

Nama = Akhmad Dian M
NIM = 1941720205
Kelas = TI-2G
Absen = 03

UAS Machine Learning Deteksi Salju *Clustering* 2021

1. Formulasi Masalah

a. Articulate Your Problem Clearly

Program ini berfungsi untuk memprediksi apakah besok akan turun salju atau tidak.

b. Identify Your Data Sources

Dataset yang digunakan adalah salju.

c. Identify Potential Learning Problems

Solusi yang ditampilkan adalah pengelompokan atribut yang memungkinkan salju turun besok.

d. Think About Potential Bias and Ethics

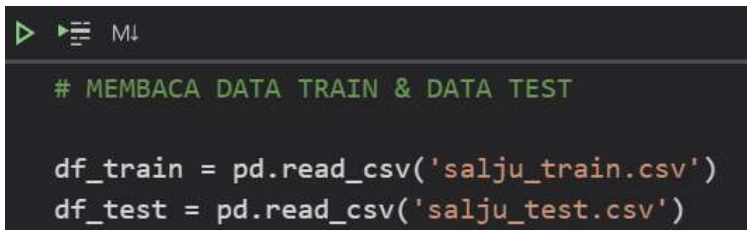
Fitur atribut yang digunakan adalah **SuhuMin, Hujan, SinarMatahari, dan Penguapan.**

2. Ekplorasi dan Persiapan Data

Pertama dilakukan persiapan dan eksplorasi data sebagai berikut:

a. Membaca Data

Disediakan 2 dataset, yaitu data train dan data test.



```
▶ ▶ M4  
# MEMBACA DATA TRAIN & DATA TEST  
  
df_train = pd.read_csv('salju_train.csv')  
df_test = pd.read_csv('salju_test.csv')
```

b. Menampilkan 5 Data Teratas

Untuk memastikan data sudah berhasil dibaca, ditampilkan 5 data teratas dari data train.

```

# MENAMPILKAN 5 DATA TERATAS PADA DATA TRAIN
df_train.head()

```

	id	Tanggal	Kelokasi	SuhuMin	SuhuMax	Hujan	Pengupan	SinarMatahari	ArahAnginTerkencang	KecepatanAnginTerkencang	...	Kelambahan	Kelambahan3pm	Tekanan
0	1	01/06/2014	14	16.6	15.5	0.0	Non	Non	Non	Non	...	14.0	...	76.0
1	1	01/07/2014	138	9.8	12.0	0.0	0.0	7.2	Non	Non	...	98.0	...	91.0
2	1	01/01/2013	248	16.2	12.0	0.0	Non	Non	100	100	...	41.0	...	41.0
3	4	09/06/2012	136	7.1	14.0	0.0	0.0	10.0	100	100	...	14.0	...	17.0
4	1	21/10/2010	17	5.0	18.2	0.0	0.0	12.0	0	0	...	11.0	...	11.0

c. Menampilkan Jumlah Baris dan Kolom

Dapat dilihat terdapat 109095 baris dan 24 kolom.

```

# MENAMPILKAN JUMLAH BARIS DAN KOLOM
df_train.shape

```

(109095, 24)

d. Menampilkan Informasi

Ditampilkan informasi tipe data untuk setiap kolom

```

# MENAMPILKAN INFORMASI TIPE DATA SETIAP KOLOM
df_train.info()

```

#	Column	Non-Null Count	Dtype
0	id	109095 non-null	int64
1	Tanggal	109095 non-null	object
2	Kelokasi	109095 non-null	object
3	SuhuMin	107973 non-null	float64
4	SuhuMax	108166 non-null	float64
5	Hujan	106664 non-null	float64
6	Pengupan	62671 non-null	float64
7	SinarMatahari	56716 non-null	float64
8	ArahAnginTerkencang	181351 non-null	object
9	KecepatanAnginTerkencang	181399 non-null	float64
10	ArahAngin3pm	101172 non-null	object
11	ArahAngin3pm	105398 non-null	object
12	KecepatanAngin3pm	107742 non-null	float64
13	KecepatanAngin3pm	106792 non-null	float64
14	Kelambahan3pm	107693 non-null	float64
15	Kelambahan3pm	105721 non-null	float64
16	Tekanan3pm	97768 non-null	float64
17	Tekanan3pm	97787 non-null	float64
18	Awan3pm	67251 non-null	float64
19	Awan3pm	64624 non-null	float64

e. Menampilkan Deskripsi Data

Menampilkan jumlah baris, rerata, standar deviasi, nilai min & max, dll.

```

# MENAMPILKAN DESKRIPSI SETIAP, NILAI MIN & MAX, DLL.
df_train.describe()

