

DATA SCIENCE PROJECT DOCUMENTATION

***Analisis Data Historis Bitcoin Menggunakan Metode Data Science untuk
Mengidentifikasi Pola Harga dan Hubungan Antar Variabel***



Kelompok 02 :

11423006 Samuel Alex Bonaparte Sirait

11423011 Ruth Heppi Evelin Sinambela

11423028 Grace Advryanti Tampubolon

Sarjana Terapan Teknologi Rekayasa Perangkat Lunak

Institut Teknologi Del

2025

DAFTAR ISI

BAB I.....	3
PENDAHULUAN.....	3
1.1 Latar Belakang.....	3
1.2 Tujuan.....	3
1.3 Rumusan Masalah.....	4
BAB II.....	5
PEMBAHASAN.....	5
2.1 Metode Pengerjaan.....	5
2.1.1 Data Collection.....	5
2.1.1.1 Implementasi di Google Colab (Tahapan data collection).....	8
2.1.1.1.1 Visualisasi awal sebelum data di bersihkan :.....	9
2.1.2 Data Processing and Techniques.....	11
2.1.2.1 Data Cleaning.....	11
2.1.2.2 Data Transformation.....	15
2.1.2.3 Data Integration.....	17
2.1.3.4 Feature Reduction.....	17
2.1.4 Data Visualization.....	20
2.1.5 Statistical Analysis.....	25
BAB III.....	29
KESIMPULAN.....	29
DAFTAR PUSTAKA.....	31

BAB I

PENDAHULUAN

1.1 Latar Belakang

Pertumbuhan aset digital seperti Bitcoin mengalami peningkatan pesat seiring dengan perkembangan teknologi blockchain dan meningkatnya minat masyarakat terhadap investasi kripto. Bitcoin, sebagai mata uang kripto pertama dan paling populer di dunia, memiliki peran penting dalam membentuk ekosistem keuangan digital modern. Nilainya yang fluktuatif dan dipengaruhi oleh berbagai faktor seperti volume transaksi, tingkat kesulitan penambangan, serta kapitalisasi pasar menjadikannya objek menarik untuk dianalisis secara mendalam [1].

Namun, di balik popularitasnya, pergerakan harga Bitcoin yang tidak stabil menjadi tantangan bagi investor dan peneliti dalam memahami pola perubahan nilai dan faktor-faktor yang mempengaruhinya. Oleh karena itu, diperlukan analisis data yang komprehensif untuk mengidentifikasi tren pasar, aktivitas jaringan, dan hubungan antar variabel yang mempengaruhi performa Bitcoin dari waktu ke waktu [2].

Dalam proyek ini, dilakukan analisis data Bitcoin dengan tahapan utama seperti Data Collection, Data Visualization, Data Processing (*Advanced Preprocessing*), dan Statistical Analysis[3]. Melalui visualisasi interaktif dan penerapan teknik preprocessing lanjutan, diharapkan diperoleh insight yang bermanfaat bagi pengambilan keputusan strategis, terutama dalam memahami perilaku pasar kripto dan faktor-faktor yang berdampak pada nilai Bitcoin secara global [4].

1.2 Tujuan

Proyek ini bertujuan untuk melakukan analisis dan visualisasi data Bitcoin guna memahami tren pasar, aktivitas jaringan, serta faktor-faktor yang mempengaruhi pergerakan harga Bitcoin. Secara khusus, tujuan dari proyek ini adalah untuk:

1. Mengumpulkan dan memahami struktur data historis Bitcoin yang mencakup variabel seperti harga pasar, kapitalisasi, volume transaksi, dan tingkat kesulitan penambangan.
2. Melakukan proses data cleaning dan preprocessing lanjutan untuk memastikan konsistensi dan kualitas data sebelum dilakukan analisis.
3. Membuat visualisasi interaktif dalam bentuk grafik dan diagram guna menampilkan pola perubahan harga, volume transaksi, serta indikator jaringan dari waktu ke waktu.

4. Melakukan analisis statistik (parametrik dan non-parametrik) untuk mengevaluasi hubungan antara variabel-variabel utama seperti harga, hash rate, dan volume transaksi.
5. Menyajikan interpretasi hasil dan insight analitik yang dapat digunakan untuk memahami perilaku pasar Bitcoin serta mendukung pengambilan keputusan dalam investasi dan riset aset kripto.
6. Menyajikan interpretasi hasil dan insight bisnis yang dapat digunakan untuk meningkatkan strategi operasional dan kepuasan pelanggan.

1.3 Rumusan Masalah

Berdasarkan latar belakang dan tujuan yang telah dijelaskan, maka rumusan masalah dalam proyek ini adalah sebagai berikut:

1. Apakah terdapat outlier atau nilai ekstrim pada data harga Bitcoin yang dapat mempengaruhi hasil analisis statistik?
2. Bagaimana visualisasi data interaktif dapat membantu mengidentifikasi tren dan pola pergerakan harga Bitcoin dari waktu ke waktu?
3. Bagaimana penerapan teknik preprocessing lanjutan seperti normalisasi, deteksi outlier, dan seleksi fitur dapat meningkatkan kualitas data sebelum dilakukan analisis statistik?
4. Bagaimana hasil uji statistik parametrik dan non-parametrik dalam menggambarkan korelasi antar variabel utama dalam dataset Bitcoin?

BAB II

PEMBAHASAN

2.1 Metode Pengerjaan

Proyek analisis data ini dikerjakan menggunakan Google Colab sebagai lingkungan pemrograman utama. Google Colab dipilih karena merupakan platform *cloud-based* yang disediakan oleh Google dan mendukung eksekusi kode Python secara langsung di peramban (*browser*). Dengan menggunakan Colab, seluruh proses analisis dapat dilakukan tanpa perlu melakukan instalasi perangkat lunak tambahan di komputer lokal.

2.1.1 Data Collection

Dataset yang digunakan dalam proyek ini berasal dari situs Kaggle melalui tautan berikut: <https://www.kaggle.com/code/aditeloo/bitcoin-price-prediction>, dengan judul “Bitcoin Price Prediction” yang dibuat oleh pengguna @aditeloo. Dataset ini bersifat terbuka (*open source*) dan dapat diakses secara bebas, sehingga siapa pun dapat mengunduh dan menggunakannya untuk keperluan analisis data. Tidak dilakukan proses scraping dari situs manapun karena dataset telah tersedia dalam format CSV di platform Kaggle. Dengan demikian, proses pengambilan data cukup dilakukan dengan mengunduh file secara langsung dari sumber resminya tanpa perlu menggunakan skrip tambahan. Format CSV ini juga memudahkan tahap pra-pemrosesan karena data sudah terstruktur dan siap untuk dianalisis.

Dataset ini merepresentasikan catatan harga dan aktivitas jaringan Bitcoin dalam rentang waktu tertentu. Setiap baris menggambarkan kondisi pasar, volume transaksi, serta parameter teknis jaringan Bitcoin pada tanggal tertentu.

Daftar Fitur dalam Dataset adalah sebagai berikut :

Tabel 1. Daftar Fitur Dataset

NO	Fitur yang ada	Penjelasan
1.	date	Tanggal pencatatan data Bitcoin.
2.	btc_market_price	Harga pasar rata-rata Bitcoin pada tanggal tersebut (dalam USD)

NO	Fitur yang ada	Penjelasan
3.	btc_total_bitcoins	Jumlah total Bitcoin yang telah ditambang (beredar) hingga tanggal tersebut.
4.	btc_market_cap	Kapitalisasi pasar Bitcoin, dihitung dari harga pasar \times jumlah total Bitcoin yang beredar.
5.	btc_trade_volume	Total volume perdagangan Bitcoin harian di pasar (dalam BTC).
6.	btc_blocks_size	Total ukuran semua blok yang ditambang dalam satu hari (dalam megabyte).
7.	btc_avg_block_size	Rata-rata ukuran tiap blok yang ditambang per hari.
8.	btc_n_orphaned_blocks	Jumlah blok yatim (orphaned blocks) yang tidak masuk ke blockchain utama.
9.	btc_n_transactions_per_block	Rata-rata jumlah transaksi yang tercatat di setiap blok pada hari tersebut.
10.	btc_median_confirmation_time	Waktu median konfirmasi transaksi di jaringan Bitcoin (dalam menit).
11.	btc_hash_rate	Kekuatan komputasi jaringan Bitcoin (hash rate) dalam satuan TH/s.
12.	btc_difficulty	Tingkat kesulitan untuk menambang blok baru dalam jaringan Bitcoin.
13.	btc_miners_revenue	Total pendapatan penambang (termasuk reward dan biaya transaksi) per hari (USD).
14.	btc_transaction_fees	Total biaya transaksi yang dibayarkan oleh pengguna ke penambang per hari (USD).
15.	btc_cost_per_transaction_percent	Persentase biaya transaksi terhadap total volume transaksi.

NO	Fitur yang ada	Penjelasan
16.	btc_cost_per_transaction	Biaya rata-rata per transaksi Bitcoin (USD per transaksi).
17.	btc_n_unique_addresses	Jumlah alamat Bitcoin unik yang aktif pada hari tersebut (menggambarkan aktivitas pengguna).
18.	btc_n_transactions	Jumlah total transaksi Bitcoin yang terjadi pada hari tersebut.
19.	btc_n_transactions_total	Jumlah kumulatif semua transaksi Bitcoin yang pernah terjadi sampai tanggal tersebut.
20.	btc_n_transactions_excluding_popular	Jumlah transaksi harian tanpa memasukkan transaksi dari alamat populer (misalnya bursa besar).
21.	btc_n_transactions_excluding_chains_longer_than_100	Jumlah transaksi yang tidak termasuk dalam rantai transaksi lebih dari 100 langkah (untuk menyaring aktivitas tidak normal).
22.	btc_output_volume	Total nilai Bitcoin yang ditransfer melalui transaksi pada hari itu (dalam BTC).
23.	btc_estimated_transaction_volume	Estimasi total volume transaksi ekonomi nyata
24.	btc_estimated_transaction_volume_usd	Estimasi volume transaksi ekonomi nyata yang dikonversi ke dalam USD.

