Laporan Proyek Akhir

Location-Based Species Presence Prediction

Disusun Oleh:

12S22003 – Yohana Natalia Siahaan

12S22016 – Desri Stevie Natalie Dabukke

12S22017 – Lenna Febriana

INSTITUT TEKNOLOGI DEL

FAKULTAS INFORMATIKA DAN TEKNIK ELEKTRO

TAHUN AJARAN 2024/2025

BAB 1 PENDAHULUAN

Keanekaragaman hayati merupakan salah satu aspek penting dalam menjaga keseimbangan ekosistem dan keberlanjutan lingkungan hidup. Salah satu tantangan utama keanekaragaman hayati adalah mengetahui spesies apa saja yang ada di suatu wilayah geografis tertentu. Dengan kemajuan teknologi informasi dan ketersediaan data lingkungan berskala besar, kini terbuka peluang untuk mengembangkan sistem prediksi kehadiran spesies secara otomatis.

Salah satu pendekatan yang digunakan dalam konteks ini adalah *Species Distribution Modelling (SDM)*, yaitu metode yang memodelkan hubungan antara kondisi lingkungan dengan keberadaan spesies tertentu. Pendekatan ini sangat berguna untuk menghasilkan peta distribusi spesies, mendukung pelestarian, dan membantu proses identifikasi spesies. Melalui kompetisi GeoLifeCLEF 2025, tersedia data observasi kehadiran spesies serta data lingkungan seperti *satellite images, climatic time series, land cover,* dan *human footprint* yang dapat dimanfaatkan untuk membangun model prediktif berbasis lokasi.

Proyek ini dirancang untuk menjawab tantangan tersebut dengan mengembangkan model *machine learning* yang mampu memprediksi spesies tumbuhan yang mungkin hadir di suatu titik lokasi berdasarkan data data yang diberikan kompetisi GeoLifeCLEF 2025.

BAB 2 BUSINESS UNDERSTANDING

2.1 Objektif Bisnis

Tujuan dari kompetisi ini adalah untuk membantu organisasi atau lembaga yang peduli terhadap lingkungan dalam memprediksi keberadaan spesies di suatu lokasi berdasarkan data yang tersedia. Dengan menggunakan model mechine learning, dapat diketahui apakah suatu spesies mungkin ada di tempat dan waktu tertentu, meskipun tidak ada pengamatan langsung. Ini penting agar pihak-pihak terkait bisa mengambil keputusan yang lebih tepat dalam menjaga keanekaragaman hayati, seperti menentukan daerah yang perlu dilindungi atau dipantau lebih lanjut.

2.2 Tujuan Teknis

Tujuan teknis untuk proyek ini berfokus pada pembangunan model prediktif yang dapat mengidentifikasi spesies tanaman yang ada di lokasi tertentu berdasarkan serangkaian variabel lingkungan dan data deret waktu. Berikut adalah tujuan teknis utama yang ingin dicapai:

1. Data Integration and Preprocessing

   Proyek ini melibatkan integrasi berbagai jenis data, termasuk deret waktu iklim, dan variabel lingkungan. Langkah-langkah persiapan data akan mencakup pembersihan data, penanganan nilai yang hilang, dan pengkodean data kategorikal untuk memastikan bahwa model dapat belajar dengan efektif dari berbagai jenis data.
2. Model Development

   Mengembangkan model pembelajaran mesin yang dapat memprediksi dengan akurat spesies tanaman yang ada di suatu lokasi berdasarkan fitur yang diberikan. Model ini akan memanfaatkan berbagai prediktor, seperti data lingkungan untuk membuat prediksi yang akurat untuk data Presence-Absence (PA).
3. Evaluation

   Evaluasi model akan dilakukan dengan mengukur seberapa akurat model dalam memprediksi spesies yang ada di suatu lokasi. Hal ini dihitung berdasarkan seberapa besar kecocokan antara spesies yang diprediksi dan yang sebenarnya ada di lokasi tersebut. Model akan terus diperbaiki dengan cara menyesuaikan pengaturan (hyperparameter), memperbaiki fitur data, dan menggunakan teknik-teknik canggih seperti pembelajaran ansambel atau metode pembelajaran mendalam.
4. Deployment

Setelah model diselesaikan dan dioptimalkan, model tersebut akan diterapkan untuk memprediksi spesies tanaman dalam skenario dunia nyata, memfasilitasi perencanaan konservasi dan pemantauan biodiversitas. Model ini juga akan diintegrasikan ke dalam sistem untuk alat identifikasi spesies, yang dapat digunakan oleh peneliti lapangan dan ilmuwan warga negara.

