Gunung Djati Conference Series, Volume 3 (2021)
Mini Seminar Kelas *Data Mining* 2020
ISSN: 2774-6585
Website: https://conferences.uinsgd.ac.id/gdcs

Klasterisasi Data Pemain *Premier League* 2020 dengan Algoritma DBSCAN

(*Premier League 2020 Player Data Clustering with the DBSCAN Algorithm*)

Muhammad Thariq Sabiq Bilhaq[1], Aisyah Amini Nur[2], Akbar Hidayatullah Harahap[3], Ihsan Muttaqin Bin Abdul Malik[4], Muhammad Irfan Nur Imam[5], Muhammad Nur Sidiq[6]

[1]Teknik Informatika, UIN Sunan Gunung Djati Bandung, 1177050077@student.uinsgd.ac.id
[2]Teknik Informatika, UIN Sunan Gunung Djati Bandung, 1177050010@student.uinsgd.ac.id
[3]Teknik Informatika, UIN Sunan Gunung Djati Bandung, 1177050012@student.uinsgd.ac.id
[4]Teknik Informatika, UIN Sunan Gunung Djati Bandung, 1177050053@student.uinsgd.ac.id
[5]Teknik Informatika, UIN Sunan Gunung Djati Bandung, 1177050071@student.uinsgd.ac.id
[6]Teknik Informatika, UIN Sunan Gunung Djati Bandung, 1167050104@student.uinsgd.ac.id

Abstrak

Premiere League merupakan event bergengsi yang diminati banyak orang. Dan Goals pemain sangat penting dalam hal ini.Riset jumlah pemain goals berdasarkan usia ini bisa dilakukan, salah satunya dengan Clustering. Clustering merupakan salah satu metode untuk mengelompokkan objek yang sesuai dengan kemiripannya. Dan untuk algoritma yang dipakai adalah DBSCAN. Algoritma ini dapat menyelesaikan masalah yang rumit dan memiliki kepadatan yang tepat sehingga dapat membantu untuk mendapatkan data clustering pada tema kali ini.

Kata kunci: Clustering, Data Mining, DBSCAN.

Abstract

Premiere League is prestigious event that is in demand by many people. And player goals are very important in this regard. Research on the number of players based on this age can be done. One of which is by clustering. Clustering is a method for grouping objects according to their similarities. And the algorithm used DBSCAN. This algorithm can solve complex problems and has the right density so that it can help to get clustering data on this theme.

Keywords: *Clustering, Data Mining, DBSCAN.*

1 Pendahuluan

Sepak bola telah menjadi permainan yang selalu menjadi olahraga paling populer untuk dimainkan dan dilihat di sebagian besar negara Eropa dan Amerika Selatan. Popularitas olahraga, bagaimanapun [1]. Liga Utama Inggris (Premier League), sebelumnya disebut dengan F.A. Premier League adalah sebuah kompetisi liga sepak bola profesional di Inggris yang merupakan kompetisi antar klub tingkat tertinggi di negara Inggris. Saat ini mendapat dukungan sponsor dari Barclays Bank sehingga nama resminya menjadi Barclays Premier League. Di luar Inggris Raya biasa disebut dengan English Premier League (https://id.wikipedia.org/wiki/Liga_Utama_Inggris).Penting untuk diketahui setiap statistik para pemainnya untuk seorang pelatih. Masalah umum sering terjadi kala pergantian pemain

setiap musim. Untuk mengatasi masalah ini, strategi pengubahan permainan dipikirkan yang mengarah pada penciptaan ide. Didukung dengan kumpulan data lengkap dari semua statistik pemain.

Riset dilakukan yang berdasarkan jumlah goals yang dicetak oleh pemain menjadikan acuan sebagai bahan olah data cluster. Metode clustering adalah sebuah metode untuk membuat kelompok dari obyek pada suatu cara tentang mengelompokkan objek sesuai dengan kemiripannya dalam kelompok-kelompok yang berbeda. Pada clustering kita memiliki banyak pilihan akan metode yang dapat digunakan. Algoritma yang digunakan pada penelitian ini menggunakan algoritma DBSCAN metode yang tangguh untuk mendeteksi data yang menyimpang, data yang menyimpang disebut noise (outlier) [2].

