LAPORAN TRAINING DATA DAN PREDIKSI MENGGUNAKAN METODE SUPPORT VECTOR MACHINE

"Disusun untuk memenuhi Ujian Akhir Semester Genap Mata Kuliah Pengantar Kercerdasan Artificial"

Dosen Pengampu:

Cakra Adipura Wicaksana, S.T., M.T

Disusun Oleh:

Kelompok 9

1. 3337210002 Ahmad Jumhadi
2. 3337210022 Raka Adi Prasetya
3. 3337210063 A'idah Eka Septiana

Kelas: A

Program Studi Informatika

Fakultas Teknik

Universitas Sultan Ageng Tirtayasa

2023

I. INTRODUCTION

Penggunaan Machine Learning dapat digunakan untuk memprediksi, karena mampu untuk memproses data dalam jumlah besar, menemukan pola perilaku atau hubungan non-eksplisit antara data yang diproses yang tidak terlihat secara langsung, memungkinkan menggunakan model yang merepresentasikan fenomena untuk membuat prediksi atas data baru yang diperoleh darinya, salah satunya adalah melakukan prediksi tentang fenomena hujan.

Hujan merupakan salah satu cuaca di bumi. Hujan sendiri terkadang tidak menentu. Bagi sebagian orang, hujan adalah sesuatu yang dinantikan. Namun bagi sebagian lainnya, hujan terkadang dipandang sebagai kendala dalam menjalankan aktivitas sehari-hari, terutama bagi wisatawan yang ingin berlibur mengunjungi berbagai objek wisata yang berbeda.

Oleh sebab itu, prediksi mengenai hujan diperlukan. Prakiraan hujan dapat mempermudah aktivitas sehari-hari. Prakiraan hujan juga dapat membantu memandu dan mendorong wisatawan untuk mengunjungi dan menghindari daerah tertentu.

Fenomena meteorologi adalah bidang yang menghasilkan data dalam jumlah besar dan sulit untuk membuat prediksi tentang peristiwa yang akan terjadi karena banyaknya variabel yang bergantung padanya. Untuk melakukan ini, kumpulan data sampel diambil sebagai contoh untuk mendeskripsikan pengukuran tema yang dikumpulkan tentang curah hujan pada kota – kota utama di Australia pada tahun 2008 – 2017.

Support Vector Machine adalah salah satu metode dari Machine Learning yang dapat digunakan untuk memprediksi, dan peramalan time series, karena SVM memiliki fungsi kernel sehingga dapat menyelesaikan permasalahan non – linear.

II. METODE YANG DIGUNAKAN

Dalam mentraining dan memprediksi dataset pada kesempatan kali ini, kelompok kami menggunakan metode Support Vector Machine (SVM), yaitu teknik yang relatif baru untuk melakukan prediksi, baik dalam kasus klasifikasi maupun regresi. Support Vector Machine masuk kelas supervised learning, dimana dalam implementasinya perlu adanya tahap pelatihan menggunakan sequential training SVM dan disusul tahap pengujian

(Santosa, 2015). Konsep klasifikasi dengan Support Vector Machine adalah mencari hyperplane terbaik yang berfungsi sebagai pemisah dua kelas data. Support Vector Machine mampu bekerja pada dataset yang berdimensi tinggi dengan menggunakan kernel trik. Support Vector Machine hanya menggunakan beberapa titik data terpilih yang berkontribusi (support vector) untuk membentuk model yang akan digunakan dalam proses klasifikasi.

III. DATA SET

Dataset yang digunakan dalam project ini berisi pengamatan cuaca pada tahun 2008 – 2017, pada berbagai lokasi di seluruh Australia. Data yang tersedia dalam dataset ini berjumlah 142,193 baris dan 24 kolom. Dataset yang didapat bersumber dari: https://www.kaggle.com/datasets/filhypedeeplearning/australia-rain-tomorrow (diakses pada 18 Juni 2023)

Attribute Information

Nama	Deskripsi	Banyak Data	Missing Values	Type
Date	Tanggal observasi	142,193	0	String
Location	Lokasi observasi	142,193	0	String
MinTemp	Suhu minimum dalam 24 jam hingga 9 pagi.	141,556	637	Float
MaxTemp	Suhu maksimum dalam 24 jam dari jam 9 pagi.	141,871	322	Float
Rainfall	Curah hujan (curah hujan) dalam 24 jam sampai jam 9 pagi.	140,787	1,406	Float
Evaporation	Evaporasi dari "Kelas A" dalam 24 jam, hingga jam 9 pagi.	81,350	60,843	Float
Sunshine	Sinar matahari cerah dalam 24 jam hingga tengah malam.	74,377	67,816	Float

WindGustDir	Arah hembusan angin terkuat dalam 24 jam hingga tengah malam.	132,863	9,330	String
WindGustSpeed	Kecepatan hembusan angin terkuat dalam 24 jam hingga tengah malam.	132,923	9,270	Float
WindDir9am	Arah angin rata-rata lebih dari 10 menit sebelum jam 9 pagi	132,180	10,013	String
WindDir3pm	Arah hembusan angin pada pukul 15.00 WIB.	138,415	3,778	String
WindSpeed9am	Kecepatan angin (dalam kilometer per jam) rata-rata lebih dari 10 menit sebelum jam 9 pagi.	140,845	1,348	Float
WindSpeed3pm	Kecepatan angin (dalam kilometer per jam) rata-rata lebih dari 10 menit sebelum jam 3 sore.	139,563	2,630	Float
Humidity9am	Kelembaban relatif (dalam persen) pada jam 9 pagi	140,419	1,774	Float
Humidity3pm	Kelembaban relatif (dalam persen) pada pukul 3 sore.	138,583	3,610	Float
Pressure9am	Tekanan atmosfer (hpa) berkurang menjadi permukaan laut rata-rata pada pukul 9 pagi.	128,179	14,014	Float
Pressure3pm	Tekanan atmosfer (hpa) berkurang menjadi permukaan laut rata-rata pada pukul 3 sore.	128,212	13,981	Float
Cloud9am	Bagian langit yang tertutup awan pada pukul 09.00. Ini diukur dalam ”oktas”, yang merupakan satuan per delapan. Ini mencatat berapa perdelapan langit tertutup	88,563	53,657	Float

	awan. Ukuran 0 menunjukkan langit yang benar-benar cerah, sedangkan angka 8 menunjukkan bahwa langit benar-benar mendung.			
Cloud3pm	Bagian langit yang tertutup awan pada pukul 15.00. lihat Cloud9am untuk deskripsi nilai	85,099	57,094	Float
Temp9am	Suhu (C) pada jam 9 pagi.	141,289	904	Float
Temp3pm	Suhu (C) pada pukul 3 sore.	139,467	2,726	Float
RainToday	Integer 1 jika curah hujan (dalam milimeter) pada 24 jam sampai jam 9 pagi melebihi 1 mm, jika tidak maka 0.	140,787	1,406	String
RISK_MM	Variabel target kontinu, jumlah hujan yang tercatat pada hari berikutnya.	142,193	0	Float
RainTomorrow	Variabel target biner apakah hujan atau tidak pada hari berikutnya.	142,193	0	String

