Clustering Data

Disusun oleh :

Mila Putri Kartika Dewi (1301174218)

Dzaka Triadi Mahdiyah (1301152728)

Wilrades Christofel (1301170753)

PROGRAM STUDI S1 INFORMATIKA

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY

BANDUNG

2020

1. Dataset

Dalam tugas analisi ini, kelompok kami menggunakan dataset dari UCI Machine Learning Repository, dataset yang kami gunakan adalah :

1.1.Exasens
1.2.Accepted Papers

2. Pre-processing

2.1. Exasens

Data *Exasens* merupakan dataset mengenai pengelompokan sampel *saliva*(air liur) dengan kategori COPD, Asthma, Infected, dan HC. Pada dataset ini masih terindikasi data noisy dan tergolong data yang tidak layak. Data ini masih terdapat *missing value* dan atribut yang tidak jelas dikarenakan atribut tersebut tidak terdapat pada deskripsi dari informasi data *Exasens.*

Dengan adanya permasalahan tersbut maka dibutuhkan pre-processing data, sehingga data *Exasens* dapat diproses untuk tahap berikutnya. Berikut tahapan dari *pre-processig* data.

2.1.1. Missing value

Missing value terdapat pada variabel “Imagunary Part” dan “Real Part”, untuk variable lain seperti “Unnamed” diabaikan karena tidak akan digunakan untuk porses selanjutnya.

```
data.isna().sum()
```

```
Diagnosis          0
ID                 0
Imaginary Part    299
Unnamed: 3        299
Real Part         299
Unnamed: 5        299
Gender             0
Age                0
Smoking            0
```

Dari data tersebut dapat diketahui *missing value* pada variable “Imagunary Part” dan “Real Part” memiliki jumlah yang sangat banyak yaitu 299, untuk itu

proses pengisian data dengan memasukan nilai *mean* pada variable tersebut dilakukan dengan cara sebagai beikut :

```
data['Imaginary Part'] = data['Imaginary Part'].fillna(data['Imaginary Part'].mean())
data['Imaginary Part Avg'] = data['Imaginary Part Avg'].fillna(data['Imaginary Part'].mean())
data['Real Part'] = data['Real Part'].fillna(data['Real Part Avg'].mean())
data['Real Part Avg'] = data['Real Part Avg'].fillna(data['Real Part Avg'].mean())
```

2.1.2. Label Encoding

Pada *variable* “Diagnosis” memiliki type data object atau kategorikal sehingga perlu dilakukan pengelompokan berdasarkan dari nilai *variable* “Diagnosis”. Diketahui *variable* “Diagnosis” terdiri dari kelompok COPD, Asthma, Infected, dan HC. Dalam mempermudah proses *clustering* perlu dilakukan perubahan pada nilai tersebut dalam format angka. Untuk mengatasi permasalahan tersebut maka, dilakukan proses *LabelEncoder* untuk penentuan dari kelompok *saliva*(air liur) dengan format angka. Berikut code penjelasannya.

```
labelencoder = LabelEncoder()
data['Diagnosis'] = labelencoder.fit_transform(data['Diagnosis'])
data
```

