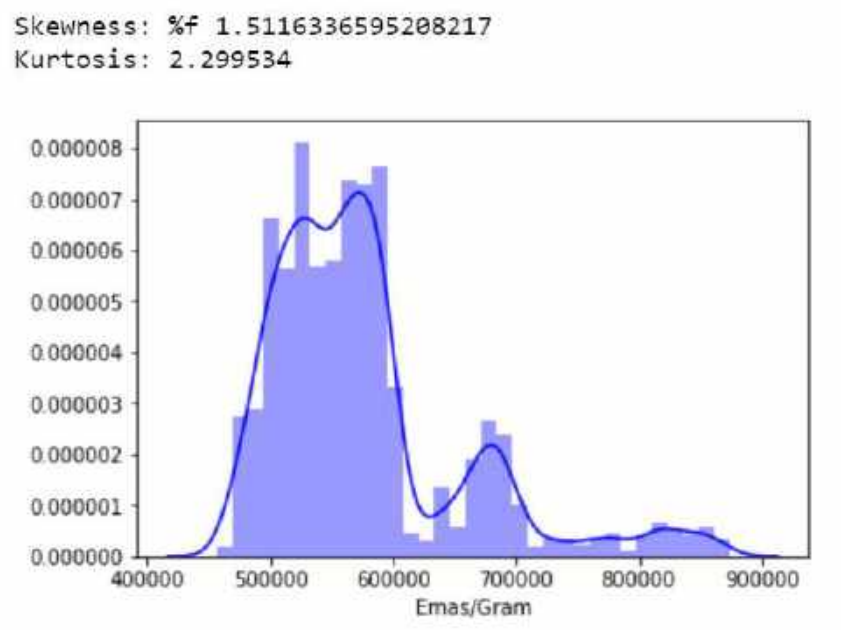
Nilai masing-masing dari penggambaran matriks diatas dijelaskan pada tabel 7.7

Tabel 7.7. Korelasi Nilai Atribut

Atribut	Nilai Korelasi
Kurs IDR/USD	0.635034
Perak/Gram	0.546610
IHSG (Level)	0.224111
Minyak Mentah (IDR)	-0.027358

Tabel 7.7 menampilkan korelasi dari masing-masing atribut. Nilai Kurs IDR/USD memiliki nilai korelasi paling tinggi dengan nilai 0.635034 dibandingkan nilai korelasi atribut lainnya artinya nilai Kurs Rupiah terhadap dollar memiliki pengaruh paling besar dalam menentukan harga emas pada dataset penelitian ini, harga perak memiliki nilai korelasi sebesar 0.546610, nilai IHSG 0.224111 dan korelasi paling kecil adalah harga minyak mentah yaitu sebesar -0.027358.

Uji nilai kurtosis dan Skewness



Gambar 7.2. Uji *Kurtosis* dan *Skewness*

Dari gambar 7.2. diperoleh nilai *skewness* sebesar 1.512. Nilai skewness menunjukkan simetri maka dikatakan data membentuk distribusi normal, apabila kemiringan distribusi data agak condong ke kanan ditunjukkan dengan nilai *skewness negative*, selanjutnya apabila kemiringan distribusi data condong ke kiri yang ditunjukkan bahwa nilai *skewness positif*. Maka kesimpulannya dataset ini lebih condong ke kiri karena nilai *skewness positif*. Nilai Kurtosis = 2.296, aturannya jika nilai kurtosis dekat nol maka data cenderung normal, apabila nilai kurtosis negatif berarti datanya tumpul atau cenderung melebar ke bawah, sebaliknya apabila nilai kurtosis positif maka datanya bersifat runcing atau cenderung mengelompok (homogen). Dari dataset harga emas berdasarkan nilai kurtosis pada gambar 7.3 cenderung meruncing membentuk lonceng artinya data yang digunakan bersifat homogen.

7.4. *Modelling*

1. Membangun Model *Decision Tree Regressor*

Tahap paling penting dari penelitian ini adalah menguji dataset dengan menggunakan berbagai model regresi untuk melihat akurasi nilai Error pada model yang dibangun berdasarkan dataset penelitian. Pada *phyton* terdapat *Library sklearn* fungsi ini untuk mengkover banyak model *machine learning* salah satu nya adalah *Decision Tree.* Untuk pemanggilan *Library sklearn* adalah sebagai berikut

```
from sklearn.tree import DecisionTreeRegressor
```

Dari *listing code* diatas model *decision tree* untuk *regresi* di import untuk menguji dataset. Proses selanjutnya melakukan training model terhadap fitur dan labelnya.

```
df_model = DecisionTreeRegressor(random_state=1)
df_model.fit (X,y)
```

Hasil ketika di run:

```
DecisionTreeRegressor(criterion='mse', max_depth=None,
     max_features=none, max_leaf_node=None,
     min_impurity_decrease=0.0, min_impurity_slit=None,
     min_sample_leaf=1, min_sample_split=2,
     min_weight_fraction_leaf=0.0,presort=False,
     random_state=1,splitter='best')
```

Data diatas menjelaskan proses *training* yang dilakukan dengan model *decision tree* dimana kriteria yang dicari adalah nilai *Mean Square Error* (MSE).