IV. PROSES KERJA

1. Import Libraries yang dibutuhkan

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import datetime
import missingno as msno
from sklearn import svm
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.metrics import f1_score
from sklearn.metrics import average_precision_score
from sklearn.metrics import accuracy_score
from sklearn.metrics import confusion_matrix, classification_report, precision_score, recall_score
import pickle
```

2. Data Loading

Kemudian lakukan data loading yang digunakan untuk membaca data dari file atau sumber data dan mengubahnya menjadi format yang dapat dengan mudah dimanipulasi dan dianalisis dalam bahasa pemrograman.

In [2]:
```
weather_data = pd.read_csv("australia_rain_tomorrow_raw.csv")
pd.set_option("display.max_columns", 24)
```

In [3]:
```
df = weather_data.copy()
```

In [4]:
```
# Menampilkan 5 data pertama
df.head()
```

Out[4]:

	Date	Location	MinTemp	MaxTemp	Rainfall	Evaporation	Sunshine	WindGustDir	WindGustSpeed	WindDir9am	WindDir3pm	WindSpeed9am	WindSp
0	01/12/2008	Albury	13.4	22.9	0.6	NaN	NaN	W	44.0	W	WNW	20.0	
1	02/12/2008	Albury	7.4	25.1	0.0	NaN	NaN	WNW	44.0	NNW	WSW	4.0	
2	03/12/2008	Albury	12.9	25.7	0.0	NaN	NaN	WSW	46.0	W	WSW	19.0	
3	04/12/2008	Albury	9.2	28.0	0.0	NaN	NaN	NE	24.0	SE	E	11.0	
4	05/12/2008	Albury	17.5	32.3	1.0	NaN	NaN	W	41.0	ENE	NW	7.0	

In [5]:
```
# Menampilkan 5 data terakhir
df.tail()
```

Out[5]:

	Date	Location	MinTemp	MaxTemp	Rainfall	Evaporation	Sunshine	WindGustDir	WindGustSpeed	WindDir9am	WindDir3pm	WindSpeed9am	W
142188	20/06/2017	Uluru	3.5	21.8	0.0	NaN	NaN	E	31.0	ESE	E	15.0	
142189	21/06/2017	Uluru	2.8	23.4	0.0	NaN	NaN	E	31.0	SE	ENE	13.0	
142190	22/06/2017	Uluru	3.6	25.3	0.0	NaN	NaN	NNW	22.0	SE	N	13.0	
142191	23/06/2017	Uluru	5.4	26.9	0.0	NaN	NaN	N	37.0	SE	WNW	9.0	
142192	24/06/2017	Uluru	7.8	27.0	0.0	NaN	NaN	SE	28.0	SSE	N	13.0	

In [6]:
```
# Memeriksa ukuran dari dataset (jumlah baris, jumlah kolom)
df.shape
```

Out[6]: (142193, 24)

In [7]:
```
# Ringkasan dataset (tipe masing-masing kolom pada data)
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 142193 entries, 0 to 142192
Data columns (total 24 columns):
 #   Column         Non-Null Count   Dtype
---  ------         --------------   -----
 0   Date           142193 non-null  object
 1   Location       142193 non-null  object
 2   MinTemp        141556 non-null  float64
 3   MaxTemp        141871 non-null  float64
 4   Rainfall       140787 non-null  float64
 5   Evaporation    81350 non-null   float64
 6   Sunshine       74377 non-null   float64
 7   WindGustDir    132863 non-null  object
 8   WindGustSpeed  132923 non-null  float64
 9   WindDir9am     132180 non-null  object
 10  WindDir3pm     138415 non-null  object
 11  WindSpeed9am   140845 non-null  float64
 12  WindSpeed3pm   139563 non-null  float64
 13  Humidity9am    140419 non-null  float64
 14  Humidity3pm    138583 non-null  float64
 15  Pressure9am    128179 non-null  float64
 16  Pressure3pm    128212 non-null  float64
 17  Cloud9am       88536 non-null   float64
 18  Cloud3pm       85099 non-null   float64
 19  Temp9am        141289 non-null  float64
 20  Temp3pm        139467 non-null  float64
 21  RainToday      140787 non-null  object
 22  RISK_MM        142193 non-null  float64
 23  RainTomorrow   142193 non-null  object
dtypes: float64(17), object(7)
memory usage: 26.0+ MB
```

Berdasarkan proses diatas, diketahui bahwa dataset berjumlah 142,193 baris dengan 24 kolom, yang mana terdapat 17 kolom bertipe data float dan 7 kolom bertipe data string.

3. Data Cleaning

Selanjutnya, masuk ke dalam tahap data cleaning untuk menganalisis data, yang berguna untuk membersihkan, mengubah, atau menghapus data yang tidak lengkap, tidak akurat, atau tidak relevan dalam sebuah dataset. Dengan tujuan untuk memastikan bahwa data yang digunakan untuk analisis atau pemodelan lebih berkualitas, valid, dan informative.

In [8]:
```
# Memeriksa adanya duplikasi dalam data
df.duplicated().sum()
```

Out[8]:
```
0
```

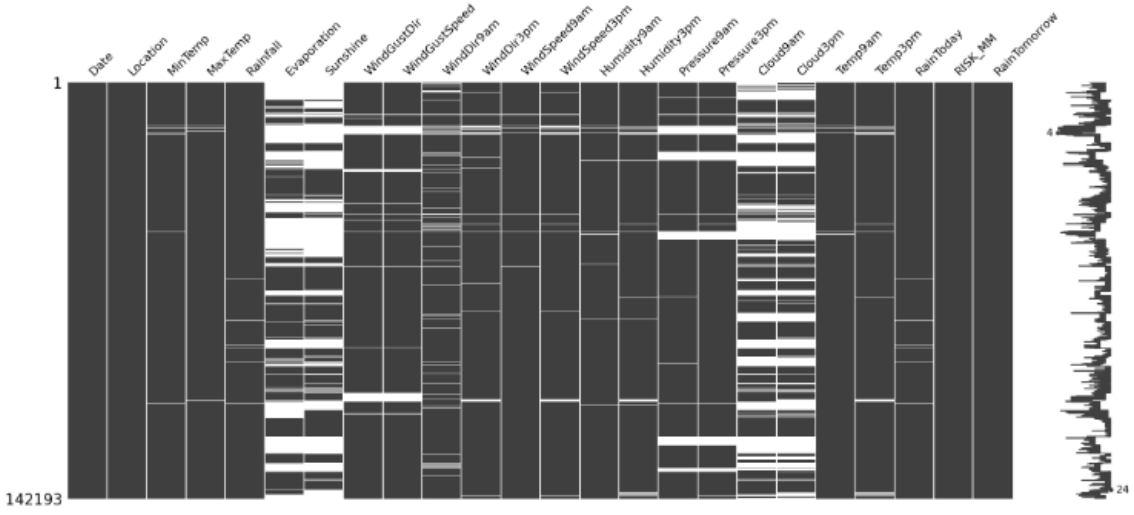
In [9]:
```
# Membuat dataframe jumlah missing values dan persentasenya
nan_sum = df.isnull().sum()
nan_sum_percentage = np.asarray((nan_sum/len(df))*100)
nan_sum = pd.DataFrame(nan_sum)
nan_sum.rename(columns={0:"nan_sum"}, inplace=True)
nan_sum["percentage"] = nan_sum_percentage

nan_sum
```

Out[9]:

	nan_sum	percentage
Date	0	0.000000
Location	0	0.000000
MinTemp	637	0.447983
MaxTemp	322	0.226453
Rainfall	1406	0.988797
Evaporation	60843	42.789026
Sunshine	67816	47.692924
WindGustDir	9330	6.561504
WindGustSpeed	9270	6.519308
WindDir9am	10013	7.041838
WindDir3pm	3778	2.656952
WindSpeed9am	1348	0.948007
WindSpeed3pm	2630	1.849599
Humidity9am	1774	1.247600
Humidity3pm	3610	2.538803
Pressure9am	14014	9.855619
Pressure3pm	13981	9.832411
Cloud9am	53657	37.735332
Cloud3pm	57094	40.152469
Temp9am	904	0.635756
Temp3pm	2726	1.917113
RainToday	1406	0.988797
RISK_MM	0	0.000000
RainTomorrow	0	0.000000

Berdasarkan tabel diatas, diketahui tidak terdapat duplikasi dalam data, tetapi terdapat adanya missing values, terutama pada kolom Evaporation, Sunshine, Cloud9am, dan Cloud3pm. Missing Values keseluruhan data dapat dilihat pada visualisasi dibawah:

```
In [10]: msno.matrix(df)
Out[10]: <Axes: >
```



Kemudian kami melakukan teknik **Moving Average** untuk mengatasi missing values pada setiap kolom numerik. Teknik ini digunakan karena dataset adalah data deret waktu, serta berdasarkan waktu dan tempat juga diduga memiliki perbedaan nilai kondisi cuaca tertentu.

```
In [11]: # Memisahkan kolom kategorik dan numerik
categorical = [cols for cols in df.columns if df[cols].dtypes=='object']
numerical = [cols for cols in df.columns if df[cols].dtypes=='float64']
```

In [12]:
```
# Mengisi missing values setiap kolom dengan moving average (rolling)
for cols in numerical:
    if df[cols].isnull().sum() != 0:
        while True:
            df[cols] = df[cols].fillna(df[cols].rolling(min_periods = 1, window = 7, center=True).mean())
            if df[cols].isnull().sum() == 0:
                    break
```

In [13]:
```
# Memeriksa kembali apakah masih terdapat missing values
nan_sum = df.isnull().sum()
nan_sum_percentage = np.asarray((nan_sum/len(df))*100)
nan_sum = pd.DataFrame(nan_sum)
nan_sum.rename(columns={0:"nan_sum"}, inplace=True)
nan_sum["percentage"] = nan_sum_percentage