2.3 Rencana Proyek

Proyek ini dirancang untuk diselesaikan dalam jangka waktu 5 minggu, dimulai pada minggu ke-11 hingga minggu ke-15 kalender akademik kampus. Setiap minggunya difokuskan pada fase berbeda dalam siklus Data Science untuk memastikan alur kerja yang sistematis dan hasil yang optimal. Berikut adalah timeline dari pengerjaan proyek ini :

Task	Minggu				
	Minggu 1	Minggu 2	Minggu 3	Minggu 4	Minggu 5
Business Understanding					
Data Understanding					
Data Cleaning & Preparation					
Label Construction					
Modeling					
Evaluation & Improvement					
Final Integration & Reporting					
Presentation					

Rencana proyek ini menguraikan langkah-langkah yang diperlukan untuk mencapai tujuan-tujuan yang telah disebutkan di atas. Proyek ini akan dilaksanakan dalam beberapa fase:

1. Persiapan Data
    a. Mengumpulkan dan meninjau dataset (catatan Presence-Only dan Presence-Absence).
    b. Melakukan persiapan data (pembersihan, pengkodean, ekstraksi fitur).

c. Mengeksplorasi visualisasi data dan melakukan analisis eksplorasi data (EDA) untuk memahami hubungan antara prediktor dan spesies tanaman.

2. Pengembangan dan Pelatihan Model
    a. Memilih algoritma pembelajaran mesin yang sesuai (misalnya, random forests, gradient boosting, jaringan saraf).
    b. Melatih dan memvalidasi model menggunakan dataset pelatihan.
    c. Mengevaluasi kinerja model menggunakan metrik evaluasi kompetisi (samples-averaged score).
    d. Iterasi pengembangan model, meningkatkan dengan teknik seperti tuning hyperparameter, validasi silang, dan pemilihan fitur.
3. Evaluasi dan Pengujian
    a. Menguji model pada data uji yang disediakan untuk memeriksa kekuatan prediksi.
    b. Menyempurnakan model berdasarkan hasil pengujian dan metrik evaluasi.
    c. Mempersiapkan file pengiriman dengan format yang diperlukan (surveyId, prediksi).

BAB 3 DATA UNDERSTANDING

3.1 Pengumpulan Data

Pada tahap ini, data yang digunakan dalam penelitian ini diambil dari file CSV yang berisi data Presence-Absence (PA). Proses pengumpulan data dimulai dengan memuat dataset menggunakan pustaka pandas. Dataset tersebut dimuat dengan fungsi read_csv(), yang memungkinkan kita untuk mengakses informasi yang tercatat dalam file CSV. Setelah data dimuat, beberapa baris pertama dari dataset dapat diperiksa dengan menggunakan metode head() untuk memastikan bahwa data sudah benar-benar terbaca dan memahami struktur data yang ada. Berikut adalah kode untuk memuat dataset:

```
[3]: import pandas as pd

file_path = r'D:\Sem 6\DAMI\Proyek\GLC25_PA_metadata_train.csv'
# Memuat data
data = pd.read_csv(file_path)

data.head()
```

[3]:

	lon	lat	year	geoUncertaintyInM	areaInM2	region	country	speciesId	surveyId
0	3.099038	43.134956	2021	5.0	100.0	MEDITERRANEAN	France	6874.0	212
1	3.099038	43.134956	2021	5.0	100.0	MEDITERRANEAN	France	476.0	212
2	3.099038	43.134956	2021	5.0	100.0	MEDITERRANEAN	France	11157.0	212
3	3.099038	43.134956	2021	5.0	100.0	MEDITERRANEAN	France	8784.0	212
4	3.099038	43.134956	2021	5.0	100.0	MEDITERRANEAN	France	4530.0	212

