Modul Praktikum 4

Penambangan Data

PROGRAM STUDI DATA SAINS

FAKULTAS SAINS

INSTITUT TEKNOLOGI SUMATERA

Tahun 2024

Pengurangan Data

Dalam penambangan data, teknik yang disebut reduksi data digunakan untuk meminimalkan ukuran kumpulan data sambil mempertahankan informasi yang paling penting. Ini mungkin berguna dalam kasus-kasus di mana kumpulan data terlalu besar untuk ditangani dengan baik atau mengandung sejumlah besar data yang berlebihan atau informasi yang tidak diperlukan. Ada beberapa teknik reduksi data yang dapat digunakan dalam data pertambangan:

Pengambilan Sampel Data: Teknik ini melibatkan pemilihan sebagian data untuk digunakan, daripada menggunakan seluruh dataset. Ini dapat berguna untuk mengurangi ukuran dataset sambil tetap mempertahankan tren dan pola keseluruhan dalam data.

Pengurangan Dimensi: Teknik ini melibatkan pengurangan jumlah fitur dalam dataset, baik dengan menghapus fitur yang tidak relevan atau dengan menggabungkan beberapa fitur menjadi satu fitur.

Kompresi Data: Teknik ini melibatkan penggunaan teknik seperti lossy atau lossless kompresi untuk mengurangi ukuran kumpulan data.

Diskritisasi Data: Teknik ini melibatkan konversi data kontinyu menjadi data diskrit dengan membagi rentang nilai yang mungkin ke dalam interval atau wadah.

Pemilihan Fitur: Teknik ini melibatkan pemilihan subset fitur dari kumpulan data yang paling relevan dengan tugas yang sedang dihadapi.

Reduksi data merupakan langkah penting dalam penambangan data, karena dapat membantu meningkatkan efisiensi dan kinerja algoritma pembelajaran mesin dengan mengurangi ukuran kumpulan data. Namun, penting untuk menyadari adanya trade-off antara ukuran dan akurasi data, dan menilai risiko serta manfaatnya dengan cermat sebelum menerapkannya.

Keuntungan Reduksi Data dalam Penambangan Data

- Peningkatan efisiensi: Pengurangan data dapat membantu meningkatkan efisiensi mesin algoritma pembelajaran dengan mengurangi ukuran kumpulan data. Hal ini dapat membuatnya lebih cepat dan lebih praktis untuk bekerja dengan kumpulan data besar.
- Peningkatan kinerja: Pengurangan data dapat membantu meningkatkan kinerja mesin algoritma pembelajaran dengan menghilangkan informasi yang tidak relevan atau berlebihan dari kumpulan data. Ini dapat membantu membuat model lebih akurat dan kuat.
- Mengurangi biaya penyimpanan: Pengurangan data dapat membantu mengurangi biaya penyimpanan yang terkait dengan kumpulan data besar dengan mengurangi ukuran data.

• Peningkatan interpretabilitas: Pengurangan data dapat membantu meningkatkan interpretabilitas hasil dengan menghilangkan informasi yang tidak relevan atau berlebihan dari kumpulan data.

Transformasi Data

Dalam konteks penambangan data, transformasi data adalah tindakan mengubah data yang belum diproses data ke dalam format yang sesuai untuk analisis dan pemodelan. Transformasi data bertujuan untuk mempersiapkan data untuk penambangan data guna mengekstrak pengetahuan dan wawasan yang berharga. Transformasi data adalah fase krusial dalam proses penambangan data karena menjamin bahwa data bebas dari kesalahan -dan bebas inkonsistensi dan dalam format yang dapat digunakan untuk analisis dan pemodelan. Dengan menurunkan dimensionalitas data dan menskalakannya ke rentang nilai umum, transformasi data dapat juga membantu dalam meningkatkan kinerja algoritma penambangan data. Beberapa teknik untuk Transformasi data meliputi:

Perataan Data: Dengan menggunakan teknik tertentu, proses “perataan data” dilakukan untuk menghilangkan noise dari dataset. Perataan data memungkinkan untuk menarik perhatian aspek penting dalam dataset. Data dapat diubah selama pengumpulan untuk menghapus atau mengurangi variasi atau jenis gangguan lainnya. Ide di balik penghalusan data adalah bahwa ia dapat mengenali perubahan kecil untuk membantu prediksi berbagai pola dan tren.

Konstruksi Atribut: Menggunakan set atribut yang disediakan, atribut baru dibangun dan diterapkan untuk membantu proses penambangan. Ini menyederhanakan data awal dan meningkatkan penambangan efektivitas.