```

	id	SuhuMin	SuhuMax	Hujan	Pengupan	SinarMatahari	KecepatanAnginTerkencang	KecepatanAngin3pm	KecepatanAngin3pm	Kelambahan
count	109095.000000	107973.000000	108166.000000	106664.000000	62671.000000	56716.000000	181399.000000	107742.000000	106792.000000	107693.000000
mean	54549.000000	12.136189	15.113819	2.123095	5.862489	7.509627	88.812081	14.812115	10.877579	50.899517
std	10000.150186	6.384433	7.380196	8.548119	4.181410	1.708842	11.447514	8.518881	8.818184	18.999512
min	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	7.000000	0.000000	0.000000	0.000000
25%	13773.500000	7.000000	17.000000	0.000000	2.000000	4.000000	11.000000	7.000000	11.000000	57.000000
50%	54549.000000	12.000000	22.000000	0.000000	4.000000	8.000000	20.000000	13.000000	10.000000	70.000000
75%	93321.500000	16.000000	26.000000	0.000000	7.000000	10.000000	40.000000	18.000000	14.000000	82.000000
max	109095.000000	31.000000	27.000000	171.000000	145.000000	14.000000	171.000000	100.000000	87.000000	100.000000

f. Menampilkan Data Yang Kosong

```

In [10]:
# MENAMPILKAN JUMLAH DATA YANG MEMILIKI DUPLIKAT

df_train.isnull().sum()

id                0
Tanggal          0
KodeKasual       0
Subahli         1152
Gudang          929
Rider           2431
Pengantar       47024
SiapaMekahari   52579
ArakAngInterkawang  7056
KecamatanAngInterkawang  7496
ArakAngRpm      7823
ArakAngRpm      3197
KecamatanAngRpm  1355
KecamatanAngRpm  7584
KecamatanAngRpm  1882
KecamatanAngRpm  2479
KecamatanAngRpm  3167
KecamatanAngRpm  3120
ArakRpm         4184
ArakRpm         4471
ArakRpm         1326
ArakRpm         2396
ArakRpm         2421
ArakRpm         2421
dtype: object

```

g. Menampilkan Data Yang Duplikat

```

In [11]:
# MENAMPILKAN JUMLAH DATA YANG MEMILIKI DUPLIKAT

df_train.duplicated().sum()

0

```

h. Mengisi Nilai Kosong

Karena terdapat banyak sekali data dengan nilai NaN sehingga tidak memungkinkan untuk didrop semua, dilakukan *impute missing values*. Untuk atribut dengan numeric value diisi dengan nilai mean (nilai rerata), sementara atribut dengan string value diisi dengan modus (value yang sering muncul).

```

In [12]:
# Mengisi nilai yang kosong

df_train.fillna(df_train.mean(), inplace=True) # untuk data numeric
df_train = df_train.fillna(df_train.mode().iloc[0]) # untuk data string
df_train.head()

```

id	Tanggal	KodeKasual	Subahli	Subahli	Subahli	Pengantar	KecamatanAngInterkawang	KecamatanAngInterkawang	KecamatanAngInterkawang	KecamatanAngInterkawang	KecamatanAngInterkawang	KecamatanAngInterkawang	KecamatanAngInterkawang	KecamatanAngInterkawang	KecamatanAngInterkawang
0	01/06/2018	14	18.4	15.9	4.2	1.40000	1.40000	1.40000	1.40000	1.40000	1.40000	1.40000	1.40000	1.40000	1.40000
1	15/07/2019	110	8.6	11.0	8.6	1.40000	1.40000	1.40000	1.40000	1.40000	1.40000	1.40000	1.40000	1.40000	1.40000
2	16/06/2013	140	18.2	11.0	8.6	1.40000	1.40000	1.40000	1.40000	1.40000	1.40000	1.40000	1.40000	1.40000	1.40000
3	06/06/2012	136	7.2	11.0	8.6	1.40000	1.40000	1.40000	1.40000	1.40000	1.40000	1.40000	1.40000	1.40000	1.40000
4	09/10/2019	12	1.8	11.0	8.6	1.40000	1.40000	1.40000	1.40000	1.40000	1.40000	1.40000	1.40000	1.40000	1.40000