3.2 Penelaahan Data

Setelah data berhasil dimuat, langkah berikutnya adalah melakukan eksplorasi data untuk memahami struktur dan kualitas dari dataset yang digunakan. Langkah awal yang dilakukan adalah memeriksa informasi umum dataset, seperti tipe data setiap kolom dan jumlah nilai yang tidak kosong (non-null). Untuk itu, digunakan fungsi data.info() yang memberikan gambaran mengenai tipe data dan apakah terdapat nilai yang hilang pada setiap kolom.

```
[5]: # Memeriksa informasi dataset, termasuk tipe data dan jumlah nilai non-null
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1483637 entries, 0 to 1483636
Data columns (total 9 columns):
 #   Column             Non-Null Count    Dtype
---  ------             --------------    -----
 0   lon                1483637 non-null  float64
 1   lat                1483637 non-null  float64
 2   year               1483637 non-null  int64
 3   geoUncertaintyInM  1471141 non-null  float64
 4   areaInM2           1300365 non-null  float64
 5   region             1483637 non-null  object
 6   country            1483637 non-null  object
 7   speciesId          1483637 non-null  float64
 8   surveyId           1483637 non-null  int64
dtypes: float64(5), int64(2), object(2)
memory usage: 101.9+ MB
```

Selanjutnya, untuk mengetahui ringkasan statistik dari kolom-kolom numerik dalam dataset, digunakan fungsi data.describe(). Fungsi ini akan menampilkan informasi seperti nilai rata-rata, standar deviasi, nilai minimum, maksimum, serta kuartil dari masing-masing kolom numerik. Hal ini membantu dalam memahami distribusi data secara keseluruhan.

```
[7]: # Menampilkan statistik deskriptif untuk data numerik
     data.describe()
```

[7]:

	lon	lat	year	geoUncertaintyInM	areaInM2	speciesId	surveyId
count	1.483637e+06	1.483637e+06	1.483637e+06	1.471141e+06	1300365.0	1.483637e+06	1.483637e+06
mean	9.024850e+00	5.242552e+01	2.018179e+03	7.483046e+00	-inf	5.811442e+03	1.969927e+06
std	4.437247e+00	4.853112e+00	1.094525e+00	6.190821e+00	NaN	3.392812e+03	1.135967e+06
min	-9.018346e+00	3.511705e+01	2.017000e+03	0.000000e+00	-inf	2.000000e+00	2.120000e+02
25%	6.052593e+00	4.856769e+01	2.017000e+03	3.000000e+00	25.0	2.847000e+03	9.768390e+05
50%	9.380850e+00	5.515287e+01	2.018000e+03	1.000000e+01	79.0	6.079000e+03	1.976273e+06
75%	1.041882e+01	5.626047e+01	2.019000e+03	1.000000e+01	79.0	8.818000e+03	2.960059e+06
max	2.915421e+01	5.989300e+01	2.021000e+03	1.000000e+02	8000.0	1.125400e+04	3.919655e+06

Untuk mengecek apakah terdapat nilai yang hilang (missing values) pada dataset, digunakan fungsi data.isnull().sum(). Fungsi ini akan menunjukkan jumlah nilai yang hilang di setiap kolom. Jika ditemukan kolom dengan jumlah nilai hilang yang terlalu banyak, maka kolom tersebut akan dihapus dari dataset. Sebagai contoh, kolom geoUncertaintyInM dan areaInM2 dihapus menggunakan fungsi data.drop() karena memiliki terlalu banyak data yang kosong.

```
[9]: # Memeriksa jumlah nilai yang hilang pada setiap kolom
     data.isnull().sum()
```

```
[9]: lon                     0
     lat                     0
     year                    0
     geoUncertaintyInM   12496
     areaInM2           183272
     region                  0
     country                 0
     speciesId               0
     surveyId                0
     dtype: int64
```

```
[11]: # Menghapus kolom dengan banyak missing values
      data.drop(columns=['geoUncertaintyInM', 'areaInM2'], inplace=True)

      # Periksa kembali apakah masih ada nilai yang hilang
      data.isnull().sum()
```

```
[11]: lon          0
      lat          0
      year         0
      region       0
      country      0
      speciesId    0
      surveyId     0
      dtype: int64
```

3.3 Validasi Data

Pada bagian validasi data, kita memeriksa hubungan antar fitur numerik dan memproses kolom kategorikal dengan melakukan One-Hot Encoding. Kita pertama-tama memisahkan kolom-kolom numerik dan kategorikal untuk lebih memahami tipe data yang ada. Menggunakan fungsi select_dtypes(), kita dapat memilih kolom numerik dan kategorikal untuk analisis lebih lanjut.