Setelah modelnya di *training* selanjutnya adalah melakukan prediksi dengan menyertakan parameter fiturnya dan parameter prediksinya. Namun sebelumnya dataset yang dibagi dua dengan membaginya menjadi data *training* dan data *testing*. Hasil nilai prediksinya pada tabel 7.8

Tabel 7.8. Nilai Prediksi dengan *Decission Tree*

Nilai Aktual	Prediksi
478114	478114
489501	489501
496694	496694
495950	495950
491648	491648

Dari tabel 7.8 antara nilai aktual dan nilai prediksi tidak ada perbedaan karena data *training* yang digunakan sama dengan data yang diprediksi. Setelah melihat model prediksi pada *decision tree* selanjtnya adalah melakukan evaluasi berdasarkan niai *Mean absolute error* (MAE) dan *Root Mean Square Error* (RMSE). Pada pengujian kali ini data dibagi dua yaitu data *training* dan data *testing* dengan *Library* di *phyton*.

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,y,
random_state=0, test_size = 0.4)
#membuat model dengan split data 40% - 60%
df_model = DecisionTreeRegressor(random_state=1)
df_model.fit(X_train,y_train)
```

Model tersebut dilakukan evaluasi dan validasi berdasarkan nilai MAE, MSE dan RMSE, hasilnya bisa dilihat pada tabel 7.9

Tabel 7.9. Evaluasi dan Validasi *Decision Tree*

Metode	MAE	RMSE
Decision Tree tanpa pembagian *training* dan *testing*	0	0
Decision Tree dengan pembagian *training* dan *testing* 40% - 60%	1682.652	8774.938

Berdasarkan Tabel 7.9 dengan model *Decission Tree* tanpa pembagian data menjadi data training dan data testing menghasilkan nilai MAE dan RMSE tidak ada *Error* yaitu 0 dikarenakan *machine learning* mengerjakan prediksi yang sama antara data *training* dan data *testing*nya dan dataset yang digunakan sedikit. Kemudian data dibagi menjadi dua dengan data *training* dan data *testing* nilai MAE yang dihasilkan adalah 1682.652 dan nilai RMSE 8774.9 dan jika diperhatikan sebetulnya untuk nilai MAE dan RMSE tersebut cukup baik untuk dataset ini menggunakan pembagian data *training* dan data *testing* pada model *decision tree.*

Untuk menguji nilai *Error* tersebut diperlukan optimasi model dengan tujuan mengetahui parameter paling optimal untuk mendapatkan nilai *Error* yang lebih baik. Pada model *decision tree* ada beberapa *set up* parameter yang bisa dilakukan untuk menguji model salah satunya dengan mengubah nilai *maksimum leaf node,* karena pada *Decision Tree maksimum leaf node* nya bisa dikontrol, maka dari itu penulis melakukan optimasi pada nilai *maksimum leaf node,* jumlah *maksimum leaf node* yang digunakan adalah 10, 100, 300, 500 dan 1000. Hasil dari pengujian dengan melakukan optimasi *maksimum leaf node* di jelaskan pada tabel 7.10

Tabel 7.10. Perbandingan Nilai Parameter *Maksimum Leaf Node*

Maksimum leaf node	**MAE**	**RMSE**
None	1682.652	8774.938
5	35593.287	48684.457
10	19881.379	29046.457
100	2893.080	8957.604
300	1789.550	8692.497
500	1745.135	8768.997
1000	1745.135	8768.997

Dari Tabel 7.10 jika diperhatikan semakin banyak maksimum leaf node tidak menjadikan nilai MAE dan RMSE semakin kecil, dari tabel 7.10 disimpulkan nilai MAE dan RMSE paling optimal untuk dataset penelitian ini adalah dengan model *decision tree* dan *maksimun leaf node* sebanyak 10.

2. Membangun Model *Random Forest Regressor*

Model kedua dengan menggunakan algoritma *Random Forest,* seperti namanya *random forest* adalah kumpulan dari algortima *Decision Tree.* Modul yang di import pada *machine learning* adalah sebagai berikut:

```
from sklearn.ensemble import RandomForestRegressor
```

Pada model *Random Forest* langkah pertama adalah menentukan *n_estimator* yaitu feature untuk menentukan jumlah *decision tree* yang akan dibangun oleh *machine learning*. Percobaan pertama dengan menggunakan 100 *n_estimator*

```
from sklearn.ensemble import RandomForestRegressor
rf_model =
RandomForestRegressor(n_estimator=100,random_state=1)
rf_model.fit(X_train,y_train)
```

Hasil ketika di run:

