Tugas 5: Tugas Mandiri 5 – DBSCAN Clustering

Amaya Eshia - 0110224102[*]

[1] Teknik Informatika, STT Terpadu Nurul Fikri, Depok

[*]E-mail: name@institution.edu – **0110224102@student.nurulfikri.ac.id**

Abstract. Penelitian ini bertujuan untuk mengimplementasikan algoritma DBSCAN (Density-Based Spatial Clustering of Applications with Noise) dalam menganalisis pola distribusi spasial listing Airbnb di New York City tahun 2019. Dataset yang digunakan adalah AB_NYC_2019.csv yang memuat informasi geografis berupa latitude dan longitude dari setiap listing properti. Metodologi penelitian diawali dengan preprocessing data yang mencakup pembersihan data dan filtering harga untuk mengeliminasi outlier ekstrem. Visualisasi awal dilakukan menggunakan heatmap berbasis Folium untuk mengidentifikasi area dengan kepadatan tinggi. Penentuan parameter epsilon optimal dilakukan melalui metode Elbow dengan K-Distance Graph menggunakan algoritma Nearest Neighbors dan metrik Haversine yang sesuai untuk data koordinat geografis. Model DBSCAN diimplementasikan dengan parameter eps=0.2 km (radius cluster 200 meter) dan min_samples=20, menggunakan algoritma ball_tree untuk efisiensi komputasi. Evaluasi model dilakukan menggunakan Silhouette Score untuk mengukur kualitas clustering, dengan pengambilan sampel untuk optimalisasi waktu komputasi. Hasil clustering divisualisasikan melalui scatter plot yang membedakan cluster valid dengan noise points. Penelitian ini berhasil mengidentifikasi area-area dengan konsentrasi listing Airbnb yang tinggi di wilayah New York City, memberikan insight tentang pola distribusi geografis properti sewa jangka pendek.

Kata Kunci: DBSCAN, Clustering, Spatial Analysis, Airbnb, Haversine Distance, K-Distance Graph, Silhouette Score, Geospatial Data.

1. Import Library

```
[1]
✓ 3 d
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.cluster import DBSCAN

[2]
✓ 36 d
from google.colab import drive
drive.mount('/content/gdrive')

Mounted at /content/gdrive

[3]
✓ 0 d
path = "/content/gdrive/MyDrive/Praktikum Machine Learning_Amaya Eshia_0110224102_Ai02/Praktikum 1
```

Tahap pertama adalah mengimpor seluruh library yang dibutuhkan untuk proses clustering dan visualisasi data. Library pandas dan numpy digunakan untuk manipulasi dan komputasi data. Matplotlib dan seaborn berfungsi untuk visualisasi statistik. Algoritma DBSCAN diimpor dari scikit-learn sebagai metode clustering utama. Selain itu, dilakukan mounting Google Drive untuk mengakses dataset yang tersimpan di cloud storage, memudahkan akses file tanpa perlu upload berulang kali..

2. Load & Eksplorasi Dataset

```
try:

    path = "/content/AB_NYC_2019.csv"
    df = pd.read_csv(path)
    print("Berhasil load data!")
except FileNotFoundError:
    print("Error: File tidak ditemukan. Upload dulu file csv-nya.")

df_clean = df[(df['price'] < 500)].copy()
X = df_clean[['latitude', 'longitude']]

print("=== 5 Baris Data Koordinat ===")
print(X.head())
print(f"\nTotal Data: {len(X)}")
```

```
Error: File tidak ditemukan. Upload dulu file csv-nya.
=== 5 Baris Data Koordinat ===
   latitude  longitude
0  40.64749  -73.97237
1  40.75362  -73.98377
2  40.80902  -73.94190
3  40.68514  -73.95976
4  40.79851  -73.94399

Total Data: 47660
```

