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

DAFTAR ISI

BAB 1 PENDAHULUAN

1.1 Latar Belakang

Keanekaragaman hayati adalah salah satu unsur penting yang menjaga keseimbangan ekosistem dan keberlanjutan lingkungan hidup. Salah satu tantangan besar dalam pelestarian keanekaragaman hayati adalah mengetahui spesies apa saja yang hidup di suatu wilayah. Pengetahuan ini sangat penting untuk merencanakan langkah-langkah pelestarian yang tepat dan mengelola sumber daya alam secara berkelanjutan.

Dengan adanya kemajuan teknologi, kini tersedia lebih banyak data dan alat untuk menganalisis lingkungan. Salah satu metode yang dapat digunakan untuk mempelajari keberadaan spesies di suatu daerah adalah Random Forest. Random Forest adalah metode yang menggunakan banyak pohon keputusan untuk memprediksi sesuatu berdasarkan data. Dalam hal ini, metode ini digunakan untuk mencari tahu hubungan antara kondisi lingkungan dengan adanya spesies tertentu.

Dengan bantuan metode ini, bisa dibuat sistem yang secara otomatis menebak spesies apa saja yang mungkin ada di suatu tempat.

1.2 Rumusan Masalah

Masalah utama yang akan dibahas dalam penelitian ini adalah:

1. Bagaimana cara mengembangkan model yang dapat memprediksi spesies tumbuhan yang kemungkinan ada di suatu lokasi berdasarkan data lingkungan yang tersedia?
2. Bagaimana cara memanfaatkan data lingkungan, seperti Data yang digunakan mencakup informasi seperti Presence-Absence (PA), Presence-Only (PO), serta data lingkungan terkait seperti citra satelit, data iklim, dan karakteristik tanah yang diperoleh dari dataset GLC25-PA-train-soilgrids dan GLC25-PO-train-soilgrids, untuk meningkatkan akurasi model prediksi distribusi spesies tumbuhan?

1.3 Tujuan

Tujuan dari penelitian ini adalah:

1. Mengembangkan model yang mampu memprediksi spesies tumbuhan yang kemungkinan ada di suatu lokasi berdasarkan data lingkungan yang tersedia, seperti citra satelit, data iklim, dan faktor lingkungan lainnya.
2. Mengeksplorasi metode yang paling efektif dalam mengolah dan memanfaatkan data lingkungan untuk meningkatkan akurasi prediksi distribusi spesies tumbuhan.

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 machine 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

Secara teknis, proyek ini bertujuan untuk membangun sebuah model machine learning yang mampu memprediksi spesies tumbuhan yang kemungkinan besar terdapat pada suatu lokasi geografis tertentu. Prediksi ini akan dilakukan dengan menggunakan empat jenis data utama yang disediakan oleh penyelenggara kompetisi, yaitu data Presence-Absence (PA), Presence-Only (PO), GLC25-PA-train-soilgrids, dan GLC25-PO-train-soilgrids.

Data PA digunakan sebagai dasar utama karena memuat informasi yang lebih lengkap mengenai kehadiran maupun ketidakhadiran spesies pada lokasi tertentu. Sementara itu, data PO akan dimanfaatkan sebagai sumber tambahan untuk memperkaya distribusi spasial dari spesies yang tercatat, meskipun hanya berisi informasi kehadiran.

GLC25-PA-train-soilgrids dan GLC25-PO-train-soilgrids akan digunakan untuk memperkaya model dengan informasi terkait kondisi lingkungan dan tanah di lokasi tertentu. Data GLC25 soilgrids memberikan informasi tentang karakteristik tanah ini, seperti kelembaban, pH, tekstur tanah (misalnya proporsi pasir, lempung, dan debu), serta kandungan unsur penting seperti nitrogen dan karbon organik, berpengaruh signifikan pada kemampuan suatu spesies untuk bertahan dan berkembang di lokasi tertentu.

Keempat data ini akan digabungkan (merge) berdasarkan surveyid dengan target species id, yang merupakan kunci utama untuk menghubungkan informasi antara data Presence-Absence (PA), Presence-Only (PO), GLC25-PA-train-soilgrids, dan GLC25-PO-train-soilgrids.. Penggabungan ini akan memungkinkan model untuk memanfaatkan informasi koordinat lokasi, waktu observasi (day-of-year), serta fitur lingkungan dan karakteristik tanah yang lebih komprehensif. Dengan menggabungkan informasi ini, model diharapkan mampu menangkap pola distribusi spesies secara lebih representatif.

Model yang dibangun akan dievaluasi menggunakan micro F1-score, yaitu metrik yang mengukur keseimbangan antara presisi dan recall pada tingkat sampel. Metrik ini akan digunakan untuk menilai seberapa tepat model dalam memprediksi kumpulan spesies yang benar-benar hadir di setiap lokasi survei.