```
RandomForestRegressor(bootstrap=True, criterion='mse',
    max_depth=None, max_features='auto',
    max_leaf_node=None, min_impurity_decrease=0.0,
    min_impurity_slit=None, min_sample_leaf=1,
    min_sample_split=2, min_weight_fraction_leaf=0.0,
    n_estimator=100, n_jobs=None, oob_score=False,
    random_state=1, verbose=0, warm_start=False)
```

Pengujian dataset menggunakan model *Random Forest* dengan 100 *n_estimator* di dapatkan nilai MAE sebesar 4798,724 dan RMSE 7970,587. Jika diperhatikan nilai MAE dan RMSE yang dihasilkan dengan *Random Forest Regresi* nilai MAE dan RMSE nya lebih baik daripada dengan model

Decission tree hal ini tidak diherankan karena model *random forest* mengakomodir sejumlah *decision tree.*

Tahap selanjutnya adalah melakukan optimalisasi pada model *random forest* dengan melakukan *set up* pada *n_estimator* sebanyak 50, 100, 200, 500, 1000 dan 5000. Hasil nilai yang di dapatkan dijelaskan pada tabel 7.11

Tabel 7.11. *Optimasi n_estimator Random Forest*

N_estimator	MAE	RMSE
50	1432.905	8020.948
100	1448.969	8170.991
200	1433.861	8242.426
500	1410.229	8178.203
1000	1401.017	8133.987
5000	1385.931	8075.002

Pada Tabel 7.11 dengan melakukan optimasi pada jumlah *n_estimator* dengan nilai 50, 100, 200, 500, 1000 dan 5000 nilai optimal di dapatkan pada n_estimator 500 dan jika dilihat semakin besar *n_estimator* yang digunakan MAE dan RMSE yang didapatkan tidak selalu baik dan optimal.

3. Membangun Model *AdaBoost Regressor*

Algoritma AdaBoost dapat meningkatkan kinerja pohon keputusan, AdaBoost dapat meningkatkan kinerja pengklasifikasi yang lemah dengan memperkuat pelatihan pada sampel yang diklasifikasikan salah. *Dataset* akan diuji dengan menggunakan model *AdaBoost Regressor.* pemanggilan *Library phyton* dengan model ini adalah sebagai berikut:

```
from sklearn.ensemble import AdaBoostRegressor
adaboost = AdaBoostRegressor(n_estimators = 100,
random_state = 1)
adaboost.fit(X_train, y_train)
```

Hasil ketika di run:

```
AdaBoostRegressor(base_estimator=None, learning_rate=1.0,
loss='linear', n_estimator=100, random_state=1
```

Hasil dengan pengujian menggunakan model *AdaBoostRegressor* ini diperoleh nilai MAE sebesar 8607.989 dan nilai RMSE 14087.7023. Dari hasil test set pengujian hasil ini tentu jika dibandingkan dengan algoritma sebelumnya yaitu *random forest* nilai MAE dan RMSE nya masih lebih besar. Maka dari itu langkah selanjutnya dengan melakukan *hyperparameter* dengan merubah nilai *n_estimator* dengan nilai 50, 100, 200, 500, 1000 dan 5.000.

Tabel 7.12. *Optimasi n_estimator AdaBoost Regressor*

N_estimator	MAE	RMSE
50	8607.989	14087.702
100	8607.989	14087.702
200	8607.989	14087.702
500	8607.989	14087.702
1000	8607.989	14087.702
5000	8607.989	14087.702

Kesimpulan yang dapat diperoleh dengan melakukan optimasi *n_estimator* pada model *AdaBoost Regressor* adalah tidak ada pengaruh dengan melakukan perubahan pada *n_estimator* pada hasil MAE dan RMSE nya tetap sama.

4. Membangun Model *Gradient Boosting Regressor*

Sebagaimana model sebelumnya model algoritma yang digunakan untuk meningkatkan nilai akurasi pada kasus regresi adalah *Gradient Boosting.* Model ini dapat digunakan untuk masalah regresi dan klasifikasi dengan menghasilkan model prediksi yang terdiri dari *ensemble* model prediksi lemah pada pohon keputusan. Pemanggilan *Library phyton* dengan model ini adalah sebagai berikut:

```
from sklearn.ensemble import GradientBoostingRegressor
Gradient = GradientBoostingRegressor(n_estimators = 100,
random_state = 1)
Gradient.fit(X_train, y_train)
```

Hasil ketika di run:

```
GradientBoostingRegressor (alpha=0.9,
    criterion='friedmen_mse', init=None,
    learning_rate=0.1, loss='ls', max_depth=3,
    max_features=None, max_leaf_node=None,
    min_impurity_decrease=0.0, min_impurity_slit=None,
    min_sample_leaf=1, min_sample_split=2,
    min_weight_fraction_leaf=0.0, n_estimator=100,
    n_iter_no_change=None, presort='auto',
    random_state=1, subsample=1.0, to;=0.0001,
    validation_fraction=0.1, verbose=0, warm_start=False)
```

Pengujian model dengan algoritma *Gradient Boosting Regressor* diperoleh nilai MAE 1442.409 dan nilai RMSE sebesar 7218.991. Selanjutnya adalah melakukan set up parameter yang sama yaitu *n_estimator* dengan nilai 50, 100, 200, 500, 1000 dan 5000.

Tabel 7.13. *Optimasi n_estimator Gradient Boosting Regressor*

N_estimator	MAE	RMSE
50	1683.638	7473.156
100	1442.409	7218.991
200	1442.409	7217.882
500	1442.409	7217.882
1000	1442.409	7217.882
5000	1442.409	7217.882

Pada tabel 7.13 dengan melakukan optimasi pada nilai *n_estimator* semakin besar nilai *n_estimator* nya maka nilai MAE dan RMSE nya sama.

7.5. *Evaluation*

1. Pemilihan Model Terbaik

Setelah melakukan pecobaan dengan empat model algoritma regresi yaitu *Decision Tree Regressor, Random Forest Regresor, AdaBoost Regressor dan Gradient Boosting Regressor* langkah selanjutnya adalah melihat model terbaik berdasarkan parameter yang dijadikan nilai optimasi ke empat model tersebut untuk menguji dataset penelitian harga emas.

Tabel 7.14. Perbandingan Model Algoritma

Model	Parameter		MAE	RMSE
Decision Tree Regressor	*Maksimum leaf node*	None	1682.652	8774.938
		5	35593.287	48684.457
		10	19881.379	29046.457
		100	2893.080	8957.604
		300	1789.550	8692.497
		500	1745.135	8768.997
		1000	1745.135	8768.997
Random Forest Regressor	*N_estimator*	50	1432.905	8020.948
		100	1448.969	8170.991
		200	1433.861	8242.426
		500	1410.229	8178.203
		1000	1401.017	8133.987
		5000	1385.931	8075.002
AdaBoost Regressor	*N_estimator*	50	8607.989	14087.702
		100	8607.989	14087.702
		200	8607.989	14087.702
		500	8607.989	14087.702
		1000	8607.989	14087.702
		5000	8607.989	14087.702
Gradient Boosting Regressor	*N_estimator*	50	1683.638	7473.156
		100	1442.409	7218.991
		200	1480.980	7217.882
		500	1480.980	7217.882
		1000	1480.980	7217.882
		5000	1480.980	7217.882

Dari Tabel 7.14 pada dataset penelitian *history* harga emas algoritma terpilih adalah *Gradient Boosting Regressor* dengan mengubah *set up* parameter nilai *n_estimator* = 100 adalah model terbaik berdasarakan dataset penelitian ini dengan nilai MAE 1442.409 dan nilai RMSE 7218.991.

2. Evaluasi Nilai *R-Squared*

Evaluasi selanjutnya adalah melihat nilai *R-Square* dari masing-masing model yang dijadikan pada penelitian ini pengaruh yang diberikan variabel bebas atau variabel independent (X) yaitu IHSG, Minyak Mentah, Perak/Gram dan nilai Kurs IDR/USD terhadap terhadap variabel terikat atau *variabel dependent* (Y) yaitu harga emas, atau dengan kata lain, nilai koefisien determinasi atau R Square ini berguna untuk memprediksi dan melihat seberapa besar kontribusi pengaruh yang diberikan variabel X secara simultan (bersama-sama) terhadap variabel Y.

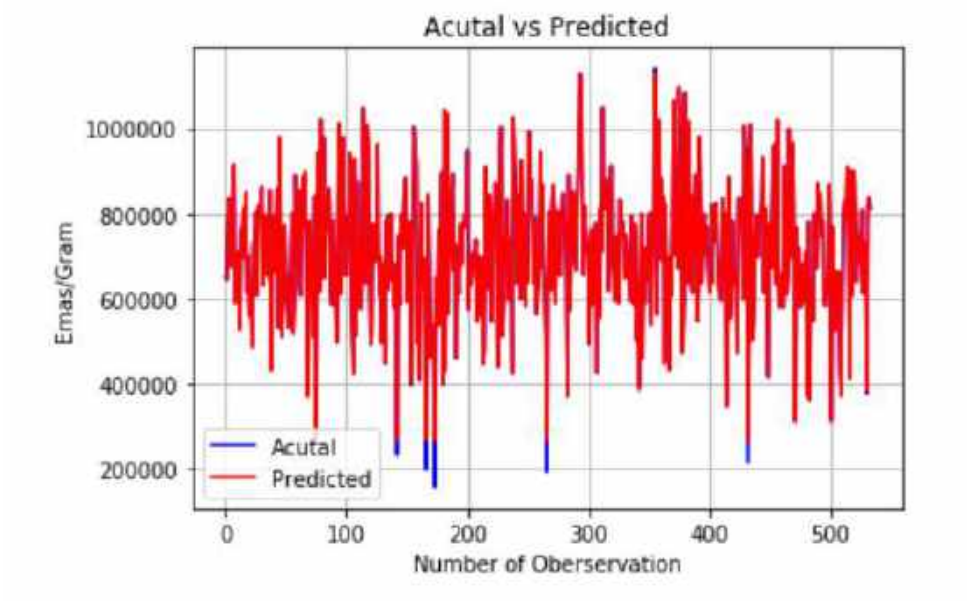
Tabel 7.15. Perbandingan Nilai *R-Squared*

Model	**Nilai R^2**
Decision Tree Regressor	0.9971
Random Forest Regressor	0.9974
AdaBoost Regressor	0.9922
Gradient Boosting Regressor	0.9980

Berdasarkan Tabel 7.15 bisa diambil kesimpulan bahwa variabel bebas atau *variabel independent* (X) yaitu IHSG, Minyak Mentah, Perak/Gram dan nilai Kurs IDR/USD berpengaruh sangat besar terhadap prediksi harga emas dengan *R-Squared* paling tinggi adalah model *Gradient Boosting Regressor* nilainya hampir mendekati angka satu yaitu sebesar 0.9980.

3. Hasil Prediksi

Selanjutnya melihat hasil prediksi dari dari data aktual terhadap prediksinya.



Gambar 7.3. Nilai Aktual Terhadap Prediksi

Gambar 7.3. menjelaskan bahwa perbandingan data aktual dengan data prediksi hampir tidak ada celah atau prediksi yang melenceng dari nilai aktualnya dimana ketika *trend* emas secara aktual menurun data prediksinya juga menurun berdasarkan dataset 1 Januari 2015 – Juni 2020.

A. *Building Decision Tree Feature Scaling*