5 rows x 14 columns

i. Mengecek Data Sudah Tidak Null

```

In [13]:
# Mengecek Data Sudah Tidak Null

df_train.isnull().sum()

id                0
Tanggal          0
KodeKasual       0
Subahli         1152
Gudang          929
Rider           2431
Pengantar       47024
SiapaMekahari   52579
ArakAngInterkawang  7056
KecamatanAngInterkawang  7496
ArakAngRpm      7823
ArakAngRpm      3197
KecamatanAngRpm  1355
KecamatanAngRpm  7584
KecamatanAngRpm  1882
KecamatanAngRpm  2479
KecamatanAngRpm  3167
KecamatanAngRpm  3120
ArakRpm         4184
ArakRpm         4471
ArakRpm         1326
ArakRpm         2396
ArakRpm         2421
ArakRpm         2421
dtype: object

```

j. Drop Kolom

Dilakukan penghapusan data untuk atribut yang tidak dibutuhkan, yaitu ID, Tanggal, dan Kode Lokasi.

```
▶ M1

# HAPUS KOLOM YANG TIDAK DIPERLUKAN

cols = [0,1,2]
df_train.drop(df_train.columns[cols], axis=1, inplace=True)
```

k. Menemukan Outlier

Karena ingin menemukan outlier untuk data yang numerik, maka dibuat dataframe baru untuk atribut yang bernilai numerik.

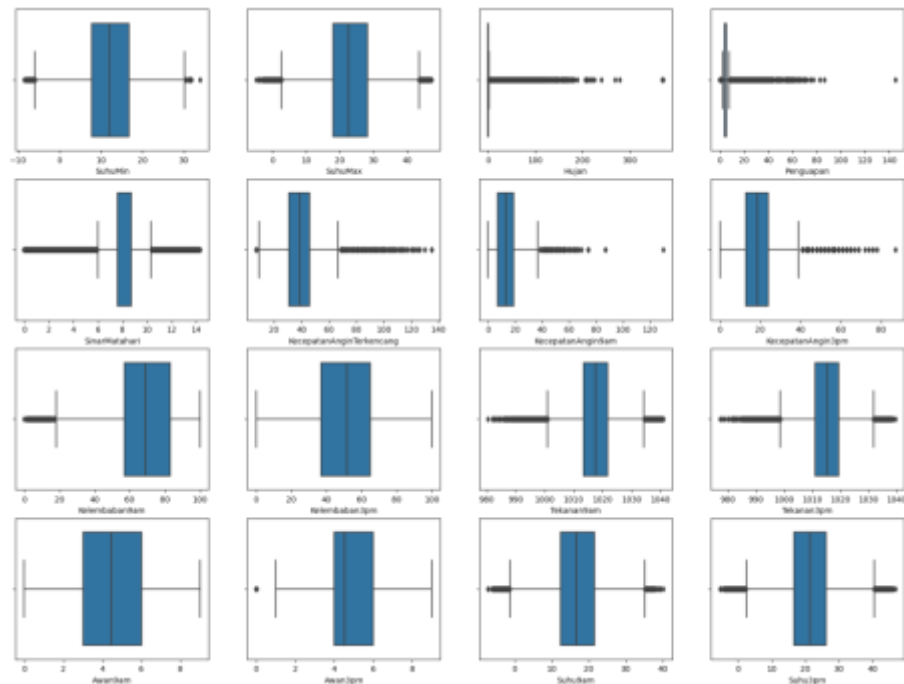
```
▶ M1

# MENEMUKAN OUTLIER

df_train_num = df_train.iloc[:, [0,1,2,3,4,6,9,10,11,12,13,14,15,16,17,18]]
df_train_num.columns

fig, axes = plt.subplots(ncols = 4, nrows = 4, figsize=(20,15))

for i, ax in zip(df_train_num.columns, axes.flat):
    sns.boxplot(x=df_train_num[i], ax=ax)
plt.show()
```