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 :

Tabel 1 Timeline Pengerjaan Proyek

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 (Presence-Only, Presence-Absence, GLC25-PA-train-soilgrids, dan GLC25-PO-train-soilgrids).
    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 (random forests).
    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

Langkah awal dalam tahap persiapan data adalah memuat keempat dataset utama yang disediakan, yaitu:

a. GLC25_PA_metadata_train.csv (Presence-Absence Train),
b. GLC25_P0_metadata_train.csv (Presence-Only Train),
c. GLC25-PA-train-soilgrids.csv
d. GLC25-PO-train-soilgrids.csv

Pada tahap ini, data dari berbagai file CSV dimuat dan diperiksa secara awal untuk memastikan bahwa struktur fitur di setiap dataset serupa, serta untuk mempermudah pengunduhan data yang relevan.

```
import pandas as pd

# Load the datasets with the new names
PA_metadata_train_file = 'GLC25_PA_metadata_train.csv'
P0_metadata_train_file = 'GLC25_P0_metadata_train.csv'
PA_soil_train_file = 'GLC25-PA-train-soilgrids.csv'
P0_soil_train_file = 'GLC25-PO-train-soilgrids.csv'
```

```
PA_metadata_train = pd.read_csv(PA_metadata_train_file)
P0_metadata_train = pd.read_csv(P0_metadata_train_file)
PA_soil_train = pd.read_csv(PA_soil_train_file)
P0_soil_train = pd.read_csv(P0_soil_train_file)
```

```
PA_metadata_train_top500 = PA_metadata_train.head(500)
P0_metadata_train_top500 = P0_metadata_train.head(500)
PA_soil_train_top500 = PA_soil_train.head(500)
P0_soil_train_top500 = P0_soil_train.head(500)
```

```
PA_metadata_train_top500.to_csv('500_PA_metadata_train.csv', index=False)
P0_metadata_train_top500.to_csv('500_P0_metadata_train.csv', index=False)
PA_soil_train_top500.to_csv('500_PA_soil_train.csv', index=False)
P0_soil_train_top500.to_csv('500_P0_soil_train.csv', index=False)
```

```
from google.colab import files

# Upload the files directly using this method
files.download('500_PA_metadata_train.csv')
files.download('500_P0_metadata_train.csv')
files.download('500_PA_soil_train.csv')
files.download('500_P0_soil_train.csv')
```

3.2 Penelaahan Data

Pada tahap awal eksplorasi, dilakukan pemeriksaan terhadap empat dataset utama: PA_metadata_train_top500, PO_metadata_train_top500, PA_soil_train_top500, dan PO_soil_train_top500. Pemeriksaan mencakup struktur data dengan .info(), tampilan awal data dengan .head(), serta identifikasi nilai hilang menggunakan .isnull().sum(). Nilai kosong pada kolom numerik akan diisi dengan rata-rata, sementara data kategorikal ditandai untuk proses encoding pada tahap praproses selanjutnya. Langkah ini bertujuan memastikan data siap digunakan untuk analisis atau pemodelan.

```
# Menelaah Data

# Memeriksa struktur dataset
PA_metadata_train_top500.info()
P0_metadata_train_top500.info()
PA_soil_train_top500.info()
P0_soil_train_top500.info()

# Memeriksa beberapa baris pertama
PA_metadata_train_top500.head()
P0_metadata_train_top500.head()
PA_soil_train_top500.head()
P0_soil_train_top500.head()

# Memeriksa nilai yang hilang
PA_metadata_train_top500.isnull().sum()
P0_metadata_train_top500.isnull().sum()
PA_soil_train_top500.isnull().sum()
P0 soil train top500.isnull().sum()
```

BAB 4 DATA PREPARATION

4.1 Memilah Data

Pada tahap ini, data yang akan digunakan dalam analisis telah dipilih berdasarkan kriteria yang relevan dengan tujuan penelitian. Data yang digunakan dalam penelitian ini terdiri dari beberapa dataset yang telah ditentukan sebelumnya, termasuk dataset PA metadata, PA soil, P0 metadata, dan P0 soil. Setiap dataset mengandung informasi yang diperlukan untuk analisis, dan kolom-kolom yang tidak relevan atau tidak sesuai dengan tujuan penelitian akan dibuang. Proses pemilahan data ini bertujuan untuk memastikan hanya data yang relevan yang digunakan dalam analisis selanjutnya. Dengan memilih kolom-kolom yang penting dan membuang data yang tidak diperlukan, kualitas data dapat ditingkatkan, yang pada akhirnya mendukung akurasi hasil analisis.

Langkah pertama dalam pemilahan data adalah memeriksa nilai yang hilang pada setiap kolom untuk menentukan apakah ada kolom yang perlu dibersihkan atau diimputasi. Proses ini dilakukan untuk setiap dataset yang digunakan dalam analisis, termasuk PA metadata, PA soil, P0 metadata, dan P0 soil, untuk memastikan bahwa data yang dipilih sudah siap digunakan.

```
# Memeriksa nilai yang hilang pada setiap kolom
PA_metadata_train_top500_missing = PA_metadata_train_top500.isnull().sum()
P0_metadata_train_top500_missing = P0_metadata_train_top500.isnull().sum()
PA_soil_train_top500_missing = PA_soil_train_top500.isnull().sum()
P0_soil_train_top500_missing = P0_soil_train_top500.isnull().sum()

# Menampilkan jumlah nilai yang hilang per kolom
print("Nilai yang hilang pada PA_metadata_train_top500:")
print(PA_metadata_train_top500_missing)

print("Nilai yang hilang pada P0_metadata_train_top500:")
print(P0_metadata_train_top500_missing)

print("Nilai yang hilang pada PA_soil_train_top500:")
print(PA_soil_train_top500_missing)

print("Nilai yang hilang pada P0_soil_train_top500:")
print(P0_soil_train_top500_missing)
```

