# MACHINE LEARNING
# LAPORAN TUGAS BESAR


Disusun Oleh:

Ilham Wahyu Adli (1301173380)

IF 41-06

## PROGRAM STUDI S1 INFORMATIKA
## FAKULTAS INFORMATIKA
## TELKOM UNIVERSITY
## BANDUNG
## 2020

# 1 Preprocessing Data

Pada tahap ini akan dilakukan preprocessing data yang akan mencari data-data yang dapat merusak algoritma. Seperti: pencarian duplikat, data kosong, relevansi dll. Tahap ini sangat penting dalam pembangunan model klasifikasi maupun *clustering*.

## 1.1 Sample Dataset

Dataset yang digunakan adalah dataset air_bnb listing dengan jumlah atribut sebanyak 16 dan 22552 record data. Load dataset dapat dilakukan menggunakan library pandas dengan cara memasukan syntax seperti berikut:

In [145]:
```
dataset = pd.read_csv('air_bnb.csv')
dataset
```

Out[145]:

| | id | name | host_id | host_name | neighbourhood_group | neighbourhood | latitude | longitude | room_type | price | minimum_nights | nun |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 2015 | Berlin-Mitte Value! Quiet courtyard/very central | 2217 | Ian | Mitte | Brunnenstr. Süd | 52.534537 | 13.402557 | Entire home/apt | 60 | 4 | |
| 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 | |
| 3 | 3309 | BerlinSpot Schöneberg near KaDeWe | 4108 | Jana | Tempelhof - Schöneberg | Schöneberg-Nord | 52.498855 | 13.349065 | Private room | 26 | 5 | |
| 4 | 7071 | BrightRoom with sunny greenview! | 17391 | Bright | Pankow | Helmholtzplatz | 52.543157 | 13.415091 | Private room | 42 | 2 | |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 22547 | 29856708 | Cozy Apartment right in the center of Berlin | 87555909 | Ulisses | Mitte | Brunnenstr. Süd | 52.533865 | 13.400731 | Entire home/apt | 60 | 2 | |

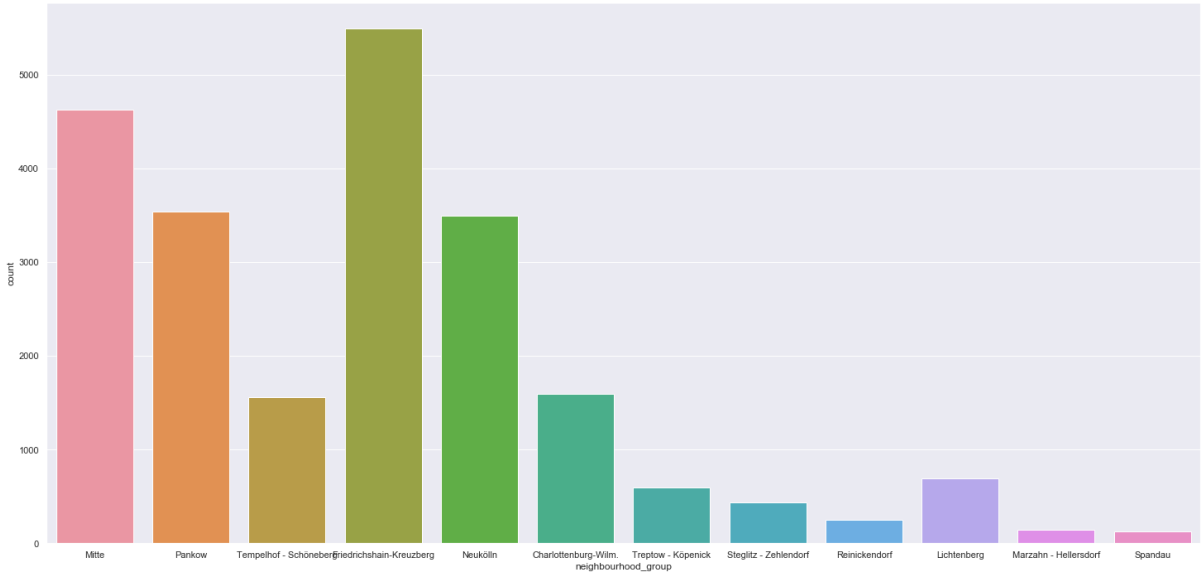
## 1.2 Eksplorasi Dataset

Tahap ini dilakukan untuk mencari tau nilai dari data yang dimiliki, apakah data yang akan diolah memiliki nilai yang tidak relevan.

### 1.2.1 Neighbourhood_group

```
import seaborn as sns
from numpy import median
ng = dataset.iloc[:,[4]]
print(Counter(ng["neighbourhood_group"]))
sns.set(style="ticks", color_codes=True)
sns.set(rc={'figure.figsize':(25,12)})
ax = sns.countplot(x=ng["neighbourhood_group"], data=ng)
```

```
Counter({'Friedrichshain-Kreuzberg': 5497, 'Mitte': 4631, 'Pankow': 3541, 'Neukölln': 3499, 'Charlottenburg-Wilm.': 1592, 'Tempelhof - Schöneberg': 1560, 'Lichtenberg': 688, 'Treptow - Köpenick': 595, 'Steglitz - Zehlendorf': 437, 'Reinickendorf': 247, 'Marzahn - Hellersdorf': 141, 'Spandau': 124})
```

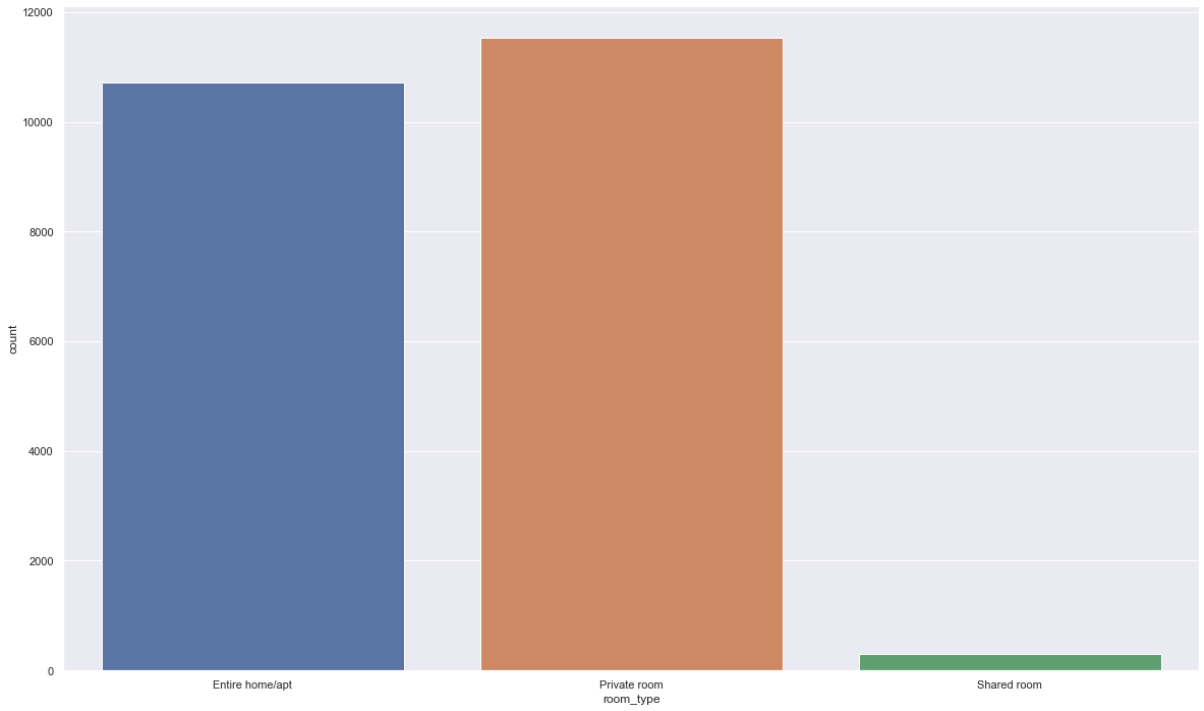



Nilai pada data tidak ada yang memiliki nilai yang tidak relevan semuanya mengenai neighbourhood group.

### 1.2.2 Room type

```
import seaborn as sns
from numpy import median
ng = dataset.iloc[:,[8]]
print(Counter(ng["room_type"]))
sns.set(style="ticks", color_codes=True)
sns.set(rc={'figure.figsize':(20,12)})
ax = sns.countplot(x=ng["room_type"], data=ng)
```

```
Counter({'Private room': 11534, 'Entire home/apt': 10722, 'Shared room': 296})
```



Menggunakan cara yang sama seperti neighbourhood_group, atribut ini tidak memiliki value yang bermasalah.

### 1.2.3 Price

```
from matplotlib.pyplot import figure
figure(figsize=(10,2))
ng = dataset.iloc[:,[9]]
y = []
for i in range(len(ng)):
    y.append(0)
plt.scatter(ng, y, s = 100, c = 'black')
plt.title('Price')
plt.show()
```


Selanjutnya atribut price, dapat dilihat terdapat price yang memiliki angka yang sangat jauh dari yang lain yaitu ketika harga > 2000. Maka akan dieksplorasi lebih mendalam pada atribut price.