nan_sum
```

Out[13]:

	nan_sum	percentage
Date	0	0.000000
Location	0	0.000000
MinTemp	0	0.000000
MaxTemp	0	0.000000
Rainfall	0	0.000000
Evaporation	0	0.000000
Sunshine	0	0.000000
WindGustDir	9330	6.581504
WindGustSpeed	0	0.000000
WindDir9am	10013	7.061838
WindDir3pm	3778	2.664952
WindSpeed9am	0	0.000000
WindSpeed3pm	0	0.000000
Humidity9am	0	0.000000
Humidity3pm	0	0.000000
Pressure9am	0	0.000000
Pressure3pm	0	0.000000
Cloud9am	0	0.000000
Cloud3pm	0	0.000000
Temp9am	0	0.000000
Temp3pm	0	0.000000
RainToday	1406	0.988797
RISK_MM	0	0.000000
RainTomorrow	0	0.000000

Missing values pada kolom numerik sudah teratasi. Selanjutnya untuk kolom yang masih terdapat missing values akan dihilangkan, terutama pada kolom RainToday dan RainTomorrow karena hanya mengandung 2% missing values, yang mana ini tidak akan memberikan pengaruh yang signifikan pada analisis data. Di sisi lain, kolom RainTomorrow adalah kolom target sehingga ada kemungkinan bias dalam prediksi mendatang.

In [14]:
```
# Menghilangkan missing values pada kolom RainToday dan RainTomorrow
df.dropna(subset=['RainToday','RainTomorrow'], axis=0, inplace=True)
```

Kemudian, untuk kolom yang mengandung <10% missing values, maka missing values pada kolom tersebut akan dihilangkan.

In [15]:
```
# Menghilangkan missing values pada kolom kategorik lain karena mengandung <10% missing values
df.dropna(subset=['WindGustDir','WindDir9am','WindDir3pm'], axis=0, inplace=True)
```

In [16]:
```
# Memeriksa kembali apakah masih terdapat missing values
nan_sum = df.isnull().sum()
nan_sum_percentage = np.asarray((nan_sum/len(df))*100)
nan_sum = pd.DataFrame(nan_sum)
nan_sum.rename(columns={0:"nan_sum"}, inplace=True)
nan_sum["percentage"] = nan_sum_percentage

nan_sum
```

Out[16]:

	nan_sum	percentage
Date	0	0.0
Location	0	0.0
MinTemp	0	0.0
MaxTemp	0	0.0
Rainfall	0	0.0
Evaporation	0	0.0
Sunshine	0	0.0
WindGustDir	0	0.0
WindGustSpeed	0	0.0
WindDir9am	0	0.0
WindDir3pm	0	0.0
WindSpeed9am	0	0.0
WindSpeed3pm	0	0.0
Humidity9am	0	0.0
Humidity3pm	0	0.0
Pressure9am	0	0.0
Pressure3pm	0	0.0
Cloud9am	0	0.0
Cloud3pm	0	0.0
Temp9am	0	0.0
Temp3pm	0	0.0
RainToday	0	0.0
RISK_MM	0	0.0
RainTomorrow	0	0.0

Dilihat dari tabel diatas, didapatkan bahwa missing values sudah terselesaikan. Selanjutnya, kolom RainToday dan RainTomorrow akan diubah menjadi kolom numerik, yaitu dengan mengganti nilai "Yes" dengan angka 1 dan "No" dengan angka 0. Selain itu, format kolom Date akan diubah menjadi format datetime dan kolom bulan dan tahun akan dibuat.

In [17]:
```
df['RainTomorrow'] = df['RainTomorrow'].map({'Yes': 1, 'No': 0})
df['RainToday'] = df['RainToday'].map({'Yes': 1, 'No': 0})
```

In [18]:
```
# Mengubah format kolom Date menjadi datetime
df["Date"] = pd.to_datetime(df["Date"], format="%d/%m/%Y")
# Membuat kolom baru
df["year"] = df["Date"].dt.year
df["month"] = df["Date"].dt.month
df["day"] = df["Date"].dt.day
```

In [19]:
```
# Memeriksa kembali ringkasan dataset (tipe masing-masing kolom pada data dan jumlah baris/kolom)
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 123710 entries, 0 to 142192
Data columns (total 27 columns):
 #   Column         Non-Null Count   Dtype
---  ------         --------------   -----
 0   Date           123710 non-null  datetime64[ns]
 1   Location       123710 non-null  object
 2   MinTemp        123710 non-null  float64
 3   MaxTemp        123710 non-null  float64
 4   Rainfall       123710 non-null  float64
 5   Evaporation    123710 non-null  float64
 6   Sunshine       123710 non-null  float64
 7   WindGustDir    123710 non-null  object
 8   WindGustSpeed  123710 non-null  float64
 9   WindDir9am     123710 non-null  object
 10  WindDir3pm     123710 non-null  object
 11  WindSpeed9am   123710 non-null  float64
 12  WindSpeed3pm   123710 non-null  float64
 13  Humidity9am    123710 non-null  float64
 14  Humidity3pm    123710 non-null  float64
 15  Pressure9am    123710 non-null  float64
 16  Pressure3pm    123710 non-null  float64
 17  Cloud9am       123710 non-null  float64
 18  Cloud3pm       123710 non-null  float64
 19  Temp9am        123710 non-null  float64
 20  Temp3pm        123710 non-null  float64
 21  RainToday      123710 non-null  int64
 22  RISK_MM        123710 non-null  float64
 23  RainTomorrow   123710 non-null  int64
 24  year           123710 non-null  int32
 25  month          123710 non-null  int32
 26  day            123710 non-null  int32
dtypes: datetime64[ns](1), float64(17), int32(3), int64(2), object(4)
memory usage: 25.0+ MB
```

4. Eksplorasi Analisis Data

Kemudian, pada tahap ini kami akan menggali dan memahami data secara mendalam untuk mengidentifikasi pola, tren, hubungan, dan informasi yang tersembunyi dalam dataset. Dengan tujuan untuk mendapatkan pemahaman yang lebih baik tentang data yang ada, mengidentifikasi pertanyaan atau hipotesis yang menarik, dan mempersiapkan data untuk analisis lanjutan atau pemodelan.

```
In [20]: pd.set_option("display.max_columns", 26)
```

```
In [21]: df.head()
```

Out[21]:

	Date	Location	MinTemp	MaxTemp	Rainfall	Evaporation	Sunshine	WindGustDir	WindGustSpeed	WindDir9am	WindDir3pm	WindSpeed9am	WindSpeed3
0	2008-12-01	Albury	13.4	22.9	0.6	12.0	12.3	W	44.0	W	WNW	20.0	2
1	2008-12-02	Albury	7.4	25.1	0.0	12.0	12.3	WNW	44.0	NNW	WSW	4.0	2
2	2008-12-03	Albury	12.9	25.7	0.0	12.0	12.3	WSW	46.0	W	WSW	19.0	2
3	2008-12-04	Albury	9.2	28.0	0.0	12.0	12.3	NE	24.0	SE	E	11.0	
4	2008-12-05	Albury	17.5	32.3	1.0	12.0	12.3	W	41.0	ENE	NW	7.0	2

5 rows × 27 columns


```
In [22]: df.tail()
```

Out[22]:

	Date	Location	MinTemp	MaxTemp	Rainfall	Evaporation	Sunshine	WindGustDir	WindGustSpeed	WindDir9am	WindDir3pm	WindSpeed9am	WindS
142188	2017-06-20	Uluru	3.5	21.8	0.0	6.4	10.7	E	31.0	ESE	E	15.0	
142189	2017-06-21	Uluru	2.8	23.4	0.0	6.4	10.7	E	31.0	SE	ENE	13.0	
142190	2017-06-22	Uluru	3.6	25.3	0.0	6.4	10.7	NNW	22.0	SE	N	13.0	
142191	2017-06-23	Uluru	5.4	26.9	0.0	6.4	10.7	N	37.0	SE	WNW	9.0	
142192	2017-06-24	Uluru	7.8	27.0	0.0	6.4	10.7	SE	28.0	SSE	N	13.0	

5 rows × 27 columns

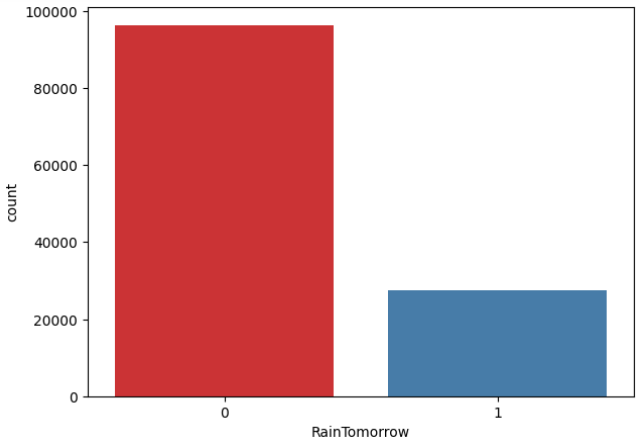
5. Frekuensi nilai pada kolom rain tomorrow.

Selanjutnya, kolom RainTomorrow akan dijadikan target untuk memprediksi. Sehingga projek ini didasari atas penelitian tentang kondisi cuaca hari itu, apakah keesokan harinya akan hujan (yes) atau tidak (no).