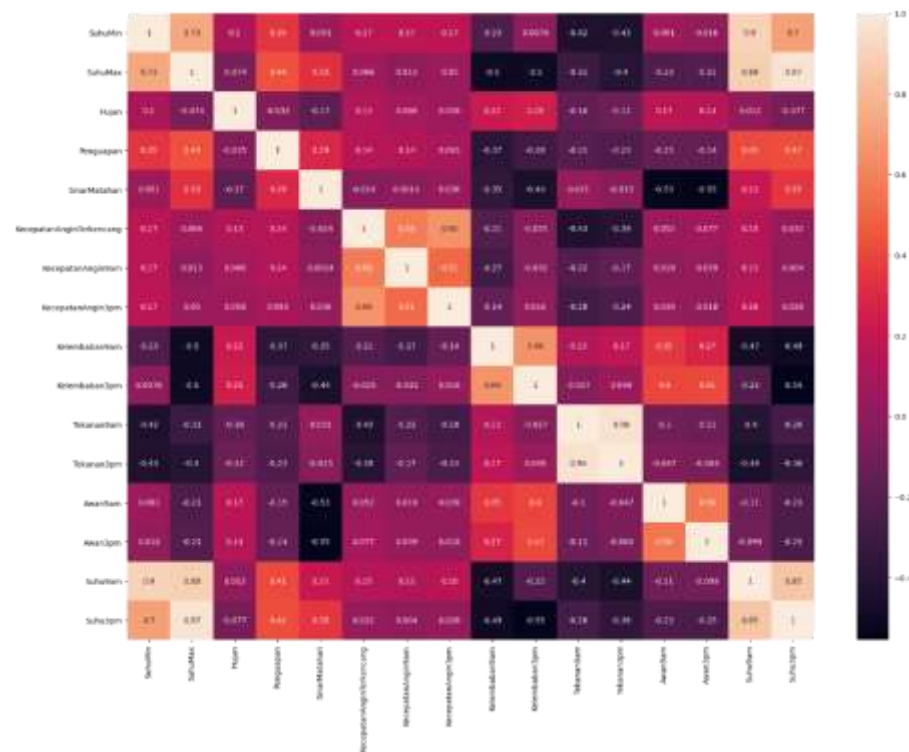


l. Membuat Matriks Korelasi

Matriks korelasi juga dibuat untuk atribut yang bernilai numerik.

```
▶ Mi
# MEMBUAT MATRIKS KORELASI

mtrxCor1 = df_train_num.corr()
plt.figure(figsize=(20,15))
sns.heatmap(mtrxCor1, annot=True)
plt.show()
```

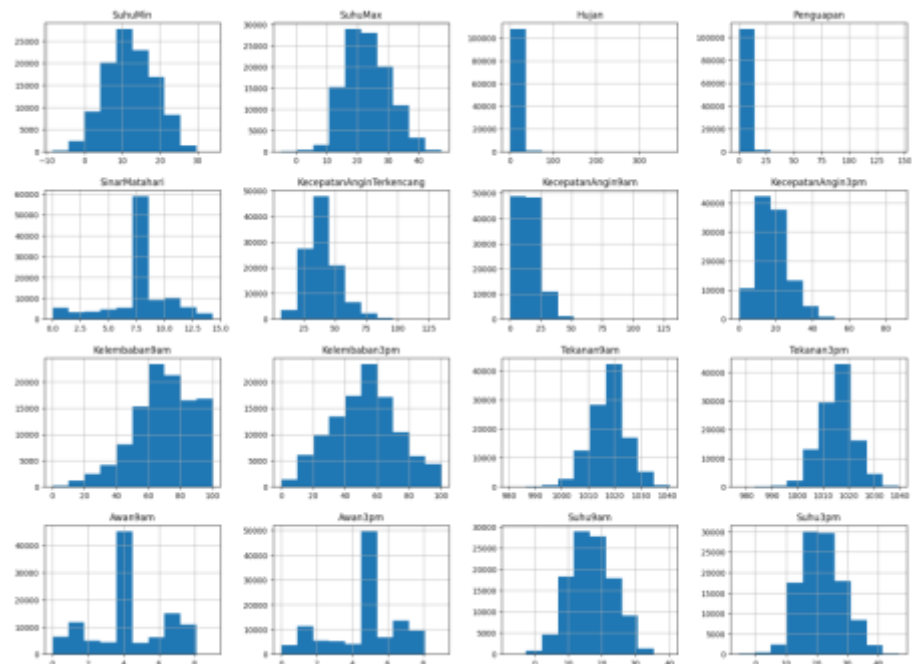


m. Menampilkan Distribusi Data Dengan Histogram

Persebaran data juga dicari untuk atribut yang bernilai numerik.