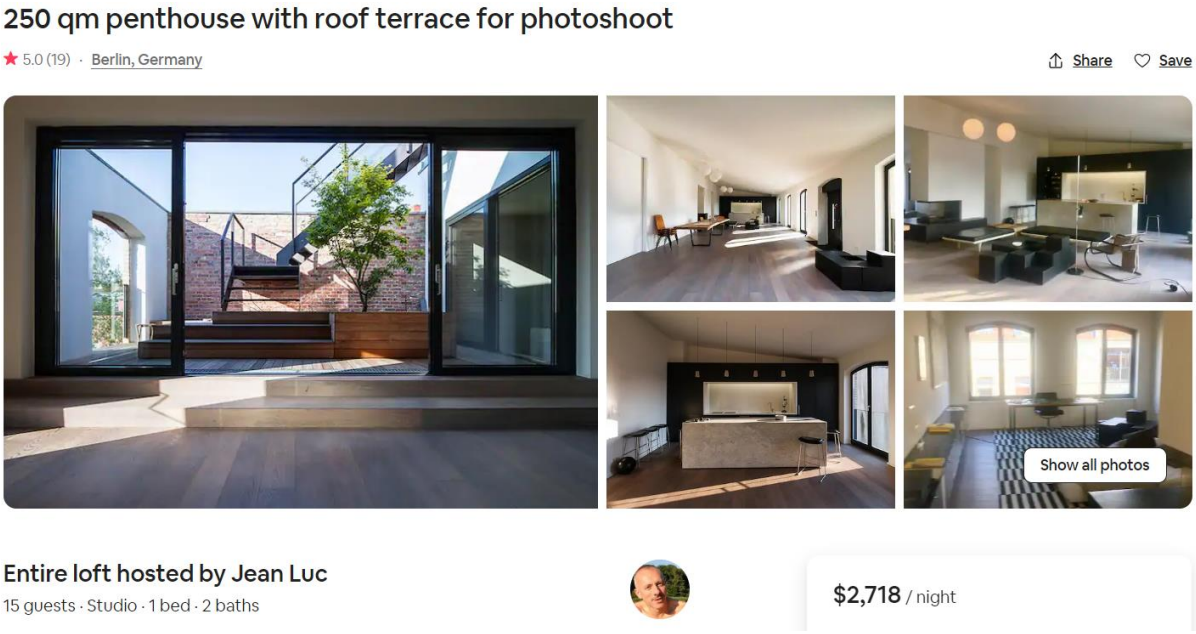
```
unusualPrice = []
unusualPrice.extend(dataset[dataset['price'] > 2000].index.tolist())
unusualPrice.sort()
print(unusualPrice)
```

```
[766, 1990, 4117, 4573, 4744, 6655, 9528, 15665, 16915, 17250, 19372, 19373, 19407, 19409, 19410, 19411, 19412, 19413, 19414, 1
9415, 19655, 19656, 19657, 19658, 19659, 19660, 19661, 19662, 19663, 19700, 19729, 21389]
```

```
dataUnusualPrice = []
for i in range (len(unusualPrice)):
    print(dataset.iloc[unusualPrice[i],:])
```

```
Name: 766, dtype: object
id                                                              2860420
name                              250 qm penthouse with roof terrace for photoshoot
host_id                                                        14623546
host_name                                                      Jean Luc
neighbourhood_group                                 Friedrichshain-Kreuzberg
neighbourhood                                     Tempelhofer Vorstadt
latitude                                                         52.4935
longitude                                                        13.4205
room_type                                                 Entire home/apt
price                                                               2500
minimum_nights                                                         1
number_of_reviews                                                     19
last_review                                                   2016-04-13
reviews_per_month                                                   0.35
calculated_host_listings_count                                         1
availability_365                                                      89
Name: 1990, dtype: object
id                                      6408850
name                            12 m2 of relaxation ***
```

Salah satu datanya adalah “250 qm penthouse with roof terrace for photoshoot” setelah melakukan cross check pada website Airbnb didapatkan listing



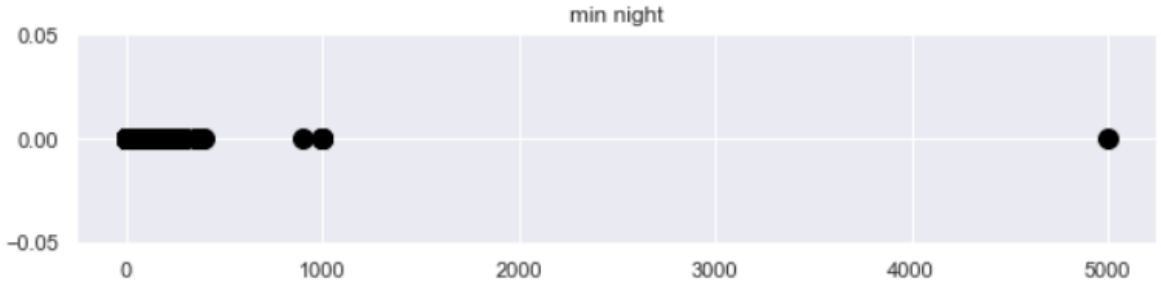

*Sumber 1: https://www.airbnb.com/rooms/2860420?s=_Dg2&source_impression_id=p3_1587298398_vhVs%2FO%2BJr7DV9IOZ&guests=1&adults=1*

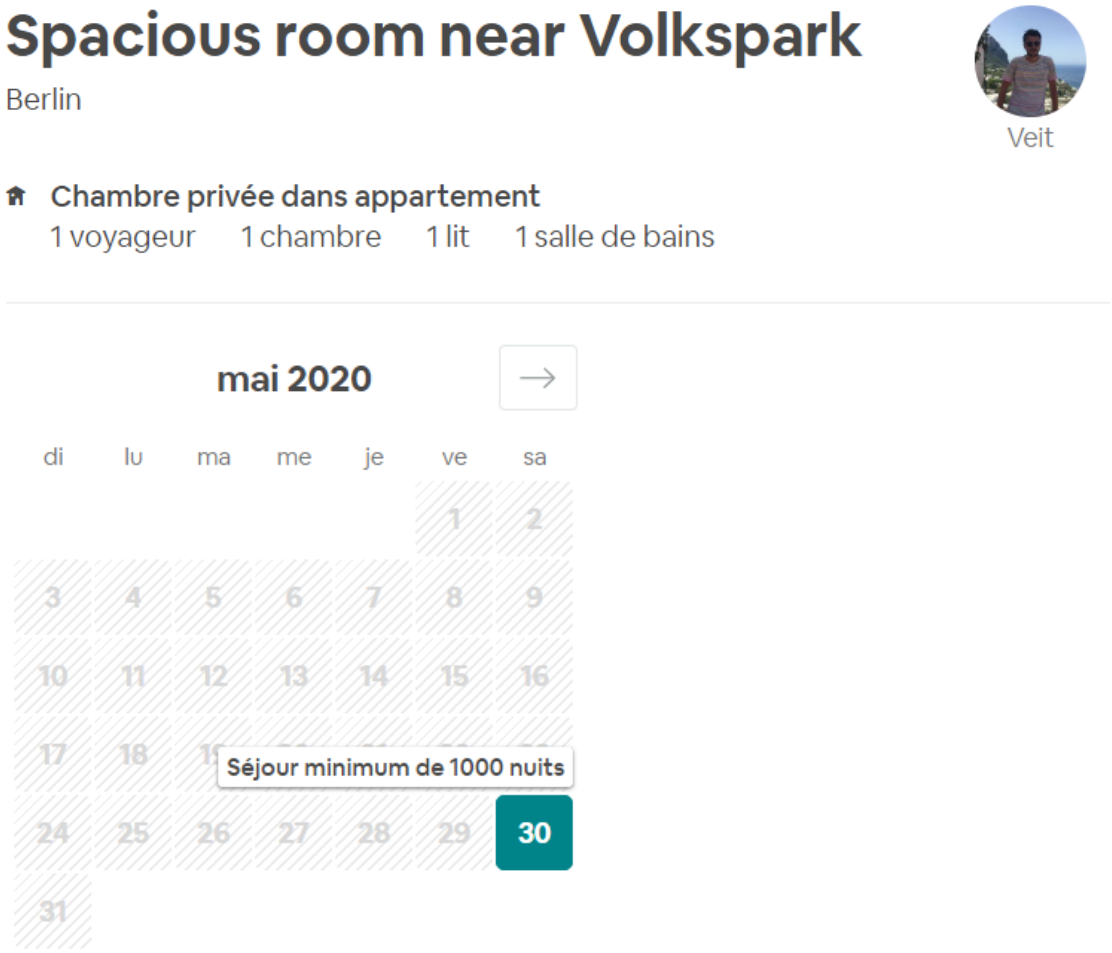
Hal ini membuktikan price dengan range > 2000 dapat digunakan sebagai penentuan calon customer dalam *clustering* seperti calon customer yang memiliki production house.

### 1.2.4 Minimum Night

Minimum night juga memiliki nilai yang tidak wajar dan akan dilakukan crosscheck seperti price.

```
from matplotlib.pyplot import figure
from pylab import *
figure(figsize=(10,2))
ng = dataset.iloc[:,[10]]
y = []
for i in range(len(ng)):
    y.append(0)
plt.scatter(ng, y, s = 100, c = 'black')
plt.title('min night')
plt.show()
```



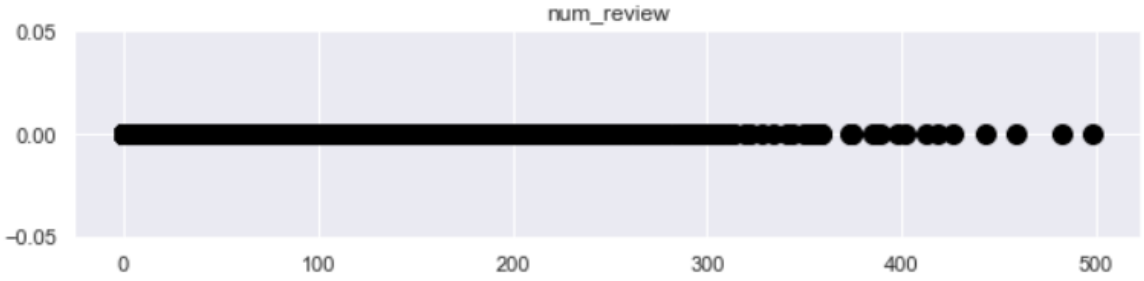



*Sumber 2: https://fr.airbnb.ca/rooms/6704144?source_impression_id=p3_1587299751_2XQ3Xx2y56vzCqhA&guests=1&adults=1&check_in=2020-05-30*

Dan pilihan minimal malam adalah 1000, sehingga data ini diperlukan dalam penentuan calon customer dalam *clustering* karena dalam tahap *clustering* yang akan dilakukan terdapat calon customer yang menyewa dengan waktu yang sangat lama seperti kostan atau kontrakan.

### 1.2.5 Number of Review

```
from matplotlib.pyplot import figure
from pylab import *
figure(figsize=(10,2))
ng = dataset.iloc[:,[11]]
y = []
for i in range(len(ng)):
    y.append(0)
plt.scatter(ng, y, s = 100, c = 'black')
plt.title('num_review')
plt.show()
```




```
for i in range (len(unusualReview)):
    print(dataset.iloc[unusualReview[i],:])
```

```
id                                             292864
name                           Lounge Room –  Alex in 5 Min
host_id                                        286494
host_name                                    Project A
neighbourhood_group                             Pankow
neighbourhood                 Prenzlauer Berg Südwest
latitude                                        52.53
longitude                                     13.4176
room_type                                Private room
price                                             50
minimum_nights                                     1
number_of_reviews                                498
last_review                               2018-10-24
reviews_per_month                               6.29
calculated_host_listings_count                     3
availability_365                                 242
Name: 327, dtype: object
```

www.airbnb.co.id › Jerman › Berlin ▾

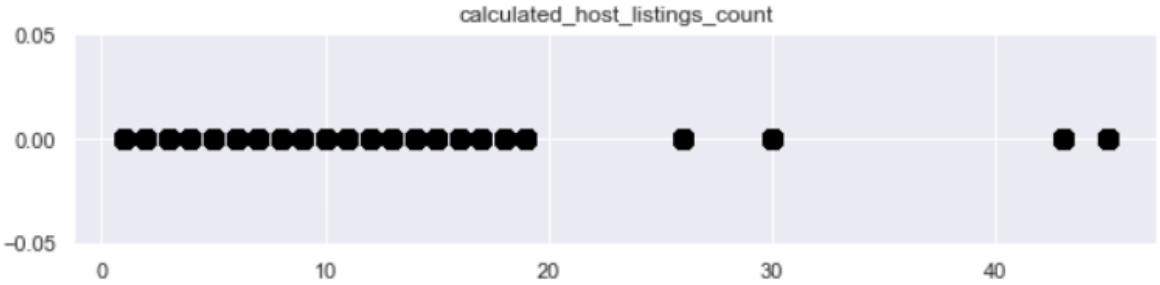
Lounge Room – Alex in 5 Min - Apartemen untuk Disewakan ...