[21]:
```
# MENGHITUNG KORELASI
# Memisahkan kolom numerik dan kategorikal
numerical_columns = data.select_dtypes(include=['float64', 'int64']).columns
categorical_columns = data.select_dtypes(include=['object']).columns

# Menampilkan kolom-kolom numerik dan kategorikal
print("Kolom Numerik:")
print(numerical_columns)

print("\nKolom Kategorikal:")
print(categorical_columns)
```

```
Kolom Numerik:
Index(['lon', 'lat', 'year', 'speciesId', 'surveyId'], dtype='object')

Kolom Kategorikal:
Index(['region', 'country'], dtype='object')
```

Setelah itu, untuk kolom kategorikal seperti region dan country, kita melakukan One-Hot Encoding untuk mengubah kolom tersebut menjadi format numerik yang bisa diproses oleh model pembelajaran mesin. Fungsi pd.get_dummies() digunakan untuk melakukan encoding pada kolom kategorikal.

[23]:
```
# Menggunakan One-Hot Encoding untuk kolom kategorikal
data_encoded = pd.get_dummies(data, columns=categorical_columns)

# Memeriksa data setelah encoding
data_encoded.head()
```

[23]:

	lon	lat	year	speciesId	surveyId	region_ALPINE	region_ATLANTIC	region_BLACK SEA	region_BOREAL	region_CONTINENTAL	...	country_Norway	country
0	3.099038	43.134956	2021	6874.0	212	False	False	False	False	False	...	False	
1	3.099038	43.134956	2021	476.0	212	False	False	False	False	False	...	False	
2	3.099038	43.134956	2021	11157.0	212	False	False	False	False	False	...	False	
3	3.099038	43.134956	2021	8784.0	212	False	False	False	False	False	...	False	
4	3.099038	43.134956	2021	4530.0	212	False	False	False	False	False	...	False	

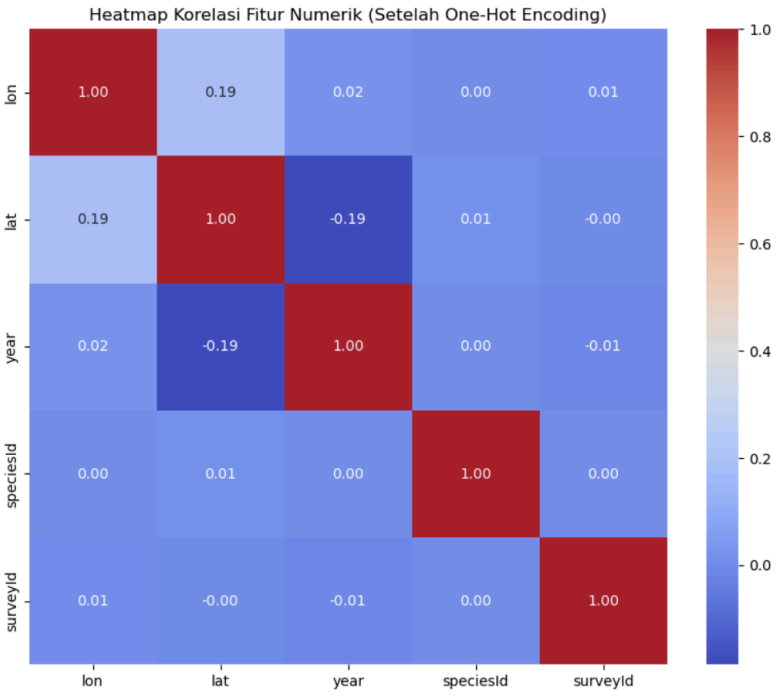
5 rows × 42 columns

Kemudian, kita memeriksa korelasi antar fitur numerik yang ada setelah encoding, dengan menghitung matriks korelasi menggunakan numeric_data.corr(). Korelasi antar fitur numerik ini kemudian divisualisasikan menggunakan heatmap dengan bantuan seaborn.heatmap(). Visualisasi korelasi ini membantu kita memahami hubungan antara fitur-fitur yang dapat memengaruhi model.