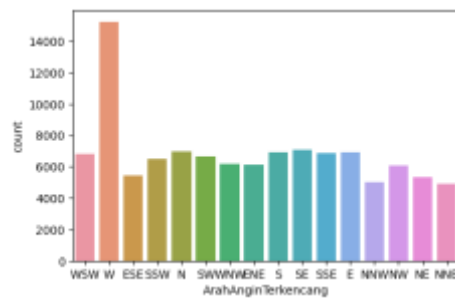
```
▶ Mi
# MENAMPILKAN DISTRIBUSI DATA TIAP KOLOM MENGGUNAKAN HISTOGRAM

hist = df_train_num.hist(figsize=(20,15))
```

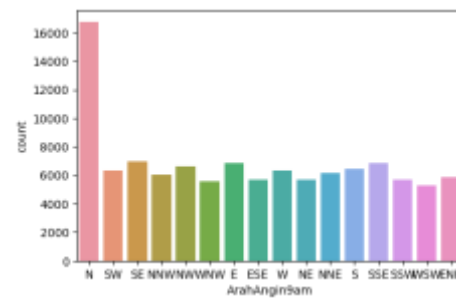


- n. Menampilkan Jumlah Data Atribut Bernilai String**
Data yang bernilai string ditampilkan dengan countplot.

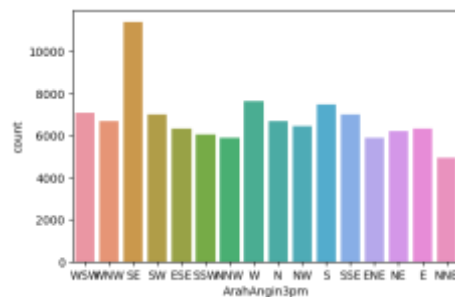
Arah Angin Terkencang



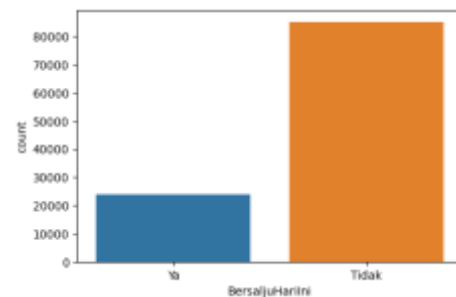
Arah Angin 9am



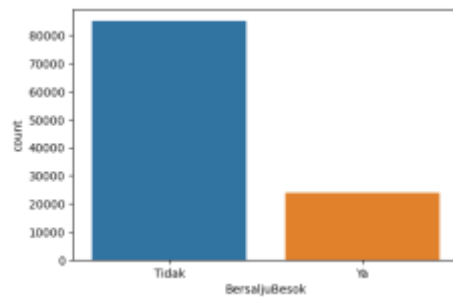
Arah Angin 3pm



Bersalju Hari Ini

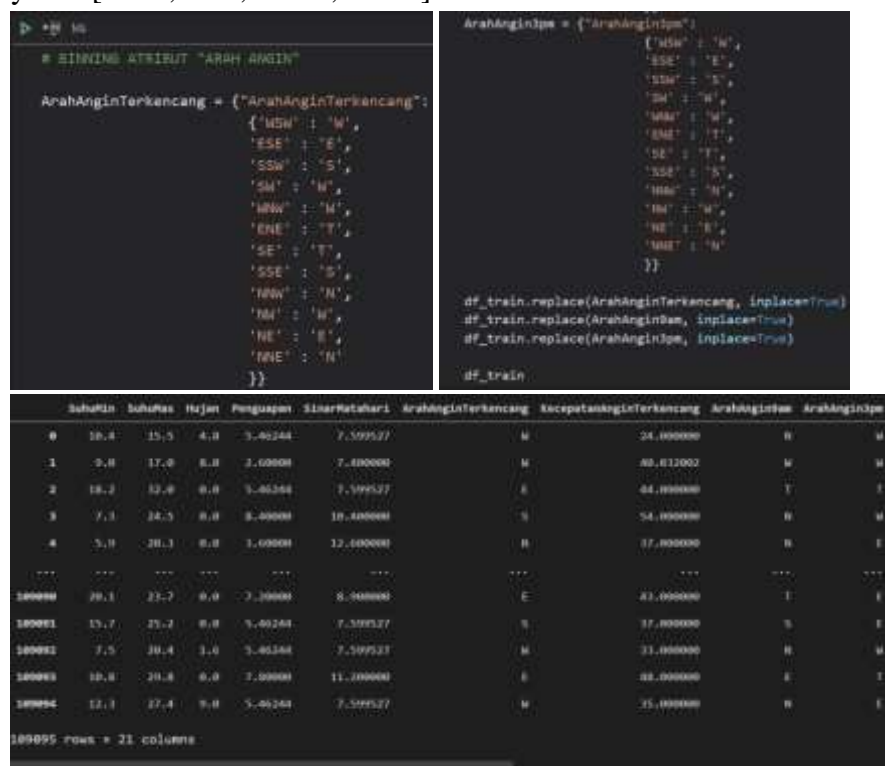


Bersalju Besok



o. Binning

Karena atribut “Arah Angin” memiliki berbagai macam kategori, maka dilakukan Binning sehingga arah angin hanya terbagi menjadi 4 kategori yaitu: [West, East, South, North]



