

Nama = Haura Athaya Salka
NIM = 1301183454
Kelas = IF-42-04
Link = <https://youtu.be/usHBmXeBHFE>

Tugas Besar MK Pembelajaran Mesin *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

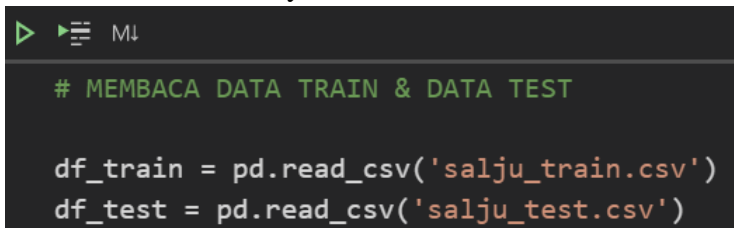
Fitur atribut yang digunakan adalah **SuhuMin**, **Hujan**, **SinarMatahari**, dan **Penguapan**.

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.



```
▶ ▶≡ M↓  
  
# 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	KodeLokasi	SuhuMin	SuhuMax	Hujan	Penguapan	SinarMatahari	ArahAnginTerkencang	KecepatanAnginTerkencang	...	Kelembaban9am	Kelembaban3pm	Tekanan9am
0	1	01/06/2014	C4	10.4	15.5	4.8	NaN	NaN	WSW	24.0	...	78.0	76.0	1020.
1	2	15/07/2014	C10	9.0	17.0	8.0	2.6	7.4	NaN	NaN	...	80.0	61.0	1015.
2	3	16/02/2011	C46	18.2	32.0	0.0	NaN	NaN	ESE	44.0	...	62.0	42.0	Na.
3	4	08/08/2012	C36	7.3	24.5	0.0	8.4	10.4	SSW	54.0	...	25.0	17.0	1019.
4	5	29/10/2016	C7	5.9	20.3	0.0	3.6	12.6	N	37.0	...	55.0	48.0	1019.

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()
```

```
<class 'pandas.core.frame.DataFrame'>
```

```
RangeIndex: 109095 entries, 0 to 109094
```

```
Data columns (total 24 columns):
```

#	Column	Non-Null Count	Dtype
0	id	109095 non-null	int64
1	Tanggal	109095 non-null	object
2	KodeLokasi	109095 non-null	object
3	SuhuMin	107973 non-null	float64
4	SuhuMax	108166 non-null	float64
5	Hujan	106664 non-null	float64
6	Penguapan	62071 non-null	float64
7	SinarMatahari	56716 non-null	float64
8	ArahAnginTerkencang	101351 non-null	object
9	KecepatanAnginTerkencang	101399 non-null	float64
10	ArahAngin9am	101172 non-null	object
11	ArahAngin3pm	105898 non-null	object
12	KecepatanAngin9am	107742 non-null	float64
13	KecepatanAngin3pm	106792 non-null	float64
14	Kelembaban9am	107093 non-null	float64
15	Kelembaban3pm	105721 non-null	float64
16	Tekanan9am	97768 non-null	float64
17	Tekanan3pm	97787 non-null	float64
18	Awan9am	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 RERATA, NILAI MIN & MAX, DLL.
```

```
df_train.describe()
```

	id	SuhuMin	SuhuMax	Hujan	Penguapan	SinarMatahari	KecepatanAnginTerkencang	KecepatanAngin9am	KecepatanAngin3pm	Kelembaban9am
count	109095.000000	107973.000000	108166.000000	106664.000000	62071.000000	56716.000000	101399.000000	107742.000000	106792.000000	107093.000000
mean	54548.000000	12.196183	23.214819	2.385005	5.462440	7.599527	40.032002	14.052115	18.677579	60.89557
std	31493.158146	6.389419	7.106596	8.588155	4.201638	3.789042	13.617554	8.926092	8.830199	18.99552
min	1.000000	-8.500000	-4.800000	0.000000	0.000000	0.000000	7.000000	0.000000	0.000000	0.000000
25%	27274.500000	7.600000	17.900000	0.000000	2.600000	4.800000	31.000000	7.000000	13.000000	57.000000
50%	54548.000000	12.000000	22.600000	0.000000	4.800000	8.400000	39.000000	13.000000	19.000000	70.000000
75%	81821.500000	16.800000	28.200000	0.000000	7.400000	10.600000	48.000000	19.000000	24.000000	83.000000
max	109095.000000	33.900000	47.300000	371.000000	145.000000	14.300000	135.000000	130.000000	87.000000	100.000000

f. Menampilkan Data Yang Kosong