```
import matplotlib.pyplot as plt
import numpy as np
def gini(p):
      return (p)*(1 - (p)) + (1 - p)*(1 - (1-p))
def entropy(p):
      return - p*np.log2(p) - (1 - p)*np.log2((1 - p))
def error(p):
      return 1 - np.max([p, 1 - p])
x = np.arange(0.0, 1.0, 0.01)
ent = [entropy(p) if p != 0 else None for p in x]
sc_ent = [e*0.5 if e else None for e in ent]
err = [error(i) for i in x]
fig = plt.figure()
ax = plt.subplot(111)
for i, lab, ls, c, in zip([ent, sc_ent, gini(x), err],
                  ['Entropy', 'Entropy (scaled)',
                        'Gini Impurity',
                        'Misclassification Error'],
                  ['-', '-', '--', '-.'],
                  ['black', 'lightgray',
                        'red', 'green', 'cyan']):
      line = ax.plot(x, i, label=lab,
            linestyle=ls, lw=2, color=c)
ax.legend(loc='upper center', bbox_to_anchor=(0.5, 1.15),
            ncol=5, fancybox=True, shadow=False)
ax.axhline(y=0.5, linewidth=1, color='k', linestyle='--')
ax.axhline(y=1.0, linewidth=1, color='k', linestyle='--')
plt.ylim([0, 1.1])
plt.xlabel('p(i=1)')
plt.ylabel('Impurity Index')
plt.show()

from sklearn.tree import DecisionTreeClassifier
tree = DecisionTreeClassifier(criterion='gini',
                              max_depth=4,
                              random_state=1)
tree.fit(X_train, y_train)
```

```
X_combined = np.vstack((X_train, X_test))
y_combined = np.hstack((y_train, y_test))
plot_decision_regions(X_combined,
                      y_combined,
                      classifier=tree,
                      test_idx=range(105, 150))
plt.xlabel('petal length [cm]')
plt.ylabel('petal width [cm]')
plt.legend(loc='upper left')
plt.show()

#menambahkan format PNG untuk decision tree di direktori
local
#install http://www.graphviz.org/Download.php
pip3 install graphviz
pip3 install pyparsing
from pydotplus import graph_from_dot_data
from sklearn.tree import export_graphviz
dot_data = export_graphviz(tree,
                           filled=True,
                           rounded=True,
                           class_names=['Setosa',
                                        'Versicolor',
                                        'Virginica'],
                           feature_names=['petal length',
                                          'petal width'],
                           out_file=None)
graph = graph_from_dot_data(dot_data)
graph.write_png('tree.png')
```

B. Combining Multiple Decision Trees Via Random Forests

```
from sklearn.ensemble import RandomForestClassifier
forest = RandomForestClassifier(criterion='gini',
                                n_estimators=25,
                                random_state=1,
                                n_jobs=2)
forest.fit(X_train, y_train)
```

```
plot_decision_regions(X_combined, y_combined,
classifier=forest, test_idx=range(105,150))
plt.xlabel('petal length')
plt.ylabel('petal width')
plt.legend(loc='upper left')
>>> plt.show()
```

C. *KNN Model In Scikitlearn Using A Euclidean Distance Metric*

```
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors=5, p=2,
                           metric='minkowski')
knn.fit(X_train_std, y_train)
plot_decision_regions(X_combined_std, y_combined,
                      classifier=knn,
                      test_idx=range(105,150))
plt.xlabel('petal length [standardized]')
plt.ylabel('petal width [standardized]')
plt.legend(loc='upper left')
plt.show()
```

D. *SVM Model*

```
from sklearn.svm import SVC
svm = SVC(kernel='linear', C=1.0, random_state=1)
svm.fit(X_train_std, y_train)
plot_decision_regions(X_combined_std,
                      y_combined,
                      classifier=svm,
                      test_idx=range(105, 150))
plt.xlabel('petal length [standardized]')
plt.ylabel('petal width [standardized]')
plt.legend(loc='upper left')
plt.show()
```

```
from sklearn.linear_model import SGDClassifier
ppn = SGDClassifier(loss='perceptron')
lr = SGDClassifier(loss='log')
```

```
svm = SGDClassifier(loss='hinge')
```

```
import matplotlib.pyplot as plt
import numpy as np
np.random.seed(1)
X_xor = np.random.randn(200, 2)
y_xor = np.logical_xor(X_xor[:, 0] > 0,
                       X_xor[:, 1] > 0)
y_xor = np.where(y_xor, 1, -1)
plt.scatter(X_xor[y_xor == 1, 0],
            X_xor[y_xor == 1, 1],
            c='b', marker='x',
            label='1')
plt.scatter(X_xor[y_xor == -1, 0],
            X_xor[y_xor == -1, 1],
            c='r',
            marker='s',
            label='-1')
plt.xlim([-3, 3])
plt.ylim([-3, 3])
plt.legend(loc='best')
plt.show()
```

```
svm = SVC(kernel='rbf', random_state=1, gamma=0.10,
C=10.0)
svm.fit(X_xor, y_xor)
plot_decision_regions(X_xor, y_xor, classifier=svm)
plt.legend(loc='upper left')
plt.show()
```

```
svm = SVC(kernel='rbf', random_state=1, gamma=0.2, C=1.0)
svm.fit(X_train_std, y_train)
plot_decision_regions(X_combined_std,
                      y_combined, classifier=svm,
                      test_idx=range(105,150))
plt.xlabel('petal length [standardized]')
plt.ylabel('petal width [standardized]')
plt.legend(loc='upper left')
plt.show()
```