p. Encode Categorical Variable

Dilakukan encode categorical variable untuk mengubah nilai string menjadi numerik dengan LabelEncoder.

```
# ENCODE CATEGORICAL VARIABLE

from sklearn.preprocessing import LabelEncoder

labelencoder = LabelEncoder()
df_train['ArahAnginTerkencang'] = labelencoder.fit_transform(df_train['ArahAnginTerkencang'])
labelencoder = LabelEncoder()
df_train['ArahAnginBan'] = labelencoder.fit_transform(df_train['ArahAnginBan'])
labelencoder = LabelEncoder()
df_train['ArahAngin3pm'] = labelencoder.fit_transform(df_train['ArahAngin3pm'])
labelencoder = LabelEncoder()
df_train['BersaljuHariIni'] = labelencoder.fit_transform(df_train['BersaljuHariIni'])
labelencoder = LabelEncoder()
df_train['BersaljuBesok'] = labelencoder.fit_transform(df_train['BersaljuBesok'])

df_train
```

	SuhuMin	SuhuMax	Hujan	Pengupan	SinarMatahari	ArahAnginTerkencang	KecepatanAnginTerkencang	ArahAnginBan	ArahAngin3pm	KecepatanAng
0	10.4	15.5	4.0	5.40244	7.599527	4	24.000000	1	4	
1	9.0	17.0	0.0	2.60000	7.600000	4	40.032003	4	4	
2	10.7	12.0	0.0	5.40244	7.599527	0	41.000000	2	3	
3	7.3	14.3	0.0	0.40000	10.400000	2	54.000000	1	4	
4	5.5	20.3	0.0	1.60000	12.400000	1	17.000000	1	0	
...
100000	20.1	23.7	0.0	7.20000	8.900000	0	41.000000	3	0	
100001	15.7	25.2	0.0	5.40244	7.599527	2	17.000000	2	0	
100002	7.5	20.4	1.0	5.40244	7.599527	4	17.000000	1	4	
100003	10.8	25.0	0.0	7.00000	11.200000	0	40.000000	0	3	
100004	12.3	27.4	0.0	5.40244	7.599527	4	25.000000	1	0	

100005 rows x 11 columns

q. Scalling

Karena value semua data berada pada range 0~1000, maka dilakukan scalling data untuk mempersempit range menjadi 0~1 saja.

```
# Min-Max Scalling

scaler = MinMaxScaler()

df_train.iloc[0:len(df_train), [0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20]] = scaler.fit_transform(
df_train.iloc[0:len(df_train), [0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20]])

df_train.head()
```

	SuhuMin	SuhuMax	Hujan	Pengupan	SinarMatahari	ArahAnginTerkencang	KecepatanAnginTerkencang	ArahAnginBan	ArahAngin3pm	KecepatanAng
0	0.445755	0.389635	0.012938	0.037672	0.511435	1.00	0.131012	0.25	1.00	0.0
1	0.412736	0.418626	0.021501	0.017931	0.517483	1.00	0.250061	1.00	1.00	0.1
2	0.520737	0.706334	0.000000	0.037672	0.511435	0.00	0.289062	0.75	0.75	0.1
3	0.372642	0.562500	0.000000	0.057931	0.722223	0.50	0.367380	0.25	1.00	0.1
4	0.330623	0.481706	0.000000	0.024618	0.881119	0.25	0.234175	0.25	0.00	0.1

5 rows x 11 columns

r. Save Data

Data untuk clustering sudah siap digunakan.

```
# Save to CSV

df_train.to_csv(r'C:\Users\haura\OneDrive\Documents\SEMESTER 6\Pembelajaran Mesin (Maling)\Tugas\salju\data_for_clustering.csv',
index=False, header=True)
```

3. Pemodelan

Untuk Clustering, digunakan model K-Means Clustering karena banyak digunakan dan mudah untuk diimplementasikan. Pertama, melakukan perhitungan untuk Euclidean Distance.


```

▶ ▶ Ml

# MENGHITUNG NILAI EUCLIDEAN DISTANCE

def euclidian_distance(u, v):
    return sum((p-q)**2 for p, q in zip(u, v))**0.5