```

# MENAMPILKAN JUMLAH DATA NULL SETIAP KOLON

df_train.isnull().sum()

id          0
Tanggal     0
KodeLokasi  0
SuhuMin     1122
SuhuMax     929
Hujan       2431
Penguapan   47824
SinarMatahari 52379
ArahAnginTerkencang 7744
KecepatanAnginTerkencang 7696
ArahAngin9am 7923
ArahAngin3pm 3197
KecepatanAngin9am 1353
KecepatanAngin3pm 2303
Kelembaban9am 2002
Kelembaban3pm 3374
Tekanan9am 11327
Tekanan3pm 11308
Awan9am     41844
Awan3pm     44471
Suhu9am     1348
Suhu3pm     2698
BersaljuHariIni 2431
BersaljuBesok 2431
dtype: int64

```

g. Menampilkan Data Yang Duplikat

```

# 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).

```

# MENGISI NILAI YANG KOSONG

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

df_train.head()

```

	id	Tanggal	KodeLokasi	SuhuMin	SuhuMax	Hujan	Penguapan	SinarMatahari	ArahAnginTerkencang	KecepatanAnginTerkencang	...	Kelembaban9am	Kelembaban3pm	Tekanan9am	Tekanan3pm
0	1	01/06/2014	C4	10.4	15.5	4.8	5.46244	7.599527	WSW	24.000000	...	78.0	76.0	1020.1000	1018.7000
1	2	15/07/2014	C10	9.0	17.0	8.0	2.600000	7.400000	W	40.032002	...	80.0	61.0	1015.2000	1015.2000
2	3	16/02/2011	C46	18.2	32.0	0.0	5.46244	7.599527	ESE	44.000000	...	62.0	42.0	1017.6470	1017.6470
3	4	08/08/2012	C36	7.3	24.5	0.0	8.400000	10.400000	SSW	54.000000	...	25.0	17.0	1019.2000	1019.2000
4	5	29/10/2016	C7	5.9	20.3	0.0	3.600000	12.600000	N	37.000000	...	55.0	48.0	1019.7000	1019.7000

5 rows x 24 columns

i. Mengecek Data Sudah Tidak Null

```

# DATA SUDAH TIDAK ADA YANG BERNILAI NULL

df_train.isnull().sum()

id          0
Tanggal     0
KodeLokasi  0
SuhuMin     0
SuhuMax     0
Hujan       0
Penguapan   0
SinarMatahari 0
ArahAnginTerkencang 0
KecepatanAnginTerkencang 0
ArahAngin9am 0
ArahAngin3pm 0
KecepatanAngin9am 0
KecepatanAngin3pm 0
Kelembaban9am 0
Kelembaban3pm 0
Tekanan9am  0
Tekanan3pm  0
Awan9am     0
Awan3pm     0
Suhu9am     0
Suhu3pm     0
BersaljuHariIni 0
BersaljuBesok 0
dtype: int64