Alasan pemilihan dataset Bitcoin Price Prediction ini adalah :

1. Relevansi tinggi dan aktual : Bitcoin merupakan mata uang kripto paling populer dan sering menjadi indikator utama tren aset digital global.
2. Kelengkapan fitur dan volume data besar : Dataset mencakup lebih dari 20 variabel penting dengan ribuan observasi, sesuai kriteria penilaian (≥ 20 fitur, ≥ 2000 baris).
3. Potensi analisis lanjutan : Data numerik dan time-series memungkinkan penerapan visualisasi tren, analisis korelasi, serta uji statistik parametrik dan non-parametrik.

4. Nilai bisnis dan akademik : Hasil analisis dapat memberikan wawasan bagi investor, analisis keuangan, maupun peneliti ekonomi digital.

2.1.1.1 Implementasi di Google Colab (Tahapan data collection)

Langkah awal pengerjaan menggunakan Google Colab :

- Sumber Data: Kami menggunakan dataset publik harga Bitcoin yang diambil dari situs Kaggle (<https://www.kaggle.com/code/aditeloo/bitcoin-price-prediction>).
- Metode Pengumpulan: Dataset diunduh dalam format .csv, kemudian diunggah ke Google Colab menggunakan:

```
[ ] from google.colab import files
    uploaded = files.upload()
```

Choose Files No file chosen Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.
Saving BitCoin.csv to BitCoin (1).csv

- Data kemudian dibaca menggunakan pandas:

```
[ ] import pandas as pd
    import seaborn as sns
    import numpy as np
    import matplotlib.pyplot as plt
    from sklearn.preprocessing import StandardScaler
    from sklearn.decomposition import PCA
```

- Tahap memuat dataset :

```
[ ] print("Jumlah Baris:", df.shape[0])
    print("Jumlah Kolom:", df.shape[1])
    df.head()
```

Jumlah Baris: 2986
Jumlah Kolom: 24

	Date	btc_market_price	btc_total_bitcoins	btc_market_cap	btc_trade_volume	btc_blocks_size	btc_avg_block_size	btc_n_orphaned_blocks	bt
0	2/17/2010	0.0	2043200.0	0.0	0.0	0.0	0.000235	0	
1	2/18/2010	0.0	2054650.0	0.0	0.0	0.0	0.000241	0	
2	2/19/2010	0.0	2063600.0	0.0	0.0	0.0	0.000228	0	
3	2/20/2010	0.0	2074700.0	0.0	0.0	0.0	0.000218	0	
4	2/21/2010	0.0	2085400.0	0.0	0.0	0.0	0.000234	0	

5 rows x 24 columns

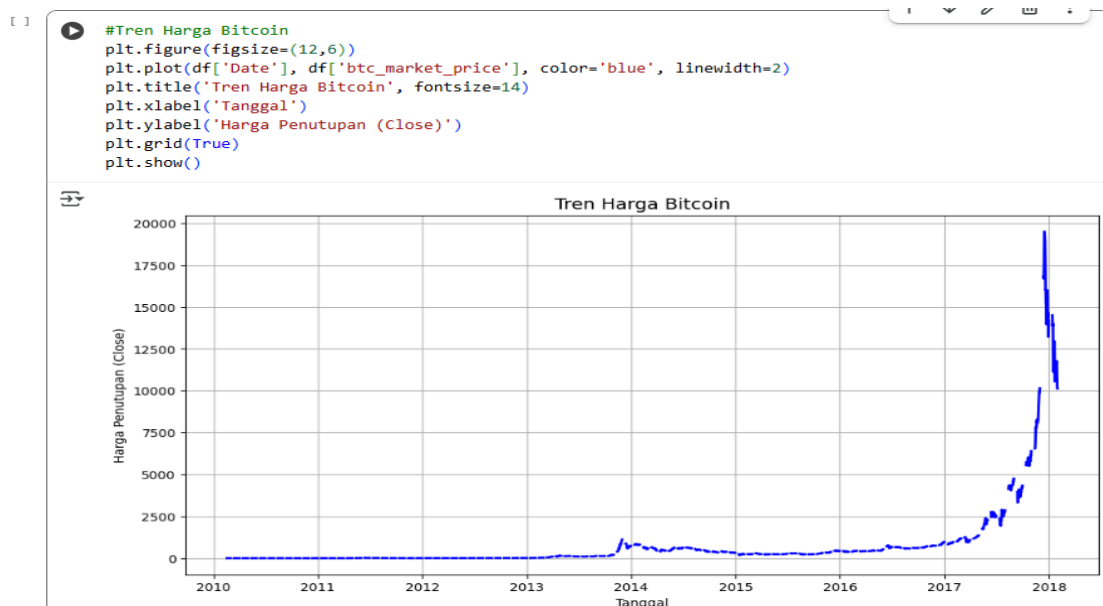
Kode tersebut berfungsi untuk menampilkan jumlah baris atau *records* yang terdapat dalam dataset. Perintah `df.shape` digunakan untuk mengetahui ukuran atau dimensi dari sebuah DataFrame, yang dikembalikan dalam bentuk tuple (baris, kolom). Nilai `df.shape[0]` berarti mengambil elemen pertama dari tuple tersebut, yaitu jumlah baris dalam dataset.

2.1.1.1.1 Visualisasi awal sebelum data di bersihkan :

Pada tahap awal, dilakukan visualisasi data sebelum proses pembersihan (data cleaning) untuk memahami kondisi asli dataset.

1. Visualisasi Awal Tren harga Bitcoin dari waktu ke waktu

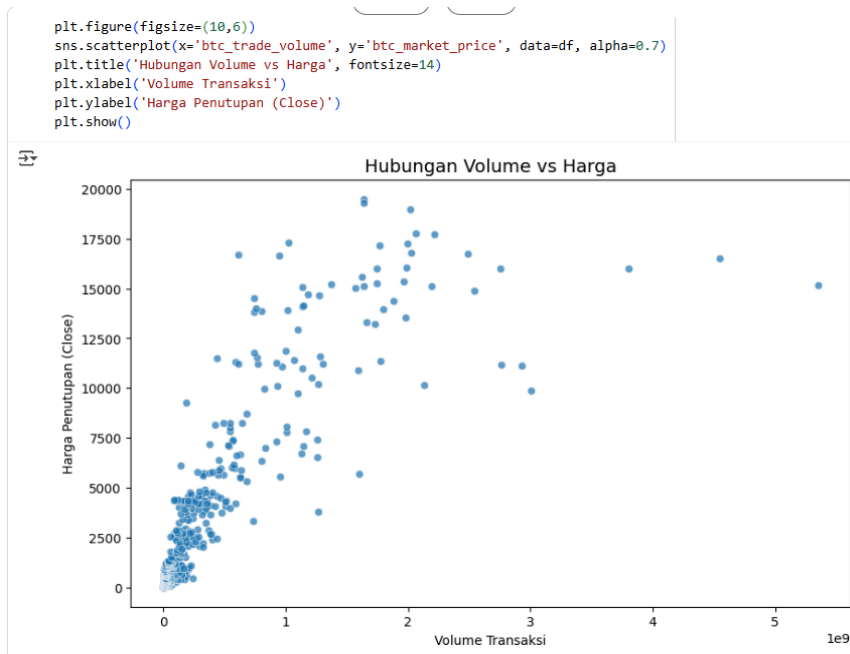
Kode ini digunakan untuk menampilkan tren harga Bitcoin dari waktu ke waktu. Grafik yang dibuat adalah line plot, di mana sumbu-X menunjukkan tanggal (Date) dan sumbu-Y menunjukkan harga pasar Bitcoin (btc_market_price).



Line plot ideal untuk menganalisis pola temporal, misalnya kenaikan, penurunan, atau fluktuasi harga Bitcoin setiap hari, sehingga pembaca bisa dengan cepat memahami tren pasar.

2. Visualisasi Awal Hubungan antara volume transaksi Bitcoin dengan harga pasar

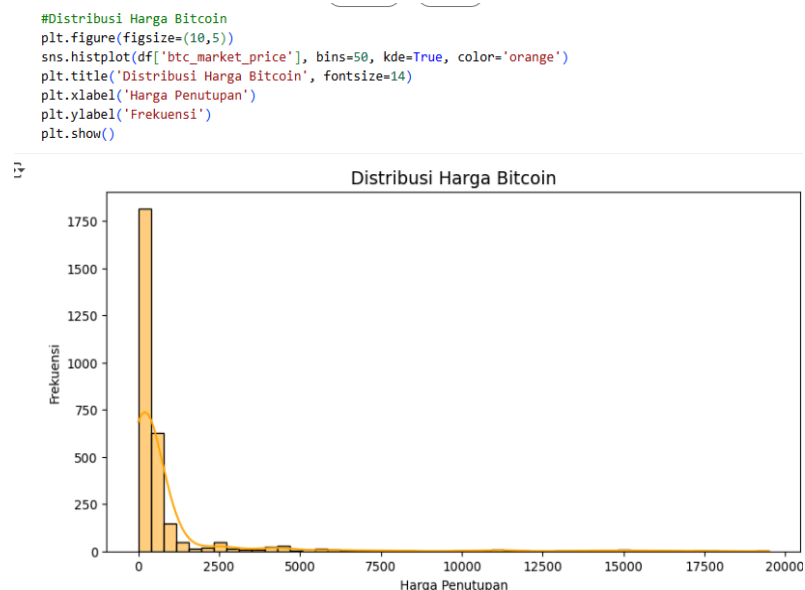
Kode ini digunakan untuk menampilkan hubungan antara volume transaksi Bitcoin dengan harga pasar. Visualisasi yang dibuat adalah scatter plot, di mana sumbu-X menunjukkan btc_trade_volume dan sumbu-Y menunjukkan btc_market_price.



Scatter plot dipilih karena efektif untuk melihat korelasi atau pola hubungan antar dua variabel numerik.

