Nama : Adriansyah Dwi Rendragraha NIM : 1301174385 Kelas : IF4110

LAPORAN TUGAS BESAR MACHINE LEARNING

Pada tugas ini diberikan problem untuk menklustering dan mengklasifikasi dari data yang diberikan. Data yang saya dapat merupakan data Airbnb. Dimana contoh head datanya adalah sebagai berikut.

```
df = pd.read_csv('air_bnb.csv')
df.head()
```

	id	name	host_id	host_name	neighbourhood_group	neighbourhood	latitude	longitude	room_type	price	minimum_nights	number_of_reviews
0	2015	Berlin-Mitte Valuel Quiet courtyard/very central	2217	Ian	Mitte	Brunnenstr. Süd	52.534537	13.402557	Entire home/apt	60	4	11
1	2695	Prenzlauer Berg close to Mauerpark	2986	Michael	Pankow	Prenzlauer Berg Nordwest	52.548513	13.404553	Private room	17	2	
2	3176	Fabulous Flat in great Location	3718	Britta	Pankow	Prenzlauer Berg Südwest	52.534996	13.417579	Entire home/apt	90	62	14
3	3309	BerlinSpot Schöneberg near KaDeWe	4108	Jana	Tempelhof - Schöneberg	Schöneberg-Nord	52.498855	13.349065	Private room	26	5	2
4	7071	BrightRoom with sunny greenview!	17391	Bright	Pankow	Helmholtzplatz	52.543157	13.415091	Private room	42	2	19

Untuk pengecekan fiturnya, pertama melakukan cek kesetiap kolom dengan melihat banyak nilai unik yang ada pada kolom tersebut. Untuk setiap kolom nya akan mendapatkan hasil sebagai berikut.

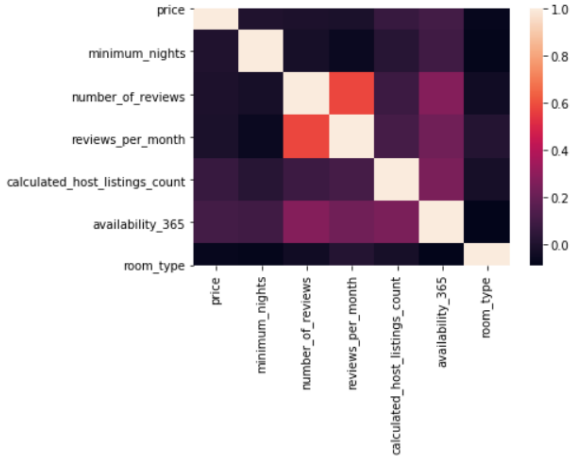
```
id : 22552
name : 21874
host_id : 19180
host_name : 5998
neighbourhood_group : 12
neighbourhood : 136
latitude : 22552
longitude : 22552
room_type : 3
price : 295
minimum_nights : 102
number_of_reviews : 306
last_review : 1313
reviews_per_month : 4682
calculated_host_listings_count : 23
availability_365 : 366
```

Dengan melihat nilai unik yang ada pada setiap kolom, langkah selanjutnya adalah memilih kolom yang akan digunakan menjadi fitur. Fitur tersebut yang nantinya akan digunakan ke dalam klasifikasi dan klustering nya. Dan juga untuk data yang bernilai “NaN” akan di convert menjadi 0. Untuk data yang bersifat kategorikal, dapat di rubah menjadi angka. Sehingga data dari berbentuk kategorikal string, menjadi label.