```

j. Drop Kolom

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

```
▶ ▶≡ M4

# 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.

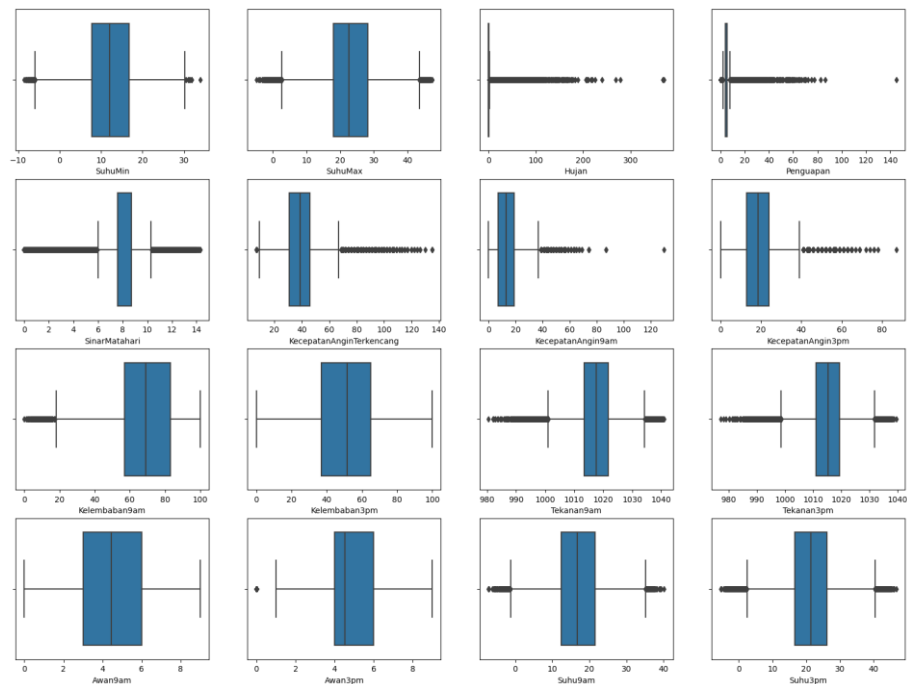
```
▶ ▶≡ M4

# 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))

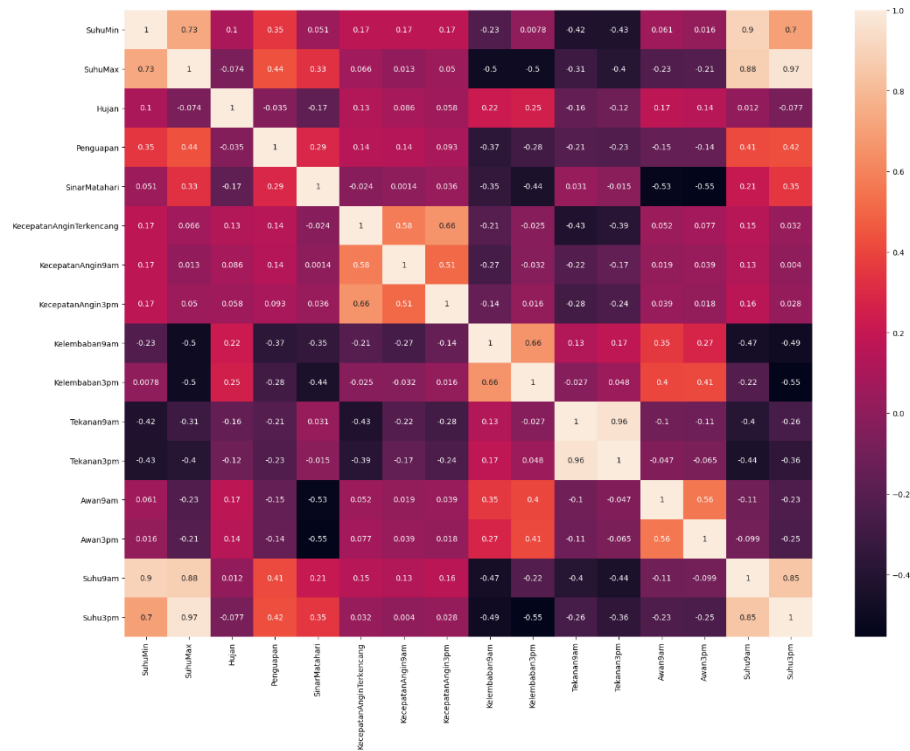
for i, ax in zip(df_train_num.columns, axes.flat):
    sns.boxplot(x=df_train_num[i], ax=ax)
plt.show()
```



l. Membuat Matriks Korelasi

Matriks korelasi juga dibuat untuk atribut yang bernilai numerik.

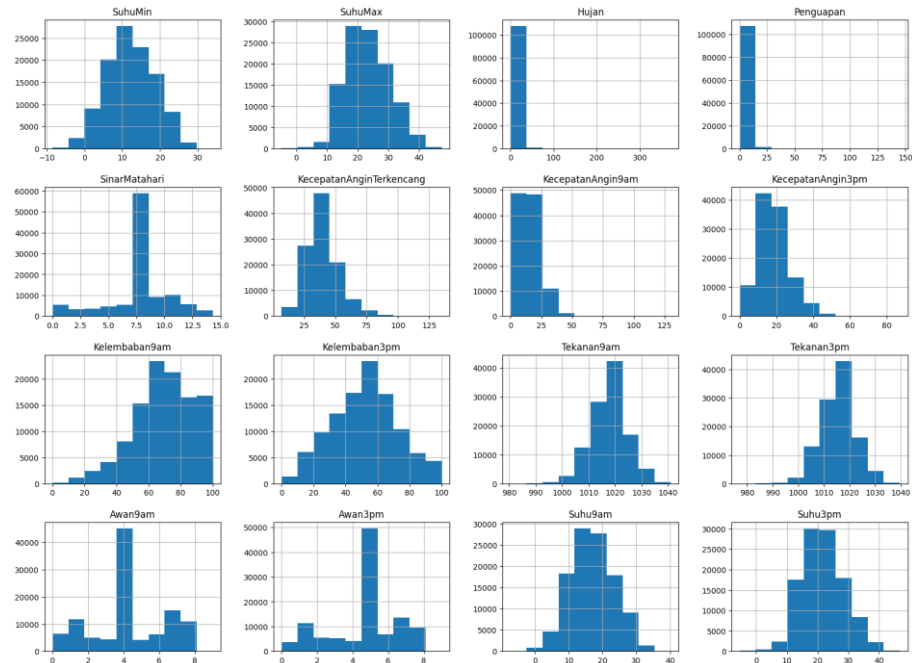
```
▶ M1  
  