3. Visualisasi Awal Distribusi Harga Pasar Bitcoin

Histogram menampilkan frekuensi kemunculan harga Bitcoin dalam interval tertentu (bin), sehingga kita bisa melihat bagaimana data tersebar. `bins=50` membagi data menjadi 50 interval untuk analisis distribusi yang lebih rinci. `kde=True` menambahkan kurva estimasi kepadatan, membantu melihat pola distribusi data secara lebih halus. Warna oranye digunakan agar visualisasi menarik dan mudah dibaca.



2.1.2 Data Processing and Techniques

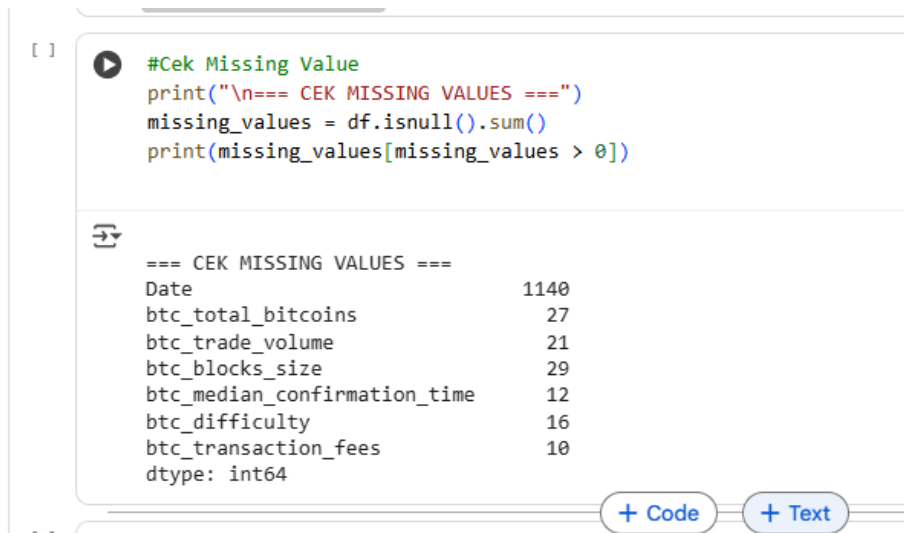
Tahap ini dilakukan untuk memastikan bahwa data yang digunakan dalam analisis memiliki kualitas yang baik dan dapat diolah dengan hasil yang akurat. Proses data preprocessing bertujuan untuk membersihkan, menyesuaikan, dan mempersiapkan data agar siap digunakan pada tahap analisis selanjutnya dari data Bitcoin.

2.1.2.1 Data Cleaning

Pada tahap ini dilakukan identifikasi dan penanganan terhadap berbagai masalah yang sering muncul dalam data mentah, seperti nilai yang hilang (missing values), data duplikat, format data yang tidak sesuai, serta nilai ekstrem (*outlier*). Proses pembersihan data sangat penting karena data yang kotor dapat menyebabkan hasil analisis menjadi bias atau tidak akurat. Dengan melakukan data cleaning, dataset akan memiliki kualitas yang lebih baik dan dapat memberikan hasil analisis yang lebih dapat dipercaya.

Tahapan yang umumnya dilakukan dalam proses ini meliputi:

1. Mengecek Missing Values



```
[ ] #Cek Missing Value
print("\n=== CEK MISSING VALUES ===")
missing_values = df.isnull().sum()
print(missing_values[missing_values > 0])
```

```
=== CEK MISSING VALUES ===
Date                1140
btc_total_bitcoins    27
btc_trade_volume      21
btc_blocks_size       29
btc_median_confirmation_time  12
btc_difficulty        16
btc_transaction_fees   10
dtype: int64
```

[+ Code](#) [+ Text](#)

Kode ini digunakan untuk memeriksa nilai yang hilang (missing values) pada dataset. Baris `df.isnull().sum()` menghitung jumlah nilai kosong di setiap kolom, sedangkan `missing_values[missing_values > 0]` menampilkan hanya kolom-kolom yang memiliki data hilang. Dengan ini, kita bisa cepat mengetahui kolom mana saja yang perlu dibersihkan sebelum analisis.

2. Menangani Missing Values

```
[ ] #Handling Missing Value
print("=== HANDLING MISSING VALUES ===")

# Select only numeric columns for imputation
df_num = df.select_dtypes(include=np.number)

df_filled = df_num.fillna(df_num.mean())

print("\nMissing values after imputation:\n", df_filled.isnull().sum())

=== HANDLING MISSING VALUES ===

Missing values after imputation:
btc_market_price      0
btc_total_bitcoins    0
btc_market_cap        0
btc_trade_volume      0
btc_blocks_size       0
btc_avg_block_size    0
btc_n_orphaned_blocks 0
btc_n_transactions_per_block 0
btc_median_confirmation_time 0
btc_hash_rate         0
btc_difficulty        0
btc_miners_revenue    0
btc_transaction_fees   0
btc_cost_per_transaction_percent 0
btc_cost_per_transaction 0
btc_n_unique_addresses 0
btc_n_transactions    0
btc_n_transactions_total 0
btc_n_transactions_excluding_popular 0
btc_n_transactions_excluding_chains_longer_than_100 0
btc_output_volume     0
btc_estimated_transaction_volume 0
btc_estimated_transaction_volume_usd 0
dtype: int64
```

Kode ini menangani nilai yang hilang pada kolom numerik dengan mengganti (imputasi) setiap missing value menggunakan rata-rata (mean) kolom tersebut. Setelah proses ini, `df_filled.isnull().sum()` digunakan untuk memastikan semua nilai kosong telah terisi, sehingga dataset siap untuk analisis lebih lanjut. Alasan utama pemilihan metode ini adalah karena kesederhanaan dan efisiensinya. Ini merupakan cara yang sangat cepat dan umum digunakan untuk memastikan dataset menjadi "lengkap" dan siap diproses oleh algoritma machine learning (yang seringkali tidak dapat menangani nilai kosong) tanpa harus menghapus seluruh baris data, yang berisiko menyebabkan kehilangan informasi berharga. Output yang menunjukkan '0' missing values untuk semua kolom mengkonfirmasi bahwa proses imputasi ini telah berhasil.

3. Menangani Outlier (Nilai Ekstrem)

Dengan menggunakan metode IQR (*Interquartile Range*) untuk mendeteksi dan menghapus *outlier* yang bisa merusak hasil analisis. Metode IQR (*Interquartile Range*) adalah salah satu teknik statistik untuk mendeteksi outlier atau data yang berada jauh dari nilai mayoritas.

```
[ ] ▶ #Cek Outliers

print("\n=== CEK OUTLIERS (IQR METHOD) ===")
numeric_cols = df.select_dtypes(include=['float64', 'int64']).columns
outlier_counts = {}

for col in numeric_cols:
    Q1 = df[col].quantile(0.25)
    Q3 = df[col].quantile(0.75)
    IQR = Q3 - Q1
    outliers = df[(df[col] < (Q1 - 1.5 * IQR)) | (df[col] > (Q3 + 1.5 * IQR))]
    if len(outliers) > 0:
        outlier_counts[col] = len(outliers)

print(pd.Series(outlier_counts).sort_values(ascending=False))
```

```
=== CEK OUTLIERS (IQR METHOD) ===
btc_n_orphaned_blocks                607
btc_cost_per_transaction_percent      434
btc_difficulty                       410
btc_cost_per_transaction              409
btc_hash_rate                        388
btc_transaction_fees                  387
btc_trade_volume                      360
btc_estimated_transaction_volume_usd  323
btc_market_cap                       298
btc_market_price                     274
btc_miners_revenue                   260
btc_output_volume                    114
btc_n_transactions_total              95
btc_median_confirmation_time          91
btc_estimated_transaction_volume      63
btc_blocks_size                      58
btc_n_unique_addresses                21
btc_n_transactions_excluding_chains_longer_than_100  2
btc_n_transactions_excluding_popular   1
btc_n_transactions                    1
dtype: int64
```

Kode ini mendeteksi outlier pada setiap kolom numerik menggunakan metode IQR (Interquartile Range). Untuk tiap kolom, nilai di bawah $Q1 - 1.5 \times IQR$ atau di atas $Q3 + 1.5 \times IQR$ dianggap outlier. Hasilnya ditampilkan dalam bentuk jumlah outlier per kolom.

```
[1] ▶ #Handling Outlier
outlier_counts_after = {}

for col in numeric_cols:
    Q1 = df_filled[col].quantile(0.25)
    Q3 = df_filled[col].quantile(0.75)
    IQR = Q3 - Q1
    outliers = df_filled[(df_filled[col] < (Q1 - 1.5 * IQR)) | (df_filled[col] > (Q3 + 1.5 * IQR))]
    if len(outliers) > 0:
        outlier_counts_after[col] = len(outliers)

if len(outlier_counts_after) == 0:
    print("Tidak ada outlier terdeteks")
else:
    print(pd.Series(outlier_counts_after).sort_values(ascending=False))
```

```
btc_n_orphaned_blocks                607
btc_cost_per_transaction_percent      434
btc_cost_per_transaction              409
btc_difficulty                       397
btc_hash_rate                        388
btc_transaction_fees                  384
btc_trade_volume                      375
btc_estimated_transaction_volume_usd  323
btc_market_cap                       298
btc_market_price                     274
btc_miners_revenue                   260
btc_output_volume                    114
btc_median_confirmation_time          96
btc_n_transactions_total              95
btc_blocks_size                      71
btc_estimated_transaction_volume      63
btc_n_unique_addresses                21
btc_n_transactions_excluding_chains_longer_than_100  2
btc_n_transactions_excluding_popular   1
btc_n_transactions                    1
dtype: int64
```

Kode ini menjalankan proses penting untuk mendeteksi outlier pada data numerik yang sebelumnya telah diisi nilai kosongnya (`df_filled`). Metode yang digunakan adalah Interquartile Range (IQR), yang merupakan teknik statistik yang sangat kokoh (robust). Alasan utama pemilihan metode ini sangat masuk akal: data metrik Bitcoin, seperti harga atau hash rate, cenderung sangat miring (skewed) dan tidak terdistribusi normal. Tidak seperti metode berbasis rata-rata, metode IQR tidak terpengaruh oleh nilai-nilai ekstrem itu sendiri sehingga dapat memberikan batasan yang lebih realistis. Secara spesifik, kode ini menghitung nilai ekstrem di setiap kolom yang berada di luar "pagar" statistik, yaitu di bawah $Q1 - 1.5 \times IQR$ atau di atas $Q3 + 1.5 \times IQR$. Kode ini tidak menghapus outlier, melainkan menghitung jumlahnya di setiap kolom dan melaporkannya sebagai laporan diagnostik yang terurut. Jika setelah memeriksa semua kolom tidak ditemukan satupun outlier, barulah kode akan mencetak pesan "Tidak ada outlier terdeteksi".