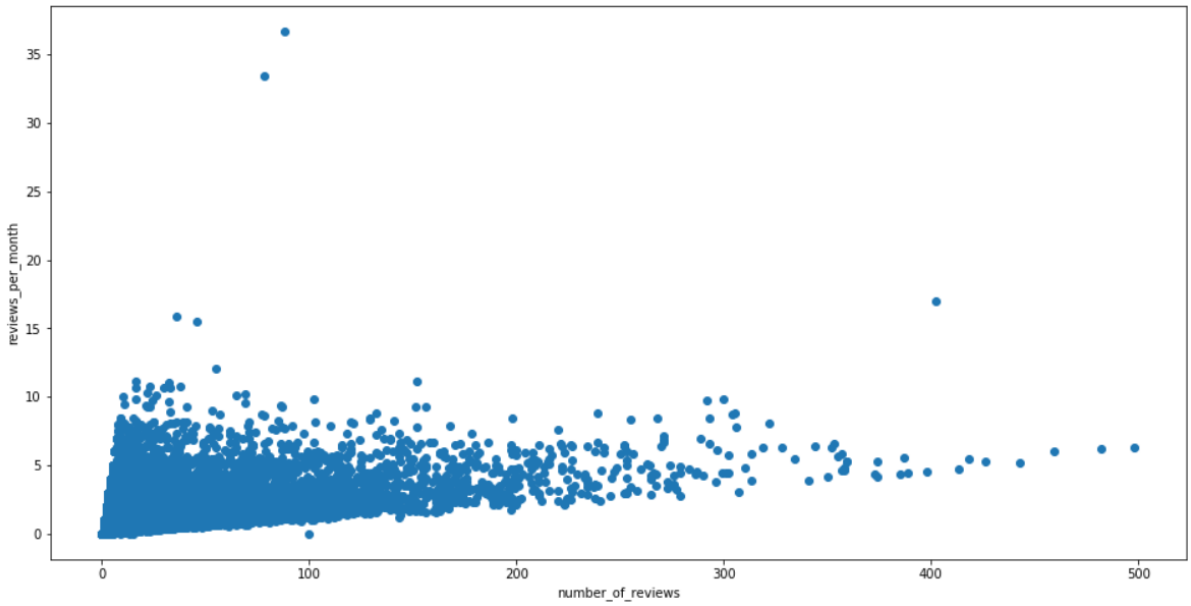
	price	minimum_nights	number_of_reviews	reviews_per_month	calculated_host_listings_count	availability_365	room_type
count	22552.000000	22552.000000	22552.000000	22552.00000	22552.000000	22552.000000	22552.000000
mean	67.143668	7.157059	17.840679	0.93845	1.918233	79.852829	0.537691
std	220.266210	40.665073	36.769624	1.43598	3.667257	119.368162	0.524254
min	0.000000	1.000000	0.000000	0.00000	1.000000	0.000000	0.000000
25%	30.000000	2.000000	1.000000	0.06000	1.000000	0.000000	0.000000
50%	45.000000	2.000000	5.000000	0.34000	1.000000	4.000000	1.000000
75%	70.000000	4.000000	16.000000	1.18000	1.000000	129.000000	1.000000
max	9000.000000	5000.000000	498.000000	36.67000	45.000000	365.000000	2.000000

Fitur yang tadi nya ada memiliki 15 kolom, sekarang sudah menjadi 7 kolom. Isi dari datanya sekarang semua sudah dalam berbentuk angka (tidak ada yang string atau kalimat). Kemudian dataframe tersebut dapat dilihat ke dalam heatmap, dimana fitur mana saja yang berkorelasi.

```
price                             -0.118139
minimum_nights                    -0.064755
number_of_reviews                 -0.040147
reviews_per_month                  0.031733
calculated_host_listings_count    -0.017058
availability_365                  -0.079867
room_type                          1.000000
Name: room_type, dtype: float64
```



Jika dilihat dalam heatmap nya, ada 1 fitur yang terlihat berkorelasi, yaitu fitur number_of_reviews dan reviews_per_month. Sehingga fitur ini akan menjadi fokus utamanya. Dan pada saat dilihat di persebaran data nya, dapat dilihat data pada fitur yang berkaitan ada yang masih outlier.