Output :

```
Nilai yang hilang pada PA_metadata_train_top500:
lon                   0
lat                   0
year                  0
geoUncertaintyInM    24
areaInM2             48
region                0
country               0
speciesId             0
surveyId              0
dtype: int64
Nilai yang hilang pada P0_metadata_train_top500:
publisher             0
year                  0
month                 0
day                   0
lat                   0
lon                   0
geoUncertaintyInM     0
taxonRank             0
date                  0
dayOfYear             0
speciesId             0
surveyId              0
dtype: int64
Nilai yang hilang pada PA_soil_train_top500:
surveyId              0
Soilgrid-bdod        47
Soilgrid-cec         43
Soilgrid-cfvo        43
Soilgrid-clay        43
Soilgrid-nitrogen    43
Soilgrid-phh2o       43
Soilgrid-sand        43
Soilgrid-silt        43
Soilgrid-soc         47
dtype: int64
Nilai yang hilang pada P0_soil_train_top500:
surveyId              0
Soilgrid-bdod        91
Soilgrid-cec         89
Soilgrid-cfvo        89
Soilgrid-clay        89
Soilgrid-nitrogen    89
Soilgrid-phh2o       89
Soilgrid-sand        89
Soilgrid-silt        89
Soilgrid-soc         91
dtype: int64
```

4.2 Membersihkan Data

Setelah memilah data, tahap selanjutnya adalah membersihkan data dengan menangani nilai yang hilang (missing values). Dalam penelitian ini, beberapa kolom memiliki nilai yang hilang yang perlu diatasi, baik dengan cara mengisi nilai tersebut menggunakan rata-rata (mean) atau median, atau dengan menghapusnya jika dianggap tidak relevan.

a. Mengatasi Missing Value pada PA metadata

   Pada dataset PA metadata, nilai yang hilang pada beberapa kolom diisi dengan menggunakan metode imputasi rata-rata (mean) atau median. Imputasi dengan rata-rata dan median menghindari penghapusan baris yang memiliki missing values, yang dapat mengurangi ukuran dataset dan mengurangi informasi yang tersedia untuk analisis. Metode ini juga efisien dan mudah diterapkan pada data numerik, terutama ketika missing values tersebar secara acak dan tidak tergantung pada pola tertentu.

```
# Menggunakan mean dan median untuk mengisi missing value pada PA metadata
PA_metadata_train_top500.loc[:, 'geoUncertaintyInM'] = PA_metadata_train_top500['geoUncertaintyInM'].fillna(PA_metadata_train_top500['geoUncertaintyInM'].mean())
PA_metadata_train_top500.loc[:, 'areaInM2'] = PA_metadata_train_top500['areaInM2'].fillna(PA_metadata_train_top500['areaInM2'].median())
```

b. Mengatasi Missing Value pada PA Soil

Untuk dataset PA soil, imputasi dengan rata-rata digunakan pada kolom seperti 'Soilgrid-bdod', 'Soilgrid-cec', dan kolom lainnya. Hal ini dilakukan karena kolom-kolom ini berisi data numerik yang dapat dipengaruhi oleh variasi alami, namun imputasi rata-rata akan mengisi nilai hilang dengan cara yang tidak mengganggu distribusi keseluruhan data. Ini membantu menjaga konsistensi data dan memungkinkan analisis yang lebih lanjut tanpa kehilangan informasi yang terlalu banyak.

```
# Mengatasi Nilai yang Hilang pada PA_soil_train_top500
PA_soil_train_top500 = PA_soil_train_top500.copy()

# Lanjutkan Imputasi
PA_soil_train_top500['Soilgrid-bdod'] = PA_soil_train_top500['Soilgrid-bdod'].fillna(PA_soil_train_top500['Soilgrid-bdod'].mean())
PA_soil_train_top500['Soilgrid-cec'] = PA_soil_train_top500['Soilgrid-cec'].fillna(PA_soil_train_top500['Soilgrid-cec'].mean())
PA_soil_train_top500['Soilgrid-cfvo'] = PA_soil_train_top500['Soilgrid-cfvo'].fillna(PA_soil_train_top500['Soilgrid-cfvo'].mean())
PA_soil_train_top500['Soilgrid-clay'] = PA_soil_train_top500['Soilgrid-clay'].fillna(PA_soil_train_top500['Soilgrid-clay'].mean())
PA_soil_train_top500['Soilgrid-nitrogen'] = PA_soil_train_top500['Soilgrid-nitrogen'].fillna(PA_soil_train_top500['Soilgrid-nitrogen'].mean())
PA_soil_train_top500['Soilgrid-phh2o'] = PA_soil_train_top500['Soilgrid-phh2o'].fillna(PA_soil_train_top500['Soilgrid-phh2o'].mean())
PA_soil_train_top500['Soilgrid-sand'] = PA_soil_train_top500['Soilgrid-sand'].fillna(PA_soil_train_top500['Soilgrid-sand'].mean())
PA_soil_train_top500['Soilgrid-silt'] = PA_soil_train_top500['Soilgrid-silt'].fillna(PA_soil_train_top500['Soilgrid-silt'].mean())
PA_soil_train_top500['Soilgrid-soc'] = PA_soil_train_top500['Soilgrid-soc'].fillna(PA_soil_train_top500['Soilgrid-soc'].mean())
```