# MEMBUAT MATRIKS KORELASI  
  
mtrxCor1 = df_train_num.corr()  
plt.figure(figsize=(20,15))  
sns.heatmap(mtrxCor1, annot=True)  
plt.show()
```



m. Menampilkan Distribusi Data Dengan Histogram

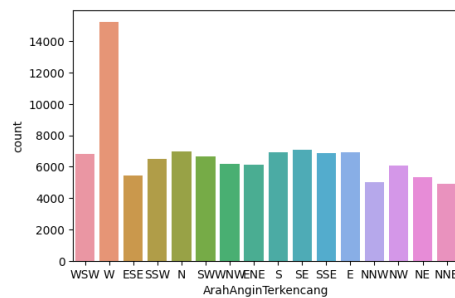
Persebaran data juga dicari untuk atribut yang bernilai numerik.

```
▶ M1  
  
# MENAMPILKAN DISTRIBUSI DATA TIAP KOLOM MENGGUNAKAN HISTOGRAM  
  
hist = df_train_num.hist(figsize=(20,15))
```

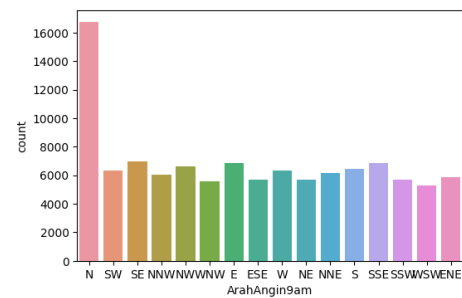


- n. Menampilkan Jumlah Data Atribut Bernilai String**
Data yang bernilai string ditampilkan dengan countplot.

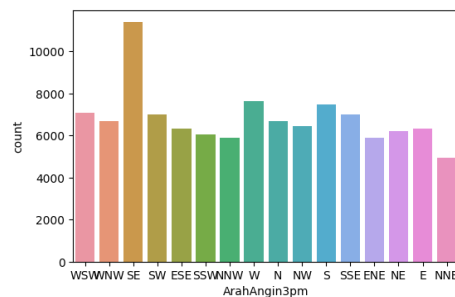
Arah Angin Terkencang



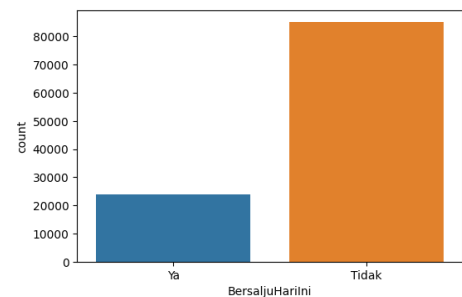
Arah Angin 9am



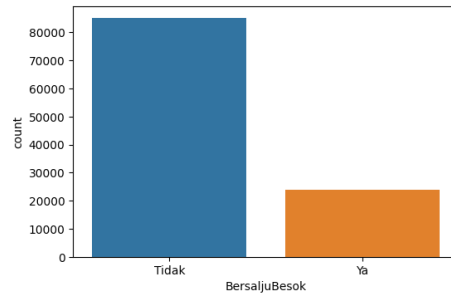
Arah Angin 3pm



Bersalju Hari Ini

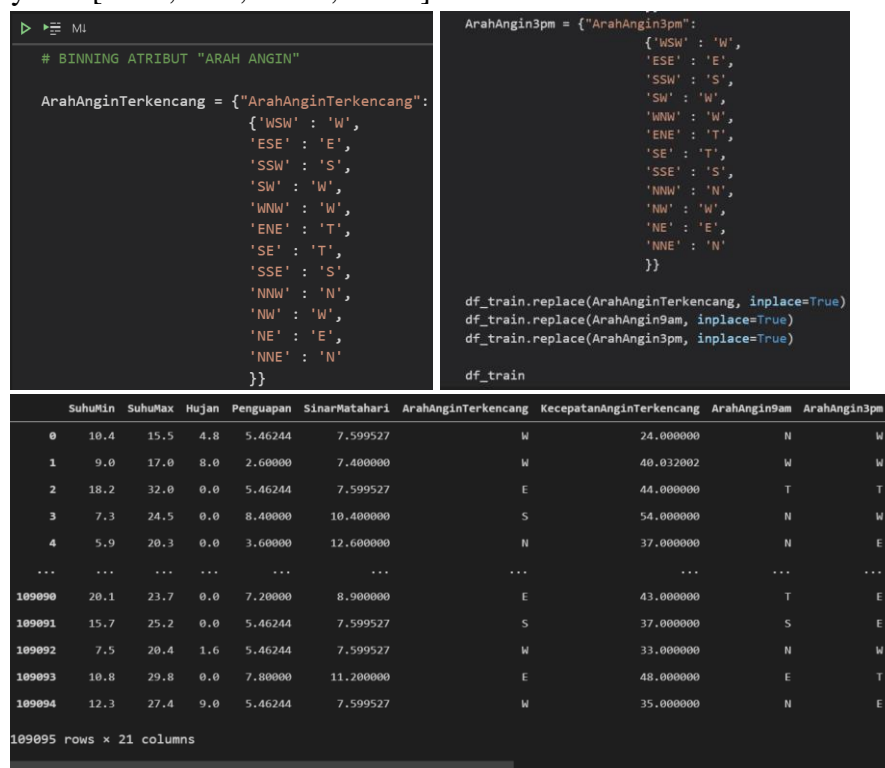


Bersalju Besok



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['ArahAnginTer kencang'] = labelencoder.fit_transform(df_train['ArahAnginTer kencang'])