4. Menampilkan Hasil Data yang Sudah Dibersihkan

```
df_filled.info()
df_filled.head()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2906 entries, 0 to 2905
Data columns (total 23 columns):
 #   Column                                     Non-Null Count  Dtype
---  -
 0   btc_market_price                         2906 non-null   float64
 1   btc_total_bitcoins                       2906 non-null   float64
 2   btc_market_cap                           2906 non-null   float64
 3   btc_trade_volume                        2906 non-null   float64
 4   btc_blocks_size                         2906 non-null   float64
 5   btc_avg_block_size                      2906 non-null   float64
 6   btc_n_orphaned_blocks                   2906 non-null   int64
 7   btc_n_transactions_per_block             2906 non-null   float64
 8   btc_median_confirmation_time             2906 non-null   float64
 9   btc_hash_rate                           2906 non-null   float64
10   btc_difficulty                          2906 non-null   float64
11   btc_miners_revenue                      2906 non-null   float64
12   btc_transaction_fees                     2906 non-null   float64
13   btc_cost_per_transaction_percent         2906 non-null   float64
14   btc_cost_per_transaction                2906 non-null   float64
15   btc_n_unique_addresses                  2906 non-null   int64
16   btc_n_transactions                      2906 non-null   int64
17   btc_n_transactions_total                 2906 non-null   int64
18   btc_n_transactions_excluding_popular     2906 non-null   int64
19   btc_n_transactions_excluding_chains_longer_than_100 2906 non-null   int64
20   btc_output_volume                       2906 non-null   float64
21   btc_estimated_transaction_volume         2906 non-null   float64
22   btc_estimated_transaction_volume_usd    2906 non-null   float64
dtypes: float64(17), int64(6)
memory usage: 522.3 KB
```

	btc_market_price	btc_total_bitcoins	btc_market_cap	btc_trade_volume	btc_blocks_size	btc_avg_block_size	btc_n_orphaned_blocks	btc_n_transactions_per_block	btc_median_confirmation_time	btc_hash_rate	...
0	0.0	2043200.0	0.0	0.0	0.0	0.000235	0	1.0	0.0	0.000029	...
1	0.0	2054650.0	0.0	0.0	0.0	0.000241	0	1.0	0.0	0.000029	...
2	0.0	2063600.0	0.0	0.0	0.0	0.000228	0	1.0	0.0	0.000023	...
3	0.0	2074700.0	0.0	0.0	0.0	0.000218	0	1.0	0.0	0.000028	...

Kode `df_filled.info()` digunakan untuk menampilkan ringkasan informasi dataset `df_filled`, termasuk jumlah baris, nama kolom, tipe data masing-masing kolom, dan jumlah nilai yang tidak kosong. Sementara `df_filled.head()` menampilkan 5 baris pertama dari dataset, sehingga kita bisa secara cepat melihat contoh data dan memastikan bahwa pengisian nilai kosong atau proses pembersihan data berjalan dengan benar.

2.1.2.2 Data Transformation

Tahap data transformation dilakukan untuk menyesuaikan skala antar fitur agar memiliki rentang nilai yang sebanding. Pada tahap ini digunakan metode feature scaling dengan pendekatan standarisasi. Proses ini bertujuan untuk menstandarkan skala data sehingga setiap variabel memiliki pengaruh yang seimbang dalam proses analisis maupun pemodelan. Dengan adanya transformasi ini, hasil analisis menjadi lebih akurat dan model dapat bekerja dengan lebih optimal.

1. Melakukan Feature Scaling dengan Standardization

- Sebelum melakukan scaling

```
[ ] #Cek Feature Scaling
feature_scaling = df_filled[numeric_cols].describe().T[['min', 'max', 'mean', 'std']]
print(feature_scaling)
```

btc_market_price	0.000000e+00
btc_total_bitcoins	2.043200e+06
btc_market_cap	0.000000e+00
btc_trade_volume	0.000000e+00
btc_blocks_size	0.000000e+00
btc_avg_block_size	2.163350e-04
btc_n_orphaned_blocks	0.000000e+00
btc_n_transactions_per_block	1.000000e+00
btc_median_confirmation_time	0.000000e+00
btc_hash_rate	2.250000e-05
btc_difficulty	2.527738e+00
btc_miners_revenue	0.000000e+00
btc_transaction_fees	0.000000e+00
btc_cost_per_transaction_percent	1.365306e-01
btc_cost_per_transaction	0.000000e+00
btc_n_unique_addresses	1.100000e+02
btc_n_transactions	1.180000e+02
btc_n_transactions_total	4.124000e+04
btc_n_transactions_excluding_popular	1.180000e+02
btc_n_transactions_excluding_chains_longer_than...	1.180000e+02
btc_output_volume	6.150000e+03
btc_estimated_transaction_volume	7.000000e+00
btc_estimated_transaction_volume_usd	0.000000e+00
	max \
btc_market_price	1.949868e+04
btc_total_bitcoins	1.683769e+07
btc_market_cap	3.270000e+11
btc_trade_volume	5.352016e+09
btc_blocks_size	1.544446e+05
btc_avg_block_size	1.110327e+00
btc_n_orphaned_blocks	7.000000e+00
btc_n_transactions_per_block	2.722625e+03
btc_median_confirmation_time	4.773333e+01
btc_hash_rate	2.160975e+07
btc_difficulty	2.600000e+12
btc_miners_revenue	5.319158e+07
btc_transaction_fees	1.495946e+03
btc_cost_per_transaction_percent	8.857143e+04
btc_cost_per_transaction	1.616861e+02

Kode ini menampilkan ringkasan statistik dasar dari semua kolom numerik sebelum dilakukan feature scaling. `describe()` memberikan informasi seperti nilai minimum (min), maksimum (max), rata-rata (mean), dan standar deviasi (std), sehingga kita bisa melihat seberapa besar variasi dan perbedaan skala antar fitur sebelum distandarisasi.

- Setelah melakukan scaling

```
#feature scaling (Standardization).
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
scaled_data = scaler.fit_transform(df_filled[numeric_cols])
df_scaled = pd.DataFrame(scaled_data, columns=numeric_cols)

after_scaling = df_scaled.describe().T[['min', 'max', 'mean', 'std']]
print(after_scaling)
```

	min	max
btc_market_price	-0.364104	8.096752
btc_total_bitcoins	-2.265254	1.274316
btc_market_cap	-0.347722	8.118249
btc_trade_volume	-0.253966	18.118090
btc_blocks_size	-0.818233	2.740908
btc_avg_block_size	-0.991623	2.152207
btc_n_orphaned_blocks	-0.432334	7.880876
btc_n_transactions_per_block	-0.972777	2.974796
btc_median_confirmation_time	-1.511282	8.105762
btc_hash_rate	-0.425521	6.965869
btc_difficulty	-0.432486	6.567587
btc_miners_revenue	-0.385318	8.997629
btc_transaction_fees	-0.534274	12.215656
btc_cost_per_transaction_percent	-0.037813	58.241328
btc_cost_per_transaction	-0.712972	7.161651
btc_n_unique_addresses	-0.927218	4.208543
btc_n_transactions	-0.981556	3.740531
btc_n_transactions_total	-0.825749	2.755258
btc_n_transactions_excluding_popular	-0.906517	3.620083
btc_n_transactions_excluding_chains_longer_than...	-0.904518	3.670692
btc_output_volume	-0.684685	15.497765
btc_estimated_transaction_volume	-0.759195	28.957304
btc_estimated_transaction_volume_usd	-0.349053	9.583236

	mean	std
btc_market_price	-3.912142e-17	1.000172
btc_total_bitcoins	3.129713e-16	1.000172
btc_market_cap	-7.824283e-17	1.000172
btc_trade_volume	1.956071e-17	1.000172
btc_blocks_size	-1.564857e-16	1.000172
btc_avg_block_size	0.000000e+00	1.000172
btc_n_orphaned_blocks	3.912142e-17	1.000172
btc_n_transactions_per_block	-7.824283e-17	1.000172
btc_median_confirmation_time	7.824283e-17	1.000172
btc_hash_rate	-3.912142e-17	1.000172
btc_difficulty	0.000000e+00	1.000172
btc_miners_revenue	0.000000e+00	1.000172
btc_transaction_fees	-7.824283e-17	1.000172
btc_cost_per_transaction_percent	0.000000e+00	1.000172
btc_cost_per_transaction	-9.780354e-17	1.000172
btc_n_unique_addresses	-7.824283e-17	1.000172
btc_n_transactions	-1.564857e-16	1.000172
btc_n_transactions_total	0.000000e+00	1.000172
btc_n_transactions_excluding_popular	7.824283e-17	1.000172
btc_n_transactions_excluding_chains_longer_than...	0.000000e+00	1.000172
btc_output_volume	-5.868213e-17	1.000172
btc_estimated_transaction_volume	5.868213e-17	1.000172
btc_estimated_transaction_volume_usd	-7.824283e-17	1.000172

Sebelum scaling (feature_scaling), setiap kolom numerik memiliki nilai minimum, maksimum, mean, dan standar deviasi yang berbeda-beda, tergantung satuan atau rentang datanya. Misalnya, harga Bitcoin bisa jutaan USD, sementara jumlah transaksi bisa ribuan.