c. Mengatasi Missing Value pada PO Soil

Pada dataset PO soil, imputasi dengan rata-rata dilakukan pada kolom-kolom seperti 'Soilgrid-bdod', 'Soilgrid-cec', dan lainnya. Proses ini dilakukan dengan cara yang sama seperti pada PA soil untuk memastikan kelancaran proses analisis.

```
# Mengatasi Nilai yang Hilang pada P0_soil_train_top500
P0_soil_train_top500_copy = P0_soil_train_top500.copy()

# Imputasi nilai yang hilang pada P0_soil_train_top500_copy dengan rata-rata
P0_soil_train_top500_copy['Soilgrid-bdod'] = P0_soil_train_top500_copy['Soilgrid-bdod'].fillna(P0_soil_train_top500_copy['Soilgrid-bdod'].mean())
P0_soil_train_top500_copy['Soilgrid-cec'] = P0_soil_train_top500_copy['Soilgrid-cec'].fillna(P0_soil_train_top500_copy['Soilgrid-cec'].mean())
P0_soil_train_top500_copy['Soilgrid-cfvo'] = P0_soil_train_top500_copy['Soilgrid-cfvo'].fillna(P0_soil_train_top500_copy['Soilgrid-cfvo'].mean())
P0_soil_train_top500_copy['Soilgrid-clay'] = P0_soil_train_top500_copy['Soilgrid-clay'].fillna(P0_soil_train_top500_copy['Soilgrid-clay'].mean())
P0_soil_train_top500_copy['Soilgrid-nitrogen'] = P0_soil_train_top500_copy['Soilgrid-nitrogen'].fillna(P0_soil_train_top500_copy['Soilgrid-nitrogen'].mean())
P0_soil_train_top500_copy['Soilgrid-phh2o'] = P0_soil_train_top500_copy['Soilgrid-phh2o'].fillna(P0_soil_train_top500_copy['Soilgrid-phh2o'].mean())
P0_soil_train_top500_copy['Soilgrid-sand'] = P0_soil_train_top500_copy['Soilgrid-sand'].fillna(P0_soil_train_top500_copy['Soilgrid-sand'].mean())
P0_soil_train_top500_copy['Soilgrid-silt'] = P0_soil_train_top500_copy['Soilgrid-silt'].fillna(P0_soil_train_top500_copy['Soilgrid-silt'].mean())
P0_soil_train_top500_copy['Soilgrid-soc'] = P0_soil_train_top500_copy['Soilgrid-soc'].fillna(P0_soil_train_top500_copy['Soilgrid-soc'].mean())
```

Seperti pada PA soil, imputasi dengan rata-rata digunakan pada PO soil untuk menghindari penghapusan data yang tidak lengkap. Imputasi ini juga mencegah adanya bias yang dapat muncul jika nilai yang hilang dibiarkan begitu saja atau dihapus. Menggunakan rata-rata untuk imputasi sangat berguna karena dapat diterapkan dengan mudah dan mengurangi kemungkinan kehilangan informasi penting dalam dataset.

4.3 Mengkonstruksi Data

Mengkonstruksi data berarti membentuk dataset yang lebih sesuai untuk analisis atau pelatihan model. Pada kasus ini, akan digabungkan beberapa dataset dan memilih kolom-kolom yang saling relevan antar dataset. Dataset digabungkan berdasarkan kolom yang memiliki kesamaan, seperti surveyId, untuk mengintegrasikan informasi yang relevan dari berbagai sumber data.

a. Menggabungkan PA metadata dan PA Soil

```
# Menggabungkan PA metadata dan PA soil berdasarkan surveyId
PA_metadata_selected = PA_metadata_train_top500[['lat', 'lon', 'geoUncertaintyInM', 'speciesId', 'surveyId']]
PA_soil_selected = PA_soil_train_top500[['surveyId', 'Soilgrid-bdod', 'Soilgrid-cec', 'Soilgrid-cfvo',
                                         'Soilgrid-clay', 'Soilgrid-nitrogen', 'Soilgrid-phh2o',
                                         'Soilgrid-sand', 'Soilgrid-silt', 'Soilgrid-soc']]

# Gabungkan PA metadata dengan PA soil berdasarkan surveyId
PA_merged = pd.merge(PA_metadata_selected, PA_soil_selected, on='surveyId', how='inner')

# Menampilkan hasil gabungan PA metadata dan PA soil
print(PA_merged.head())
# Langkah 3: Menyimpan hasil gabungan ke file CSV
PA_merged.to_csv('PA_merged.csv', index=False)
print("Data yang telah digabungkan telah disimpan.")
```

Untuk mengkonstruksi data, yang dilakukan adalah menggabungkan dua dataset PA, yaitu PA_metadata_train_top500 dan PA_soil_train_top500, berdasarkan kolom surveyId yang ada di kedua dataset tersebut. Langkah pertama memilih kolom-kolom yang relevan dari masing-masing dataset: pada dataset PA_metadata_train_top500, dipilih kolom lat, lon, geoUncertaintyInM, speciesId, dan surveyId, sedangkan pada dataset PA_soil_train_top500, dipilih kolom surveyId, serta beberapa kolom yang berkaitan dengan data tanah seperti Soilgrid-bdod, Soilgrid-cec, Soilgrid-cfvo, dan lainnya.

Kemudian, kedua dataset ini digabungkan menggunakan fungsi pd.merge() dengan parameter on='surveyId', yang berarti penggabungan dilakukan berdasarkan kolom surveyId. Metode yang digunakan adalah how='inner', yang artinya hanya baris yang memiliki kecocokan pada kolom surveyId di kedua dataset yang akan disertakan dalam hasil gabungan.