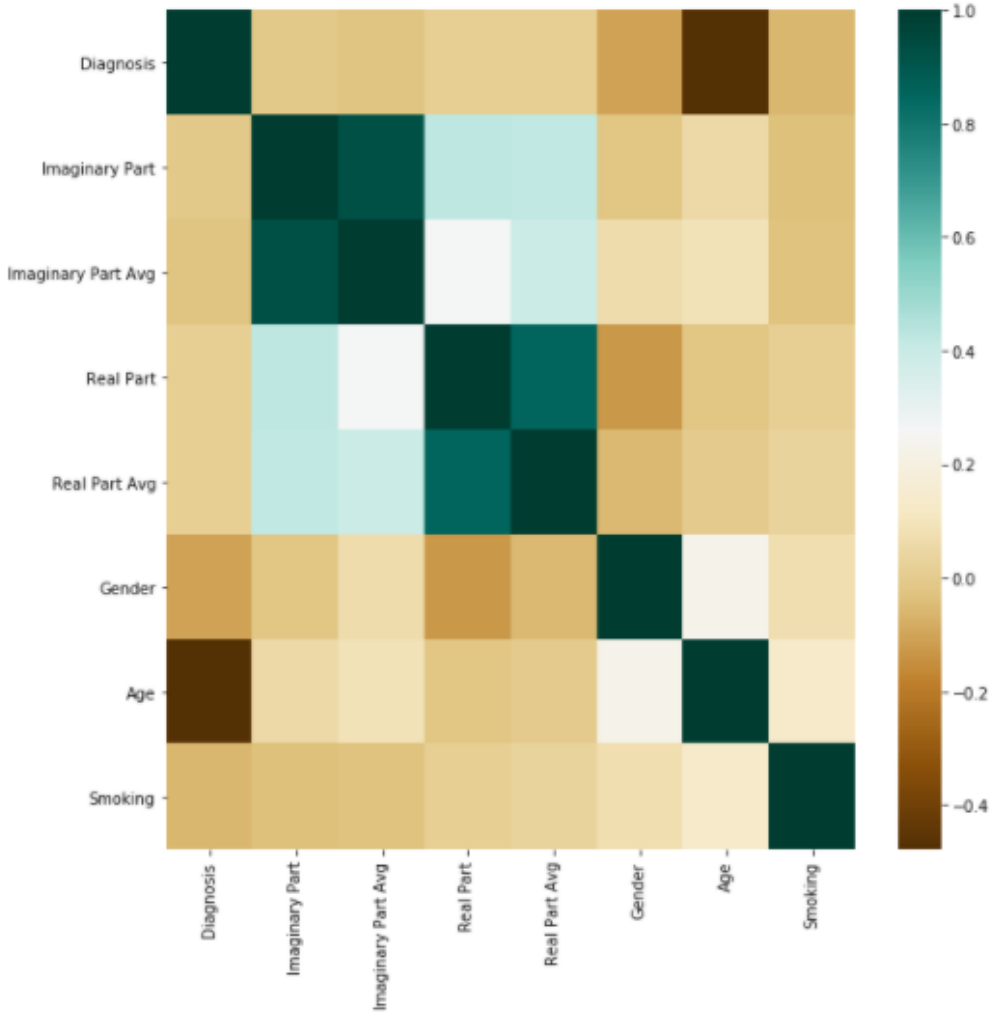
Nilai dari *variable* “Diagnosis” kemudian akan diprsentasikan kedalam angka yang bertipekan integer dengan rentan nilai 0-3. Berikut tampilan dari *variable* “Diagnosis” sebelum dan sesudah dilakukan *LabelEncoder*.

	Diagnosis
0	COPD
1	COPD
2	COPD
3	COPD
4	COPD

	Diagnosis
0	1
1	1
2	1
3	1
4	1

2.1.3. Pemilihan Variabel

Setelah semua *variable* bertypekan *integer* ataupun *float*, maka proses berikutnya adalah mencari korelasi antar semua variable pada data *Exasens*, tujuannya adalah untuk memilih variable yang paling kuat hubungannya dengan *variable* lain. Proses ini sebagai penentu untuk melakukan *clustering* terhadap *variable – variable* yang akan digunakan. Berikut hasil dari korelasi antar semua *variable.*



Dari hasil korelasi yang didapat, diketahui krelasi yang memiliki nilai tertinggi dengan kisaran 0,6 - 1 adalah *variable* “Imagunary Part” dan “Real Part”.

2.2. Accepted Papers

- Mengubah nama variabel High-Level Keyword(s) menjadi HighLevelKeywords

```
[11]: papers_new = papers_x.rename(columns={'High-Level Keyword(s)': 'HighLevelKeywords'})
```

- Mengubah kategori menjadi number

```
[12]: a = dict([e[:: -1] for e in enumerate(papers_new.Topics.unique())])
      b = dict([e[:: -1] for e in enumerate(papers_new.HighLevelKeywords.unique())])
```

```
[14]: papers_new.Topics = papers_new.Topics.map(a)
      papers_new.HighLevelKeywords = papers_new.HighLevelKeywords.map(b)
```