Agregasi Data: Proses penyimpanan dan tampilan data dalam format ringkasan disebut agregasi data. Untuk memasukkan data dari banyak sumber data ke dalam deskripsi data analisis, data mungkin berasal dari sejumlah sumber yang berbeda. Ini adalah fase penting karena volume dan kaliber data yang digunakan memiliki dampak signifikan terhadap keakuratan data temuan analisis adalah. Untuk mendapatkan temuan yang bermakna, diperlukan jumlah data berkualitas tinggi dan data yang tepat harus dikumpulkan.

Generalisasi Data: Generalisasi data mengacu pada proses transformasi data tingkat rendah atribut menjadi atribut tingkat tinggi dengan menggunakan konsep hierarki. Generalisasi data diterapkan ke data kategorikal yang memiliki jumlah nilai berbeda yang terbatas tetapi besar.

Normalisasi Data: Normalisasi data melibatkan konversi semua variabel data ke dalam nilai tertentu. kisaran biasanya antara [0,1]. Teknik yang digunakan untuk normalisasi adalah: **Min-Max Normalisasi, Normalisasi Z-Score, dan Normalisasi dengan Skala Desimal.**

Diskritisasi Data: Diskritisasi data mengacu pada proses transformasi data berkelanjutan ke dalam satu set interval data. Ini adalah metode yang sangat membantu yang dapat membantu Anda mempelajari dan memahami data dengan lebih mudah dan meningkatkan efektivitas algoritma yang diterapkan.

Keuntungan Transformasi Data dalam Penambangan Data

- Meningkatkan Kualitas Data: Transformasi data membantu meningkatkan kualitas data dengan menghapus kesalahan, ketidakkonsistenan, dan nilai yang hilang.
- Memfasilitasi Integrasi Data: Transformasi data memungkinkan integrasi data dari berbagai sumber, yang dapat meningkatkan keakuratan dan kelengkapan data.
- Meningkatkan Analisis Data: Transformasi data membantu mempersiapkan data untuk analisis dan pemodelan dengan menormalkan, mengurangi dimensionalitas, dan mendiskritisasi data.
- Meningkatkan Keamanan Data: Transformasi data dapat digunakan untuk menutupi data sensitif, atau untuk menghapus informasi sensitif dari data, yang dapat membantu meningkatkan keamanan data.
- Meningkatkan Kinerja Algoritma Penambangan Data: Transformasi data dapat meningkatkan kinerja algoritma penambangan data dengan mengurangi dimensionalitas data dan menskalakan data ke rentang nilai umum.

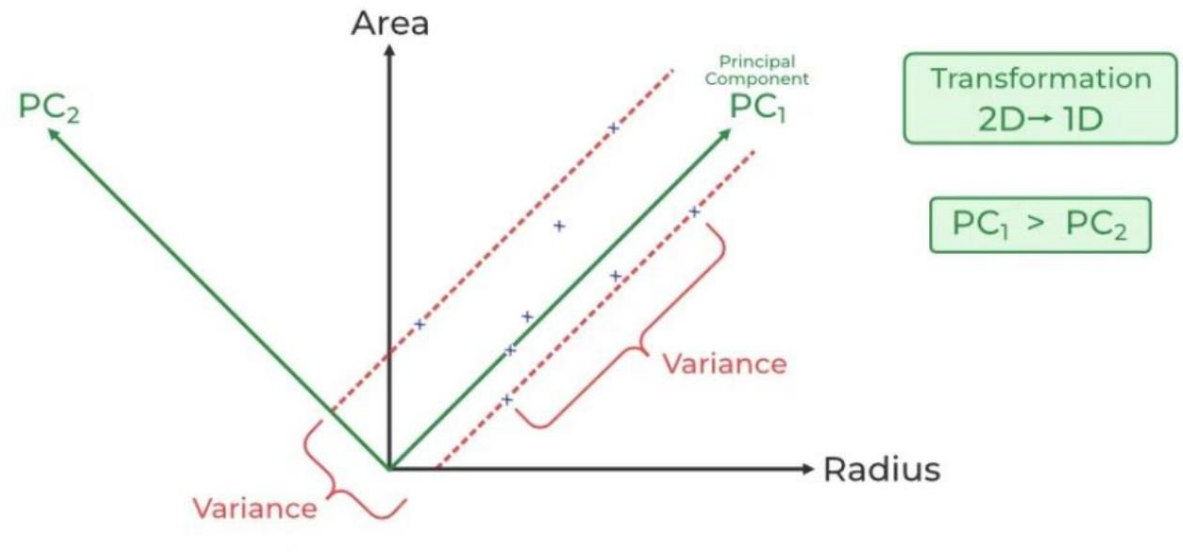
Tujuan Kerja Praktek

1. Mempelajari konsep analisis komponen utama
2. Memahami konsep analisis komponen utama dalam Python
3. Mempelajari konsep normalisasi data
4. Memahami konsep normalisasi data dalam Python