Terdapat beberapa penelitian terdahulu yang berkaitan dengan penelitian ini, antara lain: (1) analisis proyeksi pekerjaan alumni pada perguruan tinggi dengan menggunakan algoritma DBSCAN dan SUBCLU [3]; (2) analisis anomaly trafik data dengan algoritma DBSCAN dan Birch [4]; (3) perbandingan algoritma DBSCAN dan Fuzzy C-Means untuk mengelompokkan pelanggan retail [5]; (4) analisis penyebaran penyakit dengan menggunakan algoritma DBCSAN [6]; dan analisis sebaran *bandwidth* dengan menggunakan algoritma DBSCAN [7].

2 Metodologi

Density Based Spatial Clustering Algorithm with Noise (DBSCAN) adalah algoritma pengelompokan yang didasarkan pada kepadatan (density) data [8]. Konsep kepadatan yang dimaksud dalam DBSCAN adalah jumlah data yang berada dalam radius MinPts (Jumlah minimal data dalam radius ε), data tersebut masuk dalam kategori kepadatan yang diinginkan, jumlah data dalam radius tersebut termasuk data inti itu sendiri. Konsep kepadatan seperti ini melahirkan tiga macam status dari setiap data, yaitu inti (core), batas (border), dan noise (noise). Sebuah data akan dimasukkan sebagai inti jika jumlah data tetangga dan dirinya sendiri pada radius ε berjumlah ≥MinPts. Nilai radius ε dan MinPts ini ditetapkan secara mandiri. Untuk data yang jumlah tetangga dan dirinya sendiri dalam radius ε kurang dari MinPts, tetapi tetangganya menjadi inti karena kehadirannya, data tersebut dikategorikan sebagai batas. Jika jumlah tetangga dan dirinya sendiri dalam radius ε kurang dari MinPts dan tidak ada tetangga yang menjadi inti karena kehadirannya, data tersebut dikategorikan sebagai noise. Kemudian secara iteratif mengumpulkan objek yang dapat dijangkau dengan kepadatan secara langsung dari objek inti ini, yang mungkin melibatkan penggabungan cluster baru yang dapat dijangkau dengan kepadatan. Proses berakhir ketika tidak ada objek baru yang dapat ditambahkan ke cluster manapun [9].

Data dikelompokkan dalam bentuk yang berubah-ubah dengan adanya noise dalam database dimensi tinggi spasial dan non-spasial. Ide kunci dari DBSCAN adalah bahwa untuk setiap objek dari sebuah cluster, lingkungan dari radius tertentu (Eps) harus berisi setidaknya sejumlah minimum objek (MinPts), yang berarti bahwa kardinalitas dari lingkungan tersebut harus melebihi beberapa ambang batas.

Tetangga dari sembarang titik 'p' didefinisikan sebagai:

$$NEps = \{q \varepsilon D / dist(p,q) < Eps\}$$

Di sini, D adalah database objek. Jika ε –tetangga dari sebuah titik P setidaknya mengandung sejumlah kecil poin, dan maka titik ini disebut titik inti. Poin inti didefinisikan sebagai:

$$NEps(P) > MinPts$$

Gunung Djati Conference Series
ISSN 2774-6585
Volume 3 (2021)

Di sini Eps dan MinPts adalah parameter yang ditentukan pengguna yang berarti radius lingkungan dan minimum jumlah titik di titik ε-tetangga dari inti masing-masing. Jika kondisi ini tidak terpenuhi maka titik ini adalah dianggap sebagai poin non-inti.