```
In [23]: df['RainTomorrow'].value_counts()
```

```
Out[23]: RainTomorrow
         0    96318
         1    27392
         Name: count, dtype: int64
```

```
In [24]: f, ax = plt.subplots(figsize=(7, 5))
         ax = sns.countplot(x="RainTomorrow", data=df, palette="Set1")
         plt.show()
```



Berdasarkan grafik diatas, diketahui jumlah hari tanpa hujan untuk kondisi cuaca yang terekam pada dataset lebih besar dari jumlah hari tanpa hujan.

6. Statistik Deskriptif dan Analisis Persebaran Parameter Numerik

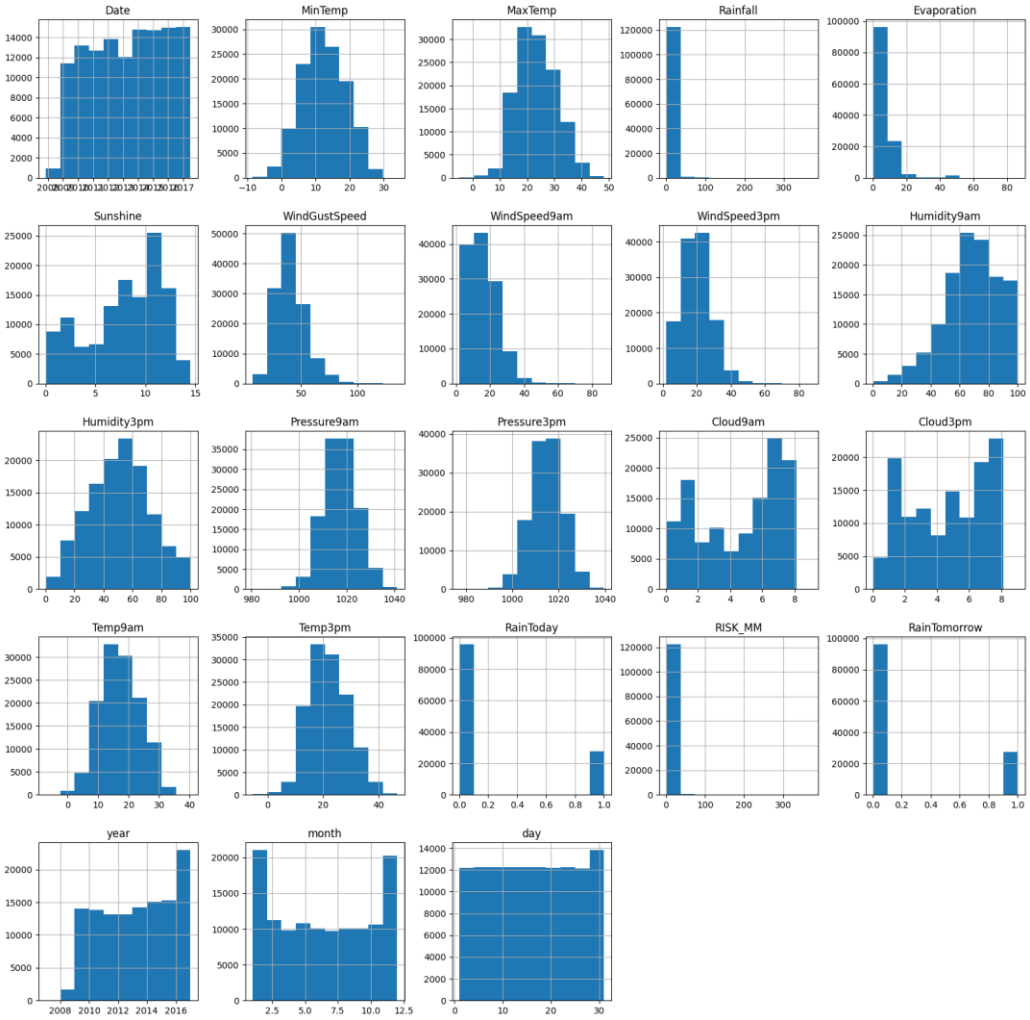
```
In [25]: df.describe()
```

Out[25]:

	Date	MinTemp	MaxTemp	Rainfall	Evaporation	Sunshine	WindGustSpeed	WindSpeed9am	WindSpeed3pm	Hum
count	123710	123710.000000	123710.000000	123710.000000	123710.000000	123710.000000	123710.000000	123710.000000	123710.000000	12371
mean	2013-04-19 03:45:49.106781952	12.416223	23.477200	2.375661	6.579938	7.926909	40.685280	15.011689	19.174578	6
min	2007-11-01 00:00:00	-8.500000	-4.800000	0.000000	0.000000	0.000000	7.000000	2.000000	2.000000	
25%	2011-01-31 00:00:00	7.800000	18.100000	0.000000	2.800000	5.400000	31.000000	9.000000	13.000000	5
50%	2013-06-20 00:00:00	12.200000	23.000000	0.000000	5.400000	8.400000	39.000000	13.000000	19.000000	6
75%	2015-06-29 00:00:00	17.100000	28.600000	0.800000	8.400000	11.200000	48.000000	20.000000	24.000000	8
max	2017-06-25 00:00:00	33.900000	48.100000	367.600000	86.200000	14.500000	135.000000	87.000000	87.000000	10
std	NaN	6.369351	7.208249	8.528039	6.268998	3.769802	13.388636	8.307285	8.571039	1

Menampilkan Histogram distribusi data:

```
In [26]: df.hist(figsize=(20, 20))
         plt.show()
```



Berdasarkan plot histogram di atas, dapat diketahui bahwa:

- Grafik MinTemp dan MaxTemp memiliki distribusi yang sama.
- Grafik Rainfall dan Evaporation memiliki nilai distribusi yang ekstrim.
- Grafik WindGustSpeed, WindGust9am, dan WindGust3pm memiliki distribusi yang mirip. Tetapi WindGust3pm memiliki nilai rata – rata lebih tinggi sehingga didapatkan kesimpulan kecepatan angina pada sore hari lebih tinggi.
- Grafik Humidity9am dan Humidity3pm memiliki distribusi yang mirip. Tetapi Grafik Humidity9am memiliki rata – rata lebih besar, sehingga dapat disimpulkan kelembapan pada pagi hari lebih tinggi, dibandingkan sore hari.
- Grafik Pressure9am dan Pressure 3pm memiliki distribusi yang mirip, dan memiliki nilai rata – rata dan nilai tengah yang tidak berbeda jauh. Sehingga disimpuljan tekanan udara pada pagi dan sore hari tidak memiliki perubahan yang signifikan.

- Grafik Cloud9am, dengan Cloud3pm dan Grafik Temp9am dan Temp3pm memiliki distribusi yang mirip. Sehingga dapat disimpulkan pada waktu pagi dan sore hari persentase langit yang tertutup awan tidak memiliki perubahan yang signifikan, tetapi untuk temperature pada sore hari lebih tinggi dibandingkan pada pagi hari

7. Persebaran lokasi berlandaskan terjadinya hujan atau tidak

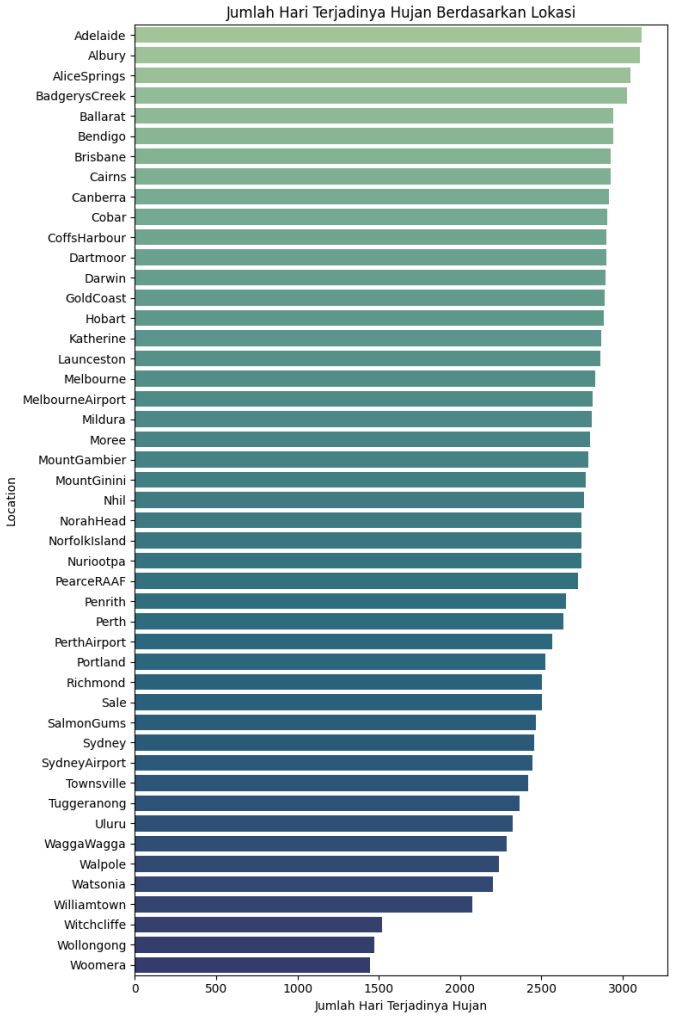
Kemudian, kami akan mencari informasi apakah perbedaan lokasi dapat mempengaruhi turunnya hujan atau tidak.

In [27]:
```
df_rain_by_loc = df.groupby('Location').count()
df_rain_by_loc = df_rain_by_loc[['RainToday']]
df_rain_by_loc.head(10)
```

Out[27]:

Location	RainToday
Adelaide	2746
Albury	2445
AliceSprings	2748
BadgerysCreek	2458
Ballarat	2895
Bendigo	2725
Brisbane	3029
Cairns	2900
Canberra	2762
Cobar	2925

In [28]:
```
plt.figure(figsize=(8, 12))
sns.barplot(x='RainToday',
            y=df_rain_by_loc.index,
            data=df_rain_by_loc.sort_values('RainToday', ascending=False),
            orient='h',
            palette='crest'
           )
plt.xlabel('Jumlah Hari Terjadinya Hujan')
plt.title('Jumlah Hari Terjadinya Hujan Berdasarkan Lokasi')
plt.tight_layout()
```



Berdasarkan plot diatas, didapatkan informasi bahwa perbedaan lokasi akan menunjukkan perbedaan jumlah hari turunnya hujan, sehingga kami mendapatkan hipotesis bahwa lokasi akan mempengaruhi turun dan tidaknya hujan.

8. Pola curah hujan didasarkan pada waktu turunnya hujan per bulan

Selanjutnya kami akan mencari tahu apakah waktu (bulan) dapat mempengaruhi turunnya hujan**.**

In [29]:
```
df_seasonality = df.copy()
df_seasonality.head()
```

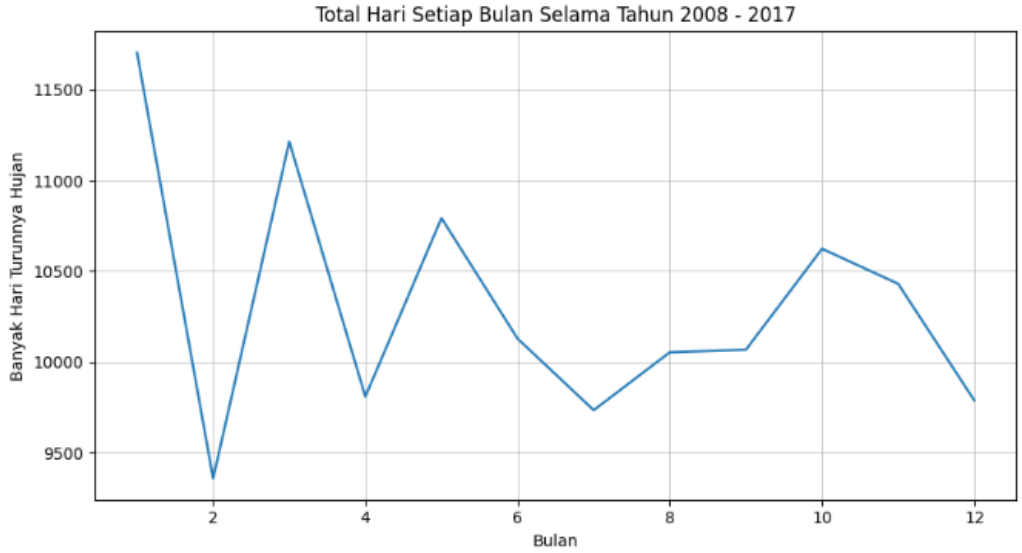
Out[29]:

	Date	Location	MinTemp	MaxTemp	Rainfall	Evaporation	Sunshine	WindGustDir	WindGustSpeed	WindDir9am	WindDir3pm	WindSpeed9am	WindSpeed3
0	2008-12-01	Albury	13.4	22.9	0.6	12.0	12.3	W	44.0	W	WNW	20.0	2
1	2008-12-02	Albury	7.4	25.1	0.0	12.0	12.3	WNW	44.0	NNW	WSW	4.0	2
2	2008-12-03	Albury	12.9	25.7	0.0	12.0	12.3	WSW	46.0	W	WSW	19.0	2
3	2008-12-04	Albury	9.2	28.0	0.0	12.0	12.3	NE	24.0	SE	E	11.0	
4	2008-12-05	Albury	17.5	32.3	1.0	12.0	12.3	W	41.0	ENE	NW	7.0	2

5 rows × 27 columns

In [30]:
```
df_seasonality_grouped = df_seasonality.groupby('month').count()
df_seasonality_grouped = df_seasonality_grouped[['RainToday']]
```

In [31]:
```
plt.figure(figsize=(9,5))
sns.lineplot(data=df_seasonality_grouped, x=df_seasonality_grouped.index, y='RainToday')
plt.title('Total Hari Setiap Bulan Selama Tahun 2008 - 2017')
plt.xlabel('Bulan')
plt.ylabel('Banyak Hari Turunnya Hujan')
plt.grid(linewidth=0.5)
plt.tight_layout()
```



Berdasakan plot diatas, di dapat informasi bahwa intensitas hujan pada bulan 2, berada pada posisi terendah dan intensitas hujan tertinggi berada pada bulan 6. Sehingga kami mendapatkan hipotesis kembali bahwa waktu (bulan) mempengaruhi turun atau tidaknya hujan.

9. Intensitas hujan berdasarkan arah angin.

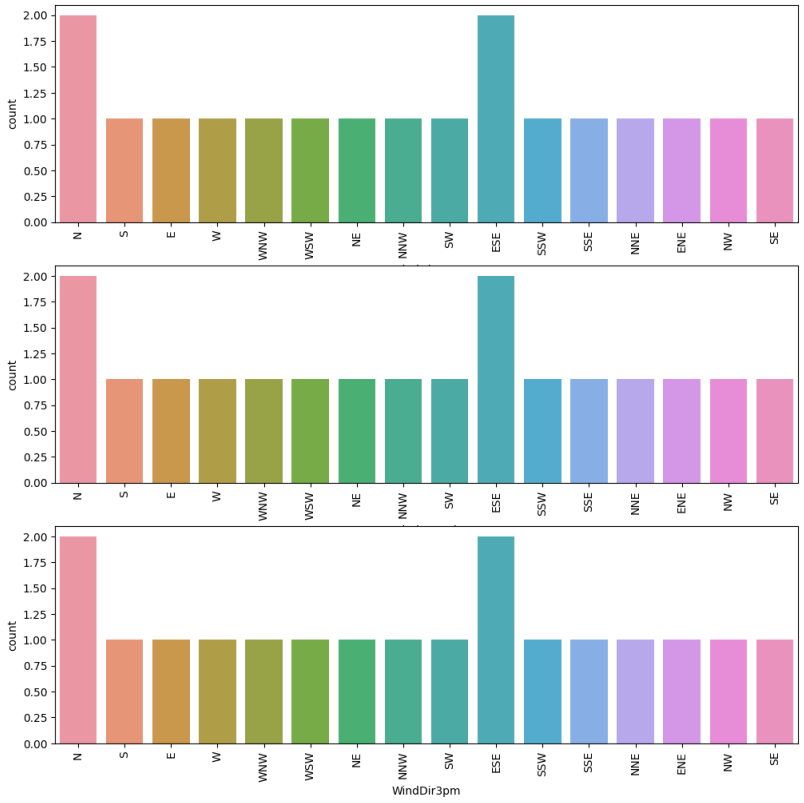
Kemudian kami akan mencari informasi kembali, apakah pada kolom WindGustDir, WinDir9am, dan WinDir3pm akan berpengaruh pada kolom target (RainTomorrow).

In [32]:
```
df_categorical = df[['RainTomorrow','WindGustDir','WindDir9am','WindDir3pm']]
df_categorical = df_categorical[df_categorical['RainTomorrow']==1]
df_categorical.head()
```

Out[32]:

	RainTomorrow	WindGustDir	WindDir9am	WindDir3pm
8	1	NNW	SE	NW
10	1	N	SSE	ESE
11	1	NNE	NE	ENE
12	1	W	NNW	NNW
15	1	ENE	SSW	E

In [33]:
```
df_categorical = pd.DataFrame({
    'WindDir9am': ['N', 'S', 'E', 'W', 'N', 'WNW', 'WSW', 'NE', 'NNW', 'SW', 'ESE', 'SSW', 'SSE', 'NNE', 'ESE', 'ENE', 'NW', 'SE
    'WindGustDir': ['N', 'S', 'E', 'W', 'N', 'WNW', 'WSW', 'NE', 'NNW', 'SW', 'ESE', 'SSW', 'SSE', 'NNE', 'ESE', 'ENE', 'NW', 'S
    'WindDir3pm': ['N', 'S', 'E', 'W', 'N', 'WNW', 'WSW', 'NE', 'NNW', 'SW', 'ESE', 'SSW', 'SSE', 'NNE', 'ESE', 'ENE', 'NW', 'SE
})

