Data Preparation

Tujuan Kode:

Kode ini bertujuan untuk mengolah data AIS, mengidentifikasi kapal yang bergerak dan stasioner di dalam area tertentu, dan menetapkan flag pergerakan kapal berdasarkan data yang telah difilter dan digabungkan dengan informasi area hexagonal.

```python
import json
from pyspark.sql import functions as F
from ais import functions as af
from aiski import functions as afki
```

json: Digunakan untuk memuat file JSON.

pyspark.sql.functions: Mengimpor fungsi-fungsi yang digunakan dalam operasi Spark DataFrame.

ais.functions dan aiski.functions: Mengimpor fungsi-fungsi khusus yang digunakan dalam proyek AIS, meskipun modul ini tidak umum, mereka mungkin merupakan bagian dari library khusus untuk proyek AIS.

```python
aoi_json = json.load(open('port.json'))
naoi_json = json.load(open('buffer.json'))
```

Memuat dua file JSON yang berisi informasi tentang Area of Interest (AOI) dan buffer area.

```python
hex_df = af.polygon_to_hex_df([('aoi', aoi_json), ('nonaoi', naoi_json)], 8) \
    .drop_duplicates(subset=['hex_ids']) \
    .rename(columns={'hex_ids': 'H3_int_index_8', 'polygon_name': 'area'})
```

Mengonversi poligon dari file JSON menjadi DataFrame dengan indeks hexagonal H3 pada resolusi 8.

Menghapus duplikat berdasarkan hex_ids.

Mengganti nama kolom hex_ids menjadi H3_int_index_8 dan polygon_name menjadi area.

```python
hex_sdf = spark.createDataFrame(hex_df)
```

Mengonversi DataFrame Pandas hex_df menjadi DataFrame Spark hex_sdf.

```python
sdf = af.get_ais(spark, start_date='2023-01-01', end_date='2023-01-31') \
    .filter(F.col('mmsi').between(201000000, 799000000))
```

Mendapatkan data AIS dari 1 Januari 2023 hingga 31 Januari 2023.

Memfilter data berdasarkan kolom mmsi (Maritime Mobile Service Identity) yang berada di antara 201000000 dan 799000000.

```python
aoi_naoi_sdf = sdf.join(hex_sdf, how='inner', on='H3_int_index_8') \
    .withColumn('stationary', F.when(F.col('sog') < 1, F.lit('Y')).otherwise(F.lit('N'
```

Menggabungkan DataFrame sdf dan hex_sdf berdasarkan kolom H3_int_index_8 dengan join tipe inner.

Menambahkan kolom baru stationary yang berisi 'Y' jika sog (speed over ground) kurang dari 1, dan 'N' jika sebaliknya.

```python
movement_flag_sdf = afki.assign_movement(
    sdf=aoi_naoi_sdf,
    ship_identifier_cols=['mmsi'],
    movement_orderby_cols=['dt_pos_utc'],
    movement_groupby_cols=['area'])
```

Menggunakan fungsi assign_movement dari modul aiski untuk menetapkan flag pergerakan kapal.

Fungsi ini menggunakan DataFrame aoi_naoi_sdf dan mengidentifikasi kapal berdasarkan kolom mmsi, mengurutkan pergerakan berdasarkan dt_pos_utc, dan mengelompokkan berdasarkan area.

Creating Polygon

Tujuan Kode:

Tujuan dari kode ini adalah untuk mengelompokkan kapal yang stasioner di dalam area yang diminati (AOI) berdasarkan posisi mereka menggunakan algoritma DBSCAN. Parameter untuk DBSCAN dihitung secara dinamis berdasarkan distribusi data kapal. Hasilnya adalah kluster kapal stasioner yang dapat dianalisis lebih lanjut.

```python
import h3.api.numpy_int as h3int
from sklearn.cluster import DBSCAN
from pyproj import Transformer
```

h3.api.numpy_int: Library untuk bekerja dengan indeks hexagonal H3.

sklearn.cluster.DBSCAN: Algoritma clustering DBSCAN dari scikit-learn.

pyproj.Transformer: Digunakan untuk transformasi koordinat geografis.

```python
def get_parameters(df, cluster_res, per):
    eps_m = round(h3int.edge_length(cluster_res) * 1000)
    df['h3_cluster_index'] = df['H3_int_index_12'].apply(lambda x: h3int.h3_to_parent(
    df_cluster = df.groupby('h3_cluster_index')['count_mmsi'].sum().reset_index().sort
    df_cluster['cumsum'] = df_cluster['count_mmsi'].cumsum()
    tot = df_cluster['count_mmsi'].sum()
    df_cluster['cumper'] = df_cluster['cumsum'] / tot
    min_points = df_cluster[df_cluster['cumper'] >= per].iloc[0]['count_mmsi']
    return eps_m, int(min_points)
```

eps_m: Jarak maksimum antar titik dalam satu kluster, dihitung berdasarkan panjang sisi hexagonal H3 pada resolusi yang diberikan.