3 Hasil dan Pembahasan

3.1 Input Data

DBSCAN merupakan algoritma clustering yang menggunakan Density-based methods atau menggunakan metode kerapatan. Dalam penelitian ini, data yang digunakan adalah data pemain premiere league 2020. Input data yang akan diolah didapat dari kaggle.

```
In [3]: dataset = pd.read_csv('dataset - 2020-09-24.csv')
        df = dataset.copy()
        df.head()
```

Out[3]:

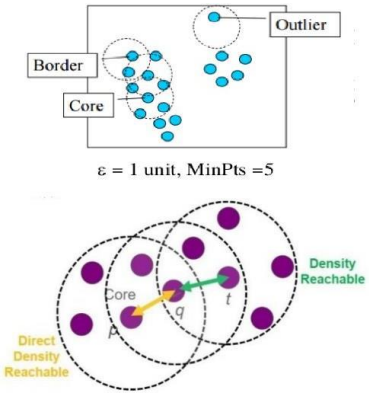
	Name	Jersey Number	Club	Position	Nationality	Age	Appearances	Wins	Losses	Goals	...	Punches	High Claims	Catches	Sweeper clearances	Throw outs	Goal Kicks
0	Bernd Leno	1.0	Arsenal	Goalkeeper	Germany	28.0	64	28	16	0	...	34.0	26.0	17.0	28.0	375.0	489.0
1	Matt Macey	33.0	Arsenal	Goalkeeper	England	26.0	0	0	0	0	...	0.0	0.0	0.0	0.0	0.0	0.0
2	Rúnar Alex Rúnarsson	13.0	Arsenal	Goalkeeper	Iceland	25.0	0	0	0	0	...	0.0	0.0	0.0	0.0	0.0	0.0
3	Héctor Bellerín	2.0	Arsenal	Defender	Spain	25.0	160	90	37	7	...	NaN	NaN	NaN	NaN	NaN	NaN
4	Kieran Tierney	3.0	Arsenal	Defender	Scotland	23.0	16	7	5	1	...	NaN	NaN	NaN	NaN	NaN	NaN

5 rows × 59 columns

Gambar 1 Input Dataset

Algoritma DBSCAN memerlukan 2 input yaitu eps dan minimum points (minPts). Cara kerja dari algoitma DBSCAN yaitu :

1. Tentukan nilai minPts dan epsilon (eps) yang akan digunakan.
2. Pilih data awal “p” secara acak.
3. Hitung jarak antara data “p” terhadap semua data menggunakan Euclidian distance.
4. Ambil semua amatan yang density-reachable dengan amatan “p”.
5. Jika amatan yang memenuhi nilai epsilon lebih dari jumlah minimal amatan dalam satu gerombol maka amatan “p” dikategorikan sebagai core points dan gerombol terbentuk.
6. Jika amatan “p” adalah border points dan tidak ada amatan yang density- reachable dengan amatan “p”, maka lanjutkan pada amatan lainnya.
7. Ulangi langkah 3 sampai 6 hingga semua amatan diproses.



Gambar 2 Cara Kerja Algoritma

3.2 Pre-processing Data

Pada tahap preprocessing terdapat beberapa hal yang dilakukan diantaranya:

1. Mengganti nilai NaN menjadi 0.
2. Kolom yang digunakan adalah kolom “Age” dan “Goals”
3. Visualisasi sebelum proses clustering

Untuk mengganti nilai NaN menjadi 0 dapat dilakukan hal berikut :

```
In [4]: # Mengisi nilai NaN dengan value 0
        df = df.fillna(value = 0)
        df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 571 entries, 0 to 570
Data columns (total 59 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   Name                    571 non-null    object
 1   Jersey Number           571 non-null    float64
 2   Club                    571 non-null    object
 3   Position                571 non-null    object
 4   Nationality             571 non-null    object
 5   Age                     571 non-null    float64
 6   Appearances             571 non-null    int64
 7   Wins                    571 non-null    int64
 8   Losses                  571 non-null    int64
 9   Goals                   571 non-null    int64
 10  Goals per match         571 non-null    float64
 11  Headed goals            571 non-null    float64
 12  Goals with right foot   571 non-null    float64
 13  Goals with left foot    571 non-null    float64
 14  Penalties scored        571 non-null    float64
 15  Freekicks scored        571 non-null    float64
 16  Shots                   571 non-null    float64
 17  Shots on target         571 non-null    float64
 18  Shooting accuracy %     571 non-null    object
 19  Hit woodwork            571 non-null    float64
 20  Big chances missed      571 non-null    float64
```