categoricalPlot = ['WindDir9am', 'WindGustDir', 'WindDir3pm']
fig, ax = plt.subplots(3, 1, figsize=(12, 12))
c = 0
while c <= 2:
    sns.countplot(x=df_categorical[categoricalPlot[c]], ax=ax[c])
    ax[c].set_xticklabels(ax[c].get_xticklabels(), rotation=90)
    c += 1
plt.show()
```



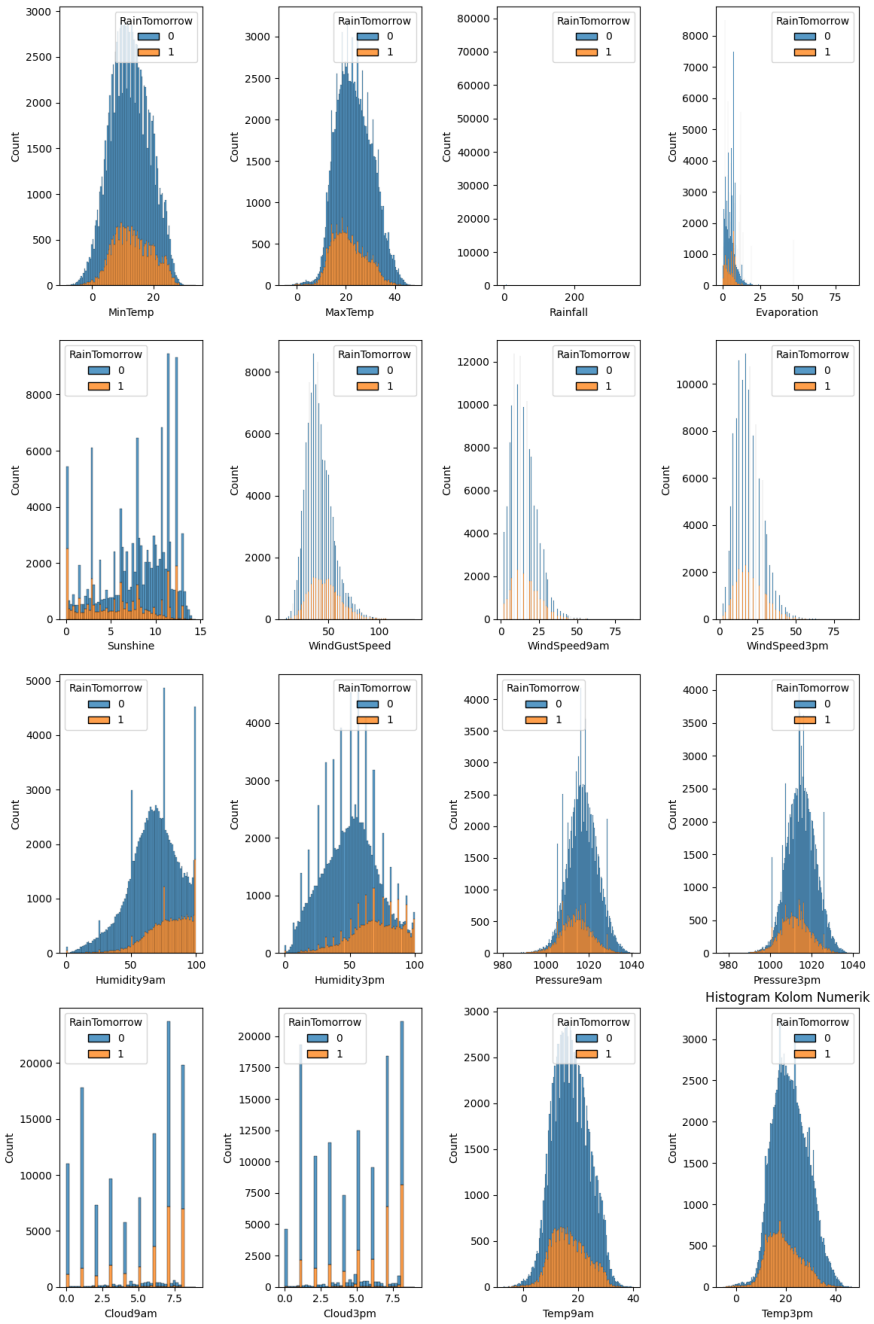
Berdasarkan grafik diatas, didapat informasi bahwa WindGustDir, WinDir9am, dan WinDir3pm tidak menunjukkan pengaruh yang signifikan kepada intensitas turunnya hujan.

10. Persebaran kolom numeric berdasarkan kolom target

Kolom RainTomorrow digunakan sebagai kolom target

In [34]:
```
cols = ['MinTemp','MaxTemp', 'Rainfall', 'Evaporation', 'Sunshine', 'WindGustSpeed', 'WindSpeed9am','WindSpeed3pm',
       'Humidity9am', 'Humidity3pm','Pressure9am','Pressure3pm','Cloud9am','Cloud3pm','Temp9am','Temp3pm']

fig, ax = plt.subplots(4, 4, figsize=(12,18))
ax = ax.reshape(-1)

for i, col in enumerate(cols):
    sns.histplot(data=df, x=col, hue='RainTomorrow', multiple='stack', ax=ax[i])