```

Selanjutnya, dilakukan looping Algoritma K-Means sampai centroid mencapai nilai yang sama.

```

> > >
def kmeans(n_neighbour, n_feat, centroids):
    # looping algoritma K-Means sampai nilai centroid sama
    while (True):
        cluster = []

        for i in range(len(X)):
            euclid = []
            #menghitung euclidean distance
            for j in range(n_neighbour):
                euclid.append(euclidian_distance(X[i][n_feat], centroids[j]))
            #menilih cluster dari nilai minimum euclidean distance
            idx = np.argmin(euclid)
            cluster.append(idx+1)
            #memasukkan cluster ke x
            X[i][n_feat] = idx+1

        #menghitung centroid baru ke tiap cluster
        group = {}
        for j in range(n_neighbour):
            group[j] = []
            for i in range(len(cluster)):
                if cluster[i] == j:
                    group[j].append(X[i][n_feat])

            #menghitung centroid baru ke tiap cluster
            new_centroids = []
            for i in range(n_neighbour):
                new_centroids.append(np.mean(group[i], axis=0).tolist())

            if (centroids == new_centroids):
                #nilai centroid
                break

        centroids = new_centroids

```

4. Evaluasi

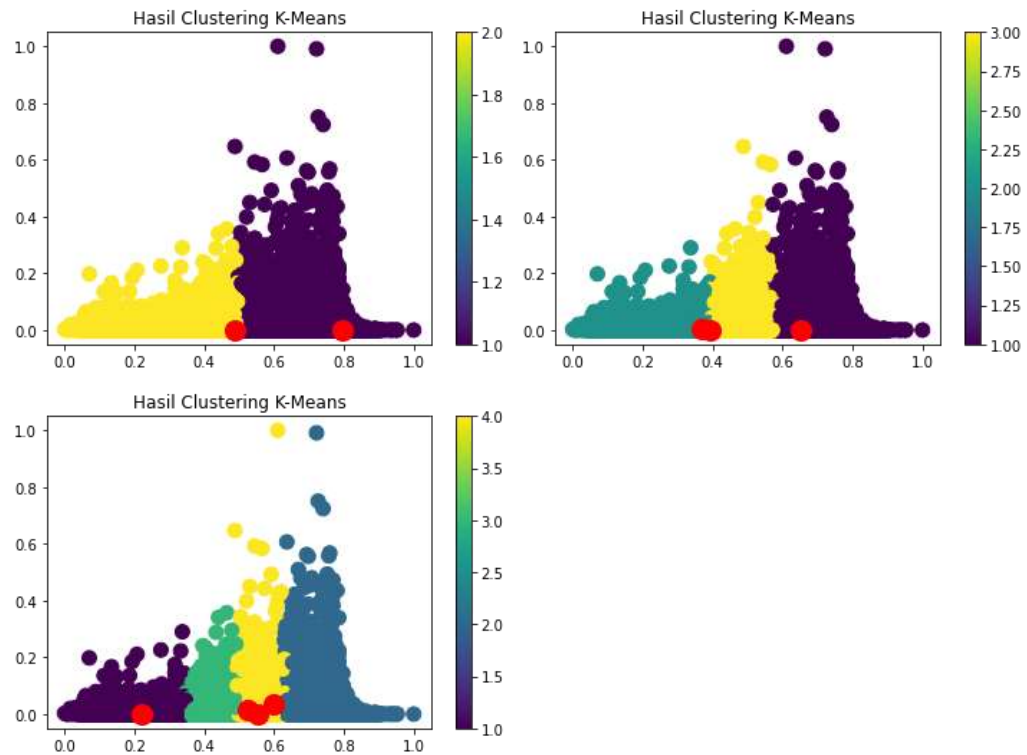
Untuk Evaluasi, dilakukan pencarian silhouette score dari hasil clustering tersebut karena silhouette score dapat mengukur kualitas dari clustering yang dilakukan.

5. Eksperimen

Dilakukan 2 eksperimen, yaitu dengan atribut SuhuMin – Hujan dan SinarMatahari – Penguapan.

Eksperimen 1: SuhuMin – Hujan

Untuk eksperimen 1 yang menggunakan fitur SuhuMin dan Hujan, dilakukan pemanggilan algoritma K-Means untuk mencari hasil clustering. Digunakan 3 nilai K untuk melihat K mana yang paling optimal dalam menemukan silhouette score.