Analisis Komponen Utama

Analisis Komponen Utama (PCA) adalah prosedur statistik yang menggunakan pendekatan ortogonal transformasi yang mengubah sekumpulan variabel berkorelasi menjadi sekumpulan variabel tidak berkorelasi. PCA bekerja pada kondisi bahwa sementara data dalam ruang dimensi yang lebih tinggi dipetakan ke data di ruang dimensi yang lebih rendah, varians data di ruang dimensi yang lebih rendah harus maksimal. Tujuan utama dari Principal Component Analysis (PCA) adalah untuk mengurangi dimensionalitas suatu dataset sambil mempertahankan pola atau hubungan yang paling penting antara variabel tanpa pengetahuan sebelumnya tentang variabel target. Dengan kata lain, PCA digunakan untuk mengurangi dimensionalitas suatu kumpulan data dengan menemukan kumpulan variabel baru yang lebih kecil

daripada kumpulan variabel asli, mempertahankan sebagian besar informasi sampel, dan berguna untuk regresi dan klasifikasi data.



Gambar 1. Analisis Komponen Utama

Dalam kerja praktek ini, kita akan belajar tentang PCA menggunakan pustaka Sklearn di Python. Sebelum menuju ke pekerjaan praktis, Anda harus mengimpor beberapa perpustakaan dan menghasilkan kumpulan data tentang data iris menggunakan distribusi normal. Data iris mengandung 4 fitur, seperti pada sumber ini kode.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.decomposition import PCA

#Set random seed untuk reproduksibilitas
np.random.seed(0)

#Buat dataset dengan fitur yang terstruktur menggunakan distribusi normal
data = {
    'Panjang Petal': np.random.normal(loc=3.5, scale=0.5, size=50),
    'Lebar Petal': np.random.normal(loc=1.0, scale=0.3, size=50),
    'Panjang Sepal': np.random.normal(loc=5.0, scale=0.5, size=50),
    'Lebar Sepal': np.random.normal(loc=1.5, scale=0.3, size=50)
}
df = pd.DataFrame(data)
print(df)
```

Lalu, visualisasikan himpunan data tersebut.

```
#Visualisasikan dataset
plt.figure(figsize=(12, 6))

#Scatter plot untuk data asli (Panjang Petal vs Lebar Petal)
plt.subplot(1, 2, 1)
plt.scatter(df['Panjang Petal'], df['Lebar Petal'], color='blue', marker='o')
plt.title('Data Awal (Panjang Petal vs Lebar Petal)')
plt.xlabel('Panjang Petal')
plt.ylabel('Lebar Petal')
plt.grid()
plt.tight_layout()
plt.show()
```

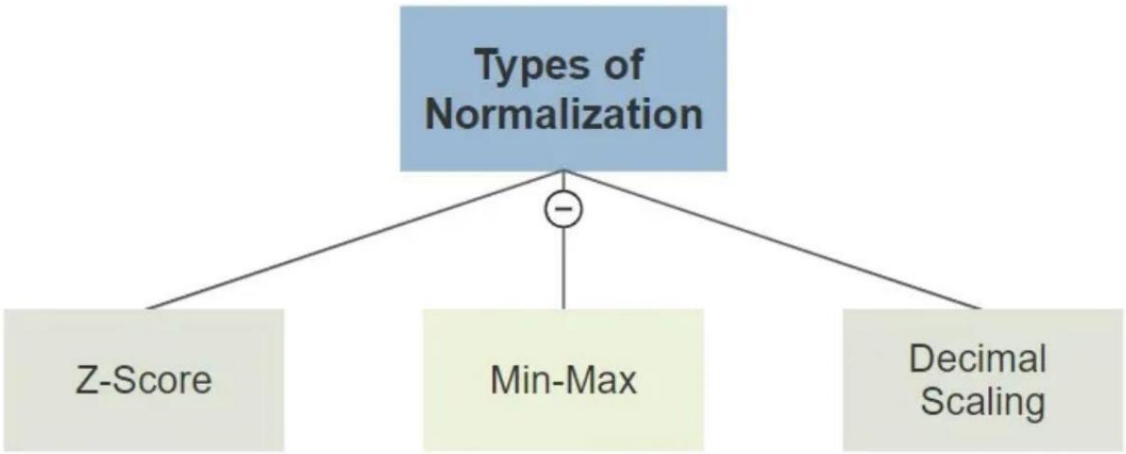
Kemudian, kita dapat melakukan dekomposisi fitur menggunakan Analisis Komponen Utama dan memvisualisasikannya

PCA dengan menggunakan kode sumber ini.