Setelah scaling dengan StandardScaler, semua kolom distandarisasi sehingga mean ≈ 0 dan standar deviasi ≈ 1 , serta rentang nilai menjadi sebanding. Hal ini membuat setiap fitur memiliki pengaruh yang seimbang ketika digunakan untuk analisis atau pemodelan, dan mencegah fitur dengan skala besar mendominasi.

2. Mengubah Kolom Date ke dalam Format Waktu

Kode ini digunakan untuk mengubah kolom 'Date' ke format waktu yang valid dan menghapus baris dengan nilai tanggal tidak sah. Langkah ini berguna agar data waktu dapat diproses dengan benar dalam analisis berbasis tanggal, seperti melihat tren atau pola waktu.


```

#Mengubah kolom Date ke format waktu

df['Date'] = pd.to_datetime(df['Date'], errors='coerce')
df = df.dropna(subset=['Date'])

```

2.1.2.3 Data Integration

Pada proyek ini, tidak dilakukan proses penggabungan data (*data integration*) karena dataset yang digunakan telah tersedia dalam kondisi lengkap dan terintegrasi dari sumber aslinya, yaitu Kaggle. Dataset tersebut sudah disusun secara sistematis oleh penyedia data, di mana seluruh informasi yang diperlukan telah tergabung dalam satu file utama tanpa adanya pemisahan antar tabel atau sumber data lain.

Selain itu, setiap atribut atau kolom di dalam dataset telah saling berhubungan dan merepresentasikan satu entitas yang sama, sehingga tidak diperlukan proses penggabungan dengan dataset eksternal untuk melengkapi informasi. Kondisi ini meminimalkan risiko terjadinya redundansi data, inkonsistensi nilai, maupun ketidaksesuaian struktur yang biasanya muncul saat menggabungkan data dari berbagai sumber. Dengan demikian, tahap data integration dapat dilewati karena dataset dari Kaggle sudah memenuhi standar kelengkapan dan konsistensi yang dibutuhkan untuk analisis.

2.1.3.4 Feature Reduction

1. Mengecek Korelasi Antar Fitur (Kolom)

```

#Cek Feature Rediction

plt.figure(figsize=(15,8))
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm')
plt.title("Heatmap Korelasi Antar Fitur")
plt.show()

threshold = 0.9
high_corr = np.where(corr_matrix > threshold)
high_corr_pairs = [
    (corr_matrix.index[x], corr_matrix.columns[y])
    for x, y in zip(*high_corr)
    if x != y and x < y
]
if len(high_corr_pairs) > 0:
    for pair in high_corr_pairs:
        print(f"{pair[0]} <--> {pair[1]}")
else:
    print("Tidak ada fitur yang terlalu berkorelasi (redundansi rendah).")

```

Kode ini dirancang untuk melakukan langkah Reduksi Fitur (Feature Reduction) dengan cara mengidentifikasi multikolinearitas, yaitu suatu kondisi di mana terdapat korelasi yang sangat tinggi antar fitur. Proses ini dimulai dengan memvisualisasikan matriks korelasi (*corr_matrix*) menggunakan *sns.heatmap* agar analis dapat melihat pola korelasi secara visual. Bagian inti dari

kode ini adalah logika penyaringan: pertama, sebuah ambang batas (threshold) 0.9 ditetapkan. Kode kemudian menggunakan np.where untuk menemukan semua "koordinat" dalam matriks di mana nilai korelasinya melebihi 0.9. Selanjutnya, kode menyaring hasil temuan tersebut untuk membuang korelasi fitur dengan dirinya sendiri ($x \neq y$) dan membuang pasangan duplikat ($x < y$). Pada akhirnya, jika ada pasangan yang lolos filter (yang berarti memiliki korelasi sangat tinggi), kode akan mencetak daftar pasangan tersebut. Jika tidak ada pasangan yang ditemukan, kode akan mencetak pesan bahwa tidak ada fitur yang berkorelasi terlalu tinggi, yang mengindikasikan redundansi data yang rendah.

Visualisasi matriks untuk menunjukkan koefisien korelasi antar setiap pasangan fitur dalam data. Setiap kotak pada grid mewakili hubungan antara dua variabel, di mana angka di dalamnya (koefisien korelasi) dan warnanya menunjukkan kekuatan dan arah hubungan tersebut. Skala warna di sebelah kanan (dari biru tua ke merah tua) adalah kuncinya: warna merah tua (mendekati 1.0) menunjukkan korelasi positif yang sangat kuat, sedangkan warna biru tua (mendekati -1.0) menunjukkan korelasi negatif yang kuat, dan warna pucat/putih (mendekati 0) berarti tidak ada korelasi linier. Insight utama dari heatmap ini adalah adanya multikolinearitas yang sangat tinggi di dalam data, yang ditandai oleh banyaknya kotak berwarna merah tua dengan nilai di atas 0.9. Kita dapat secara visual mengidentifikasi pasangan-pasangan yang sangat redundan, seperti `btc_market_price` dengan `btc_market_cap` (0.99), `btc_hash_rate` dengan `btc_difficulty` (0.99), serta sekelompok besar fitur terkait transaksi (seperti `btc_n_transactions` dan `btc_n_transactions_total`) di bagian kanan bawah yang semuanya saling berkorelasi. Visualisasi ini secara efektif mengkonfirmasi mengapa kode reduksi fitur (feature reduction) dari langkah sebelumnya diperlukan, karena ia membuktikan bahwa banyak fitur membawa informasi yang hampir identik.

2. Mengurangi Fitur yang memiliki korelasi tinggi

```
# Hilangkan satu fitur dari setiap pasangan yang berkorelasi tinggi
features_to_drop = set()
for col1, col2 in high_corr_pairs:
    if col1 not in features_to_drop and col2 not in features_to_drop:
        # Tentukan mana yang akan dihilangkan - untuk lebih sederhana, mari kita hilangkan yang kedua dalam pasangan tersebut
        features_to_drop.add(col2)

df_reduced = df_scaled.drop(columns=list(features_to_drop))

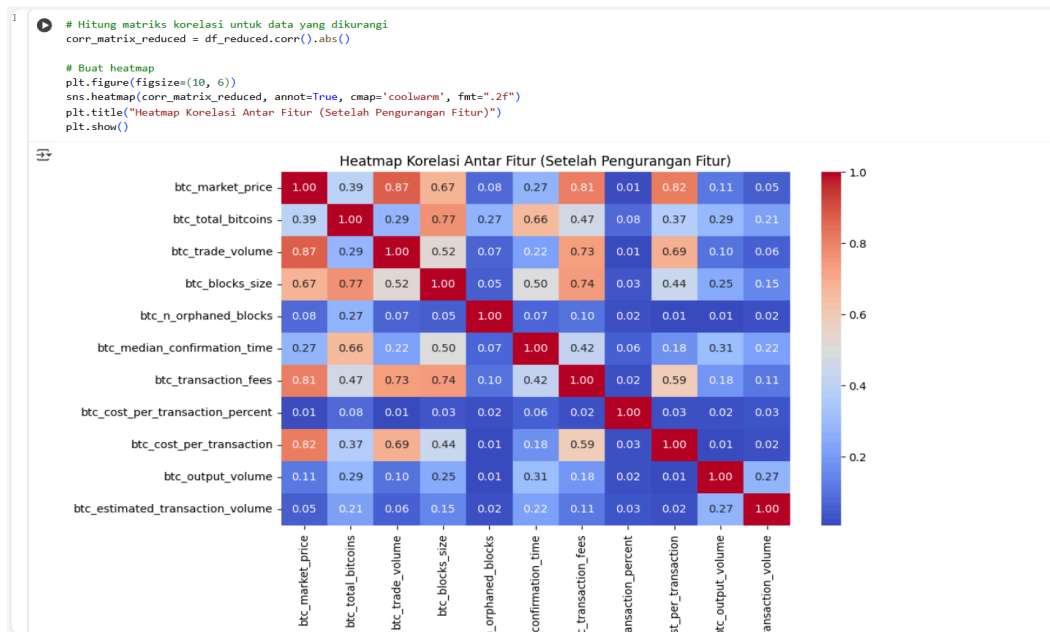
print(f"Original number of features: {df_scaled.shape[1]}")
print(f"Number of features after reduction: {df_reduced.shape[1]}")
display(df_reduced.head())
```

Original number of features: 23
Number of features after reduction: 11

	btc_market_price	btc_total_bitcoins	btc_trade_volume	btc_blocks_size	btc_n_orphaned_blocks	btc_median_confirmation_time	btc_transaction_fees	btc_cost_per_transaction_percent	btc_cost_per_transaction	bt
0	-0.364104	-2.265254	-0.253966	-0.818233	-0.432334	-1.511282	-0.514274	-0.019850	-0.712972	
1	-0.364104	-2.262515	-0.253966	-0.818233	-0.432334	-1.511282	-0.514274	0.049794	-0.712972	
2	-0.364104	-2.260373	-0.253966	-0.818233	-0.432334	-1.511282	-0.514274	0.687883	-0.712972	
3	-0.364104	-2.257718	-0.253966	-0.818233	-0.432334	-1.511282	-0.514274	12.556793	-0.712972	
4	-0.364104	-2.255158	-0.253966	-0.818233	-0.432334	-1.511282	-0.514274	0.353335	-0.712972	

Kode ini merupakan langkah eksekusi dari reduksi fitur yang bertindak berdasarkan daftar pasangan berkorelasi tinggi (`high_corr_pairs`) yang telah diidentifikasi sebelumnya. Metode yang digunakan adalah penghapusan fitur yang redundan secara terprogram. Secara spesifik, kode ini mengiterasi (melakukan looping) pada setiap pasangan fitur yang memiliki korelasi di atas 0.9. Untuk setiap pasangan, ia menerapkan strategi sederhana yaitu menambahkan fitur kedua (`col2`) dari pasangan tersebut ke dalam sebuah set bernama `features_to_drop` (penggunaan set di sini adalah untuk secara otomatis menangani duplikasi jika satu fitur muncul di beberapa pasangan). Setelah loop selesai, kode ini membuat sebuah DataFrame baru bernama `df_reduced` dengan cara membuang (`.drop()`) semua kolom yang namanya telah terkumpul di dalam set tersebut.