fig.tight_layout(pad=2.0)
plt.title('Histogram Kolom Numerik')
```

Out[34]: Text(0.5, 1.0, 'Histogram Kolom Numerik')



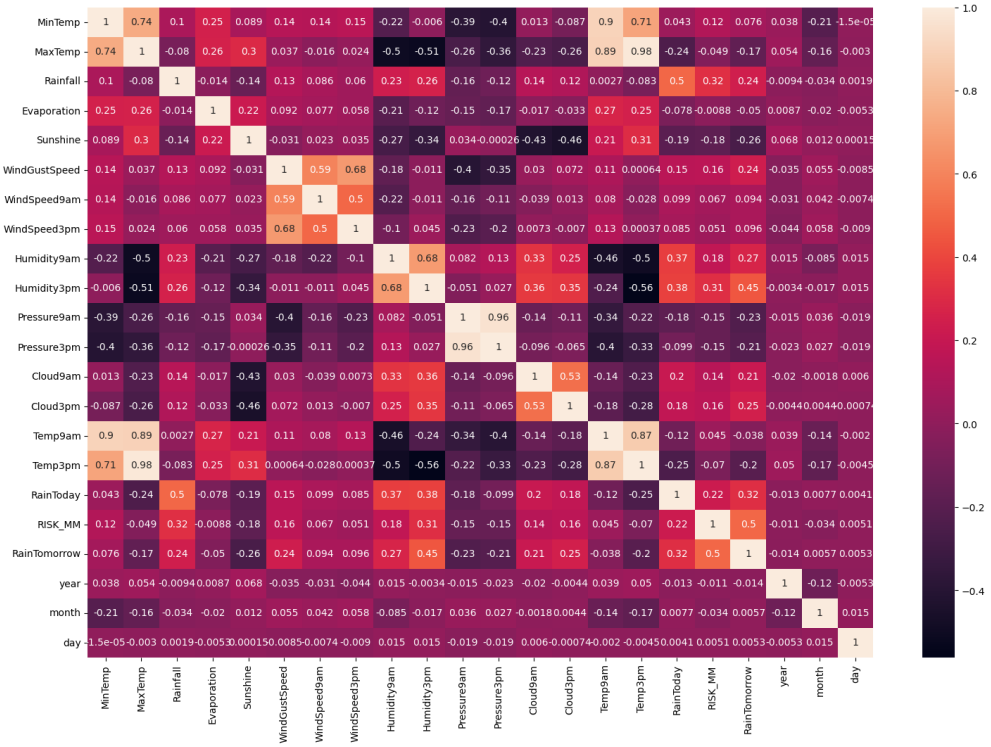
Berdasarkan grafik di atas, kami mendapatkan informasi bahwa:

- Kolom MinTemp dan MaxTemp menunjukkan bahwa semakin rendah suhu, semakin besar kemungkinan hujan. Ini diwakili oleh frekuensi suhu tinggi pada nilai rendah. Serta, hipotesis ini juga didukung oleh distribusi frekuensi pada kolom Temp9am dan Temp3pm.
- Kolom Evaporation, Sunshine, WindGustSpeed, WindSpeed9am, dan WindSpeed3pm, cenderung tidak menunjukkan pengaruh yang signifikan pada kolom RainTomorrow.
- Kolom Humidity9am dan Humidity3pm menunjukkan pengaruh yang signifikan pada kolom RainTomorrow. Hal ini direpresentasikan dengan semakin tingginya humidity, maka semakin tinggi juga frekuensi turunnya hujan.
- Kolom Pressure9am dan Pressure3pm cenderung tidak menunjukkan dampak signifikan pada kolom RainTomorrow.
- Kolom Cloud9am dan Cloud3pm cenderung menunjukkan pengaruh signifikan terhadap kolom RainTomorrow. Hal ini direpresentasikan dengan semakin tingginya frekuensi nilai persentase langit yang tertutup awan, maka frekuensi turunnya hujan juga tinggi.

11. Relasi kolom numeric dan kolom target

In [35]:
```
# Memilih kolom numerik
numeric_columns = df.select_dtypes(include=[np.number])  # Memilih kolom numerik
plt.figure(figsize=(18, 12))
sns.heatmap(numeric_columns.corr(), annot=True)
plt.xticks(rotation=90)
```

Out[35]:
```
(array([ 0.5,  1.5,  2.5,  3.5,  4.5,  5.5,  6.5,  7.5,  8.5,  9.5, 10.5,
        11.5, 12.5, 13.5, 14.5, 15.5, 16.5, 17.5, 18.5, 19.5, 20.5, 21.5]),
 [Text(0.5, 0, 'MinTemp'),
  Text(1.5, 0, 'MaxTemp'),
  Text(2.5, 0, 'Rainfall'),
  Text(3.5, 0, 'Evaporation'),
  Text(4.5, 0, 'Sunshine'),
  Text(5.5, 0, 'WindGustSpeed'),
  Text(6.5, 0, 'WindSpeed9am'),
  Text(7.5, 0, 'WindSpeed3pm'),
  Text(8.5, 0, 'Humidity9am'),
  Text(9.5, 0, 'Humidity3pm'),
  Text(10.5, 0, 'Pressure9am'),
  Text(11.5, 0, 'Pressure3pm'),
  Text(12.5, 0, 'Cloud9am'),
  Text(13.5, 0, 'Cloud3pm'),
  Text(14.5, 0, 'Temp9am'),
  Text(15.5, 0, 'Temp3pm'),
  Text(16.5, 0, 'RainToday'),
  Text(17.5, 0, 'RISK_MM'),
  Text(18.5, 0, 'RainTomorrow'),
  Text(19.5, 0, 'year'),
  Text(20.5, 0, 'month'),
  Text(21.5, 0, 'day')])
```



Berdasarkan heatmap diatas terlihat bahwa kolom Humidity3pm dan RainToday terhadap RainTomorrow memiliki nilai koefisien korelasi tertinggi. Sehingga, memperkuat informasi dari grafik sebelumnya jika semakin tinggi humiditas, maka semakin tinggi kemungkinan hujan besok, dan hujan pada hari kemarin memberikan pengaruh untuk hujan di esok hari.

12. Data Preprocessing

Untuk menyiapkan data sedemikian rupa sehingga dapat diproses secara efisien dan memberikan hasil yang lebih akurat, kami melakukan tahap data preprocessing, dan masuk kedalam tahap dibawah ini:

13. Menghapus kolom korelasi rendah terhadap kolom target

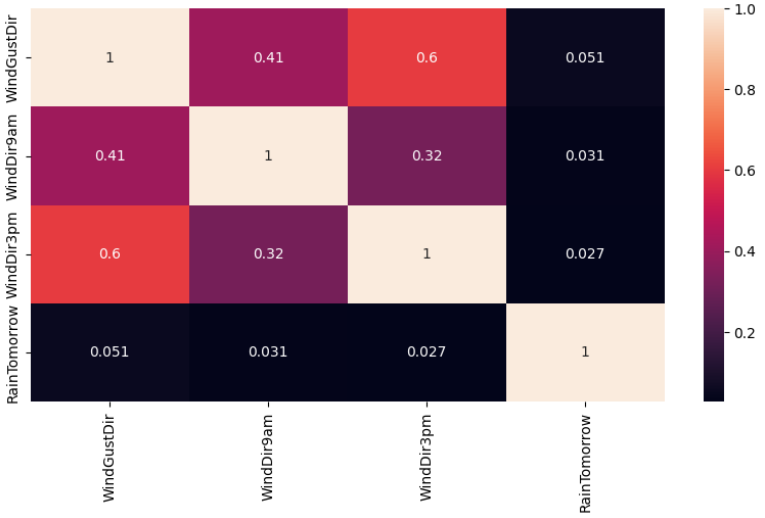
Kemudian kami akan menghapus kolom MinTemp, Evaporation, WindSpeed9am, WindSpeed3pm dan Temp9am karena memiliki nilai terendah terhadap kolom Rain Tomorrow berdasarkan heatmap diatas.

In [36]:
```
#prepocesing data
df.drop(['MinTemp','Evaporation','WindSpeed9am','WindSpeed3pm','Temp9am'], axis=1, inplace=True)
```

14. Menghapus kolom katagorik yang diduga tidak berpengaruh

Hapus kolom WindGustDir, WindDir9am, WindDir3pm, dan rain tomorrow karena tidak memiliki pengaruh yang signifikan.

```
In [37]: # Mengubah kolom kategorik menjadi kolom numerik
dff = df.copy()
cat = ['WindGustDir','WindDir9am','WindDir3pm','RainTomorrow']
le =  LabelEncoder()
for i in cat:
        dff[i] = le.fit_transform(dff[i])

dff = dff[['WindGustDir','WindDir9am','WindDir3pm','RainTomorrow']]

plt.figure(figsize=(10,5))
sns.heatmap(dff.corr(), annot=True)
plt.xticks(rotation=90)
plt.show()
```



15. **Menghapus kolom dengan nilai koefisien korelasi tinggi satu sama lain**

Pada heatmap sebelumnya diketahui, kolom Temp3pm dan Pressure3pm termasuk kedalam kolom yang memiliki nilai korelasi paling tinggi sehingga harus dihapus.

```
In [38]: df.drop(['Temp3pm','Pressure3pm'], axis=1, inplace=True)
```

```
In [39]: # Drop kolom date, year, day karena tidak dibutuhkan
df.drop(['Date','year','day'], axis=1, inplace=True)
```

In [40]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 123710 entries, 0 to 142192
Data columns (total 17 columns):
 #   Column         Non-Null Count   Dtype
---  ------         --------------   -----
 0   Location       123710 non-null  object
 1   MaxTemp        123710 non-null  float64
 2   Rainfall       123710 non-null  float64
 3   Sunshine       123710 non-null  float64
 4   WindGustDir    123710 non-null  object
 5   WindGustSpeed  123710 non-null  float64
 6   WindDir9am     123710 non-null  object
 7   WindDir3pm     123710 non-null  object
 8   Humidity9am    123710 non-null  float64
 9   Humidity3pm    123710 non-null  float64
 10  Pressure9am    123710 non-null  float64
 11  Cloud9am       123710 non-null  float64
 12  Cloud3pm       123710 non-null  float64
 13  RainToday      123710 non-null  int64
 14  RISK_MM        123710 non-null  float64
 15  RainTomorrow   123710 non-null  int64
 16  month          123710 non-null  int32
dtypes: float64(10), int32(1), int64(2), object(4)
memory usage: 16.5+ MB
```


In [41]:
```
# Reset index setelah dilakukan dropping dengan tujuan agar index berurutan
df.reset_index(drop=True, inplace=True)
```

16. Standarisasi Data

Kemudian, masuk kedalam tahap standarisasi data untuk mengubah skala variabel numerik sehingga memiliki mean nol dan simpangan baku sama dengan satu. Dalam proses standarisasi, setiap nilai dalam variabel numerik dikurangi dengan rerata variabel tersebut, kemudian hasilnya dibagi dengan simpangan baku variabel.

In [42]:
```
columns = ['MaxTemp', 'Rainfall', 'Sunshine', 'WindGustSpeed',
       'Humidity9am', 'Humidity3pm', 'Pressure9am', 'Cloud9am', 'Cloud3pm',
       'month', 'RainToday', 'RainTomorrow','Location']
```

In [43]:
```
data = df[columns]

data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 123710 entries, 0 to 123709
Data columns (total 13 columns):
 #   Column         Non-Null Count   Dtype
---  ------         --------------   -----
 0   MaxTemp        123710 non-null  float64
 1   Rainfall       123710 non-null  float64
 2   Sunshine       123710 non-null  float64
 3   WindGustSpeed  123710 non-null  float64
 4   Humidity9am    123710 non-null  float64
 5   Humidity3pm    123710 non-null  float64
 6   Pressure9am    123710 non-null  float64
 7   Cloud9am       123710 non-null  float64
 8   Cloud3pm       123710 non-null  float64
 9   month          123710 non-null  int32
 10  RainToday      123710 non-null  int64
 11  RainTomorrow   123710 non-null  int64
 12  Location       123710 non-null  object
dtypes: float64(9), int32(1), int64(2), object(1)
memory usage: 11.8+ MB
```

In [44]:
```
scaler = StandardScaler()
scaler.fit(data.drop(['RainToday','RainTomorrow','Location'], axis=1))
scaled_features = scaler.transform(data.drop(['RainToday','RainTomorrow','Location'], axis=1))
df_feat = pd.DataFrame(scaled_features,columns = data.columns[:-3])
df_feat.head()
```