[25]:
```
# Memilih hanya kolom numerik
numeric_data = data_encoded.select_dtypes(include=['float64', 'int64'])

# Menghitung korelasi antar fitur numerik
correlation_matrix = numeric_data.corr()

# Menampilkan heatmap korelasi
plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap="coolwarm", fmt='.2f')
plt.title('Heatmap Korelasi Fitur Numerik (Setelah One-Hot Encoding)')
plt.show()
```



BAB 4 DATA PREPARATION

4.1 Memilah Data

Pada tahap pertama persiapan data, kita perlu memisahkan kolom fitur dan label. Fitur adalah variabel yang digunakan untuk memprediksi speciesId yang akan diprediksi oleh model, dan label adalah kolom yang berisi speciesId. Sebelum melakukan pemisahan fitur dan label, kita juga perlu melakukan One-Hot Encoding pada kolom region dan country, yang merupakan data kategorikal. One-Hot Encoding mengubah kolom kategorikal tersebut menjadi variabel numerik yang dapat diproses oleh model pembelajaran mesin. Berikut adalah kode untuk melakukan One-Hot Encoding dan memilih kolom yang relevan setelah encoding:

[27]:
```
# Melakukan One-Hot Encoding pada kolom 'region' dan 'country'
data_encoded = pd.get_dummies(data, columns=['region', 'country'])

# Menampilkan data setelah encoding
data_encoded.head()
```

[27]:

	lon	lat	year	speciesId	surveyId	region_ALPINE	region_ATLANTIC	region_BLACK SEA	region_BOREAL	region_CONTINENTAL	...	country_Norway	countr
0	3.099038	43.134956	2021	6874.0	212	False	False	False	False	False	...	False	
1	3.099038	43.134956	2021	476.0	212	False	False	False	False	False	...	False	
2	3.099038	43.134956	2021	11157.0	212	False	False	False	False	False	...	False	
3	3.099038	43.134956	2021	8784.0	212	False	False	False	False	False	...	False	
4	3.099038	43.134956	2021	4530.0	212	False	False	False	False	False	...	False	

Setelah melakukan One-Hot Encoding, kita akan memilih kolom yang relevan untuk analisis lebih lanjut. Kolom yang relevan adalah kolom-kolom yang berisi informasi yang dapat digunakan untuk memprediksi speciesId. Misalnya, kita akan memilih kolom lon, lat, year, dan kolom hasil encoding dari region dan country, serta kolom speciesId sebagai label. Kolom surveyId tidak diperlukan untuk model dan akan dihapus.

4.2 Membersihkan Data

Pada tahap pembersihan data, kita perlu memastikan bahwa data yang digunakan tidak mengandung nilai yang hilang (missing values) atau duplikasi, serta sudah dalam format yang sesuai untuk pelatihan model. Dalam hal ini, kita menggunakan One-Hot Encoding untuk mengubah kolom kategorikal seperti region dan country menjadi format numerik yang dapat diproses oleh model pembelajaran mesin. Selain itu, kita perlu menangani missing values dengan cara mengisinya dengan nilai yang relevan, seperti rata-rata untuk kolom numerik.

```
# Menghapus duplikasi data
data_encoded = data_encoded.drop_duplicates()

# Menangani missing values (mengisi dengan rata-rata untuk kolom numerik)
data_encoded.fillna(data_encoded.mean(), inplace=True)

# Menampilkan beberapa baris setelah pembersihan data
data_encoded.head()
```

4.3 Mengkonstruksi Data

Mengkonstruksi data bertujuan untuk memilih fitur yang relevan untuk model. Setelah melakukan One-Hot Encoding dan membersihkan data, kita memilih kolom-kolom yang relevan untuk analisis lebih lanjut. Dalam hal ini, kita memilih kolom lon, lat, year, dan kolom speciesId sebagai label, serta kolom-kolom hasil encoding dari region dan country yang relevan.

Setelah memilih kolom-kolom yang relevan, kita menstandarisasi fitur numerik seperti lon, lat, dan year untuk memastikan bahwa setiap fitur berada pada skala yang sama. Fitur yang memiliki skala yang sangat berbeda bisa mempengaruhi performa model, sehingga standarisasi penting dilakukan.

[30]:
```
# Memilih kolom yang relevan setelah One-Hot Encoding
relevant_columns = ['lon', 'lat', 'year', 'speciesId'] + [col for col in data_encoded.columns if col not in ['surveyId', 'speciesId']]

# Data dengan kolom relevan
data_preprocessed = data_encoded[relevant_columns]

# Menampilkan beberapa baris dari data yang sudah dipilih
data_preprocessed.head()
```

[30]:

	lon	lat	year	speciesId	lon	lat	year	region_ALPINE	region_ATLANTIC	region_BLACK SEA	...	country_Norway	country_Poland	country_Port
0	3.099038	43.134956	2021	6874.0	3.099038	43.134956	2021	False	False	False	...	False	False	I
1	3.099038	43.134956	2021	476.0	3.099038	43.134956	2021	False	False	False	...	False	False	I
2	3.099038	43.134956	2021	11157.0	3.099038	43.134956	2021	False	False	False	...	False	False	I
3	3.099038	43.134956	2021	8784.0	3.099038	43.134956	2021	False	False	False	...	False	False	I
4	3.099038	43.134956	2021	4530.0	3.099038	43.134956	2021	False	False	False	...	False	False	I