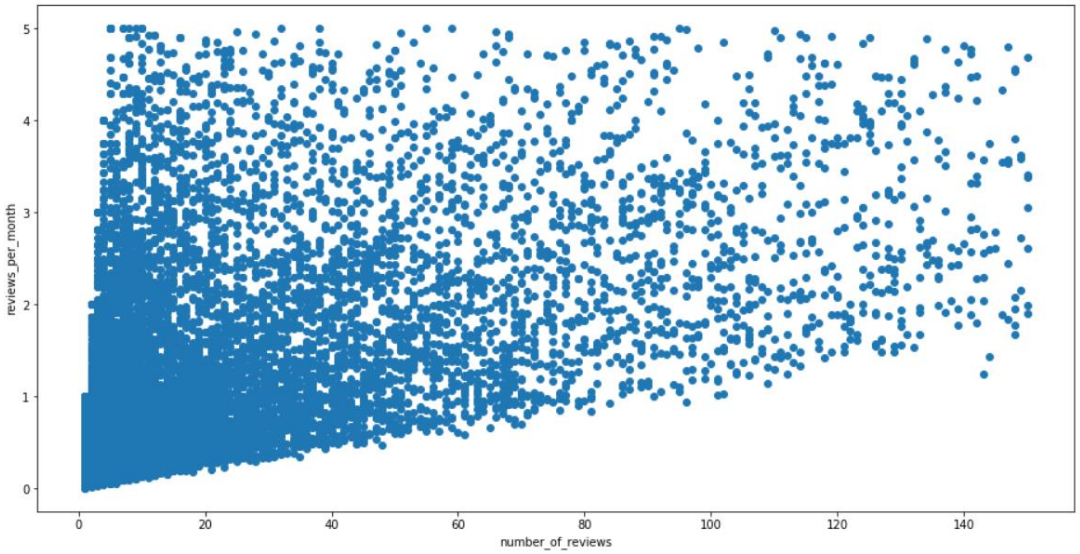


Dalam kasus ini, saya mengurangi data tersebut dengan cara manual. Dengan mengambil acuan dari persebaran data diatas.

```
feat = Test_df[Test_df.number_of_reviews <= 150]
feat = feat[feat.reviews_per_month <= 5]
feat = feat[feat.reviews_per_month != 0]
feat.shape
```

```
(17802, 7)
```

Setelah dilakukan pengurangan data, persebaran datanya terlihat jauh lebih baik dari sebelumnya



Karena sudah didapatkan hasil dari persebaran nya, maka feature yang sudah di dapat, disimpan ke dalam variable data_feature untuk data nya, dan label_feature untuk labelnya.

```
data_feature = feat.drop('room_type', axis=1)
label_feature = feat['room_type']
data_feature.to_csv('data_feature.csv')
label_feature.to_csv('label_feature.csv')
```

CLASSIFICATION

Pada klasifikasi ini saya menggunakan 2 model classifier, yaitu [1] SVM dan [2] Random Forest.

Sebelum melakukan klasifikasi, dapat dicek terlebih dahulu tiap label memiliki berapa banyak data.

```
label_feature.value_counts()
```

```
1    9137
0    8454
2     211
Name: room_type, dtype: int64
```

Setelah dilihat, ternyata datanya imbalance, dengan label 1 memiliki 9137 data, label 0 memiliki 8454 data, dan label 2 memiliki 211 data. Untuk mendapat data yang balance, saya melakukan undersampling terhadap data pada label 1 dan label 0. Undersampling yang digunakan adalah Instance Hardness Threshold. Kenapa saya menggunakan undersampling ini, dikarenakan setelah mencoba yang lain, hasil yang paling baik didapat dengan undersampling Instance Hardness Threshold. Setelah dilakukan undersampling, data pada label terlihat lebih balance.

```
label_feature_class.value_counts()
```

```
1    428
0    260
2    211
Name: room_type, dtype: int64
```

Selanjutnya data dilakukan normalisasi atau standarisasi. Sehingga lebih baik performansi nya dalam melakukan klasifikasi.

```
st = StandardScaler().fit(data_feature_class)
data_feature_class = pd.DataFrame(st.transform(data_feature_class))

data_feature_class.head()
```