Gambar 3 Isi Nilai NaN menjadi 0

Isi pada setiap kolom yang memiliki nilai NaN telah diubah menjadi 0, dibuktikan dengan jumlah *non – null* yang jumlah nya sama. Kemudian proses selanjutnya, memilih kolom apa saja yang akan dipakai untuk dilakukan clustering. Kolom yang akan digunakan untuk clustering yaitu kolom “Age” dan juga kolom “Goals”.

```
In [5]: sample = df[['Age', 'Goals']]
        sample
```

Out[5]:

	Age	Goals
0	28.0	0
1	26.0	0
2	25.0	0
3	25.0	7
4	23.0	1
...	...	...
566	20.0	3
567	29.0	32
568	24.0	5
569	18.0	0
570	20.0	0

571 rows × 2 columns

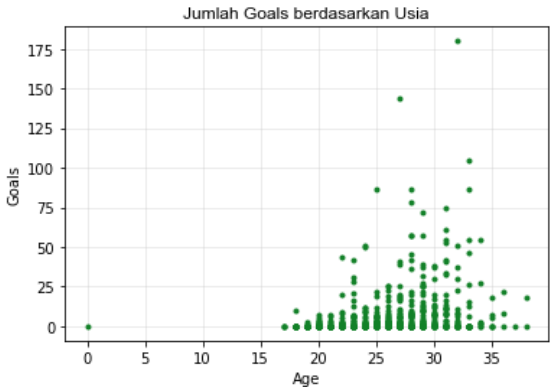
Gambar 4 Tabel yang akan digunakan untuk Clustering

Gunung Djati Conference Series
ISSN 2774-6585
Volume 3 (2021)

Selanjutnya proses visualisasi data pada kolom “Age” dan “Goals” untuk menunjukan tampilan setiap poin sebelum proses clustering.

```
In [6]: # visualisasi point sebelum clustering

_ = plt.plot(df['Age'], df['Goals'],
             marker = '.', linewidth = 0, color = '#128128')
_ = plt.grid(which = 'major', color = '#cccccc', alpha = 0.45)
_ = plt.title('Jumlah Goals berdasarkan Usia', family = 'arial', fontsize = 12)
_ = plt.xlabel('Age')
_ = plt.ylabel('Goals')
_ = plt.show()
```



Gambar 5 Visualisasi Data sebelum Proses Clustering

3.3 Implementasi Algoritma

Setelah proses preprocessing, kemudian masuk ke implementasi dari algoritma DBSCAN. Hal yang pertama dilakukan adalah membuat 2 kolom yang sudah ditentukan tadi ke dalam suatu array.

```
In [7]: dbscan_data = df[['Age', 'Goals']]

        # Mengubah / memastikan type value ke dalam float
        dbscan_data = dbscan_data.values.astype('float32', copy = False)
        dbscan_data

Out[7]: array([[28.,  0.],
               [26.,  0.],
               [25.,  0.],
               ...,
               [24.,  5.],
               [18.,  0.],
               [20.,  0.]], dtype=float32)
```

Gambar 6 Input Kolom kedalam array

Kemudian setelah dimasukkan ke dalam array, dilakukan normalisasi data.

```
In [8]: # Normalisasi Data
        dbscan_data_scaler = StandardScaler().fit(dbscan_data)
        dbscan_data = dbscan_data_scaler.transform(dbscan_data)
        dbscan_data

Out[8]: array([[ 0.50117993, -0.4362705 ],
               [ 0.05681077, -0.4362705 ],
               [-0.1653738 , -0.4362705 ],
               ...,
               [-0.38755837, -0.13857636],
               [-1.7206658 , -0.4362705 ],
               [-1.2762966 , -0.4362705 ]], dtype=float32)
```

Gambar 7 Normalisasi Data

Selanjutnya masukkan dbscan_data kedalam Algoritma DBSCAN

```
In [20]: # Model DBSCAN
         '''
         eps = float, default=0.5
         min_samples = int, default=5

         untuk dokumetasi lebih : https://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html

         '''

         model = DBSCAN(eps = 0.5, min_samples = 5, metric = 'euclidean').fit(dbscan_data)
         model

Out[20]: DBSCAN()
```

Gambar 8 Input data kedalam Algoritma DBSCAN

Pada model DBSCAN diatas, eps yang digunakan yaitu nilai default yang didapat dari dokumentasi library DBSCAN yaitu 0.5 sedangakn min_samples merupakan jumlah sampel (atau bobot total) di lingkungan untuk suatu titik yang akan dianggap sebagai titik inti atau minPts. Ini termasuk intinya sendiri.