Visualisasi dari Korelasi Fitur yang di kurangi



Kode ini bertujuan untuk memverifikasi keberhasilan dari proses reduksi fitur yang dilakukan pada langkah sebelumnya. Pertama, kode tersebut menghitung sebuah matriks korelasi baru (`corr_matrix_reduced`) dengan menggunakan DataFrame yang sudah dikurangi fiturnya (`df_reduced`). Penting untuk dicatat bahwa kode ini juga menggunakan `.abs()` untuk mengambil nilai absolut dari korelasi, yang berarti ia berfokus pada kekuatan hubungan, terlepas dari apakah itu positif atau negatif. Hasil dari matriks baru ini kemudian divisualisasikan sebagai heatmap baru dengan judul "Setelah Pengurangan Fitur". Insight utama dari output heatmap ini adalah konfirmasi visual bahwa proses reduksi fitur telah berhasil. Terlihat jelas bahwa heatmap kini jauh lebih kecil (hanya 11 fitur) dan semua kotak di luar diagonal utama kini memiliki nilai korelasi di bawah ambang batas 0.9. Korelasi tertinggi yang tersisa (seperti 0.87) masih menunjukkan hubungan yang kuat, namun masalah multikolinearitas ekstrem (redundansi data) telah berhasil dihilangkan.

2.1.4 Data Visualization

Bagian ini membahas proses visualisasi data, yang bertujuan untuk menyajikan hasil pengolahan dan analisis data dalam bentuk grafik, diagram, atau tampilan visual lainnya. Visualisasi data mempermudah pemahaman pola, tren, dan hubungan antar variabel, sehingga informasi yang dihasilkan dapat diinterpretasikan secara lebih jelas, efisien, dan informatif.

1. Visualisasi Line Plot – Tren Harga Bitcoin dari Waktu ke Waktu



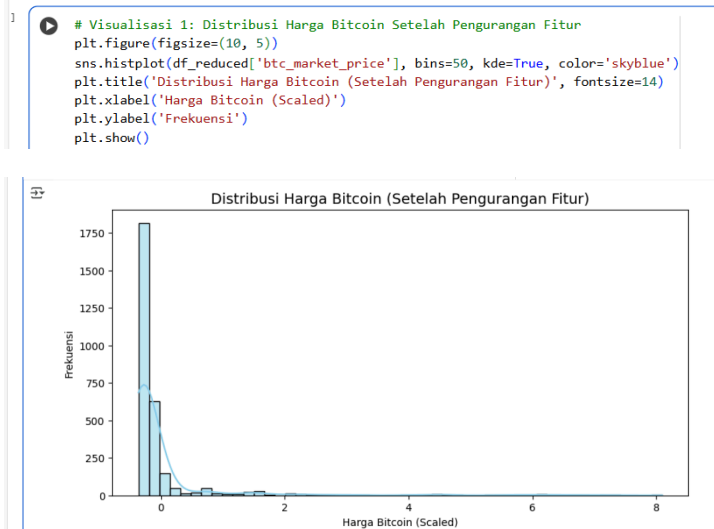
Grafik yang digunakan adalah line plot (grafik garis), yang dibuat menggunakan `sns.lineplot`. Jenis visualisasi ini paling efektif untuk menggambarkan perubahan nilai suatu variabel kontinu (`btc_market_price`) terhadap variabel waktu (`Date`). Dalam konteks ini, line plot dipilih untuk menampilkan pergerakan harga Bitcoin dari waktu ke waktu. Penggunaan garis yang menghubungkan setiap titik data secara berurutan membantu memperlihatkan dengan jelas arah tren, pola kenaikan (seperti lonjakan), dan penurunan harga secara berkesinambungan. Oleh karena itu, line plot sangat sesuai digunakan untuk menganalisis dan memvisualisasikan data deret waktu (time series) seperti riwayat harga Bitcoin.

Insight:

Dari grafik terlihat bahwa harga Bitcoin mengalami fluktuasi yang signifikan dari tahun ke tahun. Pada awal periode yang ditampilkan (sekitar tahun 2010–2016), harga Bitcoin masih relatif stabil dan berada di kisaran yang sangat rendah, seperti yang ditunjukkan oleh garis yang hampir datar di dekat bagian bawah grafik. Namun, insight utamanya adalah perubahan drastis yang dimulai sekitar tahun 2017; pada titik ini, terjadi lonjakan harga (bull run) yang sangat tajam dan eksponensial, mencapai puncaknya yang ekstrem (mendekati \$20.000) di akhir tahun 2017 atau awal 2018. Pola ini dengan jelas menunjukkan bahwa Bitcoin memiliki tingkat volatilitas yang sangat tinggi,

mencerminkan karakteristik umum dari pasar kripto yang rentan terhadap perubahan harga ekstrem.

2. Distribusi Harga Bitcoin Setelah Pengurangan Fitur



Jenis visualisasi yang dipilih dalam kode tersebut adalah histogram, yang dibuat menggunakan fungsi `sns.histplot` dari pustaka Seaborn. Alasan utama pemilihan visualisasi ini adalah karena kemampuannya yang sangat efektif untuk merepresentasikan distribusi frekuensi dari satu variabel numerik tunggal (dalam hal ini, `btc_market_price` yang sudah di-*scaling*). Histogram bekerja dengan membagi rentang data menjadi beberapa interval atau "bins" (diatur menjadi 50 dalam kode), dan kemudian menghitung berapa banyak titik data (Frekuensi) yang jatuh ke dalam setiap interval tersebut. Ini memungkinkan analisis untuk secara cepat memahami bentuk sebaran data. Penambahan parameter `kde=True`, yang memunculkan garis kurva halus (Kernel Density Estimate), semakin membantu memperjelas bentuk distribusi, seperti apakah data tersebut simetris atau miring (*skewed*).

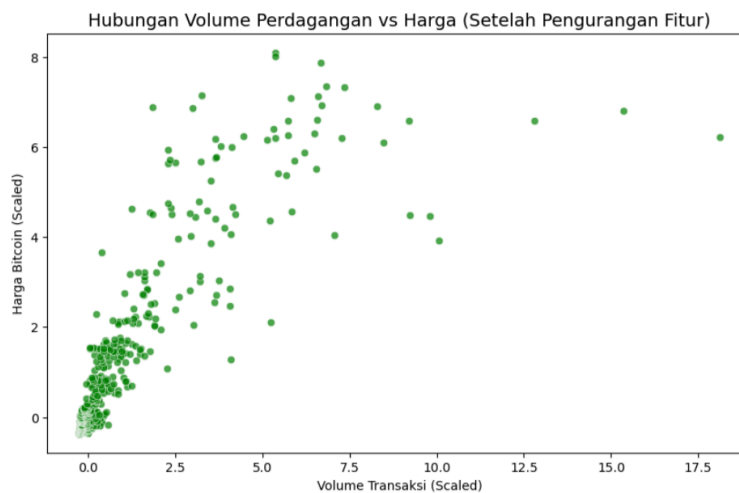
Insight:

Dari grafik histogram, terlihat jelas bahwa distribusi `btc_market_price` (yang telah di-*scaling*) sangat miring ke kanan (*right-skewed*). Pola ini ditunjukkan oleh satu batang (*bar*) frekuensi yang sangat tinggi di sisi kiri, dekat dengan nilai 0, sementara sisa datanya tersebar tipis membentuk "ekor" (*tail*) yang panjang ke arah kanan (menuju nilai-nilai yang lebih positif). Insight yang didapat adalah bahwa meskipun data telah distandardisasi (di-*scaling*), yang mengubah pusat dan skala data, bentuk distribusi aslinya tetap tidak simetris. Ini mengindikasikan bahwa untuk sebagian besar periode waktu, nilai harga Bitcoin terkonsentrasi pada nilai yang lebih rendah

(relatif terhadap meannya), dan terdapat sejumlah kecil kejadian di mana harga secara signifikan lebih tinggi daripada biasanya, yang menciptakan adanya outlier di sisi kanan dan menyebabkan ekor panjang tersebut.

3. Hubungan antara Volume Perdagangan dan Harga Bitcoin Setelah Pengurangan Fitur

```
# Hubungan antara Volume Perdagangan dan Harga Bitcoin Setelah Pengurangan Fitur
plt.figure(figsize=(10, 6))
sns.scatterplot(x='btc_trade_volume', y='btc_market_price', data=df_reduced, alpha=0.7, color='green')
plt.title('Hubungan Volume Perdagangan vs Harga (Setelah Pengurangan Fitur)', fontsize=14)
plt.xlabel('Volume Transaksi (Scaled)')
plt.ylabel('Harga Bitcoin (Scaled)')
plt.show()
```



Jenis visualisasi yang dipilih adalah scatter plot (diagram tebar), yang dibuat menggunakan `sns.scatterplot`. Alasan utama pemilihan visualisasi ini adalah karena kemampuannya yang sangat efektif untuk menunjukkan hubungan (korelasi) antara dua variabel numerik kontinu. Dalam kasus ini, plot tersebut memetakan 'Volume Transaksi (Scaled)' pada sumbu-x terhadap 'Harga Bitcoin (Scaled)' pada sumbu-y. Setiap titik hijau pada grafik mewakili satu observasi data. Visualisasi ini memungkinkan analisis untuk secara cepat mengidentifikasi pola, arah (positif atau negatif), dan kekuatan hubungan linier di antara kedua variabel tersebut, terutama setelah proses reduksi fitur dan scaling dilakukan.