Kemudian didapatkan silhouette score yang berbeda-beda.

```

> MI
# SILHOUETTE SCORE UNTUK K = 2

score = silhouette_score(xy.iloc[:,[0,n_feat-1]], xy[n_feat], metric = 'euclidean')
score

0.5525798010873928

> MI
# SILHOUETTE SCORE UNTUK K = 3

score = silhouette_score(xy.iloc[:,[0,n_feat-1]], xy[n_feat], metric = 'euclidean')
score

0.5205632552939808

> MI
# SILHOUETTE SCORE UNTUK K = 4

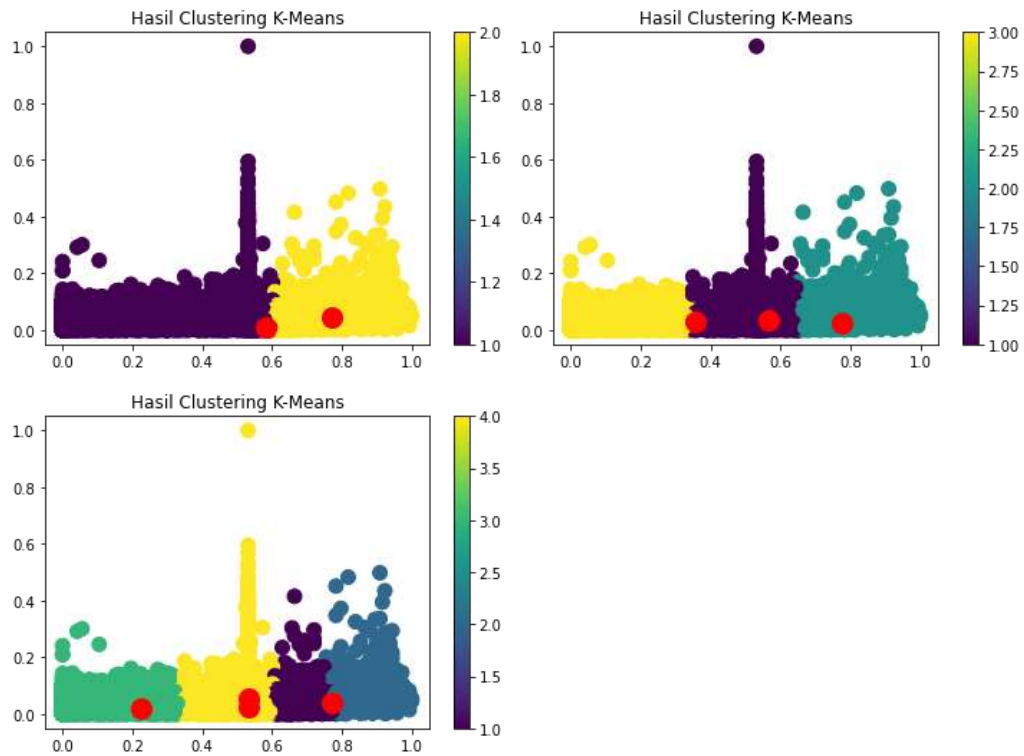
score = silhouette_score(xy.iloc[:,[0,n_feat-1]], xy[n_feat], metric = 'euclidean')
score

0.4958431326496188

```

Eksperimen 2: SinarMatahari – Penguapan.

Untuk eksperimen 2 yang menggunakan fitur SinarMatahari dan Penguapan, dilakukan pemanggilan algoritma K-Means untuk mencari hasil clustering. Digunakan 3 nilai K untuk melihat K mana yang paling optimal dalam menemukan silhouette score.



Kemudian didapatkan silhouette score yang berbeda-beda.

```
▶ MI
# SILHOUETTE SCORE UNTUK K = 2

score = silhouette_score(xy.iloc[:,[0,n_feat-1]], xy[n_feat], metric = 'euclidean')
score

0.579796175128216

▶ MI
# SILHOUETTE SCORE UNTUK K = 3

score = silhouette_score(xy.iloc[:,[0,n_feat-1]], xy[n_feat], metric = 'euclidean')
score

0.7248077677830406
```

```
▶ ML
# SILHOUETTE SCORE UNTUK K = 4

score = silhouette_score(xy.iloc[:,[0,n_feat-1]], xy[n_feat], metric = 'euclidean')
score

0.6865022224633384
```

6. Kesimpulan

Jika hasil clustering dimasukkan kedalam table, maka hasilnya sebagai berikut:

	SuhuMin – Hujan	SinarMatahari – Penguapan
K = 2	0.5525798010873928	0.579796175128216
K = 3	0.5205632552939808	0.7248077677830406
K = 4	0.4958431326496188	0.6865022224633384

Dapat disimpulkan maka hasil silhouette score yang maksimal bergantung dengan pemilihan atribut serta nilai K. Pada program ini, silhouette score yang paling besar ada fitur Sinar Matahari dan Penguapan dengan nilai K = 3.