7 Mar 2020 - Kamar pribadi dengan harga Rp830272. We are registered at the Berlin city council, so you can be save to rent a legal place. Our **room** is located ...

★★★★½ Skor: 4,5 - 612 ulasan

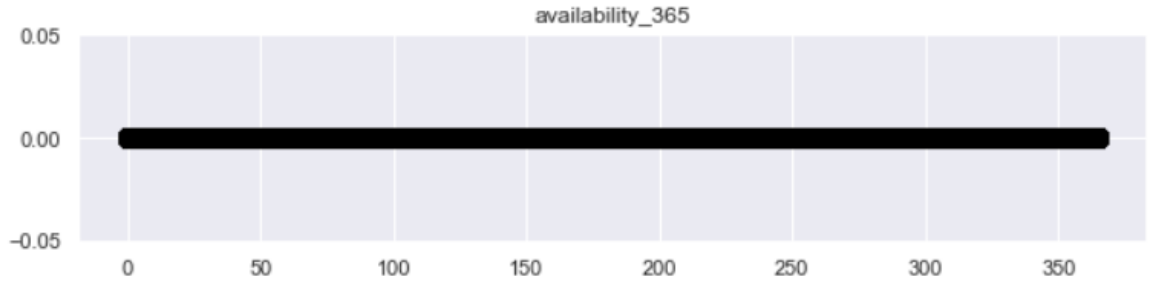
### 1.2.6 Calculated Host Listing Count

```
from matplotlib.pyplot import figure
from pylab import *
figure(figsize=(10,2))
ng = dataset.iloc[:,[14]]
y = []
for i in range(len(ng)):
    y.append(0)
plt.scatter(ng, y, s = 100, c = 'black')
plt.title('calculated_host_listings_count')
plt.show()
```




### 1.2.7 Availability 365

```
from matplotlib.pyplot import figure
from pylab import *
figure(figsize=(10,2))
ng = dataset.iloc[:,[15]]
y = []
for i in range(len(ng)):
    y.append(0)
plt.scatter(ng, y, s = 100, c = 'black')
plt.title('availability_365')
plt.show()
```




### 1.2.8 Kesimpulan

Kesimpulan dari pencarian unusual data, di dalam dataset terdapat value yang tidak wajar tetapi setelah dilakukan cross check, ternyata data listing tersebut benar dan dapat digunakan untuk melakukan klasifikasi maupun *clustering*.

## 1.3 Pencarian duplikat

Dalam library pandas juga terdapat syntax yang dapat memudahkan dalam pencarian duplikat yaitu:

```
In [112]:
1 print(len(dataset))
2 dataset.drop_duplicates()
3 print(len(dataset))
```

```
22552
22552
```

Baris 1 memberikan kita informasi berapa jumlah data didalam dataset, kemudian pada baris 2 melakukan delete data duplikat, dan baris 3 adalah jumlah data setelah dihapus. Hal ini menunjukan jika dataset tidak memiliki data yang duplikat.

### 1.4 Data Nilai Kosong

Untuk dapat melihat nilai kosong kita dapat memanggil salah satu fungsi dari *library* pandas.

```
1 dataset.isnull().sum()
```

```
id                                  0
name                               59
host_id                             0
host_name                          26
neighbourhood_group                 0
neighbourhood                       0
latitude                            0
longitude                           0
room_type                           0
price                               0
minimum_nights                      0
number_of_reviews                   0
last_review                      3908
reviews_per_month                3914
calculated_host_listings_count      0
availability_365                    0
dtype: int64
```

Dapat dilihat dataset kita memiliki nilai null pada atribut name, host_name, last_review dan reviews_per_month. Dikarenakan jumlah data awal ada 22552 record, sehingga pemilihan metode untuk menangani masalah ini adalah penghapusan record tersebut. Index dari record yang memiliki nilai null tersebut dimasukan kedalam variabel bernama *nullValue*, kemudian akan dihapus data yang duplikat di dalam list *nullValue* tersebut. Sehingga didapatkan terdapat 3965 data bernilai null. Implementasi:

```
nullValue = []
nullValue.extend(dataset[dataset['name'].isnull()].index.tolist())
nullValue.extend(dataset[dataset['host_name'].isnull()].index.tolist())
nullValue.extend(dataset[dataset['last_review'].isnull()].index.tolist())
nullValue.extend(dataset[dataset['reviews_per_month'].isnull()].index.tolist())
nullValue = list(dict.fromkeys(nullValue))
len(nullValue)
```

```
3965
```

```
#delete null
print("Awal: ", len(dataset))
dataset.drop(nullValue, inplace = True)
print("Akhir: ", len(dataset))
print(dataset.isnull().sum())
print(dataset.info())
```

```
Awal:  22552
Akhir:  18587
id                                0
name                              0
host_id                           0
host_name                         0
neighbourhood_group               0
neighbourhood                     0
latitude                          0
longitude                         0
room_type                         0
price                             0
minimum_nights                    0
number_of_reviews                 0
last_review                       0
reviews_per_month                 0
calculated_host_listings_count    0
availability_365                  0
```

Setelah melakukan penghapusan nilai kosong data yang tersisa sebanyak 18587 record. Pemilihan penghapusan data ini dipilih karena alasan pada pemilihan fitur klasifikasi. Numbers_of_revies dan revies_per_month memiliki nilai yang jauh lebih kecil dibandingkan price, availability_365, dan minimum_nights. Jika ke-2 data tersebut memiliki skor yang lebih besar penggunaan mean imputation.

## 2 Batasan Penelitian

Penelitian akan dilakukan menggunakan 2 studi kasus yang pertama adalah dengan outlier, maksud dari studi kasus yang pertama adalah kita mengambil keseluruhan data dan hanya menghilangkan data yang memiliki nilai null. Sedangkan pada kasus ke 2 adanya penggunaan outlier ketika harga > 300. Kedua studi ini akan dilakukan tahapan yang sama yaitu *classification* dan *clustering.*

Klasifikasi adalah suatu pengelompokan data dimana data yang digunakan tersebut mempunyai kelas label atau target. Sehingga algoritma-algoritma untuk

menyelesaikan masalah klasifikasi dikategorisasikan ke dalam supervised learning atau pembelajaran yang diawasi. Sedangkan *clustering* adalah tugas membagi populasi atau titik data menjadi sejumlah kelompok sehingga titik data dalam kelompok yang sama lebih mirip dengan titik data lain dalam kelompok yang sama dan tidak sama dengan titik data di kelompok yang lain. Hal ini pada dasarnya adalah kumpulan benda-benda atas dasar kesamaan dan ketidaksamaan antara mereka. *Clustering* merupakan unsupervised learning sehingga kita membiarkan program untuk membagi data tersebut menjadi beberapa cluster tanpa adanya atribut yang akan menjadi acuannya. Biasanya *clustering* digunakan untuk memprediksi jenis-jenis customer. Hal inilah yang akan dijadikan tujuan dari model *clustering* yang akan dibuat.

## 3 Klasifikasi

Algoritma yang akan digunakan pada penelitian ini adalah K-NN dan Decision Tree. Decision tree digunakan karena sangat baik untuk data dengan jumlah yang banyak, sedangkan K-NN digunakan dikarenakan komputasi yang cepat. Kedua algoritma tadi memiliki kekurangan pada decision tree kekurangannya adalah ketika memiliki outlier (maka dari itu ada 2 studi kasus) dan pada K-NN sulit untuk menentukan nilai K yang optimal.

**Classification**

| Classification Model | Pros | Cons |
|---|---|---|
| Logistic Regression | Probabilistic approach, gives informations about statistical significance of features | The Logistic Regression Assumptions |
| K-NN | Simple to understand, fast and efficient | Need to choose the number of neighbours k |
| SVM | Performant, not biased by outliers, not sensitive to overfitting | Not appropriate for non linear problems, not the best choice for large number of features |
| Kernel SVM | High performance on nonlinear problems, not biased by outliers, not sensitive to overfitting | Not the best choice for large number of features, more complex |
| Naive Bayes | Efficient, not biased by outliers, works on nonlinear problems, probabilistic approach | Based on the assumption that features have same statistical relevance |
| Decision Tree Classification | Interpretability, no need for feature scaling, works on both linear / nonlinear problems | Poor results on too small datasets, overfitting can easily occur |
| Random Forest Classification | Powerful and accurate, good performance on many problems, including non linear | No interpretability, overfitting can easily occur, need to choose the number of trees |

Machine Learning A-Z © SuperDataScience

*Gambar 1 Sumber dari superdatascience.com. https://sds-platform-private.s3-us-east-2.amazonaws.com/uploads/P14-Classification-Pros-Cons.pdf*

## 3.1 Studi Kasus 1

### 3.1.1 Identifikasi Fitur

```
klasifikasiDataset = dataset.copy()
klasifikasiDataset = klasifikasiDataset.drop(['id', 'name', 'host_id', 'host_name' ,'latitude','longitude','last_review'], a
```

```
klasifikasiDataset
```

| | neighbourhood_group | neighbourhood | room_type | price | minimum_nights | number_of_reviews | reviews_per_month | calculated_host_listings_count | avai |
|---|---|---|---|---|---|---|---|---|---|
| 0 | Mitte | Brunnenstr. Süd | Entire home/apt | 60 | 4 | 118 | 3.76 | 4 | |
| 1 | Pankow | Prenzlauer Berg Nordwest | Private room | 17 | 2 | 6 | 1.42 | 1 | |
| 2 | Pankow | Prenzlauer Berg Südwest | Entire home/apt | 90 | 62 | 143 | 1.25 | 1 | |
| 3 | Tempelhof - Schöneberg | Schöneberg-Nord | Private room | 26 | 5 | 25 | 0.39 | 1 | |
| 4 | Pankow | Helmholtzplatz | Private room | 42 | 2 | 197 | 1.75 | 1 | |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | |
| 22449 | Lichtenberg | Neu Lichtenberg | Private room | 25 | 2 | 1 | 1.00 | 1 | |
| 22463 | Treptow - Köpenick | Oberschöneweide | Entire home/apt | 99 | 1 | 5 | 5.00 | 2 | |
| 22475 | Friedrichshain-Kreuzberg | Frankfurter Allee Süd FK | Private room | 25 | 1 | 1 | 1.00 | 1 | |
| 22508 | Pankow | Prenzlauer Berg Nordwest | Entire home/apt | 70 | 1 | 1 | 1.00 | 1 | |
| 22536 | Pankow | Prenzlauer Berg Südwest | Entire home/apt | 150 | 10 | 2 | 2.00 | 1 | |

18587 rows × 9 columns

Atribut yang tidak digunakan adalah 'id', 'name', 'host_id', 'host_name', 'latitude', 'longitude', 'last_review'. Dikarenakan pada dasarnya value atribut ini memiliki nilai yang berbeda-beda sehingga akan merusak akurasi dari model yang akan dibuat. Seperti id dan host_id kedua atribut ini merupakan primary key dari listing Airbnb sehingga jika digunakan akan merusak hasil pemodelan karena tidak ada data yang mirip untuk diklasifikasikan atau dilakukan *clustering*.