```
svm = SVC(kernel='rbf', random_state=1, gamma=100.0,
C=1.0)
svm.fit(X_train_std, y_train)
plot_decision_regions(X_combined_std,
                      y_combined, classifier=svm,
                      test_idx=range(105,150))
plt.xlabel('petal length [standardized]')
plt.ylabel('petal width [standardized]')
plt.legend(loc='upper left')
plt.show()
```

Dalam buku ini dapat diharapkan pembaca dapat mempelajari berbagai algoritma *machine learning* yang berbeda yang dapat digunakan untuk mengatasi masalah *linier* dan *nonlinier*. Memungkinkan pembaca untuk memprediksi kemungkinan suatu peristiwa tertentu serta membuat prediksi yang baik. Namun, yang lebih penting lagi adalah pemilihan algoritma pembelajaran yang tepat dengn algoritma yang berbeda.

7.6 Tugas dan Proyek Latihan

Implementasikan *deployment model* dari studi kasus algoritma diatas dan pastikan dapat bekerja dengan baik.

GLOSARIUM

A

Algoritma	Rangkaian langkah-langkah logis dan sistematis yang digunakan untuk menyelesaikan suatu masalah tertentu.
Attribut	Karakteristik atau properti yang mendeskripsikan suatu objek, elemen, atau file.
Artificial Intelligence	Bidang ilmu komputer yang didedikasikan untuk menciptakan kecerdasan pada mesin.

B

Business Understanding	Tahapanan *CRISP-DM* untuk menentukan tujuan bisnis dan menilai situasi dimana organisasi dapat memahami dan memiliki tujuan apa yang ingin dicapai dari perspektif bisnis.

D

Dataset	Kumpulan data terstruktur dari data yang terorganisir dan tersimpan bersama untuk keperluan analisis atau pemrosesan.
Data Preparation	Tahapan pada metodologi *CRISP-DM* yang berguna untuk untuk memutuskan data yang akan di analisa..
Data Mining	Proses yang menggunakan satu atau lebih teknik pembelajaran komputer

	(*machine learning*) untuk menganalisis dan Mengekstrak pengetahuan secara otomatis.
Data Science	Bidang yang menggabungkan beberapa disiplin ilmu matematika, statistic dan ilmu komputer..
Data Understanding	Tahapan pada metodologi *CRISP-DM* yang berguna untuk untuk memperoleh data yang tercantum dalam sumber daya proyek.
Deployment	Tahapan pada metodologi *CRISP-DM* yang berguna untuk mengambil hasil evaluasi dan menentukan strategi penerapannya.
Domain Project	Cara untuk mengatur sumber daya bersama yang digunakan oleh beberapa aplikasi.
	E
Evaluation	Tahapan pada metodologi *CRISP-DM* yang berguna untuk menilai sejauh mana Model tersebut memenuhi tujuan bisnis.
	M
Machine Learning	Bagian dari ilmu komputer yang bertujuan untuk mengenali pola dan belajar dari data untuk menghasilkan prediksi Yang benar.
Modeling	Tahapan pada metodologi *CRISP-DM* yang berguna untuk memilih secara spesipik model yang digunakan.

INDEX

DAFTAR PUSTAKA

D. F. Azam, D. E. Ratnawati, And P. P. Adikara, "Prediksi Harga Emas Batang Menggunakan Feed Forward Neural Network Dengan Algoritme Genetika," *J. Pengemb. Teknol. Inf. Dan Ilmu Komput. Univ. Brawijaya*, Vol. 2, No. 8, Pp. 2317–2322, 2018.

S. K. Chandar, M. Sumathi, And S. N. Sivanadam, "Forecasting Gold Prices Based On Extreme Learning Machine," *Int. J. Comput. Commun. Control*, Vol. 11, No. 3, Pp. 372–380, 2016.

A. N. Sihananto And F. A. Bachtiar, "Gold Price Movement Forecasting Using Hybrid Es-Fis," *Proc. - 2017 Int. Conf. Sustain. Inf. Eng. Technol. Siet 2017*, Vol. 2018-Janua, Pp. 321–326, 2018.

Y. Sari, "Prediksi Harga Emas Menggunakan Metode Neural Network Backropagation Algoritma Conjugate Gradient," *J. Eltikom*, Vol. 1, No. 2, Pp. 64–70, 2018.

H. Mombeini And A. Yazdani-Chamzini, "Modeling Gold Price Via Artificial Neural Network," *J. Econ. Bus. Manag.*, Vol. 3, No. 7, Pp. 699–703, 2015.

Z. Alameer, M. A. Elaziz, A. A. Ewees, H. Ye, And Z. Jianhua, "Forecasting Gold Price Fluctuations Using Improved Multilayer Perceptron Neural Network And Whale Optimization Algorithm," *Resour. Policy*, Vol. 61, No. September 2018, Pp. 250–260, 2019.

B. R. Auer, "On The Performance Of Simple Trading Rules Derived From The Fractal Dynamics Of Gold And Silver Price Fluctuations,"

Financ. Res. Lett., Vol. 16, Pp. 255–267, 2016.

H. M. Nawawi, J. J. Purnama, And A. B. Hikmah, "Komparasi Algoritma Neural Network Dan Naïve Bayes Untuk Memprediksi Penyakit Jantung," *J. Pilar Nusa Mandiri*, Vol. 15, No. 2, Pp. 189–194, 2019.