Selanjutnya setelah algoritma berhasil dimasukkan, dapat dilihat berapa cluster yang dihasilkan, dapat dilakukan dengan menggunakan syntax "*model.labels_* ". Berikut jumlah cluster dan outliers yang terbentuk.

```
In [17]: outliers_df = df[model.labels_ == -1]
         clusters_df = df[model.labels_ != -1]

         colors = model.labels_
         colors_clusters = colors[colors != -1]
         color_outliers = 'black'

         clusters = Counter(model.labels_)
         print(clusters)
         print(df[model.labels_ == -1].head())
         print('Number of Clusters = {}'.format(len(clusters)-1))

         Counter({0: 544, -1: 19, 1: 8})
```

Gambar 9 Hasil Clustering dan Outlier yang terbentuk

Dan jumlah cluster yang terbentuk yaitu:

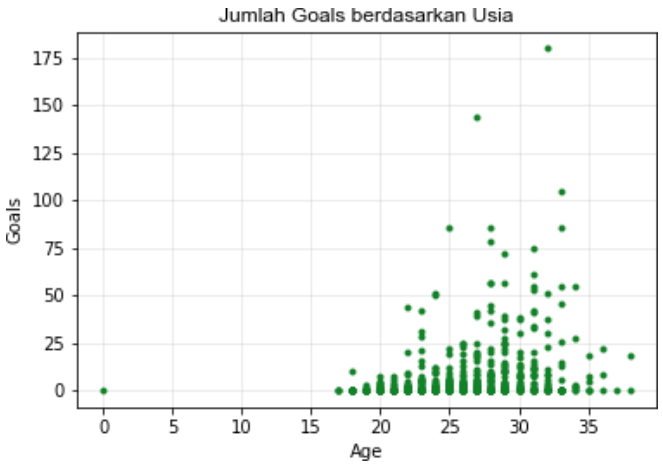
```
[5 rows x 59 columns]
Number of Clusters = 2
```

Gambar 10 Jumlah Cluster yang terbentuk

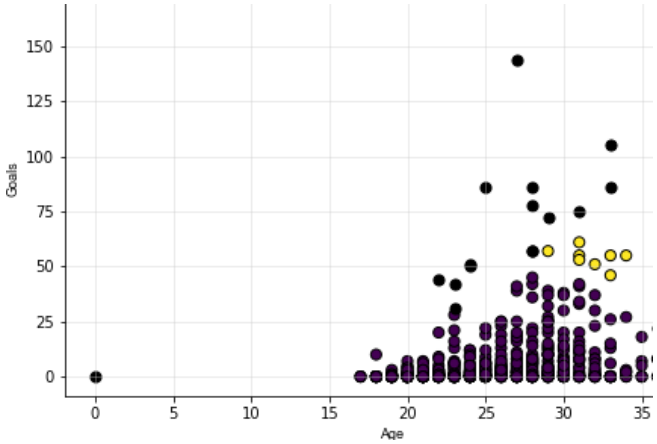
3.4 Hasil Eksperimen

Setelah didapat jumlah cluster serta outliers dari proses tersebut, kemudian dilakukan perbandingan visualisasi sebelum proses clustering dan visualisasi hasil dari clustering.