### 3.1.2 Labeling Fitur

Memberikan label pada fitur dapat membantu dalam melakukan pemodelan karena akan lebih mudah melakukan kalkulasi jika menggunakan nilai yang dapat dihitung (angka) bukan tulisan (kata/kalimat).

```
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
labelencoder = LabelEncoder()
klasifikasiDataset.iloc[:, [0]] = labelencoder.fit_transform(klasifikasiDataset.iloc[:, [0]])
klasifikasiDataset.iloc[:, [1]] = labelencoder.fit_transform(klasifikasiDataset.iloc[:, [1]])
klasifikasiDataset.iloc[:, [2]] = labelencoder.fit_transform(klasifikasiDataset.iloc[:, [2]])
# onehotencoder = OneHotEncoder(categorical_features = [0])
# test = onehotencoder.fit_transform(test).toarray()
klasifikasiDataset
```

```
C:\Anaconda\lib\site-packages\sklearn\preprocessing\label.py:235: DataConversionWarning: A column-vector y was passed when a 1d
array was expected. Please change the shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

| | neighbourhood_group | neighbourhood | room_type | price | minimum_nights | number_of_reviews | reviews_per_month | calculated_host_listings_count | availa |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 4 | 18 | 0 | 60 | 4 | 118 | 3.76 | 4 | |
| 1 | 6 | 95 | 1 | 17 | 2 | 6 | 1.42 | 1 | |
| 2 | 6 | 98 | 0 | 90 | 62 | 143 | 1.25 | 1 | |
| 3 | 10 | 110 | 1 | 26 | 5 | 25 | 0.39 | 1 | |
| 4 | 6 | 49 | 1 | 42 | 2 | 197 | 1.75 | 1 | |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 22449 | 2 | 76 | 1 | 25 | 2 | 1 | 1.00 | 1 | |
| 22463 | 11 | 84 | 0 | 99 | 1 | 5 | 5.00 | 2 | |
| 22475 | 1 | 32 | 1 | 25 | 1 | 1 | 1.00 | 1 | |
| 22508 | 6 | 95 | 0 | 70 | 1 | 1 | 1.00 | 1 | |
| 22536 | 6 | 98 | 0 | 150 | 10 | 2 | 2.00 | 1 | |

### 3.1.3 Pemilihan Fitur yang Optimal

Pemilihan fitur optimal ini dibantu oleh library sklearn yang dinamakan SelectKBest, dari seluruh atribut yang telah dipilih, akan ditentukan skor dari tiap atribut untuk memprediksi kelas dari room_type.

```
#Feature Selection
from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import chi2
atr2Sellect = klasifikasiDataset.iloc[:,[0,1,3,4,5,6,7,8]]
cls2Sellect = klasifikasiDataset.iloc[:,2]
#apply SelectKBest class to extract top 10 best features
bestfeatures = SelectKBest(score_func=chi2, k='all')
fit = bestfeatures.fit(atr2Sellect,cls2Sellect)
dfscores = pd.DataFrame(fit.scores_)
dfcolumns = pd.DataFrame(atr2Sellect.columns)
#concat two dataframes for better visualization
featureScores = pd.concat([dfcolumns,dfscores],axis=1)
featureScores.columns = ['Attribute','Score']  #naming the dataframe columns
print(featureScores.nlargest(8,'Score'))  #print 10 best features
```

```
                         Attribute          Score
2                            price  110633.472797
7                availability_365   40870.313145
3                  minimum_nights   21912.773781
6  calculated_host_listings_count    6737.238408
4                number_of_reviews    2311.423426
1                    neighbourhood     849.311030
5                reviews_per_month      27.042953
0              neighbourhood_group      18.783984
```

Dapat dilihat atribut price, availability_365, dan minimum_nights merupakan 3 atribut yang paling kuat kandidatnya untuk dijadikan fitur klasifikasi.

### 3.1.4 Pembagian Data Latih dan Data Test

Pembagian dilakukan menggunakan library dari sklearn yaitu train_test_split() yang merupakan model_selection. Dalam tugas besar ini banyaknya jumlah data test adalah sebanyak 20% dari data. Angka 20% tadi merupakan nilai yang cukup. Yang menjadi masalah pada klasifikasi dataset ini adalah kurangnya record kelas ‘Shared Room’ dengan recordnya sebanyak 218 dibandingkan record kelas lain yaitu ‘Private Room’ dengan 9534 record dan 'Entire home/apt' dengan record 8835. Hal ini yang membuat nilai akurasi dapat berkurang, terlebih lagi untuk *precission, recall* dan *f1-score* yang menggunakan nilai support dari kelas yang telah diprediksi.

### 3.1.5 Analisis Hasil Algoritma

#### 3.1.5.1 Hasil Decision Tree

```
# Making the Confusion Matrix
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
print(cm)
from sklearn.metrics import classification_report
print(classification_report(y_test, y_pred))
```

```
[[1343  379    2]
 [ 445 1486   19]
 [   6   28   10]]
              precision    recall  f1-score   support

           0       0.75      0.78      0.76      1724
           1       0.78      0.76      0.77      1950
           2       0.32      0.23      0.27        44

    accuracy                           0.76      3718
   macro avg       0.62      0.59      0.60      3718
weighted avg       0.76      0.76      0.76      3718
```

3.1.5.2 Hasil K-Nearest Neighbour

```
# Making the Confusion Matrix
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
print(cm)
from sklearn.metrics import classification_report
print(classification_report(y_test, y_pred))
```

```
[[1336  388    0]
 [ 340 1610    0]
 [   4   40    0]]
              precision    recall  f1-score   support

           0       0.80      0.77      0.78      1724
           1       0.79      0.83      0.81      1950
           2       0.00      0.00      0.00        44

    accuracy                           0.79      3718
   macro avg       0.53      0.53      0.53      3718
weighted avg       0.78      0.79      0.79      3718
```

3.1.5.3 Kesimpulan

Seperti yang telah dijabarkan dalam bagian penggunaan algoritma, Decision Tree menghasilkan hasil yang lebih optimal walaupun model tersebut memiliki akurasi yang lebih kecil dibandingkan dengan K-NN. Hal ini dikarenakan model Decision Tree berhasil memprediksi kelas Shared Room yang memiliki data support sedikit. Sedangkan K-NN tidak dapat memprediksi benar kelas Shared Room.

## 3.2 Studi Kasus 2

Langkah kerja dari studi 2 ini masih sama seperti studi case 1, tetapi ada penambahan penghapusan outlier, pada studi ini outlier yang diambil ada price > 300. Setelah penghapusan outlier data dari price berubah seperti:


Banyaknya jumlah data yang dihapus bertambah 165 data. Pembagian data latih dan data train juga sama seperti case 1.

### 3.2.1 Analisis Hasil Algoritma

#### 3.2.1.1 Decision Tree

```
# Making the Confusion Matrix
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
print(cm)
from sklearn.metrics import classification_report
print(classification_report(y_test, y_pred))
#memberikan report hasil evaluasi
```

```
[[1359  384    6]
 [ 401 1480   20]
 [   4   35    9]]
                precision    recall  f1-score   support

Entire home/apt       0.77      0.78      0.77      1749
   Private room       0.78      0.78      0.78      1901
    Shared room       0.26      0.19      0.22        48

       accuracy                           0.77      3698
      macro avg       0.60      0.58      0.59      3698
   weighted avg       0.77      0.77      0.77      3698
```

#### 3.2.1.2 K Nearest Neighbour

```
# Making the Confusion Matrix
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
print(cm)
from sklearn.metrics import classification_report
print(classification_report(y_test, y_pred))
#memberikan report hasil evaluasi
```

```
[[1359  390    0]
 [ 320 1581    0]
 [   3   45    0]]
                precision    recall  f1-score   support

Entire home/apt       0.81      0.78      0.79      1749
   Private room       0.78      0.83      0.81      1901
    Shared room       0.00      0.00      0.00        48

       accuracy                           0.80      3698
      macro avg       0.53      0.54      0.53      3698
   weighted avg       0.79      0.80      0.79      3698
```

#### 3.2.1.3 Kesimpulan

Case 2 ini membuktikan bahwa pemberian outlier dapat mempengaruhi hasil algoritma. Dapat dilihat akurasi dari kedua algoritma bertambah 1%, mungkin itu bukan nilai yang besar

dikarenakan outlier yang ditetapkan hanya 165 data. Tetapi hal ini membuktikan bahwa penambahan outlier walaupun sedikit tetap bisa menambahkan kualitas dari pemodelan.

### 3.3 Kesimpulan Hasil Eksperimen

Dari eksperiman yang telah dilakukan dapat diambil kesimpulan bahwa pengurangan outlier dapat mempengaruhi hasil pemodelan untuk melakukan klasifikasi. Sehingga Preprocessing data tahap pencarian outlier sangat penting. Hal ini ditunjukan dengan adanya penambahan akurasi sebanyak 1%.

## 4 *Clustering*

Algoritma yang akan digunakan adalah K-means dan Hiearchical *Clustering*. Dengan Studi Kasus yang sama dengan outlier dan tanpa outlier. Ada batasan dalam pengerjaan Hierarchical *Clustering* yaitu program tidak dapat dijalankan karena RAM yang dibutuhkan tidak mencukupi, sehingga pada Hierarchical *Clustering* akan dibuat model per-neighbourhood group sehingga pemodelan tetap bisa dijalankan. Pengunaan K-means dipilih karena komputasinya yang cepat dan efisien. Dan penggunaaan Hierarchical *Clustering* adalah kita dapat memilih berapa banyak jumlah yang diinginkan dari dendogramnya, sehingga kita tidak perlu melakukan kalkulasi ulang ketika ingin menghitung jumlah cluster.