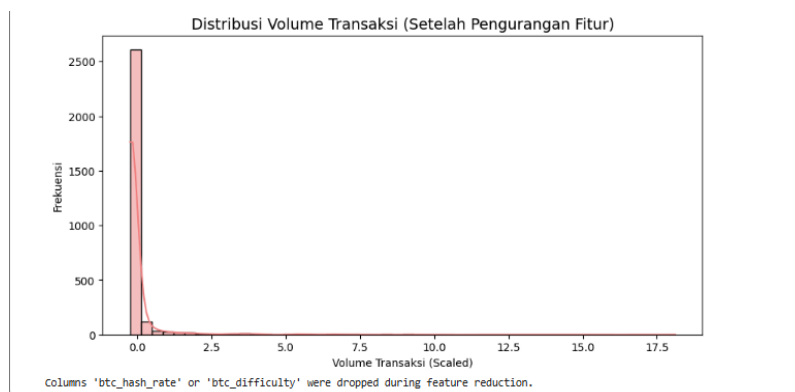
Insight:

Dari grafik scatter plot tersebut, insight utama yang terlihat adalah adanya korelasi positif yang kuat antara Volume Transaksi dan Harga Bitcoin (bahkan setelah data di-scaling dan dikurangi fiturnya). Pola ini ditunjukkan oleh sebaran titik-titik data yang secara jelas bergerak dari kiri bawah ke kanan atas. Ini mengindikasikan bahwa hari-hari dengan volume transaksi yang lebih

tinggi (nilai di sumbu-x) juga cenderung bertepatan dengan hari-hari di mana harga Bitcoin (nilai di sumbu-y) lebih tinggi. Selain itu, terlihat bahwa sebagian besar data terkonsentrasi di area kiri bawah (volume dan harga rendah), yang konsisten dengan distribusi data yang right-skewed (condong ke kanan) yang telah teridentifikasi sebelumnya.

4. Distribusi Volume Transaksi Setelah Pengurangan Fitur

```
# Distribusi Volume Transaksi Setelah Pengurangan Fitur
plt.figure(figsize=(10, 5))
sns.histplot(df_reduced['btc_trade_volume'], bins=50, kde=True, color='lightcoral')
plt.title('Distribusi Volume Transaksi (Setelah Pengurangan Fitur)', fontsize=14)
plt.xlabel('Volume Transaksi (Scaled)')
plt.ylabel('Frekuensi')
plt.show()
```



Jenis visualisasi yang dipilih dalam kode tersebut adalah histogram, yang dibuat menggunakan fungsi `sns.histplot` dari pustaka Seaborn. Alasan utama pemilihan visualisasi ini adalah karena kemampuannya yang sangat efektif untuk merepresentasikan distribusi frekuensi dari satu variabel numerik tunggal (dalam hal ini, `btc_trade_volume` yang sudah di-scaling). Histogram bekerja dengan membagi rentang data menjadi beberapa interval atau "bins" (diatur menjadi 50 dalam kode), dan kemudian menghitung berapa banyak titik data (Frekuensi) yang jatuh ke dalam setiap interval tersebut. Ini memungkinkan analisis untuk secara cepat memahami bentuk sebaran data. Penambahan parameter `kde=True`, yang memunculkan garis kurva halus (Kernel Density Estimate), semakin membantu memperjelas bentuk distribusi, seperti apakah data tersebut simetris atau miring (skewed).

Insight:

Dari grafik histogram, insight utamanya adalah bahwa distribusi Volume Transaksi (Scaled) sangat miring ke kanan (right-skewed). Pola ini terlihat sangat jelas dari adanya satu batang (bar) yang ekstrem tinggi di sisi kiri (frekuensi > 2500), yang terkonsentrasi di dekat nilai nol. Sementara itu, sisa datanya tersebar sangat tipis dan membentuk "ekor" (tail) yang panjang ke

arah kanan (menuju nilai-nilai yang lebih tinggi). Ini mengindikasikan bahwa meskipun data telah di-scaling dan fiturnya dikurangi, sifat asli data tetap ada: yaitu, pada sebagian besar periode waktu, volume transaksi berada pada level yang sangat rendah. Hanya ada sejumlah kecil kejadian di mana volume transaksi melonjak secara ekstrem, yang direpresentasikan oleh ekor panjang di sebelah kanan tersebut.

2.1.5 Statistical Analysis

Bagian ini membahas analisis statistik yang dilakukan untuk mengidentifikasi pola, hubungan, serta pengaruh antar variabel dalam data. Melalui penerapan metode statistik, hasil analisis dapat memberikan dasar yang kuat untuk pengambilan keputusan, validasi temuan, serta pemahaman yang lebih mendalam terhadap fenomena yang diteliti.

1. Mengambil Parametrik yang Penting

```
[ ] df_stat = df_stat.select_dtypes(include=['float64', 'int64']).dropna()  
    x = df_stat['btc_market_price']  
    y = df_stat['btc_trade_volume']
```

Kode ini merupakan langkah persiapan data. Intinya, kode ini pertama-tama membersihkan DataFrame `df_stat` dengan cara memilih hanya kolom-kolom bertipe numerik (yaitu `float64` dan `int64`). Segera setelah itu, metode `.dropna()` diterapkan untuk menghapus baris manapun dari kolom-kolom numerik tersebut yang memiliki nilai kosong. Setelah DataFrame bersih, kode ini mengisolasi dua kolom spesifik untuk analisis: kolom `btc_market_price` ditugaskan ke variabel `x`, dan kolom `btc_trade_volume` ditugaskan ke variabel `y`. Hasilnya adalah dua variabel (`x` dan `y`) yang siap digunakan untuk analisis statistik atau visualisasi, seperti uji korelasi atau scatter plot.

2. Melakukan Uji Parametrik dan non-Parametrik

1. Uji Parametrik dengan Menggunakan Pearson Correlation

```

import math
from scipy import stats

# Uji PARAMETRIK: Pearson Correlation
pearson_corr, pearson_p = stats.pearsonr(x, y)
effect_size_pearson = abs(pearson_corr)

# Hitung Confidence Interval untuk korelasi (95%)
# Fisher transformation
def fisher_ci(r, n, alpha=0.05):
    z = np.arctanh(r)
    se = 1 / math.sqrt(n - 3)
    z_crit = stats.norm.ppf(1 - alpha/2)
    lo_z, hi_z = z - z_crit * se, z + z_crit * se
    lo, hi = np.tanh((lo_z, hi_z))
    return lo, hi

ci_low, ci_high = fisher_ci(pearson_corr, len(df_stat))

print("=== PARAMETRIC TEST: Pearson Correlation ===")
print(f"Variabel: btc_market_price vs btc_trade_volume")
print(f"Koefisien Korelasi (r) = {pearson_corr:.4f}")
print(f"p-value = {pearson_p:.4e}")
print(f"Effect size (|r|) = {effect_size_pearson:.3f}")
print(f"95% Confidence Interval = [{ci_low:.3f}, {ci_high:.3f}]")

# Interpretasi sederhana
if pearson_p < 0.05:
    print("Interpretasi: Terdapat hubungan linear yang signifikan secara statistik antara kedua variabel (p < 0.05).")
else:
    print("Interpretasi: Tidak terdapat hubungan linear yang signifikan antara kedua variabel (p ≥ 0.05).")

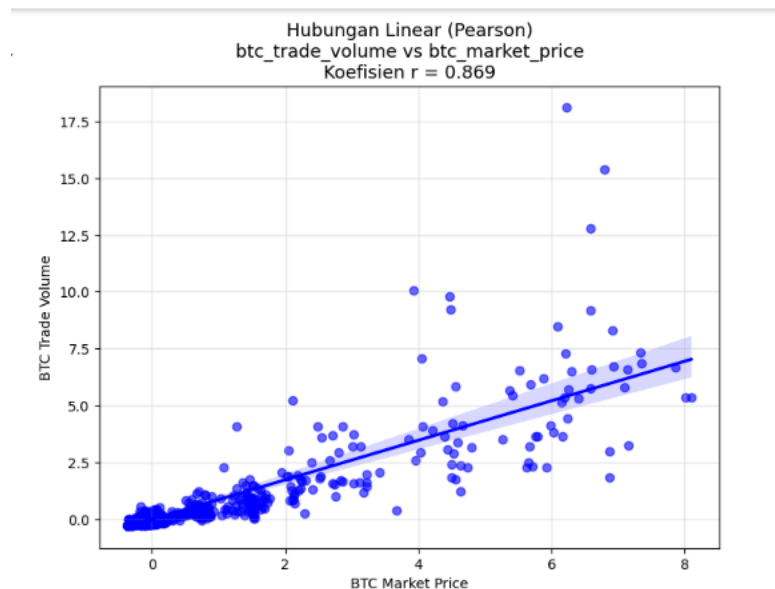
```

```

=== PARAMETRIC TEST: Pearson Correlation ===
Variabel: btc_market_price vs btc_trade_volume
Koefisien Korelasi (r) = 0.8686
p-value = 0.0000e+00
Effect size (|r|) = 0.869
95% Confidence Interval = [0.859, 0.877]
Interpretasi: Terdapat hubungan linear yang signifikan secara statistik antara kedua variabel (p < 0.05).

```

Visualisasi:



Kode ini menjalankan Uji Korelasi Pearson (`stats.pearsonr`), yang merupakan uji statistik parametrik. Tujuannya adalah untuk mengukur secara presisi seberapa kuat dan ke arah mana hubungan linier (yang membentuk pola garis lurus) antara dua variabel: harga pasar Bitcoin (`btc_market_price`) dan volume perdagangannya (`btc_trade_volume`).

Hasil output teks menunjukkan Koefisien Korelasi (r) = 0.8686. Nilai ini, yang mendekati 1.0, mengindikasikan adanya hubungan linier positif yang sangat kuat. Ini didukung oleh p-value 0.0000e+00 (nol), yang berarti temuan ini sangat signifikan secara statistik dan bukan terjadi karena kebetulan. Plot visual "Hubungan Linear (Pearson)" (grafik biru) mengkonfirmasi hal ini, menunjukkan tren titik-titik yang bergerak dari kiri bawah ke kanan atas. Namun, plot ini juga menunjukkan mengapa korelasinya tidak sempurna 1.0: data terlihat sangat miring (skewed), dengan sebagian besar titik menumpuk di kiri bawah dan beberapa outlier (pencilan) di kanan atas, yang sedikit "menarik" garis regresi.