	0	1	2	3	4	5
0	1.658136	3.935395	2.170736	-0.001309	-0.094310	1.327484
1	-0.019319	3.935395	0.938005	-0.220682	-0.212724	1.695331
2	0.798952	-0.028863	-0.374256	-0.629513	-0.331139	-0.563080
3	0.533014	3.935395	1.653784	0.008663	0.024104	1.832205
4	0.737582	0.187369	-0.294725	-0.609570	-0.331139	2.063179

Kemudian data feature dan label feature nya dibagi menjadi train dan test. Dimana train untuk melatih model nya, sedangkan test untuk mengevaluasi model yang sudah didapat. Hasil evaluasi menggunakan classification report. Dikarenakan classification report mencakup score yang dibutuhkan dalam mengevaluasi klasifikasi. Hasil nya dapat dilihat sebagai berikut.

1. Random Forest

Random Forest

```
rdcf = RandomForestClassifier(max_depth=50, random_state=0)
rdcf.fit(X_train,y_train)
y_pred = rdcf.predict(X_test)

print("Classification report")
print(classification_report(y_test,y_pred))
```

```
Classification report
              precision    recall  f1-score   support

           0       0.98      0.99      0.98        87
           1       0.90      0.94      0.92       142
           2       0.85      0.75      0.80        68

    accuracy                           0.91       297
   macro avg       0.91      0.89      0.90       297
weighted avg       0.91      0.91      0.91       297
```

2. SVM

SVM

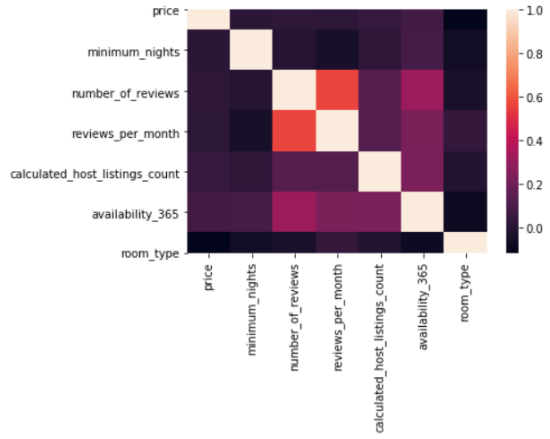
```
svm = SVC()
svm.fit(X_train,y_train)
y_pred = svm.predict(X_test)

print("Classification report")
print(classification_report(y_test,y_pred))
```

```
Classification report
              precision    recall  f1-score   support

           0       0.90      0.97      0.94        77
           1       0.85      1.00      0.92       143
           2       0.96      0.56      0.70        77

    accuracy                           0.88       297
   macro avg       0.90      0.84      0.85       297
weighted avg       0.89      0.88      0.87       297
```

CLUSTERING

Untuk tahapan clustering ini, awalnya lihat atau cek heatmap data, fitur apa saja yang berkorelasi sehingga dapat digunakan dalam clustering.

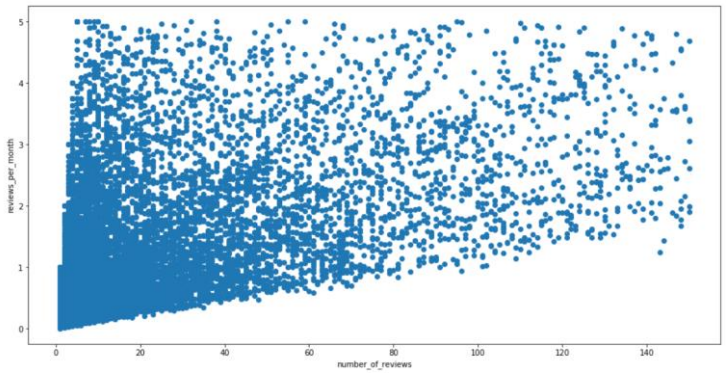
```
price                            -0.118139
minimum_nights                   -0.064755
number_of_reviews                -0.040147
reviews_per_month                 0.031733
calculated_host_listings_count   -0.017058
availability_365                 -0.079867
room_type                         1.000000
Name: room_type, dtype: float64
```