b. Menggabungkan P0 Metadata dan P0 Soil

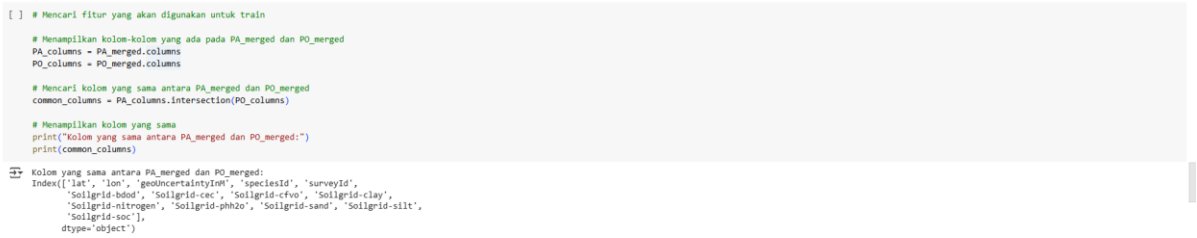
```
[ ] # Menggabungkan PO metadata dan PO soil berdasarkan surveyId
    P0_metadata_selected = P0_metadata_train_top500[['lat', 'lon', 'geoUncertaintyInM', 'speciesId', 'surveyId']]
    P0_soil_selected = P0_soil_train_top500[['surveyId', 'Soilgrid-bdod', 'Soilgrid-cec', 'Soilgrid-cfvo',
                                             'Soilgrid-clay', 'Soilgrid-nitrogen', 'Soilgrid-phh2o',
                                             'Soilgrid-sand', 'Soilgrid-silt', 'Soilgrid-soc']]

    # Gabungkan PO metadata dengan PO soil berdasarkan surveyId
    PO_merged = pd.merge(P0_metadata_selected, P0_soil_selected, on='surveyId', how='inner')

    # Menampilkan hasil gabungan PO metadata dan PO soil
    print(PO_merged.head())

    # Langkah 3: Menyimpan hasil gabungan ke file CSV
    PO_merged.to_csv('PO_merged.csv', index=False)
    print("Data yang telah digabungkan telah disimpan.")
```

Sama dengan data PA sebelumnya kali ini hal yang dilakukan adalah menggabungkan dua dataset, yaitu P0 metadata dan P0 soil, berdasarkan kolom surveyId yang ada pada kedua dataset tersebut. Langkah pertama adalah memilih kolom-kolom yang relevan dari masing-masing dataset. Pada dataset P0 metadata, dipilih kolom lat, lon, geoUncertaintyInM, speciesId, dan surveyId, sedangkan pada dataset P0 soil, dipilih kolom-kolom terkait data tanah, seperti Soilgrid-bdod, Soilgrid-cec, Soilgrid-cfvo, dan lainnya.

Proses penggabungan dilakukan dengan menggunakan fungsi pd.merge(), yang menggabungkan kedua dataset berdasarkan kolom surveyId menggunakan metode inner join, artinya hanya baris yang memiliki kecocokan pada surveyId dari kedua dataset yang akan disertakan dalam hasil gabungan.

4.4 Menentukan Label Data

Pada tahap ini, dilakukan pencarian terhadap kolom-kolom yang sama antara kedua dataset yang telah digabungkan, yaitu PA_merged dan PO_merged. Kolom yang sama antara kedua dataset ini akan menjadi kandidat fitur yang dapat digunakan untuk pelatihan model, karena kedua dataset memiliki struktur yang serupa dan data yang relevan. Dengan menggunakan fungsi intersection(), kolom yang ada di kedua dataset akan diidentifikasi, sehingga dapat dipilih kolom-kolom yang akan menjadi fitur untuk analisis lebih lanjut.



```
[ ] # Mencari fitur yang akan digunakan untuk train

    # Menampilkan kolom-kolom yang ada pada PA_merged dan PO_merged
    PA_columns = PA_merged.columns
    PO_columns = PO_merged.columns

    # Mencari kolom yang sama antara PA_merged dan PO_merged
    common_columns = PA_columns.intersection(PO_columns)

    # Menampilkan kolom yang sama
    print("Kolom yang sama antara PA_merged dan PO_merged:")
    print(common_columns)

→ Kolom yang sama antara PA_merged dan PO_merged:
  Index(['lat', 'lon', 'geoUncertaintyInM', 'speciesId', 'surveyId',
         'Soilgrid-bdod', 'Soilgrid-cec', 'Soilgrid-cfvo', 'Soilgrid-clay',
         'Soilgrid-nitrogen', 'Soilgrid-phh2o', 'Soilgrid-sand', 'Soilgrid-silt',
         'Soilgrid-soc'],
        dtype='object')
```

Setelah kolom yang relevan ditemukan, langkah berikutnya adalah mempersiapkan data ini untuk proses pelatihan model, di mana fitur-fitur ini akan digunakan untuk memprediksi label atau variabel target yang sudah ditentukan sebelumnya.

BAB 5 MODELING

5.1 Membangun Scenario Pengujian

Pada bagian ini, dijelaskan mengenai skenario pengujian yang digunakan untuk mengevaluasi performa model. Input yang digunakan adalah dataset final_merged_data, dengan fitur yang dipilih adalah semua kolom kecuali speciesId dan surveyId. Kolom speciesId menjadi target yang diprediksi, sementara surveyId digunakan untuk mengelompokkan hasil prediksi. Data dibagi menggunakan teknik StratifiedKFold yang menyesuaikan jumlah split berdasarkan ukuran kelas terkecil untuk memastikan pembagian data yang seimbang antara kelas-kelas yang ada.

Model yang digunakan adalah RandomForestClassifier, yang dioptimalkan melalui pencarian hyperparameter menggunakan GridSearchCV untuk menemukan kombinasi parameter terbaik. Proses tuning dilakukan dengan mengatur parameter seperti jumlah pohon (n_estimators), kedalaman pohon (max_depth), dan jumlah sampel minimum pada pembagian dan daun (min_samples_split dan min_samples_leaf). Setelah model dilatih, evaluasi dilakukan dengan menghitung accuracy dan F1-score, yang menunjukkan performa model.