```
    dm = np.empty((m * (m - 1)) // 2, dtype=np.double)
MemoryError: Unable to allocate 1.89 GiB for an array with shape (254285076,) and data type float64
```

*gambar 2: Error dapat dilihat ketika menconvert file kedalam ekstensi .py dan dijalankan dengan terminal*

### 4.1 Studi Kasus 1

Studi Kasus 1 dikarenakan adanya jumlah outlier sehingga ditambahkan jumlah clusternya untuk mengatasi hal ini, dikarenakan harga tertinggi mencapai 8000. Sehingga dalam kasus ini ada calon pelanggan yang berbeda dengan kasus 2

#### 4.1.1 Pemilihan Fitur

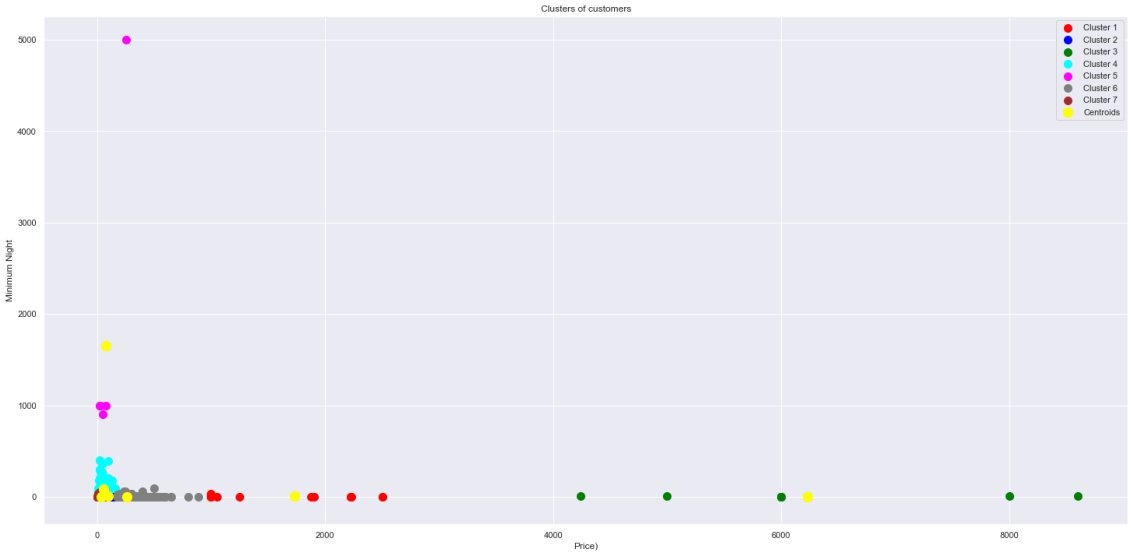
Fitur yang akan digunakan dalam *clustering* ini tidak dapat ditentukan, karena kita tidak tahu acuan apa yang harus diprediksi. Sehingga fitur yang akan digunakan adalah berdasarkan logika orang yang ingin menyewa holel, apartemen, atau rumah, yaitu: harga dan minimum_nights. Cluster calon pelanggan ini akan dibagi menjadi 7, yaitu:

1. Orang yang menyewa dengan budget terbatas dan waktu terbatas
2. Orang yang menyewa dengan budget menengah dan waktu terbatas
3. Orang yang menyewa dengan budget lebih dan waktu terbatas
4. Orang yang menyewa dengan budget lebih dan waktu yang banyak
5. Orang yang menyewa dengan budget besar dan waktu sedikit (production house skala kecil)

6. Orang yang menyewa dengan budget sangat besar dan waktu sedikit (production house skala besar)
7. Orang yang menyewa dengan durasi yang lama (kostan atau kontrakan)

### 4.1.2 Analisis Hasil Algoritma

#### 4.1.2.1 K-Means




```
from sklearn import metrics
from sklearn.metrics import pairwise_distances
label = np.zeros(len(X))
for i in range(len(clustersidx)):
    label[clustersidx[i]] = i
print("Calinski-Harabasz", metrics.calinski_harabasz_score(X, label))
print("Silhouette Coefficient", metrics.silhouette_score(X, label, metric='euclidean'))
```

```
Calinski-Harabasz 27591.134568657442
Silhouette Coefficient 0.6119252314376723
```

#### 4.1.2.2 Hierarchical *Clustering*

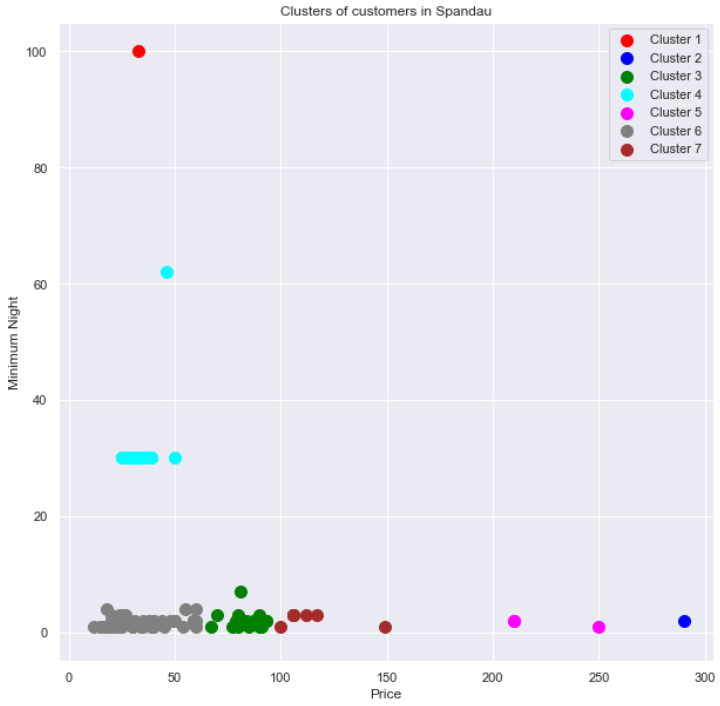
```
print(Counter(clusteringDataset["neighbourhood_group"]))
```

```
Counter({'Friedrichshain-Kreuzberg': 4589, 'Mitte': 3821, 'Pankow': 2949, 'Neukölln': 2900, 'Tempelhof - Schöneberg': 1283, 'Ch
arlottenburg-Wilm.': 1282, 'Lichtenberg': 529, 'Treptow - Köpenick': 477, 'Steglitz - Zehlendorf': 350, 'Reinickendorf': 199,
'Marzahn - Hellersdorf': 114, 'Spandau': 94})
```

*Gambar 3 List Neighbourhood Group berserta jumlah datanya*

Neighbourhood group yang diambil ada 3 yaitu: Spandau (94 record), Lichtenberg (529 Record), Tempelhof – Schöneberg (1283 record), dan Neukölln (2900). Neighbourhood Group tadi yang akan menjadi acuan perhitungan evaluasi *clustering*.

1. Spandau



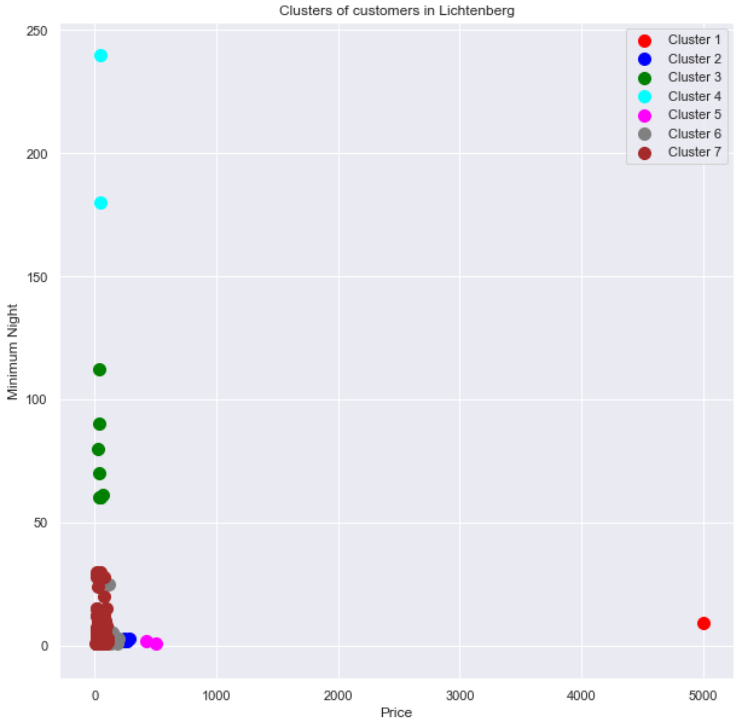

*Gambar 4 Plot hasil clustering tipe customer di Spandau*

```
label = np.zeros(len(X))
for i in range(len(clusters)):
    label[clusters[i]] = i
print("Calinski-Harabasz", metrics.calinski_harabasz_score(X, label))
print("Silhouette Coefficient", metrics.silhouette_score(X, label, metric='euclidean'))
```

```
Calinski-Harabasz 204.76587968825027
Silhouette Coefficient 0.5191917299025981
```

*Gambar 5 Hasil evaluasi model clustering di Spandau*

2. Lichtenberg



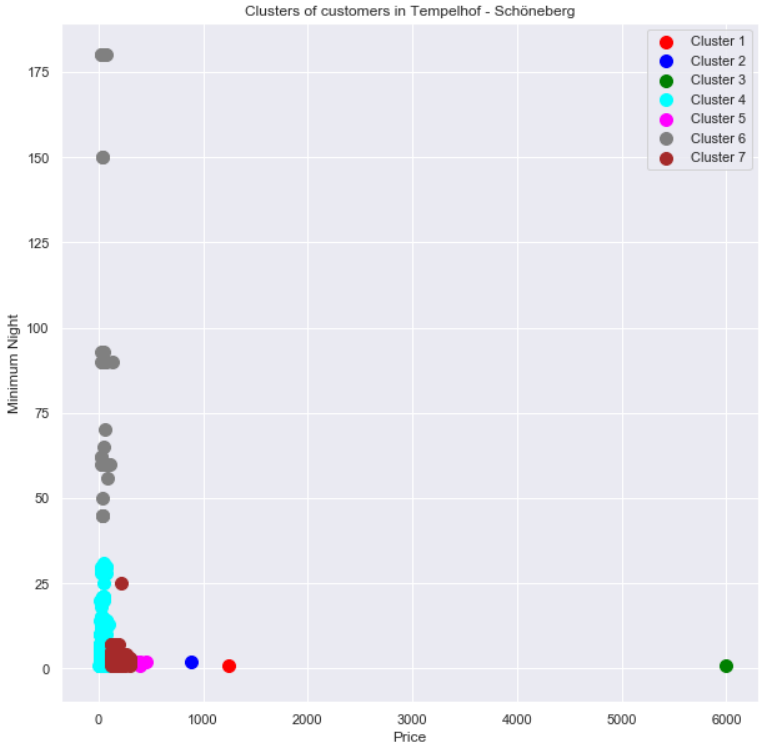

*Gambar 6 Plot hasil clustering tipe customer di Lichtenberg*