S. B. Koduri, L. Gunisetti, C. R. Ramesh, K. Mutyalu, And D. Ganesh, "Prediction Of Crop Production Using Adaboost Regression Method Prediction Of Crop Production Using Adaboost Regression Method," *J. Phys. Conf. Ser.*, 2019.

Firdaus And F. Zamzam, *Aplikasi Metodologi Penelitian*, 1st Ed. Yogyakarta: Penerbit Deepublish, 2018.

E. Tandelilin, *Portofolio Dan Investasi Teori Dan Aplikasi*, 1st Ed. Yogyakarta: Penerbit Kanisius, 2010.

S. Widoatmodjo, *Cara Sehat Investasi Pasar Modal Pengantar Menjadi Investor Profesional*, 6th Ed. Yogyakarta: Pt. Elex Media Komputindo (Kelompok Gramedia), 2008.

E. Syafputri, *Investasi Emas, Dinar, Dirham*. Depok: Penebar Plus, 2012.

A. Nugroho, *Solusi Keuangan Pribadi Sehari-Hari Cinta, Uang, Kehidupan*, 1st Ed. Jakarta: Pt Elex Media Komputindo, 2015.

A. Byna, *Monograf Analisis Komparatif Machine Learning Untuk Klasifikasi Kejadian Stunting*, 1st Ed. Banyumas: Cv. Pena Persada, 2020.

J. Sanjaya, E. Renata, V. E. Budiman, F. Anderson, And M. Ayub, "Prediksi Kelalaian Pinjaman Bank Menggunakan Random Forest Dan Adaptive Boosting," *J. Tek. Inform. Dan Sist. Inf.*, Vol. 6, No. April, Pp. 50–60, 2020.

S. Adinugroho And Y. A. Sari, *Implementasi Data Mining Menggunakan Weka*, 1st Ed. Malang: Ub Press, 2018.

H. Hermanto, S. J. Kuryanti, And S. N. Khasanah, "Comparison Of Naïve Bayes Algorithm , C4 . 5 And Random Forest For Service Classification Ojek Online," *J. Publ. Informatics Eng. Res.*, Vol. 3, No. 2, 2019.

Indrayanti, D. Sugianti, And M. A. Al Karomi, "Optimasi Parameter K Pada Algoritma K-Nearest Neighbour Untuk Klasifikasi Penyakit Diabetes Mellitus," *Pros. Snatif*, Pp. 823–829, 2017.

B. Santoso, A. I. S. Azis, And Zoharahayati, *Machine Learning & Reasoning Fuzzy Logic Algoritma, Manual, Matlab & Rapid Miner*, 1st Ed. Sleman: Penerbit Deepublish, 2020.

N. D. G. Vadivu, "Big Data Analytics For Gold Price Forecasting Based On Decision Tree Algorithm And Support Vector Regression (Svr)," *Int. J. Sci. Res.*, Vol. 4, No. 3, Pp. 2026–2030, 2015.

A. R. Muslikh, H. A. Santoso, And A. Marjuni, "Klasifikasi Data Time Series Arus Lalu Lintas Jangka Pendek Menggunakan Algoritma Adaboost Dengan Random Forest," *Briliant J. Ris. Dan Konseptual*, Vol. 4, No. 1, Pp. 78–96, 2019.

V. Rodriguez-Galiano, M. Sanchez-Castillo, M. Chica-Olmo, And M. Chica-Rivas, "Machine Learning Predictive Models For Mineral Prospectivity: An Evaluation Of Neural Networks, Random Forest, Regression Trees And Support Vector Machines," *Ore Geol. Rev.*, Vol. 71, Pp. 804–818, 2015.

V. W. Siburian, "Prediksi Harga Ponsel Menggunakan Metode Random Forest," *Annu. Res. Semin. 2018 Fak. Ilmu Komput. Unsri*, Vol. 4, No. 1, Pp. 978–979, 2018.

S. Sulistiani And T. Diani, "Prakiraan Flare Sinar-X Matahari Berdasarkan

Evolusi Daerah Aktif (Prediction Of Solar X-Ray Flares Based On Active Regions Evolution),” *J. Sains Dirgant.*, Vol. Vol. 16 No, Pp. 23–32, 2018.

P. Probst, M. Wright, And A. Boulesteix, “Hyperparameters And Tuning Strategies For Random Forest,” *Stat.Ml*, No. 2010, Pp. 1–19, 2019.

L. Yang, L. Yang, Z. Jianyong, And L. Rongxiu, “Prediction Of Component Content In Rare Earth Extraction Process Based On Esns-Adaboost,” *Ifac-Papersonline*, Vol. 51, No. 21, Pp. 42–47, 2018.

J. Cai, K. Xu, Y. Zhu, F. Hu, And L. Li, “Prediction And Analysis Of Net Ecosystem Carbon Exchange Based On Gradient Boosting Regression And Random Forest,” *Appl. Energy*, Vol. 262, No. November 2019, P. 114566, 2020.

S. Nakagawa, P. C. D. Johnson, And H. Schielzeth, “The Coefficient Of Determination R 2 And Intra-Class Correlation Coefficient From Generalized Linear Mixed-Effects Models Revisited And Expanded,” *R. Soc. Publ.*, Pp. 1–11, 2017.

M. Kayri, I. Kayri, And M. T. Gencoglu, “The Performance Comparison Of Multiple Linear Regression , Random Forest And Artificial Neural Network By Using Photovoltaic And Atmospheric Data,” *2017 14th Int. Conf. Eng. Mod. Electr. Syst.*, Pp. 1–4, 2017.

W. Putri And E. I. M. Susi, “Prediksi Parameter Hasil Pengisian Adonan Tahu Tuna Menggunakan Pendekatan Machine Learning,” *Semin. Nas. Tah. Xvi*, Pp. 319–324, 2019.