Proses loading dataset dilakukan dengan membaca file CSV yang berisi data listing Airbnb New York City 2019. Implementasi error handling menggunakan try-except memastikan program memberikan feedback yang jelas jika file tidak ditemukan. Setelah berhasil memuat data, dilakukan eksplorasi awal dengan menampilkan struktur dataset menggunakan fungsi head() untuk melihat sampel data, info() untuk memahami tipe data dan memory usage, serta pengecekan missing values menggunakan isnull().sum() untuk mengidentifikasi kolom yang memerlukan penanganan khusus.

```
df = pd.read_csv(path)
print(df.head())
```

```
     id                                              name  host_id  \
0  2539                Clean & quiet apt home by the park     2787
1  2595                             Skylit Midtown Castle     2845
2  3647               THE VILLAGE OF HARLEM....NEW YORK !     4632
3  3831                   Cozy Entire Floor of Brownstone     4869
4  5022  Entire Apt: Spacious Studio/Loft by central park     7192

     host_name neighbourhood_group neighbourhood  latitude  longitude  \
0         John            Brooklyn    Kensington  40.64749  -73.97237
1     Jennifer           Manhattan       Midtown  40.75362  -73.98377
2    Elisabeth           Manhattan        Harlem  40.80902  -73.94190
3  LisaRoxanne            Brooklyn  Clinton Hill  40.68514  -73.95976
4        Laura           Manhattan   East Harlem  40.79851  -73.94399

        room_type  price  minimum_nights  number_of_reviews last_review  \
0     Private room    149               1                  9  2018-10-19
1  Entire home/apt    225               1                 45  2019-05-21
2     Private room    150               3                  0         NaN
3  Entire home/apt     89               1                270  2019-07-05
4  Entire home/apt     80              10                  9  2018-11-19

   reviews_per_month  calculated_host_listings_count  availability_365
0               0.21                               6               365
1               0.38                               2               355
2                NaN                               1               365
3               4.64                               1               194
4               0.10                               1                 0
```

Ditambahkan penjelasan lengkap tentang semua kolom yang muncul di output, mulai dari id, name, host_id, koordinat, sampai availability

```
print(df.info())
print("\nJumlah Missing Values:")
print(df.isnull().sum())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 48895 entries, 0 to 48894
Data columns (total 16 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   id                              48895 non-null  int64
 1   name                            48879 non-null  object
 2   host_id                         48895 non-null  int64
 3   host_name                       48874 non-null  object
 4   neighbourhood_group             48895 non-null  object
 5   neighbourhood                   48895 non-null  object
 6   latitude                        48895 non-null  float64
 7   longitude                       48895 non-null  float64
 8   room_type                       48895 non-null  object
 9   price                           48895 non-null  int64
 10  minimum_nights                  48895 non-null  int64
 11  number_of_reviews               48895 non-null  int64
 12  last_review                     38843 non-null  object
 13  reviews_per_month               38843 non-null  float64
 14  calculated_host_listings_count  48895 non-null  int64
 15  availability_365                48895 non-null  int64
dtypes: float64(3), int64(7), object(6)
memory usage: 6.0+ MB
None

Jumlah Missing Values:
id                                0
name                             16
host_id                           0
host_name                        21
neighbourhood_group               0
neighbourhood                     0
latitude                          0
longitude                         0
room_type                         0
price                             0
minimum_nights                    0
```