Hasil dari evaluasi ini menunjukkan accuracy 86.7% dan F1-score 0.854, yang mencerminkan kinerja model yang baik. Selain itu, dilakukan juga analisis feature importance untuk mengidentifikasi fitur-fitur yang paling berpengaruh dalam proses prediksi. Hasil prediksi model kemudian disimpan dalam file CSV, yang mencakup surveyId dan predictions, untuk keperluan analisis lebih lanjut atau penggunaan di tahap berikutnya.

5.2 Membangun model

```
# Memeriksa apakah ada nilai kosong (NaN) di seluruh dataframe
print(final_merged_data.isnull().sum())
```

```
lat                  0
lon                  0
geoUncertaintyInM    0
speciesId            0
surveyId             0
Soilgrid-bdod       89
Soilgrid-cec        81
Soilgrid-cfvo       81
Soilgrid-clay       81
Soilgrid-nitrogen   81
Soilgrid-phh2o      81
Soilgrid-sand       81
Soilgrid-silt       81
Soilgrid-soc        89
dtype: int64
```

Pada tahap ini, dilakukan pemeriksaan terhadap nilai kosong atau NaN (Not a Number) dalam DataFrame yang telah digabungkan, yaitu final_merged_data. Fungsi isnull() digunakan untuk

memeriksa setiap elemen dalam DataFrame apakah memiliki nilai kosong, kemudian fungsi sum() menghitung jumlah nilai NaN pada setiap kolom. Hasil dari pemeriksaan ini menunjukkan bahwa kolom lat, lon, geoUncertaintyInM, speciesId, dan surveyId tidak memiliki nilai kosong (NaN = 0). Sementara itu, kolom Soilgrid-bbod, Soilgrid-cec, dan Soilgrid-soc masing-masing memiliki 89 nilai kosong, sedangkan kolom Soilgrid-cfvo, Soilgrid-clay, Soilgrid-nitrogen, Soilgrid-phh2o, Soilgrid-sand, dan Soilgrid-silt masing-masing memiliki 81 nilai kosong. Proses ini penting untuk mendeteksi data yang hilang atau tidak lengkap, yang selanjutnya dapat ditangani dengan berbagai metode, seperti mengisi nilai yang hilang dengan rata-rata (mean).

```python
import pandas as pd
from sklearn.impute import SimpleImputer

# Memastikan DataFrame Anda telah dibuat
# final_merged_data sudah ada berdasarkan kode yang Anda berikan sebelumnya

# Membuat SimpleImputer dengan strategi 'mean' untuk mengisi missing value
imputer = SimpleImputer(strategy='mean')

# Mengimputasi/mengisi missing value pada seluruh DataFrame
final_merged_data_imputed = pd.DataFrame(imputer.fit_transform(final_merged_data), columns=final_merged_data.columns)

# Menampilkan beberapa baris pertama dari hasil imputasi
print(final_merged_data_imputed.head())

# Menyimpan data yang sudah diimputasi ke dalam file CSV
final_merged_data_imputed.to_csv('final_merged_data_imputed.csv', index=False)
print("Data yang telah diimputasi dan disimpan sebagai final_merged_data_imputed.csv.")
```

```
        lat       lon  geoUncertaintyInM  speciesId  surveyId  Soilgrid-bdod  \
0  43.74605  1.573057                6.0     3383.0       1.0          145.0
1  42.12559  0.314948                5.0     1152.0       2.0          134.0
2  48.29520 -0.934518               24.9     6772.0       3.0          127.0
3  53.63367 -2.644535                8.0     3318.0       4.0          103.0
4  49.79471  7.925086               15.0     3374.0       5.0          138.0

   Soilgrid-cec  Soilgrid-cfvo  Soilgrid-clay  Soilgrid-nitrogen  \
0         212.0          130.0          303.0              134.0
1         213.0          189.0          302.0              184.0
2         211.0          109.0          187.0              257.0
3         212.0          123.0          216.0              466.0
4         195.0          131.0          310.0              198.0

   Soilgrid-phh2o  Soilgrid-sand  Soilgrid-silt  Soilgrid-soc
0            73.0          306.0          389.0         122.0
1            74.0          294.0          403.0         223.0
2            59.0          132.0          681.0         301.0
3            56.0          429.0          355.0        1044.0
4            73.0          208.0          480.0         208.0
Data yang telah diimputasi dan disimpan sebagai final_merged_data_imputed.csv.
```

Pada tahap ini, dilakukan imputasi untuk mengisi nilai kosong (NaN) dalam data yang telah digabungkan menggunakan metode pengisian dengan rata-rata (mean) untuk setiap kolom. Proses ini bertujuan untuk memastikan bahwa data yang hilang dapat diisi dengan nilai