```
label = np.zeros(len(X))
for i in range(len(clusters)):
    label[clusters[i]] = i
print("Calinski-Harabasz", metrics.calinski_harabasz_score(X, label))
print("Silhouette Coefficient", metrics.silhouette_score(X, label, metric='euclidean'))
```

```
Calinski-Harabasz 9876.229320769595
Silhouette Coefficient 0.6591590300969307
```

*Gambar 7 Hasil evaluasi model clustering di Lichtenberg*

3. Tempelhof – Schöneberg



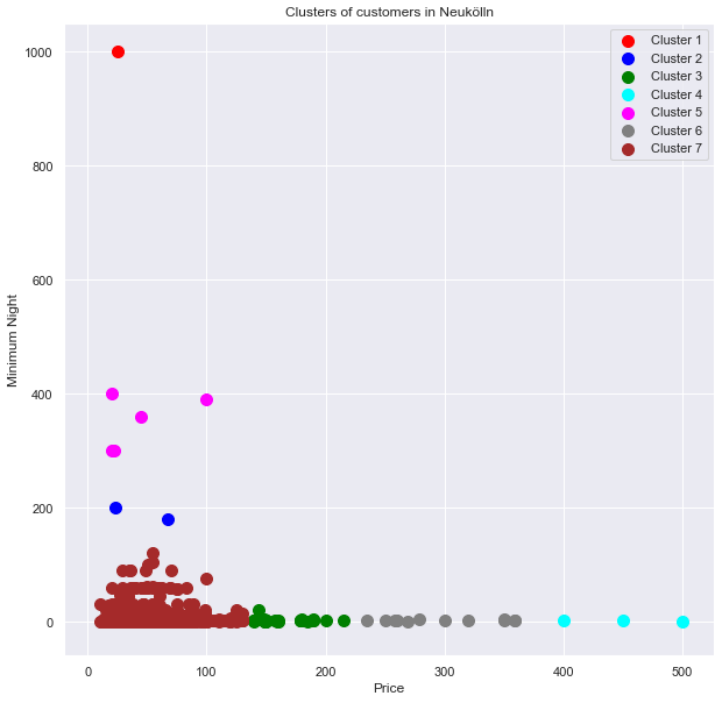

*Gambar 8 Plot hasil clustering tipe customer di Tempelhof – Schöneberg*

```
label = np.zeros(len(X))
for i in range(len(clusters)):
    label[clusters[i]] = i
print("Calinski-Harabasz", metrics.calinski_harabasz_score(X, label))
print("Silhouette Coefficient", metrics.silhouette_score(X, label, metric='euclidean'))
```

```
Calinski-Harabasz 9739.817378212852
Silhouette Coefficient 0.6637075121365007
```

*Gambar 9 Hasil evaluasi model clustering di Tempelhof - Schöneberg*

4. Neukölln




*Gambar 10 Plot hasil clustering tipe customer di Neukölln*

```
label = np.zeros(len(X))
for i in range(len(clusters)):
    label[clusters[i]] = i
print("Calinski-Harabasz", metrics.calinski_harabasz_score(X, label))
print("Silhouette Coefficient", metrics.silhouette_score(X, label, metric='euclidean'))
```

```
Calinski-Harabasz 1020.8242406440356
Silhouette Coefficient 0.7584940362235779
```

*Gambar 11 Hasil evaluasi model clustering di Neukölln*

## 4.2 Studi Kasus 2

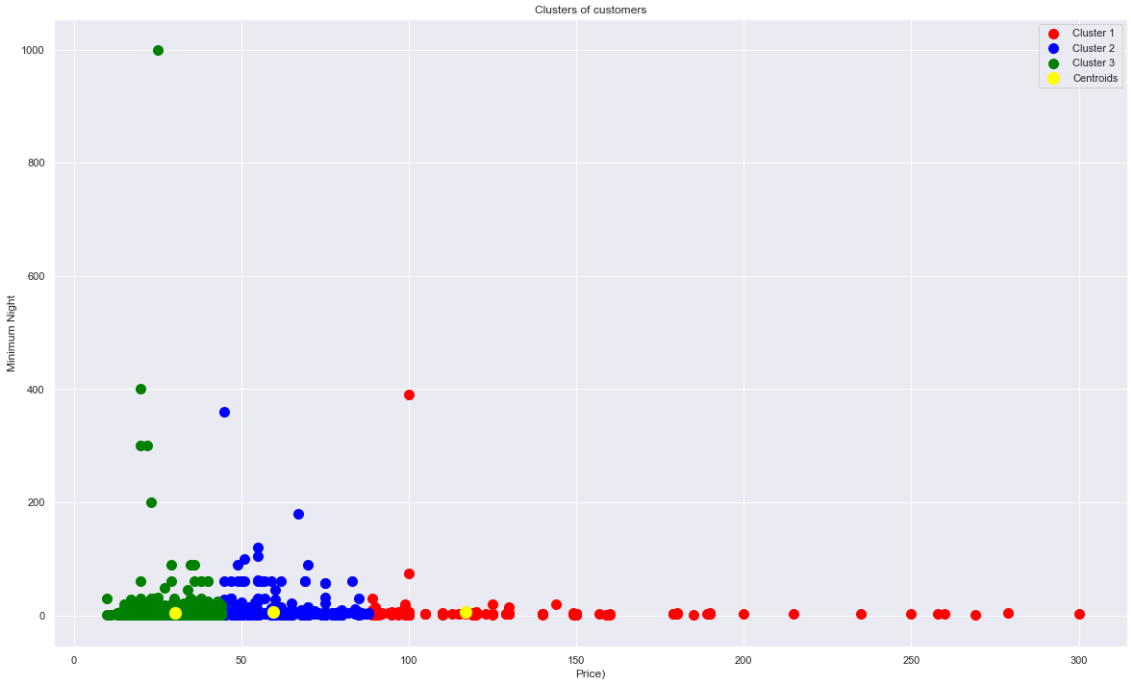
Studi Kasus 2 membatasi price dengan harga maksimum sebesar 300, sehingga calon customer yang akan dikelompokan menjadi lebih sedikit. Dalam case ini akan dipilih 3 clusters saja.

### 4.2.1 Pemilihan Fitur

Fitur yang akan digunakan dalam *clustering* ini tidak dapat ditentukan, karena kita tidak tahu acuan apa yang harus diprediksi. Sehingga fitur yang akan digunakan adalah berdasarkan logika orang yang ingin menyewa holel, apartemen, atau rumah, yaitu: harga dan minimum_nights. Cluster calon pelanggan ini akan dibagi menjadi 3.

### 4.2.2 Analisa Hasil Algoritma

#### 4.2.2.1 K-Means




```
from sklearn import metrics
from sklearn.metrics import pairwise_distances
label = np.zeros(len(X))
for i in range(len(clustersidx)):
    label[clustersidx[i]] = i
print("Calinski-Harabasz", metrics.calinski_harabasz_score(X, label))
print("Silhouette Coefficient", metrics.silhouette_score(X, label, metric='euclidean'))
```

```
Calinski-Harabasz 5040.799915396437
Silhouette Coefficient 0.5704072838621326
```

#### 4.2.2.2 Hierarchical *Clustering*

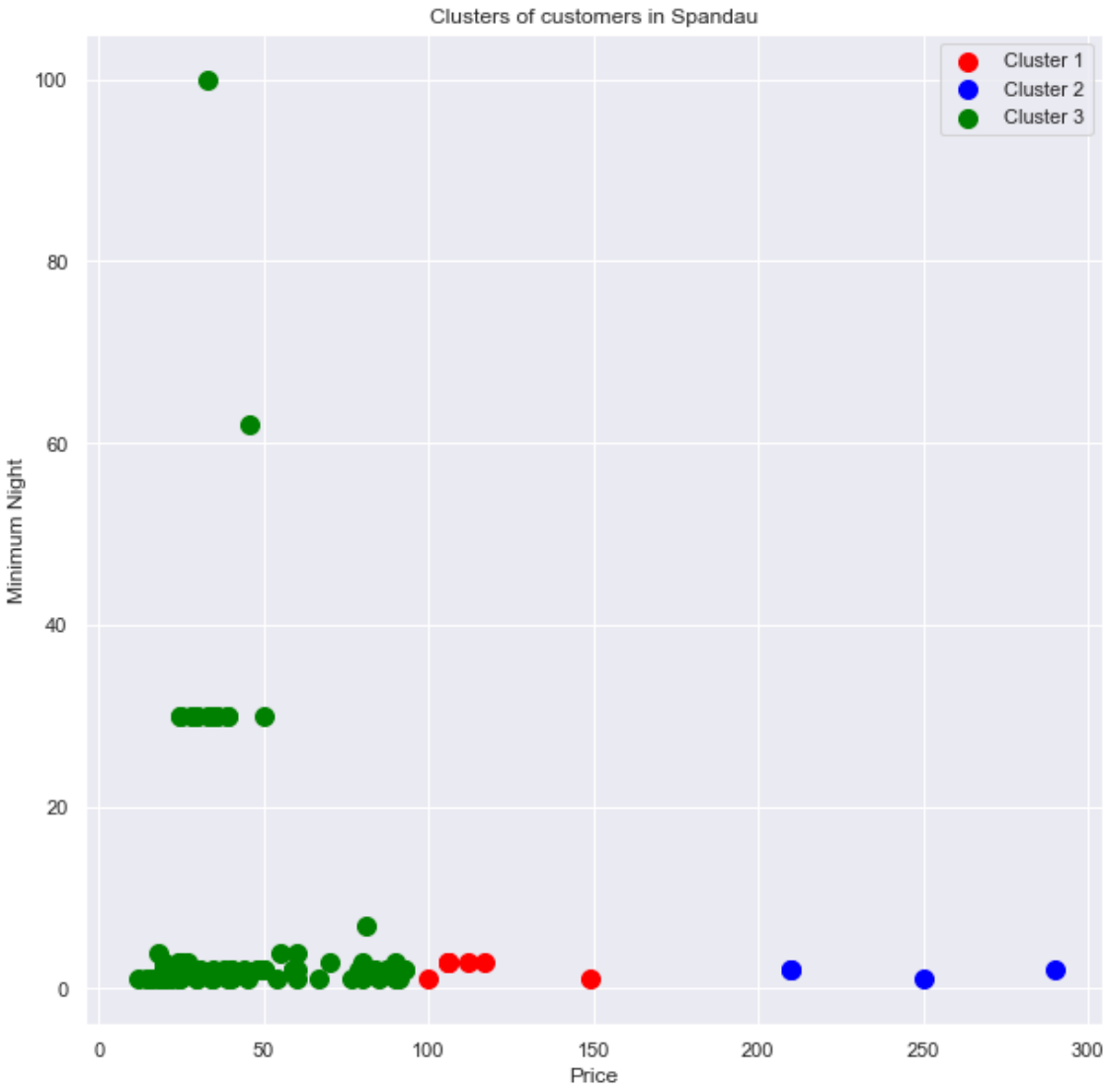
```
print(Counter(clusteringDataset["neighbourhood_group"]))
```