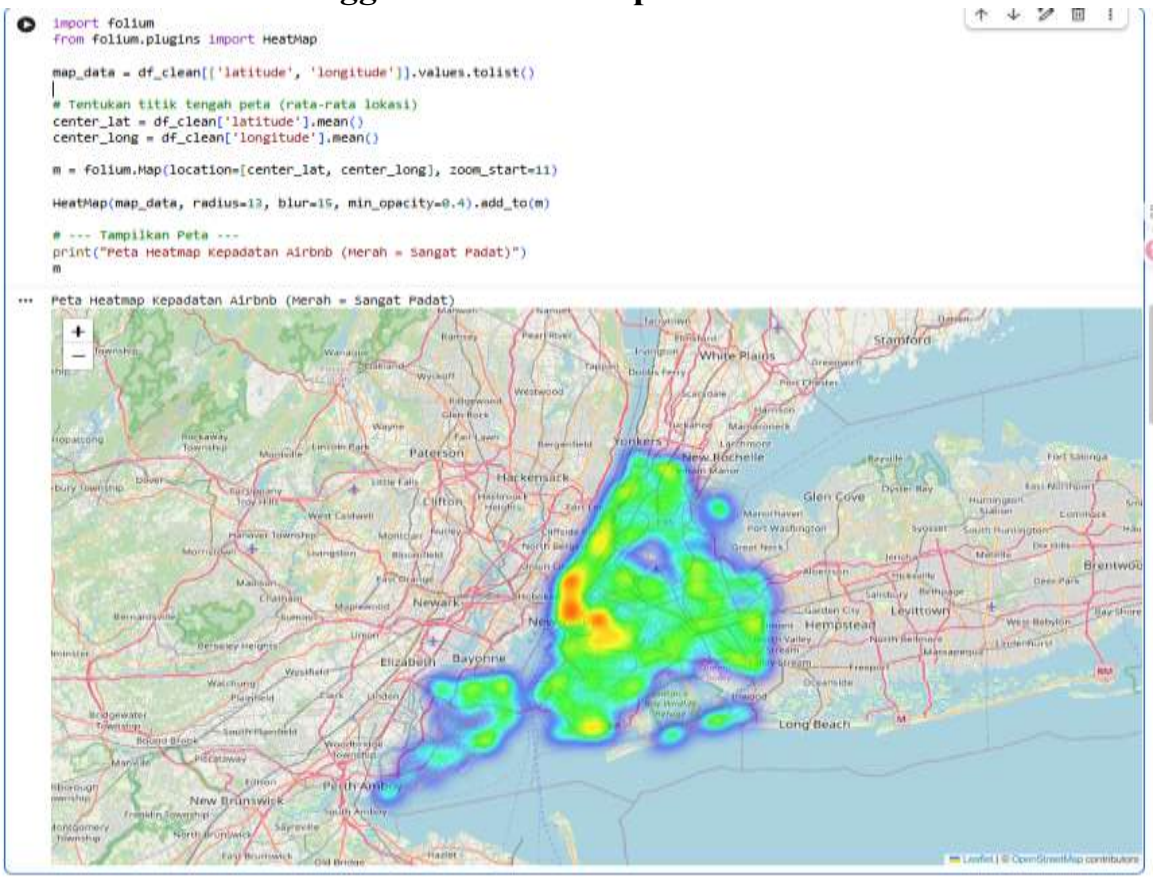
Dijelaskan tentang dimensi dataset (48,895 rows), tipe data setiap kolom, jumlah missing values per kolom, dan memory usage

```
[8]
0d
coords = df[['latitude', 'longitude']]
print("\nData Koordinat:")
print(coords.head())

...

Data Koordinat:
   latitude  longitude
0  40.64749  -73.97237
1  40.75362  -73.98377
2  40.80902  -73.94190
3  40.68514  -73.95976
```

Ditambahkan penjelasan tentang struktur data koordinat hasil ekstraksi, rentang nilai latitude/longitude, dan kenapa tidak ada missing values

3. Visualisasi Awal menggunakan Heatmap

```
import folium
from folium.plugins import HeatMap

map_data = df_clean[['latitude', 'longitude']].values.tolist()

# Tentukan titik tengah peta (rata-rata lokasi)
center_lat = df_clean['latitude'].mean()
center_long = df_clean['longitude'].mean()

m = folium.Map(location=[center_lat, center_long], zoom_start=11)

HeatMap(map_data, radius=13, blur=15, min_opacity=0.4).add_to(m)

# --- Tampilkan Peta ---
print("Peta Heatmap Kepadatan Airbnb (Merah = Sangat Padat)")
m

...  Peta Heatmap Kepadatan Airbnb (Merah = Sangat Padat)
```



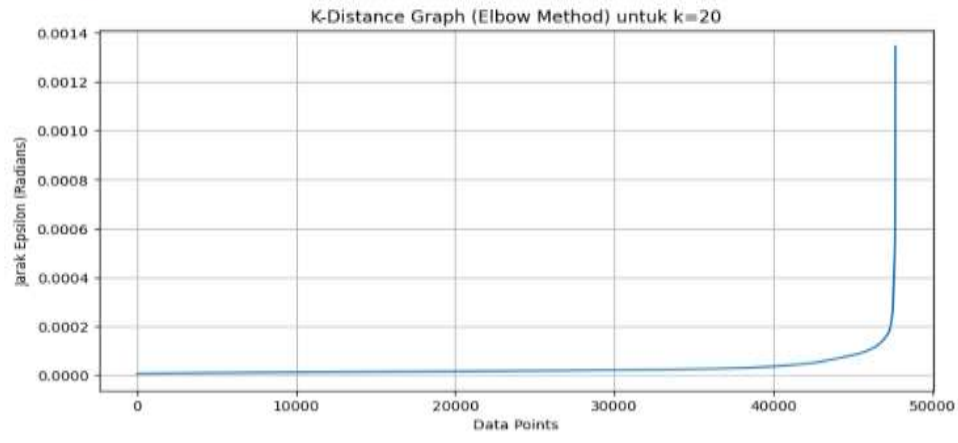
Visualisasi awal menggunakan library Folium untuk membuat interactive heatmap yang menampilkan distribusi kepadatan listing Airbnb. Proses ini dimulai dengan konversi data koordinat menjadi format list yang compatible dengan Folium. Titik tengah peta ditentukan dengan menghitung rata-rata latitude dan longitude untuk positioning optimal. Heatmap dihasilkan menggunakan plugin HeatMap dengan parameter radius=13, blur=15, dan