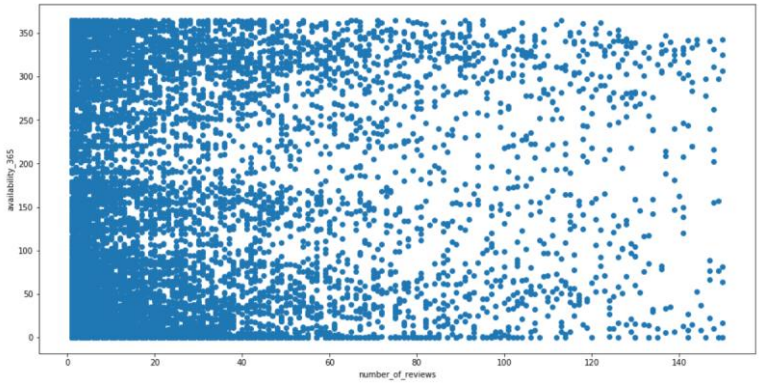
Disini saya mencoba menggunakan 2 input data yang berbeda, dimana jika dilihat dari heatmap diatas maka Input 1 (Feature 1) = [number_of_reviews , reviews_per_month] dan Input 2 (Feature 2) = [number_of_reviews , availability_365].

Jika dilihat plot persebaran data nya adalah sebagai berikut.

1. Input 1 (Feature 1) = [number_of_reviews , reviews_per_month]



2. Input 2 (Feature 2) = [number_of_reviews , availability_365]



Metode clustering yang digunakan adalah K-Means clustering, dimana nanti nya akan membentuk cluster sebanyak K yang diinputkan.

Dalam codenya, saya membentuk suatu fungsi untuk menjalankan K-Means nya.

1. Initialisasi Centroid

```
def initialCentro(K, feature, df, info):
    centro =[[],[]]
    for i in range(K):
        x = df[info].values[np.random.randint(0,feature-1)]
        centro[0].append(x[0])
        centro[1].append(x[1])

    return centro
```

Disini variable centro merupakan tempat penyimpanan titik centroids, pada fungsi disini centro di initialisasi terlebih dahulu. Dilanjut kedalam perulangan for yang melakukan pemilihan sebanyak K centroid awal dari data featurenya. Setelah didapat akan mereturn variable centro.

2. Mencari Euclidian Distance

```
def Euclidian(K,feature,centro, df, info):
    ed = []
    for i in range(feature):
        tmp = []
        x = df[info].values[i]
        for j in range(K):
            y = ((x[0]-centro[0][j])**2+(x[1]-centro[1][j])**2)
            tmp.append(mt.sqrt(y))
        ed.append(tmp)
    return ed
```

Pada fungsi Euclidian, variable ed merupakan tempat penyimpanan jarak Euclidian nya. Kemudian dalam for nya ini dilakukan perhitung Euclidian distance nya. Setelah selesai akan mereturn variable ed.

3. Main Fungsi Kluster

```
def cluster(K, iterasi, centro, feature, df, info):
    for i in range(iterasi):
        print('iterasi -',i)
        ED = Euclidian(K,feature,centro, df, info)
        C = np.argmin(ED,axis=1)+1

        out = [[],[],[]]
        for i in range(len(df)):
            x = df[info].values[i]
            if C[i] == 1:
                out[0].append([x[0],x[1]])
            elif C[i] == 2:
                out[1].append([x[0],x[1]])
            elif C[i] == 3:
                out[2].append([x[0],x[1]])

        centro =[[],[]]
        for i in range(K):
            sx,sy = np.mean(out[i], axis=0)
            centro[0].append(sx)
            centro[1].append(sy)

    return centro, out
```

Ini merupakan fungsi utama dari klusternya, for paling luar melakukan iterasi sebanyak yang diinginkan untuk mencapai hasil klustering nya. Pada variable out, merupakan variable yang digunakan untuk menyimpan data pada cluster tertentu. Kemudian dilanjutkan dengan mencari Euclidian Distance, dan variable C merupakan urutan cluster untuk setiap datanya. Masuk ke dalam for , disini variable out dimasukan data dengan penyesuaian data yang masuk ke cluster apa. Lanjut ke for berikutnya dimana menghitung rata rata titik centroids baru yang digunakan untuk iterasi selanjutnya.