```
Counter({'Friedrichshain-Kreuzberg': 4570, 'Mitte': 3791, 'Pankow': 2935, 'Neukölln': 2892, 'Tempelhof - Schöneberg': 1275, 'Ch
arlottenburg-Wilm.': 1267, 'Lichtenberg': 526, 'Treptow - Köpenick': 475, 'Steglitz - Zehlendorf': 350, 'Reinickendorf': 199,
'Marzahn - Hellersdorf': 114, 'Spandau': 94})
```

*Gambar 12 List Neighbourhood Group berserta jumlah datanya*

Neighbourhood group yang diambil ada 4 yaitu: Spandau (94 record), Lichtenberg (526 Record), Tempelhof – Schöneberg (1275 record), dan Neukölln (2892). Neighbourhood Group tadi yang akan menjadi acuan perhitungan evaluasi *clustering*.

1. Spandau



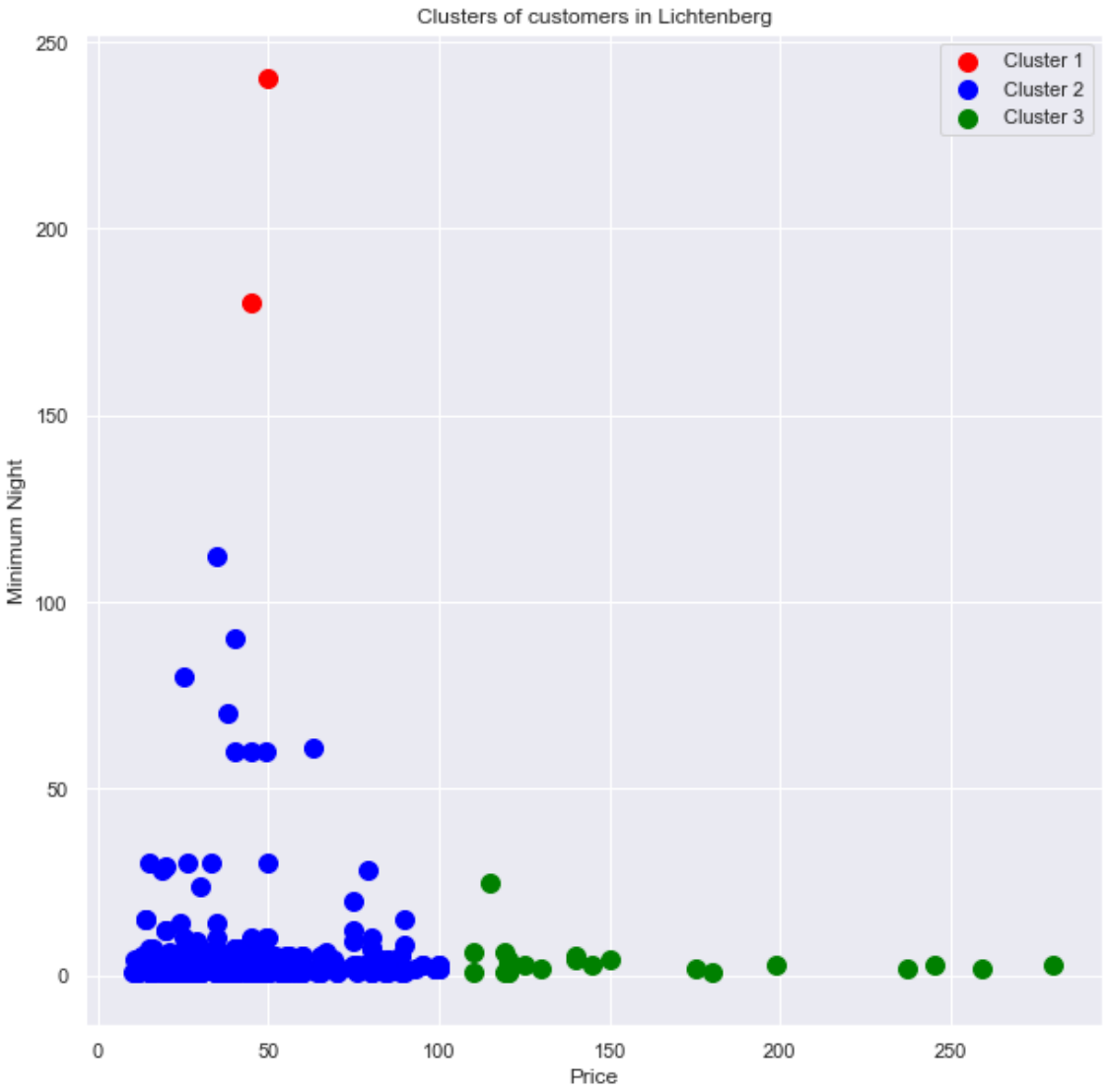

*Gambar 13 Plot hasil clustering tipe customer di Spandau*

```
label = np.zeros(len(X))
for i in range(len(clusters)):
    label[clusters[i]] = i
print("Calinski-Harabasz", metrics.calinski_harabasz_score(X, label))
print("Silhouette Coefficient", metrics.silhouette_score(X, label, metric='euclidean'))
```

```
Calinski-Harabasz 113.52658954621614
Silhouette Coefficient 0.492426692653275
```

*Gambar 14 Hasil evaluasi model clustering di Spandau*

2. Lichtenberg



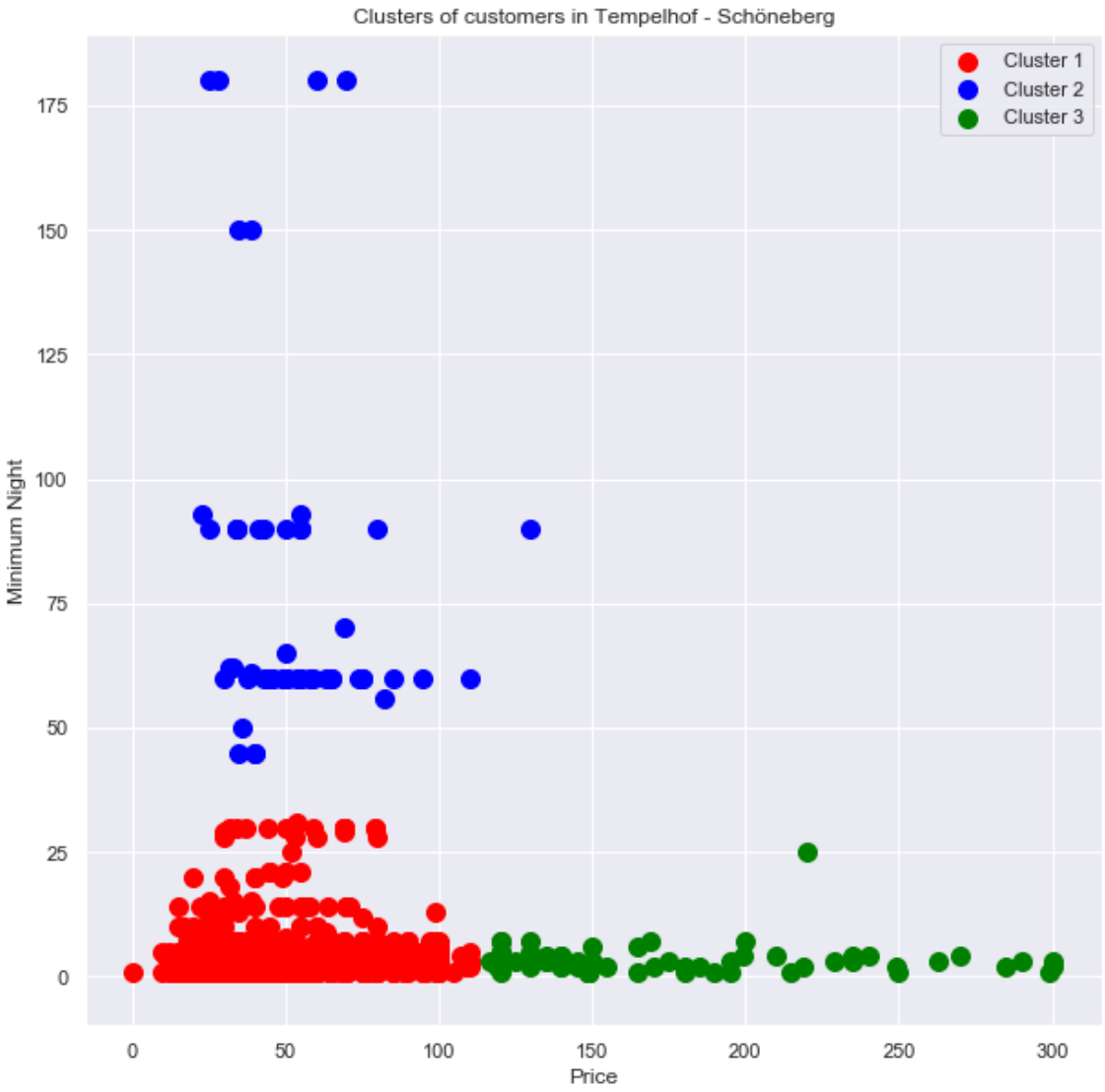

*Gambar 15 Plot hasil clustering tipe customer di Lichtenberg*

```
label = np.zeros(len(X))
for i in range(len(clusters)):
    label[clusters[i]] = i
print("Calinski-Harabasz", metrics.calinski_harabasz_score(X, label))
print("Silhouette Coefficient", metrics.silhouette_score(X, label, metric='euclidean'))
```