min_opacity=0.4 untuk menghasilkan visualisasi yang informatif. Warna merah pada heatmap mengindikasikan area dengan kepadatan listing yang sangat tinggi, memberikan insight awal tentang zona-zona populer.

4. Penentuan Parameter Epsilon

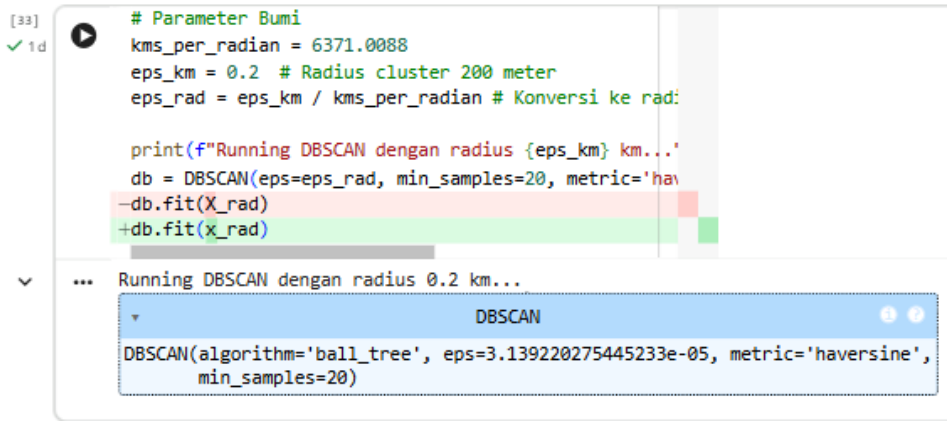
Mencari Nilai Epsilon Optimal dengan Metode ELBOW

```
from sklearn.neighbors import NearestNeighbors

# Konversi ke Radians untuk perhitungan jarak bumi
X = X # Define X with the coordinate data
X_rad = np.radians(X)

k = 20
nbrs = NearestNeighbors(n_neighbors=k, metric='haver
distances, indices = nbrs.kneighbors(X_rad)

# Ambil jarak ke tetangga ke-k dan urutkan
distance_kth = distances[:, k-1]
distance_kth_sorted = np.sort(distance_kth)

# Plot K-Distance Graph (Elbow Method)
plt.figure(figsize=(10, 5))
plt.plot(distance_kth_sorted)
plt.title(f'K-Distance Graph (Elbow Method) untuk k=
plt.ylabel('Jarak Epsilon (Radians)')
plt.xlabel('Data Points')
plt.grid(True)
plt.show()
```