labelencoder = LabelEncoder()
df_train['ArahAngin9am'] = labelencoder.fit_transform(df_train['ArahAngin9am'])
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	Penguapan	SinarMatahari	ArahAnginTer kencang	KecepatanAnginTer kencang	ArahAngin9am	ArahAngin3pm	KecepatanAng
0	10.4	15.5	4.8	5.46244	7.599527	4	24.000000	1	4	
1	9.0	17.0	8.0	2.60000	7.400000	4	40.032002	4	4	
2	18.2	32.0	0.0	5.46244	7.599527	0	44.000000	3	3	
3	7.3	24.5	0.0	8.40000	10.400000	2	54.000000	1	4	
4	5.9	20.3	0.0	3.60000	12.600000	1	37.000000	1	0	
...
109090	20.1	23.7	0.0	7.20000	8.900000	0	43.000000	3	0	
109091	15.7	25.2	0.0	5.46244	7.599527	2	37.000000	2	0	
109092	7.5	20.4	1.6	5.46244	7.599527	4	33.000000	1	4	
109093	10.8	29.8	0.0	7.80000	11.200000	0	48.000000	0	3	
109094	12.3	27.4	9.0	5.46244	7.599527	4	35.000000	1	0	

109095 rows x 21 columns

q. Scalling

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