Gunung Djati Conference Series
ISSN 2774-6585
Volume 3 (2021)



Gambar 11 Data sebelum Clustering



Gambar 12 Data Setelah Clustering

4 Simpulan

Untuk kesimpulan dari pengujian data diatas adalah algoritma DBSCAN berhasil melakukan clustering pada produktivitas goal berdasarkan usia dengan total 2 cluster, dimana algoritma DBSCAN akan menentukan sendiri cluster yang dihasilkan. Setiap objek yang yang tidak termasuk cluster akan dianggap outlier, dilihat dari grafik, outlier diberikan warna hitam dan sisanya merupakan bagian cluster. Dalam kasus ini, algoritma DBSCAN dapat dikatakan berhasil melakukan proses clustering sesuai kepadatan, namun belum memenuhi keinginan dari peneilitian ini, karena diharapkan clustering ini dapat mengelompokkan tingkat produktivitas gol (tinggi, normal, kurang). Dikarenakan untuk jumlah cluster DBSCAN ditentukan oleh algoritma itu sendiri. Mungkin akan lebih cocok jika digunakan algoritma K-Means untuk kasus ini. Namun, secara keseluruhan dapat dikatakan produktivitas gol pemain pada tahun ini cukup rendah karena dilihat dari grafik kepadatan yang lebih banyak pada bagian bawah (rendah). Kelebihan dari DBSCAN Tidak seperti K-means, DBSCAN tidak meminta kepada user untuk memasukkan nilai berapa banyak cluster yang akan dibuat. DBSCAN dapat membuat cluster dengan beragam bentuk, tidak harus berbentuk circle (lingkaran). DBSCAN dapat membedakan data mana yang outliers.

Referensi

[1] A. Gangal, A. Talnikar, A. Dalvi, V. Zope, and A. Kulkarni, “Analysis and Prediction of Football Statistics using Data Mining Techniques,” *Int. J. Comput. Appl.*, vol. 132, no. 5, pp.

8–11, 2015, doi: 10.5120/ijca2015907263.

[2] E. Prasetyo, *Data mining konsep dan aplikasi menggunakan matlab*. Yogyakarta: Andi, 2012.

[3] A. R. Aritonang, Sutarman, and P. Sihombing, "Analisis Subspace Clustering Menggunakan DBSCAN dan SUBCLU Untuk Proyeksi Pekerjaan Alumni Perguruan Tinggi," *Teknovasi*, vol. 02, pp. 33–60, 2015.

[4] M. A. Shauma, Y. Purwanto, and A. Novianty, "Deteksi Anomali Trafik Menggunakan Algoritma Birch Dan Dbscan Pada Streaming Traffic," *eProceedings Eng.*, vol. 3, no. 3, pp. 5004–5012, 2016, [Online]. Available: https://libraryeproceeding.telkomuniversity.ac.id/index.php/engineering/article/view/3132.

[5] M. I. Chanafi, D. P. Hapsari, R. K. Hapsari, and T. Indriyani, "Implementasi Algoritma Clustering Untuk Pengelompokan Pelanggan Retail Berdasarkan Skor Recency, Frequency, Dan Monetary," *Pros. Semin. Nas. Sains dan Teknol. Terap.*, vol. 1, no. 1, pp. 797–810, 2019, [Online]. Available: https://ejurnal.itats.ac.id/sntekpan/article/view/783.

[6] T. I. Hermanto and M. A. Sunandar, "Analisis Data Sebaran Penyakit Menggunakan Algoritma Density Based Spatial Clustering Of Applications With Noise," *J. Sains Komput. dan Teknol. Inf.*, vol. 3, no. 1, pp. 104–110, 2020, doi: 10.33084/jsakti.v3i1.1775.

[7] T. I. Hermanto and Y. Muhyidin, "Analisis Data Sebaran Bandwidth Menggunakan Algoritma Dbscan Untuk Menentukan Tingkat Kebutuhan Bandwidth Di Kabupaten Purwakarta," *Rabit J. Teknol. dan Sist. Inf. Univrab*, vol. 5, no. 2, pp. 130–137, 2020, doi: 10.36341/rabit.v5i2.1388.

[8] K. Khan, S. U. Rehman, K. Aziz, S. Fong, S. Sarasvady, and A. Vishwa, "DBSCAN: Past, present and future," in *5th International Conference on the Applications of Digital Information and Web Technologies, ICADIWT 2014*, 2014, pp. 232–238, doi: 10.1109/ICADIWT.2014.6814687.

[9] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan, "Automatic subspace clustering of high dimensional data for data mining applications," *SIGMOD Rec.*, vol. 27, no. 2, pp. 94–105, 1998, doi: 10.1145/276305.276314.