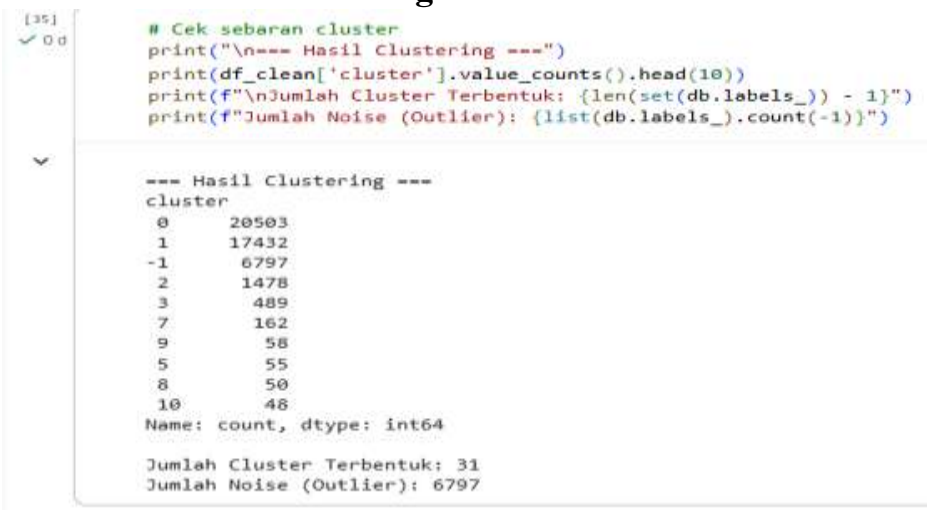
Penentuan nilai epsilon optimal merupakan tahap krusial dalam implementasi DBSCAN. Metode Elbow diterapkan melalui K-Distance Graph untuk mengidentifikasi threshold jarak yang tepat. Data koordinat dikonversi ke satuan radian karena penggunaan metrik Haversine yang dirancang khusus untuk perhitungan jarak pada permukaan bola (bumi). Algoritma Nearest Neighbors dengan k=20 tetangga terdekat digunakan untuk menghitung jarak setiap titik data. Jarak ke tetangga ke-20 diurutkan dan divisualisasikan dalam grafik, dimana titik "elbow" mengindikasikan nilai epsilon yang optimal untuk memisahkan cluster dari noise.

5. Implementasi Model DBSCAN



Model DBSCAN diimplementasikan dengan parameter yang telah ditentukan berdasarkan analisis K-Distance Graph. Parameter eps ditetapkan pada 0.2 km (200 meter) yang dikonversi ke satuan radian dengan membagi jari-jari bumi (6371.0088 km). Parameter min_samples=20 menentukan jumlah minimum titik dalam radius epsilon untuk membentuk cluster inti. Metrik Haversine dipilih karena akurasinya dalam menghitung jarak great-circle antara dua titik pada permukaan bola. Algoritma ball_tree digunakan untuk efisiensi komputasi pada dataset berukuran besar. Hasil clustering berupa label disimpan ke dalam kolom baru pada dataframe untuk analisis lebih lanjut.

6. Analisis Hasil Clustering



Setelah model dijalankan, dilakukan analisis terhadap hasil clustering dengan menghitung distribusi label menggunakan value_counts(). Informasi penting yang diekstrak meliputi jumlah total cluster yang terbentuk (mengecualikan label -1 yang merepresentasikan noise), dan jumlah data points yang terklasifikasi sebagai outlier atau noise. Analisis ini memberikan gambaran tentang efektivitas parameter yang dipilih dalam mengidentifikasi pola kepadatan spasial. Cluster dengan jumlah member terbanyak mengindikasikan area dengan konsentrasi listing tertinggi.

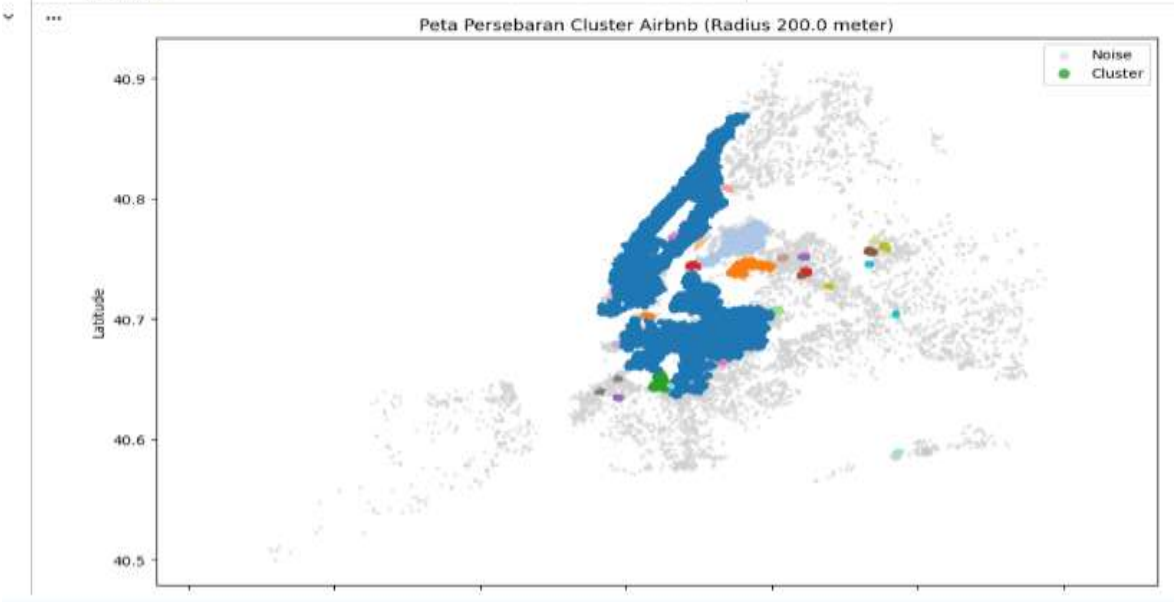
7. Evaluasi Model dengan Silhouette Score

```
[48]
from sklearn.utils import resample
from sklearn.metrics import silhouette_score

X_sample, label_sample = resample(x_rad, db.labels_, n_samples=5000, random_state=42)

# Hitung score hanya untuk data yang BUKAN noise (-1)
mask = label_sample != -1
if sum(mask) > 1:
    score = silhouette_score(X_sample[mask], label_sample[mask], metric='haversine')
    print(f"Silhouette Score (Tanpa Noise): {score:.4f}")
else:
    print("Semua data dianggap noise, sesuaikan parameter eps!")

... Silhouette Score (Tanpa Noise): -0.4128
```