```
Calinski-Harabasz 325.6886699167166
Silhouette Coefficient 0.7507448422525883
```

*Gambar 16 Hasil evaluasi model clustering di Lichtenberg*

3. Tempelhof – Schöneberg



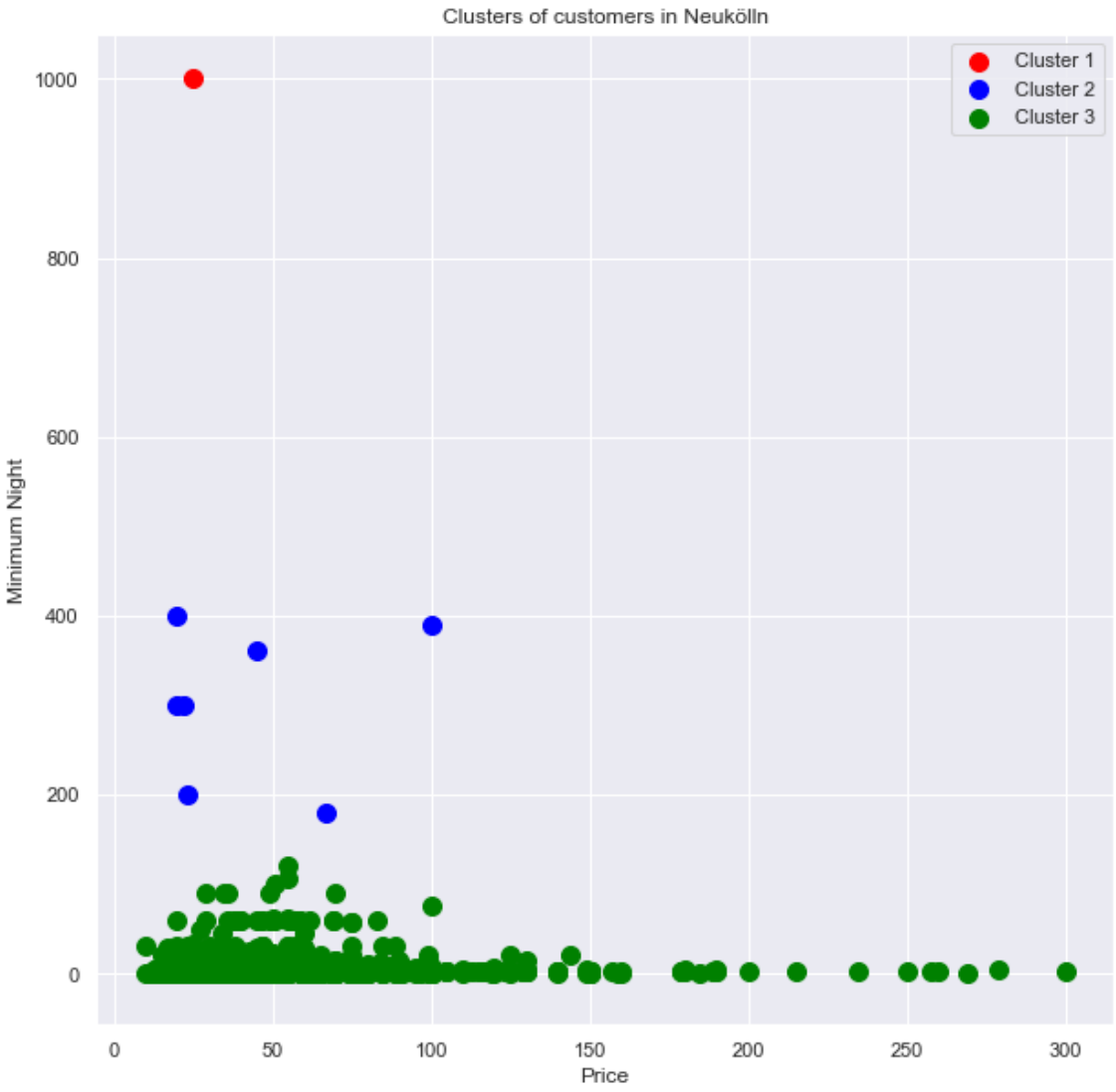

*Gambar 17 Plot hasil clustering tipe customer di Tempelhof – Schöneberg*

```
label = np.zeros(len(X))
for i in range(len(clusters)):
    label[clusters[i]] = i
print("Calinski-Harabasz", metrics.calinski_harabasz_score(X, label))
print("Silhouette Coefficient", metrics.silhouette_score(X, label, metric='euclidean'))
```

```
Calinski-Harabasz 1003.4289362034199
Silhouette Coefficient 0.6691083381247709
```

*Gambar 18 Hasil evaluasi model clustering di Tempelhof - Schöneberg*

4. Neukölln




*Gambar 19 Plot hasil clustering tipe customer di Neukölln*

```
label = np.zeros(len(X))
for i in range(len(clusters)):
    label[clusters[i]] = i
print("Calinski-Harabasz", metrics.calinski_harabasz_score(X, label))
print("Silhouette Coefficient", metrics.silhouette_score(X, label, metric='euclidean'))
```

```
Calinski-Harabasz 982.6548559570152
Silhouette Coefficient 0.905604568366408
```

*Gambar 20 Hasil evaluasi model clustering di Neukölln*

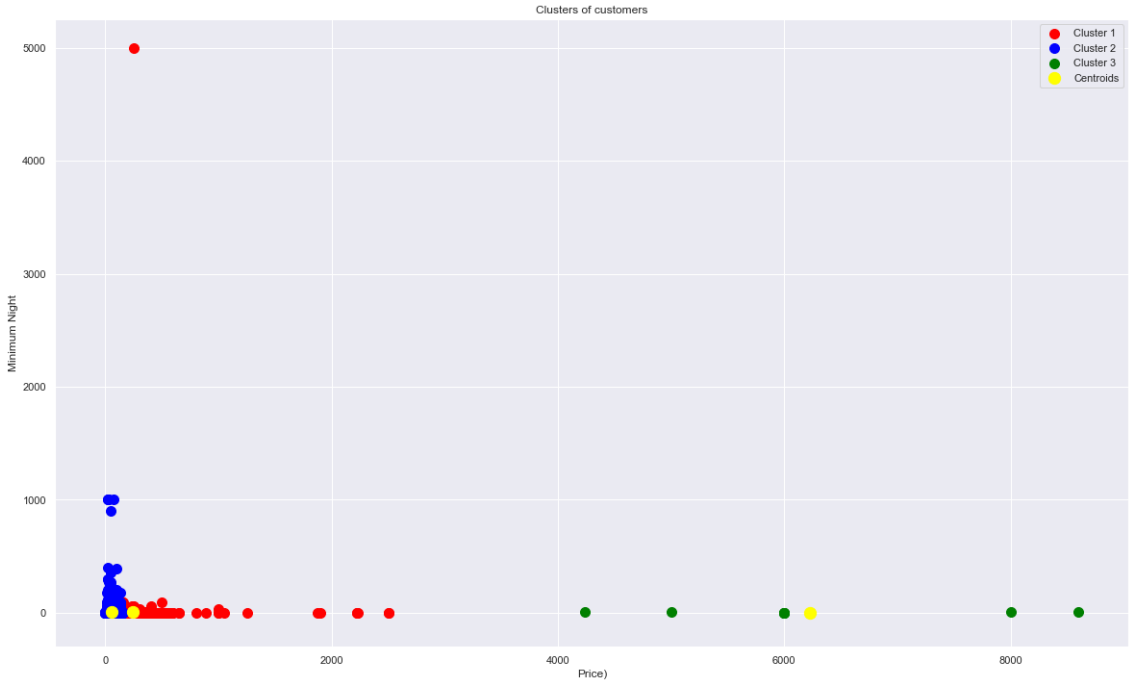
## 4.3 Kesimpulan Hasil Eksperimen

Pada *clustering* penentuan jumlah cluster terhadap data itu sangat penting, pada saat eksperimen menggunakan 7 clusters di studi kasus 2, memiliki hasil evaluasi yang berbeda dengan studi kasus 2.

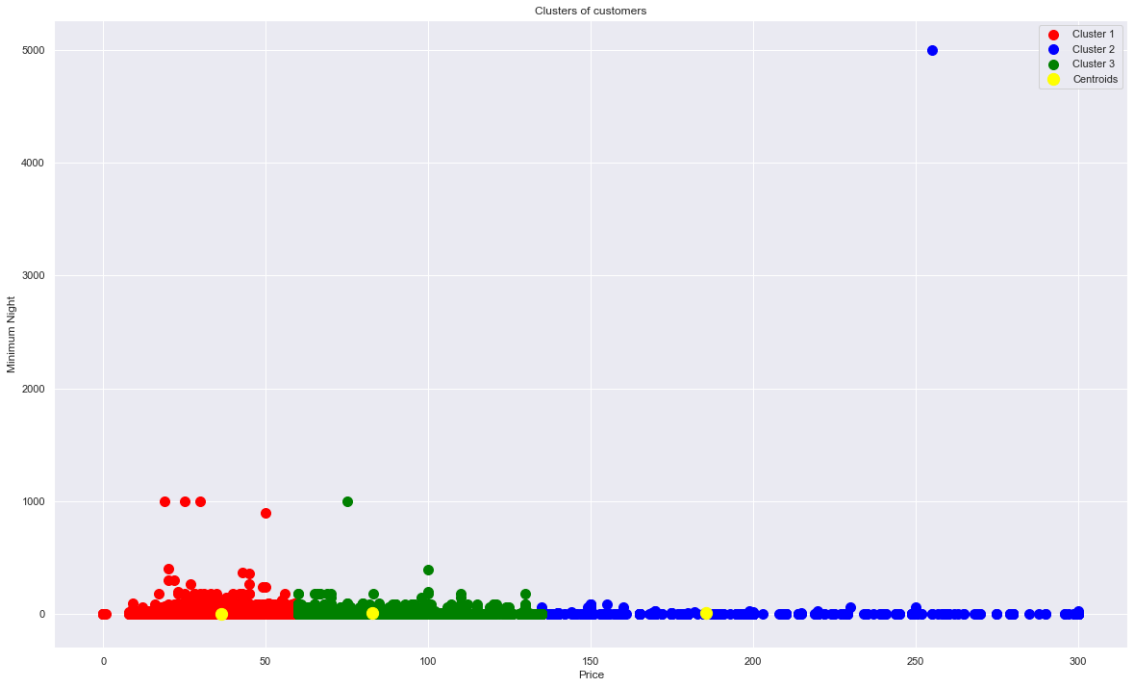
| Algoritma K-Means Cluster 7 | Studi Kasus 1 | Studi Kasus 2 |
|---|---|---|
| Calinski-Harabasz | 27591.134568657442 | 2797.4368982250567 |
| Silhouette Coefficient | 0.6119252314376723 | 0.4514751977995608 |

| Algoritma K-Means Cluster 3 | Studi Kasus 1 | Studi Kasus 2 |
|---|---|---|
| Calinski-Harabasz | 31065.541726573803 | 5120.317323632956 |
| Silhouette Coefficient | 0.7941953066199083 | 0.5499513864632493 |

Hal ini menunjukkan bahwa banyaknya jumlah cluster sangat mempengaruhi kedua eksperimen baik itu yang menggunakan outlier maupun tidak. Tetapi pada kasus *clustering* jika tidak menggunakan outlier hasilnya semakin baik karena evaluasi dari metode *clustering* adalah ketika datanya saling berdekatan dan terpisah. Jika dilakukan tanpa outlier data akan saling terpisah hal ini mengakibatkan beberapa cluster memiliki jumlah data yang lebih sedikit dibandingkan dengan yang menggunakan outlier.




*Gambar 21 K-Means 3 cluster dengan outlier*




*Gambar 22 K-Means 3 cluster tanpa outlier*