3. Clustering

3.1. Exasens

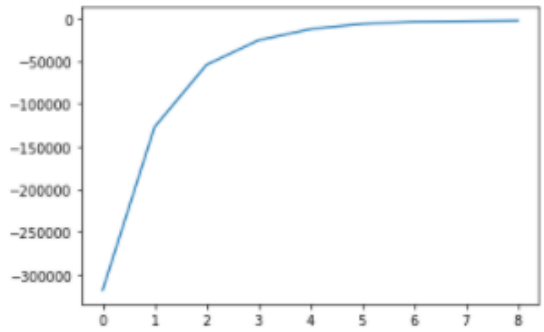
3.1.1. Penetuan N Cluster

Proses clustering pada dataset ini adalah dengan menentukan jumlah *n-cluster* terlebih dahulu sehingga *n-cluster* yang dipilih tepat dengan pembagian dari masing masing data yang ada. Berikut penjelasan dari algoritmanya.

```
scr = []
for i in range(1,10):
    score = KMeans(n_clusters=i).fit(df).score(df)
    #print(score)
    scr.append(score)

plt.plot(scr)
```

```
[<matplotlib.lines.Line2D at 0x28babc7f748>]
```



Dari teknik tersebut dapat dijelaskan bahwa nilai *n-cluster* yang cocok untuk proses clustering pada data Exasens adalah 2, karena garis melengkung pada kurva

tersebut menunjukan nilai yang cocok dalam penentuan *n-cluster* dan nilai yang cocok berada pada rentan 1-2 dan dalam kasus ini akan diambil nilai terbesar yaitu 2.

3.1.2. Proses Clustering dengan Fungsi K-Means

Sebelum diakukan *clustering* dengan *K-Means* perlu dilakukan perubahan format dari dataframe pandas ke format array, tujuannya adalah untuk mmepermudah proses clustering yang akan dilakukan. Selanjutnya dilakukan standarisasi pada data dengan tujuan akan lebih mudah dalam proses *clustering.*

```
scaler = MinMaxScaler()
x_scaled = scaler.fit_transform(x_array)
x_scaled
```

```
array([[0.14899866, 0.22578321],
       [0.10645305, 0.26272175],
       [0.12772586, 0.25862128],
       [0.08518024, 0.26272175],
       [0.10645305, 0.25450365],
       [0.08518024, 0.20524654],
       [0.06381842, 0.26272175],
       [0.19944993, 0.26397042],
       [0.14899866, 0.25862128],
```

Dari hasil *scalling* data membuat data yang dimiliki berada diantara nilai 0-1 sehingga proses tersebut nantikan akan mempremudah dalam clustering data.
Proses selanjutnya yaitu melakukan cluatering dengan menggunakan *library* k-menas, dengan *output* nilai pusat dari masing-masing cluster. Berikut tampilan code program yang dijalankan.

```
kmeans = KMeans(n_clusters = 3, random_state=123)
kmeans.fit(x_scaled)
print(kmeans.cluster_centers_)
```

```
[[0.19764313 0.26320104]
 [0.93858478 0.38235597]
 [0.09313529 0.24937968]]
```

3.2. Accepted Papers

3.2.1. Proses Clustering dengan Fungsi K-Means

```
# Menentukan dan mengkonfigurasi fungsi kmeans
kmeans = KMeans(n_clusters = 5, random_state=123)
# Menentukan kluster dari data
kmeans.fit(x_scaled)
```

```
KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
       n_clusters=5, n_init=10, n_jobs=None, precompute_distances='auto',
       random_state=123, tol=0.0001, verbose=0)
```

```
kmeans.cluster_centers_
```

```
array([[0.13632246, 0.1005117 ],
       [0.80875866, 0.81006865],
       [0.86923584, 0.27352472],
       [0.54477226, 0.06328321],
       [0.43103448, 0.49364791]])
```