E. Rahmawati, S. Hadianti, M. F. Akbar, And W. Gata, “Penerapan Algoritma Cuntuk Memprediksi Performa Vendor Online,” *Semin. Nas. Teknol. Inf. Univ. Ibn Khaldun Bogor 2018*, Pp. 224–231, 2018.

A. M. Talpur, *Congestion Detection In Software Defined Networks Using Machine Learning*. Bremen: University Of Bremenr, 2017.

I. H. Witten And E. Frank, *Practical Machine Learning Tools And Techniques*, Second. San Francisco: Diane Cerra, 2005.

D. Parbat And M. Chakraborty, "A Python Based Support Vector Regression Model For Prediction Of Covid19 Cases In India," *Elsevier*, Vol. 138, Pp. 3–7, 2020.

S. Rasckha And V. Mirjalili, *Phyton Machine Learnig*, Second. Birmingham: Packt Publishing, Ltd, 2017.

A. M. Yusuf, *Metode Penelitian Kuantitatif, Kualitatif & Penelitian Gabungan*. Jakarta: Prenada Media, 2016.

A. Juliandi, Irfan., And S. Manurung, *Metodologi Penelitian Bisnis, Konsep Dan Aplikasi: Sukses Menulis Skripsi & Tesis Mandiri*. 2014.

E. Susana, "Pengukuran Tekanan Darah Non-Invasive Tanpa Manset Menggunakan Metode Pulse Transit Time Berbasis Machine Learning Multivariat Regresi," *J. Kesehat.*, Vol. 10, No. April, Pp. 1–6, 2019.

Yose Alloisius. (2020). Implentasi CRISP-DM untuk dataset Bank Marketing. Diambil dari https://Github.Com/Yoseas/Implentasi-Crisp-Dm-Untuk-Dataset-Bank-Marketing/Tree/Master

Mauro Benetti. (2022). CRISP-DM-Rossmann. Diambil dari https://github.com/mbenetti/CRISP-DM-Rossmann/blob/master/CRISP-DMRossmann.ipynb

TENTANG PENULIS

Agung Baitul Hikmah, S. Kom, M. Kom. Lahir di Tasikmalaya, 19 Agustus 1983. Telah menyelesaikan Program Studi Magister Ilmu Komputer (S2) di STMIK Nusa Mandiri (2013). Sejak tahun 2016 telah berprofesi menjadi dosen tetap di Universitas Bina Sarana Informatika Program Studi Sistem Informasi Kampus Kota Tasikmalaya dan saat ini aktif mengajar, penulis buku ajar dan mengisi berbagai workshop terkait teknologi informasi.

Nani Purwati, S. Kom, M. Kom. Lahir di Kebumen, 01 Maret 1988. sebagai dosen pada Program Studi Sistem Informasi Kampus Yogyakarta. Dosen dengan lulusan S-1 dan S-2 IImu Komputer STMIK Nusa Mandiri.

Sri Kiswati, S.T., M.M. Saat ini sebagai dosen pada Program Studi Sistem Informasi Kampus Yogyakarta. Menyelesaikan Program Studi Magister Manajemen di Universitas Diponegoro Semarang dan lulus tahun 2010. Penulis juga aktif melakukan penelitian yang diterbitkan pada jurnal nasional maupun internasional yang telah terindex Scopus.

Pudji Widodo, M. Kom. Lahir di Temanggung, 22 Juli 1973, pendidikan S1 di Program Pascasarjana STMIK NUSA MANDIRI JAKARTA tahun 2009, S2 di Program Pascasarjana STMIK NUSA MANDIRI Jakarta 2012. Sejak tahun 1995 menjadi Dosen Universitas Bina Sarana Informatika Program Studi Sistem Informasi Kampus Yogyakarta.

Hendri Mahmud Nawawi, M. Kom. Lahir di Garut, 17 April 1994, Sebagai seorang dosen di kampus Univeristas Nusa Mandiri. Menyelesaikan S2 di Program Pascasarjana Program Studi Ilmu Komputer STMIK NUSA MANDIRI Jakarta tahun 2020.

Vincent Christian. Lahir di Tasikmalaya, 3 Agustus 2005. Sedang menyelesaikan studinya di Universitas Bina Sarana Informatika Kampus Kota Tasikmalaya mengambil Program Studi Sistem Informasi.

-oo0oo-

ALGORITMA DATA SCIENCE

Agung Baitul Hikmah, dkk

Buku ajar ini berisi tentang pengolahan algoritma *data science*, tujuan yang diharapkan penulis agar pembaca mampu memahami dan melakukan pengembangan model berbasis data dengan mengikuti suatu metodologi algoritma d*ata science*, objektif bisnis, teknis dan rencana projek data science, teknik mengumpulkan data, menganalisis data, menentukan objek atau memilah data, membersihkan data, mengkonstruksi data, membangun model dan dapat melakukan *deployment* model. Selain itu di dalam buku ini diberikan contoh studi kasus dengan menggunakan metodologi *Cross Industry Standard Process For Data Mining* (CRISP-DM) menggunakan algoritma *Support Vector Machine* (SVM), ANN (*Artificial Neural Networks*), *Gradient Boosting Regressor, Decision Tree* dan *Random Forest* dengan bahasa pemrograman python.

ISBN: 978-623-8075-77-5

TEKNOSAIN

ALGORITMA DATA SCIENCE