sehingga dataset menjadi lengkap dan siap digunakan untuk analisis lebih lanjut.

```python
import pandas as pd
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import StratifiedKFold, GridSearchCV
from sklearn.metrics import accuracy_score, f1_score
import matplotlib.pyplot as plt

# Load dataset
train_data = pd.read_csv('final_merged_data_imputed.csv')

# Prepare features and target
X = train_data.drop(columns=['speciesId', 'surveyId'])
y = train_data['speciesId']

# Remove classes with only one sample (optional)
train_data_filtered = train_data[train_data['speciesId'].isin(y.value_counts()[y.value_counts() > 1].index)]

# Prepare filtered data
X_filtered = train_data_filtered.drop(columns=['speciesId', 'surveyId'])
y_filtered = train_data_filtered['speciesId']

# Use StratifiedKFold with the adjusted n_splits based on the smallest class size
n_splits = min(5, y_filtered.value_counts().min())
sss = StratifiedKFold(n_splits=n_splits, shuffle=True, random_state=42)

# Initialize RandomForest model
model = RandomForestClassifier(random_state=42)

# Hyperparameter grid for tuning
param_grid = {
    'n_estimators': [100, 200, 500],
    'max_depth': [None, 10, 20, 30],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4],
    'max_features': ['sqrt', 'log2']
}
```

```python
# Perform GridSearchCV
grid_search = GridSearchCV(model, param_grid, cv=sss, n_jobs=-1, verbose=2)
grid_search.fit(X_filtered, y_filtered)

# Best model
best_model = grid_search.best_estimator_

# Train the best model
best_model.fit(X_filtered, y_filtered)

# Predictions and evaluation
y_pred = best_model.predict(X_filtered)

# Calculate accuracy
accuracy = accuracy_score(y_filtered, y_pred)
print(f"Accuracy with tuned model: {accuracy:.3f}")

# Feature importance plot
feature_importances = best_model.feature_importances_
plt.figure(figsize=(10, 6))
plt.barh(X_filtered.columns, feature_importances)
plt.xlabel("Feature Importance")
plt.title("Feature Importance from RandomForest")
plt.show()

# Create submission file
output = pd.DataFrame({'surveyId': train_data_filtered['surveyId'], 'predictions': y_pred})
output['predictions'] = output.groupby('surveyId')['predictions'].transform(lambda x: ' '.join(map(str, sorted(x))))
output = output.drop_duplicates(subset=['surveyId'])

# Save the result as CSV
output[['surveyId', 'predictions']].to_csv('submission.csv', index=False)
print("Prediction file saved as submission.csv")
```

```
Fitting 2 folds for each of 216 candidates, totalling 432 fits
Accuracy with tuned model: 0.867
```

Pada tahap ini, digunakan RandomForestClassifier yang dioptimalkan dengan bantuan GridSearchCV untuk menemukan kombinasi parameter terbaik. Proses ini mencakup pencarian nilai terbaik untuk beberapa parameter, seperti jumlah pohon dalam hutan

(n_estimators), kedalaman maksimum pohon (max_depth), serta jumlah sampel minimum untuk pembagian dan daun (min_samples_split dan min_samples_leaf). Setelah dilakukan pengaturan, model yang dihasilkan memiliki akurasi sebesar 86.7%, yang menunjukkan hasil yang cukup baik dalam memprediksi data.

Selain mengukur akurasi, dilakukan juga analisis terhadap feature importance, yang menunjukkan fitur mana yang paling mempengaruhi hasil prediksi. Hasil dari analisis ini membantu dalam memahami faktor-faktor penting yang digunakan oleh model dalam proses pengambilan keputusan.

Setelah model selesai dilatih dan dioptimalkan, prediksi dilakukan pada data yang sudah disaring, dan hasilnya disimpan dalam file CSV. File ini berisi surveyId dan predictions, di mana setiap surveyId memiliki hasil prediksi yang sesuai.

5.3 Evaluasi

F1 Score adalah salah satu metrik evaluasi yang digunakan untuk mengukur kinerja model klasifikasi, terutama ketika dataset memiliki distribusi kelas yang tidak seimbang. F1 Score menggabungkan dua metrik penting, yaitu Precision dan Recall, untuk memberikan gambaran yang lebih komprehensif tentang kemampuan model dalam memprediksi kelas positif.

```
from sklearn.metrics import f1_score

# Predictions and evaluation
y_pred =best_model.predict(X_filtered)

# Evaluasi dengan F1 Score (weighted average untuk seluruh kelas)
f1 = f1_score(y_filtered, y_pred, average='weighted')
print(f"F1 Score (weighted average): {f1:.3f}")
```

```
F1 Score (weighted average): 0.854
```

Untuk menghitung nilai F1 Score menggunakan metode f1_score dari pustaka sklearn.metrics., kita pertama-tama melakukan prediksi menggunakan model terbaik yang telah dilatih (best_model.predict(X_filtered)) pada data yang telah difilter (X_filtered). F1 Score di sini dihitung dengan parameter average='weighted', yang berarti F1 Score dihitung untuk setiap kelas, kemudian dihitung rata-rata berbobot berdasarkan jumlah data yang ada pada masing-masing kelas. Ini memberikan penekanan lebih pada kelas yang lebih banyak muncul dalam dataset, sehingga metrik ini cocok untuk digunakan pada masalah klasifikasi dengan kelas yang tidak seimbang.

a. y_pred = best_model.predict(X_filtered), digunakan untuk melakukan prediksi menggunakan model terbaik.

b. f1 = f1_score(y_filtered, y_pred, average='weighted'), digunakan untuk menghitung F1-score berbobot untuk hasil prediksi.
c. print(f"Weighted F1-score: {f1:.3f}"),untuk menampilkan F1-score berbobot yang dihitung dengan tiga angka desimal.

Setelah perhitungan F1 Score selesai, hasilnya dicetak dengan format tiga angka di belakang koma menggunakan print. Ini memberikan gambaran mengenai seberapa baik model mengklasifikasikan data, dengan mempertimbangkan baik precision maupun recall pada setiap kelas. Nilai yang diperoleh dari perhitungan F1 Score adalah 0,854, yang menunjukkan kinerja model yang cukup baik dalam memprediksi spesies tumbuhan di lokasi tertentu.

BAB 6 DEPLOYMENT

Deployment merupakan proses penting dalam implementasi model machine learning ke dalam sistem operasional, sehingga model tersebut dapat digunakan secara langsung untuk menghasilkan prediksi dalam konteks dunia nyata. Pada proyek ini, deployment dilakukan untuk memastikan model yang telah dilatih dapat dimanfaatkan secara optimal dalam memprediksi spesies tumbuhan yang kemungkinan besar terdapat pada suatu lokasi geografis tertentu.