```
#Lakukan PCA menggunakan scikit-learn
pca = PCA(n_components=2)
X_pca = pca.fit_transform(df)

#Buat DataFrame untuk hasil PCA
df_pca = pd.DataFrame(data=X_pca, columns=['Komponen 1', 'Komponen 2'])
print(df_pca)

#Visualisasikan hasil PCA
plt.figure(figsize=(8, 6))
plt.scatter(df_pca['Komponen 1'], df_pca['Komponen 2'], color = 'green')
plt.title('Hasil PCA')
plt.xlabel('Komponen 1')
plt.ylabel('Komponen 2')
plt.grid()

plt.tight_layout()
plt.show()
```

Normalisasi Data

Pada praktikum kali ini kita akan mempelajari tentang Normalisasi Data dengan menggunakan Min-Max Normalisasi, Normalisasi Z-skor, dan Normalisasi dengan Skala Desimal.



Gambar 2. Jenis Normalisasi

Normalisasi Min-Maks

$$' = \frac{-}{-}$$

Misalkan bahwa: adalah minimum dan adalah nilai maksimum dari suatu atribut. adalah nilai Anda ingin memplot dalam rentang baru. adalah nilai baru yang Anda dapatkan setelah menormalkan nilai lama. Normalisasi dapat dilakukan dengan menggunakan rumus di atas. Source code ini dapat digunakan untuk mencari normalisasi dataset dalam Python.

```python
import pandas as pd

#Contoh DataFrame
data = {
    'A': [4, 7, 8, 11, 25],
    'B': [15, 29, 80, 26, 57]
}
df = pd.DataFrame(data)
print(df)

#Normalisasi min-max
df_normalized = (df - df.min()) / (df.max() - df.min())
print("Data Normalized (Min-Max):\n", df_normalized)
```

Normalisasi Min-Maks juga dapat dilakukan menggunakan MinMaxScaler di Python.

```
from sklearn.preprocessing import MinMaxScaler

#Inisialisasi MinMaxScaler
scaler = MinMaxScaler()

#Normalisasi menggunakan Scikit-learn
df_normalized1 = pd.DataFrame(scaler.fit_transform(df), columns=df.columns)
print("Data Normalized (Min-Max) dengan Scikit-learn:\n", df_normalized1)
```

Normalisasi Skor Z

$$' = \frac{- ()}{()}$$

Dalam Normalisasi Z-score (atau normalisasi rata-rata nol) nilai atribut adalah dinormalisasi berdasarkan rata-rata dan deviasi standarnya. Nilai atribut adalah dinormalisasi menjadi '. Kode sumber untuk Normalisasi Z-score dalam Python menggunakan rumus di atas dapat dilihat di bawah ini.

```
#Menghitung rata-rata (mean) dan deviasi standar (standard deviation)
mean = df.mean()
std_dev = df.std()

#Menghitung Z-score menggunakan rumus
df_zscore1 = (df - mean) / std_dev

print("Data Normalized (Z-Score):\n", df_zscore1)
```

Normalisasi Z-score juga dapat dilakukan menggunakan StandardScaler di Python.

```
#Inisialisasi StandardScaler
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()

#Melakukan standarisasi
df_standardized = pd.DataFrame(scaler.fit_transform(df), columns=df.columns)
print("Data Normalized (Z-Score) dengan Scikit-learn:\n", df_standardized)
```

Normalisasi dengan Skala Desimal

Ini menormalkan nilai atribut dengan mengubah posisi titik desimalnya.

jumlah titik yang dipindahkan titik desimalnya dapat ditentukan oleh absolut nilai maksimum atribut. Nilai atribut dinormalisasi ke ' dengan komputasi

$$' = \frac{}{10}$$

di mana adalah bilangan bulat terkecil sehingga Nilai x adalah 0,05.