Insight utama dari Uji Pearson ini adalah bahwa terdapat hubungan linier positif yang sangat kuat ($r = 0.8686$) dan sangat signifikan secara statistik (p-value ≈ 0) antara harga pasar Bitcoin dan volume perdagangannya.

2. Uji Non-Parametrik dengan Menggunakan Spearman Correlation

```
#UJI NON-PARAMETRIK: Spearman Correlation

spearman_corr, spearman_p = stats.spearmanr(x, y)

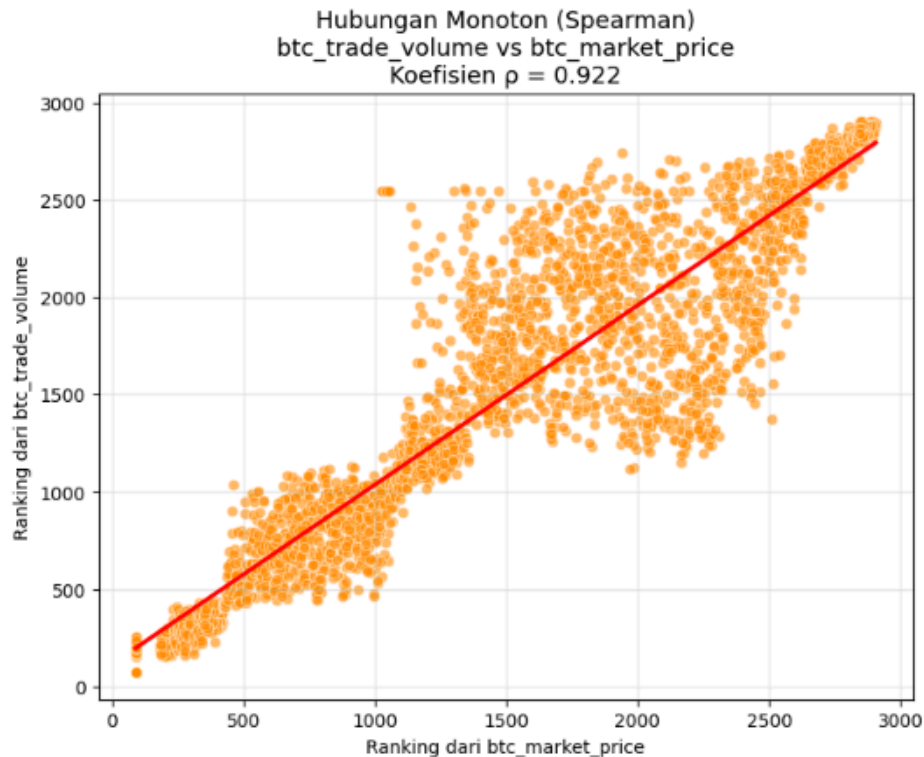
print("\n=== NON-PARAMETRIC TEST: Spearman Correlation ===")
print(f"Variabel: btc_market_price vs btc_trade_volume")
print(f"Koefisien Korelasi (p) = {spearman_corr:.4f}")
print(f"p-value = {spearman_p:.4e}")
print(f"Effect size (|p|) = {abs(spearman_corr):.3f}")

if spearman_p < 0.05:
    print("Interpretasi: Terdapat hubungan monoton signifikan antara kedua variabel (p < 0.05).")
else:
    print("Interpretasi: Tidak terdapat hubungan monoton signifikan antara kedua variabel (p ≥ 0.05).")
```



```
=== NON-PARAMETRIC TEST: Spearman Correlation ===
Variabel: btc_market_price vs btc_trade_volume
Koefisien Korelasi (p) = 0.9224
p-value = 0.0000e+00
Effect size (|p|) = 0.922
Interpretasi: Terdapat hubungan monoton signifikan antara kedua variabel (p < 0.05).
```

Visualisasi:



Kode ini juga menjalankan Uji Korelasi Spearman (`stats.spearmanr`), yang merupakan uji statistik non-parametrik. Tujuannya berbeda: ia mengukur kekuatan hubungan monotonik. Artinya, ia hanya memeriksa apakah "ketika satu variabel naik, variabel lainnya juga konsisten naik", terlepas dari apakah pola tersebut membentuk garis lurus sempurna atau tidak. Uji ini bekerja menggunakan peringkat (ranking) data, bukan nilai aktualnya.

Hasil output menunjukkan Koefisien Korelasi (ρ) = 0.9224. Nilai ini lebih tinggi daripada koefisien Pearson (0.8686). Plot visual "Hubungan Monoton (Spearman)" (grafik oranye) menjelaskan alasannya: dengan mengubah nilai aktual menjadi peringkat, dampak dari outlier dan distribusi data yang miring (skewed) dapat diabaikan. Hasilnya, sebaran titik-titik peringkat terlihat jauh lebih rapat dan membentuk pola linier yang lebih konsisten, yang menunjukkan hubungan monotonik yang bahkan lebih kuat.

Insight utama dari Uji Spearman ini adalah bahwa terdapat hubungan monotonik positif yang hampir sempurna (dengan koefisien $\rho = 0.9224$) antara harga Bitcoin dan volume perdagangannya.

BAB III

KESIMPULAN

Proyek analisis data historis Bitcoin ini telah berhasil dilaksanakan melalui serangkaian tahapan metodologi data science, mulai dari pengumpulan data, pembersihan, transformasi, reduksi fitur, visualisasi, hingga analisis statistik. Berdasarkan hasil analisis dan pembahasan, dapat ditarik beberapa kesimpulan utama yang menjawab rumusan masalah:

1. Distribusi Data dan Outlier: Analisis visualisasi menggunakan histogram (pada Bagian 2.4) menunjukkan bahwa variabel-variabel kunci seperti `btc_market_price` dan `btc_trade_volume` memiliki distribusi yang sangat miring ke kanan (right-skewed). Pola ini mengindikasikan bahwa sebagian besar data terkonsentrasi pada nilai-nilai yang lebih rendah, namun terdapat sejumlah kecil observasi dengan nilai yang sangat tinggi (ekstrem). Nilai-nilai ekstrem ini secara efektif berfungsi sebagai outlier yang mempengaruhi analisis statistik, seperti yang terlihat dari perbedaan hasil uji Pearson dan Spearman.

2. Peran Visualisasi Data: Visualisasi data interaktif terbukti sangat efektif dalam mengidentifikasi pola dan tren. Grafik garis (line plot) secara jelas menggambarkan volatilitas harga Bitcoin yang tinggi, terutama lonjakan eksponensial yang terjadi pada periode 2017-2018. Sementara itu, diagram tebar (scatter plot) berhasil menunjukkan adanya korelasi positif yang kuat antara volume perdagangan dan harga Bitcoin, di mana titik data cenderung bergerak dari kiri bawah ke kanan atas.
3. Pentingnya Preprocessing Lanjutan: Tahap preprocessing lanjutan sangat krusial untuk meningkatkan kualitas data. Penanganan missing values melalui imputasi rata-rata memastikan kelengkapan data. Lebih penting lagi, teknik reduksi fitur yang didasarkan pada matriks korelasi berhasil mengidentifikasi dan mengatasi masalah multikolinearitas yang tinggi (korelasi antar fitur > 0.9). Dengan membuang fitur-fitur yang redundan, dataset menjadi lebih efisien dan fokus, sehingga mencegah bias dalam analisis selanjutnya.
4. Hasil Uji Statistik (Korelasi): Analisis statistik berhasil mengkuantifikasi hubungan antar variabel utama.
5. Uji Korelasi Pearson (parametrik) menunjukkan adanya hubungan linear positif yang sangat kuat antara `btc_market_price` dan `btc_trade_volume`, dengan koefisien korelasi (r) sebesar 0.8686 ($p \approx 0$).
6. Uji Korelasi Spearman (non-parametrik) menunjukkan hubungan monotonik positif yang bahkan lebih kuat, dengan koefisien korelasi (ρ) sebesar 0.9224.
7. Nilai Spearman yang lebih tinggi mengonfirmasi bahwa hubungan antara harga dan volume sangat konsisten (saat satu naik, yang lain cenderung naik), dan uji non-parametrik ini lebih robust dalam menangani data yang tidak terdistribusi normal dan memiliki outlier, seperti yang telah diidentifikasi pada poin 1.

Secara keseluruhan, proyek ini berhasil mencapai tujuannya dengan mengidentifikasi pola harga yang volatil, distribusi data yang miring, serta mengonfirmasi secara statistik adanya hubungan positif yang sangat signifikan antara harga pasar Bitcoin dan volume perdagangannya.

DAFTAR PUSTAKA

- [1] D. Rifaldi, Abdul Fadlil, and Herman, “Preprocessing Techniques in Text Mining: ‘MentalHealth’ Tweet Data,” *Decod. J. Pendidik. Teknol. Inf.*, vol. 3, no. 2, pp. 161–171, 2023.
- [2] T. Gori, A. Sunyoto, and H. Al Fatta, “Preprocessing Data dan Klasifikasi untuk Prediksi Kinerja Akademik Siswa,” *J. Teknol. Inf. dan Ilmu Komput.*, vol. 11, no. 1, pp. 215–224, 2024, doi: 10.25126/jtiik.20241118074.
- [3] Rangga Gelar Guntara, “Visualisasi Data Laporan Penjualan Toko Online Melalui Pendekatan Data Science Menggunakan Google Colab,” *ULIL ALBAB J. Ilm. Multidisiplin*, vol. 2, no. 6, pp. 2091–2100, 2023, doi: 10.56799/jim.v2i6.1578.
- [4] A. T. J. H, “Preprocessing Text untuk Meminimalisir Kata yang Tidak Berarti dalam Proses Text Mining,” pp. 1–9.