```
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	Penguapan	SinarMatahari	ArahAnginTer kencang	KecepatanAnginTer kencang	ArahAngin9am	ArahAngin3pm	KecepatanAng
0	0.445755	0.389635	0.012938	0.037672	0.531435	1.00	0.132812	0.25	1.00	0.0
1	0.412736	0.418426	0.021563	0.017931	0.517483	1.00	0.258063	1.00	1.00	0.1
2	0.629717	0.706334	0.000000	0.037672	0.531435	0.00	0.289062	0.75	0.75	0.1
3	0.372642	0.562380	0.000000	0.057931	0.727273	0.50	0.367188	0.25	1.00	0.1
4	0.339623	0.481766	0.000000	0.024828	0.881119	0.25	0.234375	0.25	0.00	0.1

5 rows x 21 columns

r. Save Data

Data untuk clustering sudah siap digunakan.

```
# SAVE TO CSV

df_train.to_csv(r'C:\Users\haurya\OneDrive\Documents\SEMESTER 6\Pembelajaran Mesin (MaLing)\Tubes\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.

```
> *# Ml

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 l in range(0, n_neighbour):
                euclid.append(euclidian_distance(X[i][:n_feat], centroids[l]))
            #memilih cluster dari nilai minimum euclidean distance
            idx = np.argmin(euclid)
            cluster.append(idx+1)
            #menambahkan cluster ke X
            X[i][n_feat] = idx+1

#clustering centroid
group = {}
for j in set(cluster):
    group[j] = [i for i in range(len(cluster)) if cluster[i] == j]

#memasukkan info centroid ke tiap cluster
datax = {}
for j in range(1, n_neighbour+1):
    datax[j] = [X[group[j][i]][i:n_feat] for i in range(len(group[j]))]

#mengassign centroid baru ke tiap cluster
new_centroids = []
for l in range(1, n_neighbour+1):
    new_centroids.append(np.mean(datax[l], axis=0).tolist())

if (centroids == new_centroids):
    return centroids
#
centroids = copy.copy(new_centroids)
```

4. Evaluasi

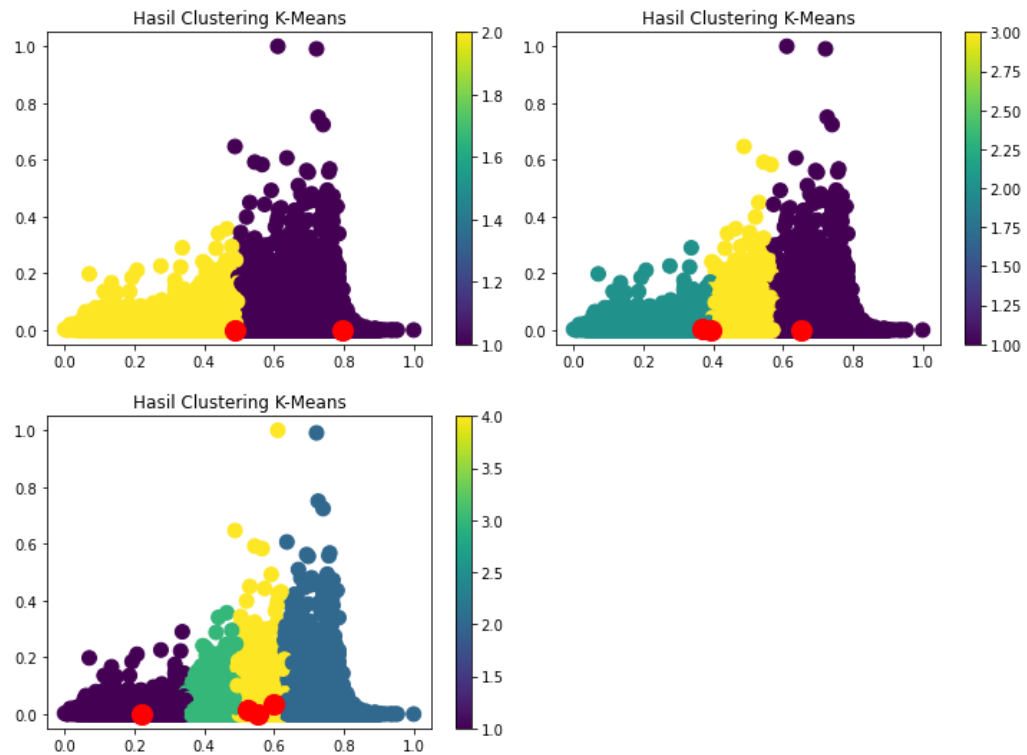
Untuk Evaluasi, dilakukan pencarian silhouette score dari hasil clustering tersebut karena silhouette score dapat mengukur kualitas dari clustering yang dilakukan.

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.