```python
import pandas as pd

#Contoh DataFrame
data = {
    'A': [123, 456, 789, 1011, 1213],
    'B': [50, 200, 300, 400, 500]
}
df1 = pd.DataFrame(data)

#Menentukan nilai maksimum
max_value = df1.max().max()

#Menentukan j (jumlah digit dari nilai maksimum)
j = len(str(max_value))

#Melakukan normalisasi dengan decimal scaling
df1_normalized = df1 / (10 ** j)
print("Data Normalized (Decimal Scaling):\n", df1_normalized)
```

Persiapan untuk Tugas Individu

Untuk tugas individu, Anda perlu menyiapkan beberapa langkah ini.

• Instal Yahoo Finance ke Python.

```
pip install yfinance
```

• Lalu unduh data yang ingin Anda gunakan, ini hanya sebagai contoh.

```
import yfinance as yf
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

#Tentukan nama Saham
ggrm = 'GGRM.JK'
bbni = 'BBNI.JK'
bbca = 'BBCA.JK'
bbri = 'BBRI.JK'
isat = 'ISAT.JK'

#tentukan nama data dan hari pertama dan hari terakhir nya
a = yf.download(ggrm, start="2023-09-24", end="2024-09-24")
b = yf.download(bbni, start="2023-09-24", end="2024-09-24")
c = yf.download(bbca, start="2023-09-24", end="2024-09-24")
d = yf.download(bbri, start="2023-09-24", end="2024-09-24")
e = yf.download(isat, start="2023-09-24", end="2024-09-24")
```

• Lakukan beberapa persiapan untuk menggabungkan dan mendapatkan dataset gabungan.

```
ggrm_close = a[['Close']]
bbni_close = b[['Close']]
bbca_close = c[['Close']]
bbri_close = d[['Close']]
isat_close = e[['Close']]
ggrm_close.rename(columns={'Close':'GGRM'}, inplace=True)
bbni_close.rename(columns={'Close':'BBNI'}, inplace=True)
bbca_close.rename(columns={'Close':'BBCA'}, inplace=True)
bbri_close.rename(columns={'Close':'BBRI'}, inplace=True)
isat_close.rename(columns={'Close':'ISAT'}, inplace=True)
```

```
data_saham = pd.concat([ggrm_close, bbni_close, bbca_close, bbri_close, isat_close], axis = 1)
print(data_saham)
```

• Kemudian, visualisasikan himpunan data tersebut menggunakan grafik.

```
#Membuat grafik harga penutupan
plt.figure(figsize=(12, 6))
plt.plot(data_saham['GGRM'], label='Saham GGRM', color='blue')
plt.plot(data_saham['BBNI'], label='Saham BBNI', color='red')
plt.plot(data_saham['BBCA'], label='Saham BBCA', color='orange')
plt.plot(data_saham['BBRI'], label='Saham BBRI', color='green')
plt.plot(data_saham['ISAT'], label='Saham ISAT', color='purple')

#Menambahkan judul dan label
plt.title(f'Grafik Harga Saham {data_saham}')
plt.xlabel('Tanggal')
plt.ylabel('Harga Saham')
plt.legend()
plt.grid()

#Menampilkan grafik
plt.show()
```

Untuk memudahkan pengkodean dalam pembuatan grafik, Anda dapat menyalin dan menempelkan kode sumber ini.

```
#Membuat grafik harga penutupan
plt.figure(figsize=(12, 6))
plt.plot(data_saham['GGRM'], label='Saham GGRM', color='blue')
plt.plot(data_saham['BBNI' ], label='Saham BBNI', color='merah')
plt.plot(data_saham['BBCA'], label='Saham BBCA', color='oranye')
plt.plot(data_saham['BBRI'], label='Saham BBRI', color='hijau')
plt.plot(data_saham['ISAT'], label='Saham ISAT', color='ungu')

#Menambahkan judul dan label
plt.title(f'Grafik Harga Saham {data_saham}')
plt.xlabel('Tanggal')
plt.ylabel('Harga Saham')
plt.legend()
plt.grid()

#Menampilkan grafik
plt.show()
```

• Membuat histogram untuk dataset.