Beberapa alasan utama pentingnya tahap deployment antara lain:

1. Efektivitas implementasi: Deployment memungkinkan model yang telah dikembangkan untuk diterapkan secara langsung dalam sistem prediksi, sehingga pengguna dapat memperoleh hasil secara real-time.
2. Maksimalisasi nilai model: Dengan mengintegrasikan model ke dalam sistem dan menyalurkan hasil prediksi ke komponen lain, nilai praktis dari model dapat dimaksimalkan dalam pengambilan keputusan atau aplikasi nyata.

Namun demikian, proses deployment model juga menghadapi sejumlah tantangan:

1. Tantangan Umum pada Perangkat Lunak:
    - Reliability (keandalan): Menjamin bahwa model dapat berjalan dengan stabil tanpa gangguan.
    - Reusability (dapat digunakan ulang): Memungkinkan penerapan kembali model untuk skenario serupa di masa depan.
    - Maintainability (kemudahan pemeliharaan): Memfasilitasi pemeliharaan dan pembaruan model secara efisien.
    - Flexibility (fleksibilitas): Mengakomodasi perubahan kebutuhan atau integrasi dengan sistem yang berbeda.
2. Tantangan Spesifik pada Machine Learning:
    - Reproducibility (reproduksibilitas): Menjamin bahwa hasil prediksi tetap konsisten ketika model dijalankan di lingkungan yang berbeda.

Proses prediksi dalam deployment model ini dilakukan dengan memanfaatkan empat jenis data utama:

1. Presence-Absence (PA): Data ini menjadi fondasi utama karena menyediakan informasi tentang kehadiran maupun ketidakhadiran spesies di lokasi tertentu.
2. Presence-Only (PO): Digunakan sebagai pelengkap untuk memperkaya informasi distribusi spasial spesies, meskipun hanya mencakup data kehadiran.

3. GLC25-PA-train-soilgrids dan GLC25-PO-train-soilgrids: Menyediakan fitur lingkungan dan karakteristik tanah, seperti pH, kelembapan, serta kandungan nitrogen dan karbon organik, yang sangat berpengaruh terhadap distribusi spesies.

Keempat dataset ini digabungkan berdasarkan survey_id, dengan species_id sebagai target prediksi. Proses penggabungan ini memungkinkan model untuk memanfaatkan informasi lokasi, waktu observasi, serta kondisi lingkungan dan tanah secara menyeluruh. Dengan integrasi ini, model dapat mengenali pola distribusi spesies secara lebih akurat.

Model yang dibangun dievaluasi menggunakan metrik micro F1-score, yaitu ukuran kinerja yang mempertimbangkan keseimbangan antara precision dan recall di seluruh sampel. Penggunaan metrik ini bertujuan untuk menilai seberapa tepat model dalam memprediksi kumpulan spesies yang benar-benar hadir di tiap lokasi survei.

Melalui proses deployment, model ini tidak hanya digunakan sebagai prototipe analisis, tetapi juga telah diintegrasikan ke dalam sistem yang mampu memberikan prediksi secara langsung, memperkuat aplikasi praktis dari penelitian ini dalam konservasi dan studi keanekaragaman hayati.

Berikut gambar model yang telah berhasil di deployment, yang dimana masih memiliki banyak kekurangan dalam pencocokan speciesId tumbuhan tersebut.

Prediksi Species Tanaman Berdasarkan Data Lingkungan

Latitude:

45.08378

Longitude:

9.882871

Geo Uncertainty (Meter):

31

Survey ID:

19

Soil Bulk Density (bdod):

130

Cation Exchange Capacity (cec):

220

Coarse Fragments Volume (cfvo):

119

Kadar Clay (%):

358

Kadar Nitrogen (%):

300

pH H_2O:

64

Kadar Pasir (%):

206

Kadar Debu (silt) (%):

434

Kandungan Karbon Organik (% SOC):

401

Prediksi Species

Hasil Prediksi

ID Survei: 19

Species ID yang diprediksi: **5749.0**

BAB 7 KESIMPULAN

Proyek ini mengembangkan model machine learning menggunakan metode Random Forest untuk memprediksi spesies tumbuhan yang kemungkinan ada di suatu lokasi berdasarkan data lingkungan. Data yang digunakan meliputi data Presence-Absence (PA), Presence-Only (PO), serta informasi tentang kondisi tanah dan lingkungan dari dataset GLC25-PA-train-soilgrids dan GLC25-PO-train-soilgrids.

Proses pembuatan model melibatkan beberapa tahap, yaitu persiapan data, pemilahan dan pembersihan data, penggabungan dataset, dan pemilihan fitur yang relevan. Model yang dibangun dioptimalkan menggunakan GridSearchCV untuk mencari hyperparameter terbaik, dan hasil evaluasi menunjukkan akurasi 86,7% serta F1-score 0,854.

Analisis fitur-fitur yang berpengaruh dalam proses prediksi juga dilakukan. Setelah model selesai, deployment dilakukan untuk memprediksi spesies tumbuhan di lokasi tertentu secara real-time, yang berguna untuk konservasi dan pelestarian keanekaragaman hayati. Meskipun model ini sudah cukup baik, masih ada tantangan dalam pencocokan speciesId yang perlu diperbaiki.

Secara keseluruhan, proyek ini menunjukkan bahwa machine learning dapat diterapkan untuk membantu dalam melestarikan keanekaragaman hayati, terutama dalam memprediksi distribusi spesies tumbuhan di suatu daerah.