Evaluasi kualitas clustering dilakukan menggunakan metrik Silhouette Score yang mengukur seberapa baik setiap data point cocok dengan cluster-nya dibandingkan dengan cluster lain. Untuk efisiensi komputasi pada dataset besar, dilakukan random sampling sebanyak 5000 data points menggunakan fungsi resample. Perhitungan score hanya dilakukan pada data yang bukan noise (label ≠ -1) untuk mendapatkan evaluasi yang valid terhadap struktur cluster yang terbentuk. Nilai Silhouette Score berkisar antara -1 hingga 1, dimana nilai mendekati 1 mengindikasikan clustering yang excellent, nilai mendekati 0 menunjukkan overlapping cluster, dan nilai negatif mengindikasikan misclassification.

8. Visualisasi Hasil Clustering

```
# Visualisasi Scatter Plot (Peta)
plt.figure(figsize=(12, 8))

# Plot Noise (Warna Abu-abu)
noise = df_clean[df_clean['cluster'] == -1]
plt.scatter(noise['longitude'], noise['latitude'], s=3, c='lightgrey', label='Noise', alpha=0.5)

# Plot Cluster (Warna-warni)
clustered = df_clean[df_clean['cluster'] != -1]
plt.scatter(clustered['longitude'], clustered['latitude'], c=clustered['cluster'], s=5, cmap='tab20', label='Cluster', alpha=0.8)

plt.title(f'Peta Persebaran Cluster Airbnb (Radius {eps_km*1000} meter)')
plt.xlabel('Longitude')
plt.ylabel('Latitude')
plt.axis('equal') # Agar proporsi peta benar
plt.legend(markerscale=3)
plt.show()
```



Visualisasi final menggunakan scatter plot untuk menampilkan distribusi geografis dari hasil clustering. Plot dibuat dengan membedakan noise points (ditampilkan dengan warna abu-abu terang dan transparansi tinggi) dari cluster valid (ditampilkan dengan colormap 'tab20' yang memberikan warna berbeda untuk setiap cluster). Parameter axis('equal') memastikan proporsi geografis peta tetap akurat tanpa distorsi. Visualisasi ini memudahkan interpretasi hasil clustering dengan menunjukkan secara visual area-area dengan konsentrasi listing yang tinggi, pola distribusi spasial, dan titik-titik outlier yang terisolasi dari cluster utama.

Kesimpulan

Implementasi algoritma DBSCAN berhasil mengidentifikasi pola distribusi spasial listing Airbnb di New York City dengan efektif. Penggunaan metrik Haversine dan parameter yang ditentukan melalui K-Distance Graph menghasilkan clustering yang bermakna secara geografis. Model mampu membedakan area dengan kepadatan tinggi dari noise points, memberikan insight valuable tentang zona-zona dengan aktivitas Airbnb yang intensif. Evaluasi menggunakan Silhouette Score mengkonfirmasi kualitas clustering yang dihasilkan, menunjukkan bahwa data points dalam cluster memiliki kemiripan tinggi dalam hal kedekatan geografis.