h3_cluster_index: Indeks cluster H3 pada resolusi yang lebih rendah (coarser resolution).

df_cluster: DataFrame yang mengelompokkan data berdasarkan h3_cluster_index dan menjumlahkan count_mmsi.

min_points: Minimum jumlah titik dalam satu kluster berdasarkan persentase kumulatif per.

```python
sdf_agg = aoi_naoi_sdf.filter((F.col('area') == 'aoi') & (F.col('stationary') == 'Y'))
df = sdf_agg.toPandas()
```

Memfilter data kapal yang berada di area of interest (aoi) dan stasioner (stationary = 'Y').

Mengelompokkan data berdasarkan H3_int_index_12 dan menghitung jumlah kapal (count_mmsi).

Mengonversi DataFrame Spark ke Pandas DataFrame untuk pemrosesan lebih lanjut.

```python
eps_m, min_points = get_parameters(df, cluster_res=8, per=0.90)
```

Menghitung parameter eps (radius maksimum untuk satu kluster) dan min_samples (jumlah minimum titik untuk membentuk kluster) untuk algoritma DBSCAN.

```python
model = DBSCAN(eps=eps_m, min_samples=min_points)
points = df['H3_int_index_12'].apply(lambda x: h3int.h3_to_geo(x))
df['latitude'], df['longitude'] = zip(*points)
transf = Transformer.from_crs(4326, epsg_to=3832, always_xy=True)
(df['long_t'], df['lat_t']) = transf.transform(df['longitude'].values, df['latitude'].
X = df[['long_t', 'lat_t']].values
sample_weight = df['count_mmsi'].values
model_fit = model.fit(X, sample_weight=sample_weight)
```

model: Membuat model DBSCAN dengan parameter eps_m dan min_points.

points: Mengonversi indeks H3 ke koordinat geografis (latitude dan longitude).

transf: Mengonversi koordinat geografis dari CRS EPSG:4326 ke EPSG:3832 (sistem koordinat yang digunakan untuk clustering).

X: Array koordinat tertransformasi untuk clustering.

sample_weight: Bobot sampel berdasarkan jumlah kapal (count_mmsi).

model_fit: Menjalankan algoritma DBSCAN pada data tertransformasi dengan bobot sampel.

Penerapan

Epsilon (eps):

Rekomendasi: Gunakan panjang sisi rata-rata dari resolusi H3 sebagai nilai epsilon (eps).

Penjelasan: Dalam DBSCAN, eps adalah jarak maksimum antara dua titik agar mereka dianggap sebagai bagian dari cluster yang sama. Rekomendasi ini menyarankan agar jarak ini diatur berdasarkan panjang sisi rata-rata dari hexagon pada resolusi H3 tertentu. Setiap resolusi H3 memiliki hexagon dengan ukuran tertentu, dan panjang sisi adalah proksi yang baik untuk jarak antara titik-titik dalam cluster yang sama.

MinPts (Minimum Points):

Rekomendasi: Gunakan frekuensi dari indeks H3 dengan jumlah titik lokasi terendah tetapi merupakan bagian dari indeks H3 teratas yang mencakup persentase tinggi dari total frekuensi.

Penjelasan: MinPts adalah jumlah minimum titik yang diperlukan untuk membentuk wilayah padat (cluster). Rekomendasi ini menyarankan untuk mengatur MinPts berdasarkan indeks H3 yang memiliki titik paling sedikit tetapi masih termasuk dalam indeks teratas yang mencakup sebagian besar data. Artinya, Anda harus melihat distribusi titik-titik di seluruh hexagon H3 dan mengidentifikasi hexagon H3 yang secara kolektif mencakup sebagian besar titik. Di antara hexagon tersebut, pilih yang memiliki titik paling sedikit sebagai ambang MinPts. Ini membantu memastikan bahwa cluster terbentuk di wilayah dengan kepadatan data yang signifikan.

Contoh Skenario

Bayangkan Anda memiliki data geospasial dan telah mengindeksnya menggunakan sistem H3 pada resolusi 8. Berikut cara memilih parameter DBSCAN berdasarkan rekomendasi:

Memilih Epsilon (eps):

Hitung panjang sisi rata-rata dari hexagon H3 pada resolusi 8.

Gunakan panjang sisi rata-rata ini (dikonversi ke meter jika perlu) sebagai nilai eps.

Memilih MinPts:

Hitung jumlah titik data di setiap hexagon H3.

Urutkan hexagon-hexagon ini berdasarkan jumlah titik yang mereka miliki.

Tentukan ambang batas (misalnya 90%) yang mencakup sebagian besar titik data Anda.

Dari hexagon teratas ini, identifikasi yang memiliki titik paling sedikit dan tetapkan jumlah ini sebagai MinPts.