```
#Membuat beberapa histogram untuk harga penutupan dan volume
fig, axs = plt.subplots(5, 1, figsize=(24, 50))

#Histogram untuk Harga Saham GGRM
axs[0].hist(data_saham['GGRM'], bins=30, color='blue', edgecolor='black')
axs[0].set_title('Histogram Harga Saham GGRM', fontsize=26)
axs[0].set_xlabel('Harga', fontsize=24)
axs[0].set_ylabel('Frekuensi', fontsize=24)
axs[0].grid(axis='y', alpha=0.75)

#Histogram untuk Harga Saham BBNI
axs[1].hist(data_saham['BBNI'], bins=30, color='red', edgecolor='black')
axs[1].set_title('Histogram Harga Saham BBNI', fontsize=26)
axs[1].set_xlabel('Harga', fontsize=24)
axs[1].set_ylabel('Frekuensi', fontsize=24)
axs[1].grid(axis='y', alpha=0.75)

#Histogram untuk Harga Saham BBCA
axs[2].hist(data_saham['BBCA'], bins=30, color='orange', edgecolor='black')
axs[2].set_title('Histogram Harga Saham BBCA', fontsize=26)
axs[2].set_xlabel('Harga', fontsize=24)
axs[2].set_ylabel('Frekuensi', fontsize=24)
axs[2].grid(axis='y', alpha=0.75)

#Histogram untuk Harga Saham BBRI
axs[3].hist(data_saham['BBRI'], bins=30, color='green', edgecolor='black')
axs[3].set_title('Histogram Harga Saham BBRI', fontsize=26)
axs[3].set_xlabel('Harga', fontsize=24)
axs[3].set_ylabel('Frekuensi', fontsize=24)
axs[3].grid(axis='y', alpha=0.75)

#Histogram untuk Harga Saham ISAT
axs[4].hist(data_saham['ISAT'], bins=30, color='purple', edgecolor='black')
axs[4].set_title('Histogram Harga Saham ISAT', fontsize=26)
axs[4].set_xlabel('Harga', fontsize=24)
axs[4].set_ylabel('Frekuensi', fontsize=24)
axs[4].grid(axis='y', alpha=0.75)

#Menampilkan grafik
plt.tight_layout()  #Menyusun layout agar tidak saling bertumpuk
plt.show()
```

Untuk memudahkan pengkodean dalam pembuatan grafik, Anda dapat menyalin dan menempelkan kode sumber ini.

```
#Membuat beberapa histogram untuk harga penutupan dan volume fig, axs =
plt.subplots(5, 1, figsize=(24, 50))

#Histogram untuk Harga Saham GGRM
axs[0].hist(data_saham['GGRM'], bins=30, color='blue', edgecolor='black')
axs[0].set_title('Histogram
Harga Saham GGRM', ukuran font=26) axs[0].set_xlabel('Harga', ukuran
font=24) axs[0].set_ylabel('Frekuensi', ukuran
font=24) axs[0].grid(axis='y', alpha=0.75 )


#Histogram untuk Harga Saham BBNI
axs[1].hist(data_saham['BBNI'], bins=30, color='red', edgecolor='black')
axs[1].set_title('Histogram
Harga Saham BBNI', ukuran font=26) axs[1].set_xlabel('Harga', ukuran
font=24) axs[1].set_ylabel('Frekuensi', ukuran
font=24) axs[1].grid(axis='y', alpha=0.75 )


#Histogram untuk Harga Saham BBCA
axs[2].hist(data_saham['BBCA'], bins=30, color='orange', edgecolor='black')
axs[2].set_title('Histogram
Harga Saham BBCA', ukuran font=26) axs[2].set_xlabel('Harga', ukuran
font=24) axs[2].set_ylabel('Frekuensi', ukuran
font=24) axs[2].grid(axis='y', alpha=0.75 )


#Histogram Harga Saham BBRI
axs[3].hist(data_saham['BBRI'], bins=30, color='green', edgecolor='black')
axs[3].set_title('Histogram
Harga Saham BBRI', ukuran font=26) axs[3].set_xlabel('Harga', ukuran
font=24) axs[3].set_ylabel('Frekuensi', ukuran
font=24) axs[3].grid(axis='y', alpha=0.75 )


#Histogram untuk Harga Saham ISAT
axs[4].hist(data_saham['ISAT'], bins=30, color='purple', edgecolor='black')
axs[4].set_title('Histogram
Harga Saham ISAT', ukuran font=26) axs[4].set_xlabel('Harga', ukuran
font=24) axs[4].set_ylabel('Frekuensi', ukuran
font=24) axs[4].grid(axis='y', alpha=0.75 )


#Menampilkan grafik
plt.tight_layout() #Menyusun layout agar tidak saling bertumpuk
plt.show()
```