```

> ML

# SILHOUETTE SCORE UNTUK K = 2

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

0.5525798010873928

> ML

# SILHOUETTE SCORE UNTUK K = 3

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

0.5205632552939808

> ML

# SILHOUETTE SCORE UNTUK K = 4

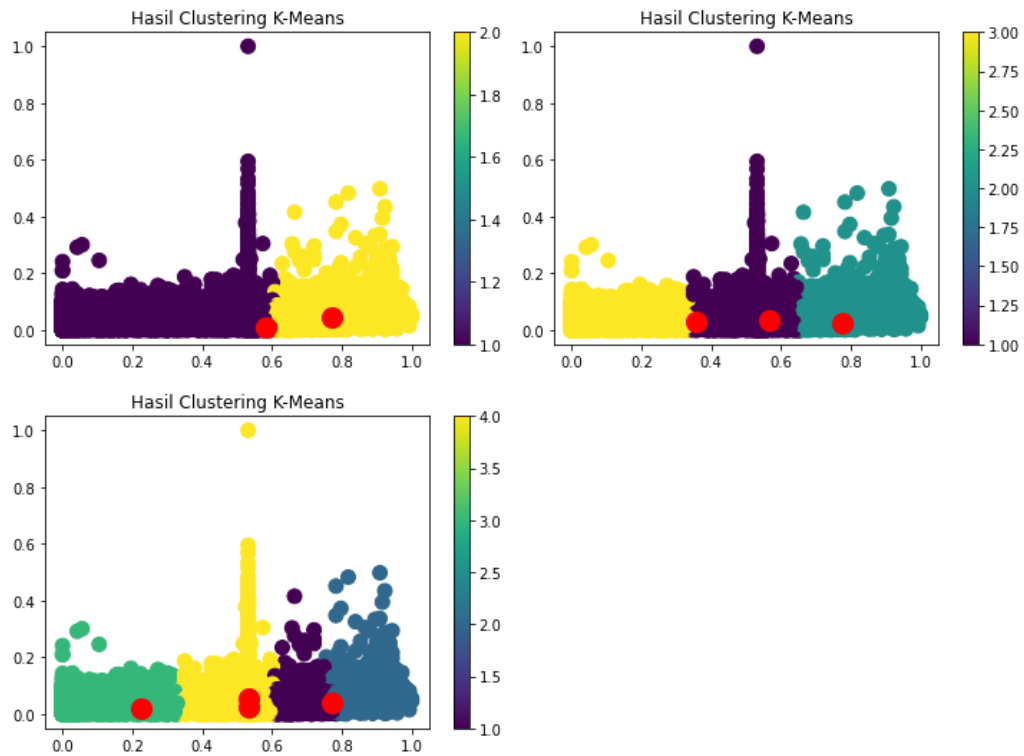
score = silhouette_score(xy.iloc[:,[0,n_feat-1]], xy[n_feat], metric = 'euclidean')
score

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.

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

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

0.579796175128216

▶ ▶ ML
# 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.