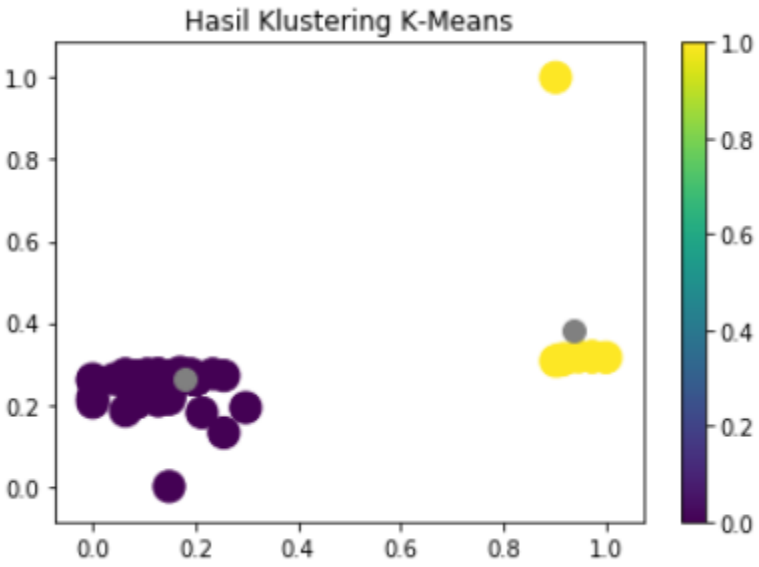
4. Hasil Clustering

4.1. Exasens

Hasil *clustering* pada dataset *Exasens* dapat dilihat pada gambar berikut.

	Imaginary Part	Real Part	kluster
0	-320.61	-495.26	2
1	-325.39	-473.73	2
2	-323.00	-476.12	2
3	-327.78	-473.73	2
4	-325.39	-478.52	2

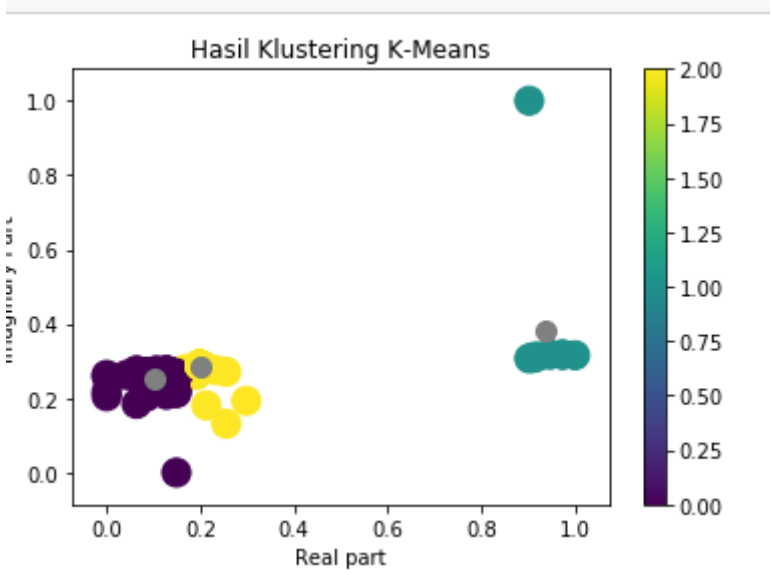
Dengan visualisasi data menggunakan *plot scatter* sehingga dapat diketahui persebaran dari data yang terlah dilakukan *clustering.*




```
df['kluster'].value_counts()
```

```
0    389
1     10
```

Kesimpulan yang didapat dari proses clustering dengan n-cluster sebanyak 2 yang telah dilakukan pada dataset *Exasens*, jumlah tertinggi cluster ada pada cluster 0 dengan jumlah 389 sedangkan cluster 1 dengan jumlah 10.



```
df['kluster'].value_counts()
```

```
2    310
0     79
1     10
```

Kesimpulan yang didapat dari proses clustering dengan n-cluster sebanyak 3 yang telah dilakukan pada dataset *Exasens*, jumlah tertinggi cluster ada pada cluster 0 dengan jumlah 79 sedangkan cluster 1 dengan jumlah 10 dan cluster 2 sebanyak 310.

4.2. Accepted Papers

Menampilkan visualisai hasil cluster K-Means

```
fig, ax = plt.subplots()
sct = ax.scatter(x_scaled[:,1], x_scaled[:,0], s = 100,
c = papers_new.kluster, marker = "o", alpha = 0.5)
centers = kmeans.cluster_centers_
ax.scatter(centers[:,1], centers[:,0], c='blue', s=200, alpha=0.5);
plt.title("Hasil Klustering K-Means")
plt.xlabel("Scaled Topics")
plt.ylabel("Scaled HighLevelKeywords")
plt.show()
```