4. Visualisasi Hasil Kluster

```
def plot(K, centro, out):
    xa = []
    for i in range(len(out)):
        xa.append(np.array(out[i]))
    out = np.array(xa)
    centro = np.array(centro)
    color=['r','b','g']
    labels=['cluster1','cluster2','cluster3']
    for i in range(K):
        plt.scatter(out[i][:,0],out[i][:,1],c=color[i],label=labels[i])
    plt.scatter(centro[0,:],centro[1,:],s=200,c='yellow',label='Centroids')
    plt.xlabel('feature-x')
    plt.ylabel('feature-y')
    plt.legend()
    plt.show()
```

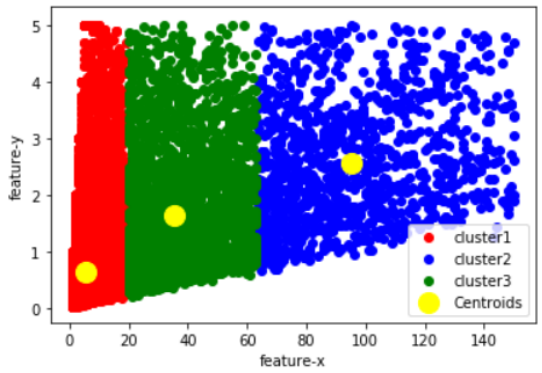
Pada fungsi ini, melakukan plot terhadap data dan centroid sesuai hasil dari klustering. Sebelum dilakukan plot, variable centro dan out diubah ke dalam array terlebih dahulu. Setelah itu, baru dilakukan visualisasi.

Kemudian setelah dilakukan pembangunan fungsi dan procedure yang dibutuhkan, maka akan dicoba clustering dengan Feature 1 dan Feature 2. Dimana dengan parameter sebagai berikut.

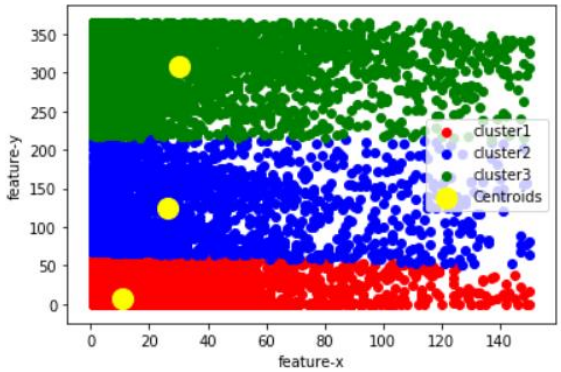
```
# Parameter
K = 3
iterasi = 10
```

Disini saya menggunakan K (banyak cluster) sebanyak 3 dengan iterasi 10 kali. Dengan hasil untuk Feature 1 dan Feature 2 adalah sebagai berikut.

1. Feature 1



2. Feature 2



KESIMPULAN

Dari percobaan yang telah dilakukan, pada klasifikasi data imbalance sangat mempengaruhi performansi dari model yang telah dibangun. Pada klustering, pemilihan fitur dan algoritma klustering dapat menentukan apakah kluster ini dapat mengelompokkan secara baik atau tidak. Penanganan dataset yang ingin dipakai juga menjadi krusial, dikarenakan jika salah dalam penanganan nya akan berakibat langsung ke klasifikasi dan klusteringnya.

Saran dan improvement ke depannya dari hasil percobaan saya ini adalah.

1. Dapat melakukan pemilihan data atau fitur yang lebih baik lagi.
2. Penanganan data outliner dan data imbalance yang lebih maksimal.
3. Penggunaan dan penentuan algoritam klustering yang beragam dan juga secara komputasi lebih efisien.