[32]:
```
from sklearn.preprocessing import StandardScaler

# Inisialisasi scaler
scaler = StandardScaler()

# Menstandarisasi fitur numerik
data_preprocessed[['lon', 'lat', 'year']] = scaler.fit_transform(data_preprocessed[['lon', 'lat', 'year']])

# Menampilkan data setelah standarisasi
data_preprocessed.head()
```

```
C:\Users\ASUS\AppData\Local\Temp\ipykernel_10868\537075296.py:7: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  data_preprocessed[['lon', 'lat', 'year']] = scaler.fit_transform(data_preprocessed[['lon', 'lat', 'year']])
```

[32]:

	lon	lat	year	speciesId	lon	lat	year	region_ALPINE	region_ATLANTIC	region_BLACK SEA	...	country_Norway	country_Poland	cou
0	-1.335471	-1.914352	2.577744	6874.0	-1.335471	-1.914352	2.577744	False	False	False	...	False	False	
1	-1.335471	-1.914352	2.577744	476.0	-1.335471	-1.914352	2.577744	False	False	False	...	False	False	
2	-1.335471	-1.914352	2.577744	11157.0	-1.335471	-1.914352	2.577744	False	False	False	...	False	False	
3	-1.335471	-1.914352	2.577744	8784.0	-1.335471	-1.914352	2.577744	False	False	False	...	False	False	
4	-1.335471	-1.914352	2.577744	4530.0	-1.335471	-1.914352	2.577744	False	False	False	...	False	False	

Proses ini memastikan bahwa hanya fitur yang relevan yang dipilih dan semua fitur numerik distandarisasi, yang membuat data siap digunakan untuk pelatihan model.

4.4 Menentukan Label Data

Setelah memilih fitur yang relevan dan menstandarisasi data, kita memisahkan fitur (X) dan label (y). Dalam kasus ini, kolom speciesId merupakan label yang akan diprediksi oleh model, sementara sisanya adalah fitur yang digunakan untuk melakukan prediksi. Pemisahan ini sangat penting karena model hanya akan belajar untuk memprediksi nilai pada kolom speciesId berdasarkan input dari fitur lainnya.

Kode berikut digunakan untuk memisahkan label (y) dari fitur (X):

[36]:
```
# Memisahkan fitur (X) dan label (y)
X = data_preprocessed.drop('speciesId', axis=1)  # Fitur
y = data_preprocessed['speciesId']  # Label (ID spesies)
```

Tujuan dari langkah ini adalah untuk memisahkan variabel target (speciesId) agar model dapat belajar untuk memprediksi nilai tersebut berdasarkan input dari fitur-fitur lainnya.

4.5 Mengintegrasikan Data

Langkah terakhir dalam persiapan data adalah mengintegrasikan data, yaitu membagi dataset menjadi data pelatihan dan data pengujian. Pembagian ini penting karena memungkinkan kita untuk melatih model menggunakan satu bagian dari data (pelatihan) dan mengevaluasi kinerjanya pada data lain yang belum pernah dilihat sebelumnya (pengujian). Pembagian ini

memastikan bahwa model dapat diuji dengan data yang tidak terlibat dalam pelatihan, memberikan gambaran yang lebih realistis tentang seberapa baik model bekerja pada data baru.

Kode berikut digunakan untuk membagi data menjadi pelatihan dan pengujian:

```
from sklearn.model_selection import train_test_split

# Membagi data menjadi pelatihan dan pengujian (80% untuk pelatihan, 20% untuk pengujian)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Menampilkan ukuran data pelatihan dan pengujian
print(f"Ukuran Data Pelatihan: {X_train.shape}")
print(f"Ukuran Data Pengujian: {X_test.shape}")
```

```
Ukuran Data Pelatihan: (1186909, 43)
Ukuran Data Pengujian: (296728, 43)
```

Dengan membagi data menjadi set pelatihan dan pengujian, kita memastikan bahwa model dapat dievaluasi secara objektif, dan kita dapat memeriksa apakah model dapat menggeneralisasi dengan baik ke data yang tidak terlihat sebelumnya.