Out[44]:

	MaxTemp	Rainfall	Sunshine	WindGustSpeed	Humidity9am	Humidity3pm	Pressure9am	Cloud9am	Cloud3pm	month
0	-0.080075	-0.208215	1.160037	0.247578	0.174560	-1.371006	-1.349823	1.220705	-0.982267	1.602307
1	0.225132	-0.278572	1.160037	0.247578	-1.234348	-1.228013	-0.939373	1.041376	0.157780	1.602307
2	0.308370	-0.278572	1.160037	0.396959	-1.547438	-0.989691	-1.363976	1.041376	-0.982267	1.602307
3	0.627450	-0.278572	1.160037	-1.246232	-1.182166	-1.656991	0.051368	0.264284	0.157780	1.602307
4	1.223992	-0.161311	1.160037	0.023507	0.748559	-0.846699	-0.911066	0.862047	1.297827	1.602307

In [45]:
```
df_feat['Location'] = data['Location']
df_feat['RainToday'] = data['RainToday']

df_feat.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 123710 entries, 0 to 123709
Data columns (total 12 columns):
 #   Column         Non-Null Count   Dtype
---  ------         --------------   -----
 0   MaxTemp        123710 non-null  float64
 1   Rainfall       123710 non-null  float64
 2   Sunshine       123710 non-null  float64
 3   WindGustSpeed  123710 non-null  float64
 4   Humidity9am    123710 non-null  float64
 5   Humidity3pm    123710 non-null  float64
 6   Pressure9am    123710 non-null  float64
 7   Cloud9am       123710 non-null  float64
 8   Cloud3pm       123710 non-null  float64
 9   month          123710 non-null  float64
 10  Location       123710 non-null  object
 11  RainToday      123710 non-null  int64
dtypes: float64(10), int64(1), object(1)
memory usage: 11.3+ MB
```


17. Encoder untuk kolom 'Location'

Selanjutnya masuk kedalam tahap ini untuk mengubah nilai kategori dalam kolom tersebut menjadi representasi numerik. Hal ini berguna dalam pemrosesan data karena beberapa algoritma atau model hanya dapat menerima input dalam bentuk numerik.

In [46]:
```
df_dummies = pd.get_dummies(df_feat, columns=['Location'])
df_dummies.head()
```

Out[46]:

	MaxTemp	Rainfall	Sunshine	WindGustSpeed	Humidity9am	Humidity3pm	Pressure9am	Cloud9am	Cloud3pm	month	RainToday	Location_Adelaide	L
0	-0.080075	-0.208215	1.160037	0.247578	0.174560	-1.371006	-1.349823	1.220705	-0.982267	1.602307	0	False	
1	0.225132	-0.278572	1.160037	0.247578	-1.234348	-1.228013	-0.939373	1.041376	0.157780	1.602307	0	False	
2	0.308370	-0.278572	1.160037	0.396959	-1.547438	-0.989691	-1.363976	1.041376	-0.982267	1.602307	0	False	
3	0.627450	-0.278572	1.160037	-1.246232	-1.182166	-1.656991	0.051368	0.264284	0.157780	1.602307	0	False	
4	1.223992	-0.161311	1.160037	0.023507	0.748559	-0.846699	-0.911066	0.862047	1.297827	1.602307	0	False	

5 rows × 58 columns

In [47]:
```
df_final = df_dummies.copy()
```

18. Splitting Dataset

Kemudian kami membagi dataset menjadi dua yaitu data training dan data uji dengan tujuan kita dapat memvalidasi kemampuan model dalam melakukan prediksi atau analisis pada data baru yang belum pernah dilihat sebelumnya. Dan kami membagi data training dan data testing dengan rasio 70:30.

In [48]:
```
X = df_final
y = df.RainTomorrow
```

In [49]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=0)
```

In [50]:
```
print("Training set sebesar {0:0.2f}%".format(len(X_train)/len(df.index)*100))
print("Test set sebesar {0:0.2f}%".format(len(X_test)/len(df.index)*100))
```

```
Training set sebesar 70.00%
Test set sebesar 30.00%
```

In [51]:
```
print('Jumlah training set:', len(X_train))
print('Jumlah testing set:', len(X_test))
```

```
Jumlah training set: 86597
Jumlah testing set: 37113
```

In [52]:
```
print("Jumlah hari turunnya hujan: {0} ({1:2.2f}%)".format(len(df.loc[df["RainTomorrow"] == 1]), (len(df.loc[df["RainTomorrow"]==
print("Jumlah hari turunnya hujan : {0} ({1:2.2f}%)\n".format(len(df.loc[df["RainTomorrow"] == 0]), (len(df.loc[df["RainTomorrow"

print("Training True: {0} ({1:2.2f}%)".format(len(y_train[y_train[:] == 1]), (len(y_train[y_train[:] == 1])/len(y_train)*100.0)))
print("Training False: {0} ({1:2.2f}%)\n".format(len(y_train[y_train[:] == 0]), (len(y_train[y_train[:] == 0])/len(y_train)*100.0

print("Testing True: {0} ({1:2.2f}%)".format(len(y_test[y_test[:] == 1]), (len(y_test[y_test[:] == 1])/len(y_test)*100.0)))
print("Testing False: {0} ({1:2.2f}%)".format(len(y_test[y_test[:] == 0]), (len(y_test[y_test[:] == 0])/len(y_test)*100.0)))
```

```
Jumlah hari turunnya hujan: 27392 (22.14%)
Jumlah hari turunnya hujan : 96318 (77.86%)

Training True: 19095 (22.05%)
Training False: 67502 (77.95%)

Testing True: 8297 (22.36%)
Testing False: 28816 (77.64%)
```

19. Pendefinisian dan Pelatihan Model Support Vector Machine

Pendefinisian dan Pelatihan Model SVM

In [53]:
```
# Pendefisian Model SVM
svm = svm.SVC(class_weight='balanced', kernel='linear')
```

In [54]:
```
# Pendefisian Model SVM
svm.fit(X_train, y_train)
```

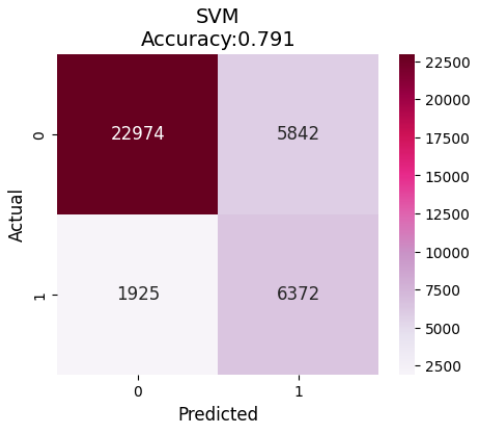
Out[54]:
```
SVC(class_weight='balanced', kernel='linear')
```
In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [55]:
```
y_pred_svm = svm.predict(X_test)
```

20. Evaluasi Model Support Vector Machine

In [56]:
```
# Confusion Matrix
cm = confusion_matrix(y_test, y_pred_svm)
```

In [57]:
```
plt.figure(figsize=(5,4))
sns.heatmap(cm, annot=True, fmt=".0f", cmap="PuRd", annot_kws={"fontsize":12})
plt.title("SVM\nAccuracy:{:.3f}".format(accuracy_score(y_test, y_pred_svm)), fontsize=14)
plt.xlabel("Predicted", fontsize=12)
plt.ylabel("Actual", fontsize=12)
plt.show()

print("Classification Report:\n", classification_report(y_test, y_pred_svm))
print(f'Accuracy Score: {accuracy_score(y_test, y_pred_svm)}')
print(f'Precision Score: {precision_score(y_test, y_pred_svm)}')
print(f'Recall Score: {recall_score(y_test, y_pred_svm)}')
print("F1 Score:", f1_score(y_test, y_pred_svm))
```




```
Classification Report:
              precision    recall  f1-score   support

           0       0.92      0.80      0.86     28816
           1       0.52      0.77      0.62      8297

    accuracy                           0.79     37113
   macro avg       0.72      0.78      0.74     37113
weighted avg       0.83      0.79      0.80     37113

Accuracy Score: 0.790720232802522
Precision Score: 0.5216964139512036
Recall Score: 0.7679884295528504
F1 Score: 0.6213251426064064
```

Dari model diatas, dapat kita ketahui bahwa nilai akurasinya adalah 0.79, precision score sebesar 0.52, recall score sebesar 0.77, dan F1 Score sebesar 0.62.

21. Feature Importance

Selanjutnya kami akan mengevaluasi kontribusi relatif dari setiap fitur atau variabel dalam mempengaruhi hasil prediksi model. Metode ini membantu kami dalam memahami mana fitur yang paling penting atau informatif dalam memprediksi target.

In [58]:
```
print('Nilai Intersep model:', svm.intercept_)
print('')
print('Nilai koefisien model regresi logistik:', svm.coef_)
```

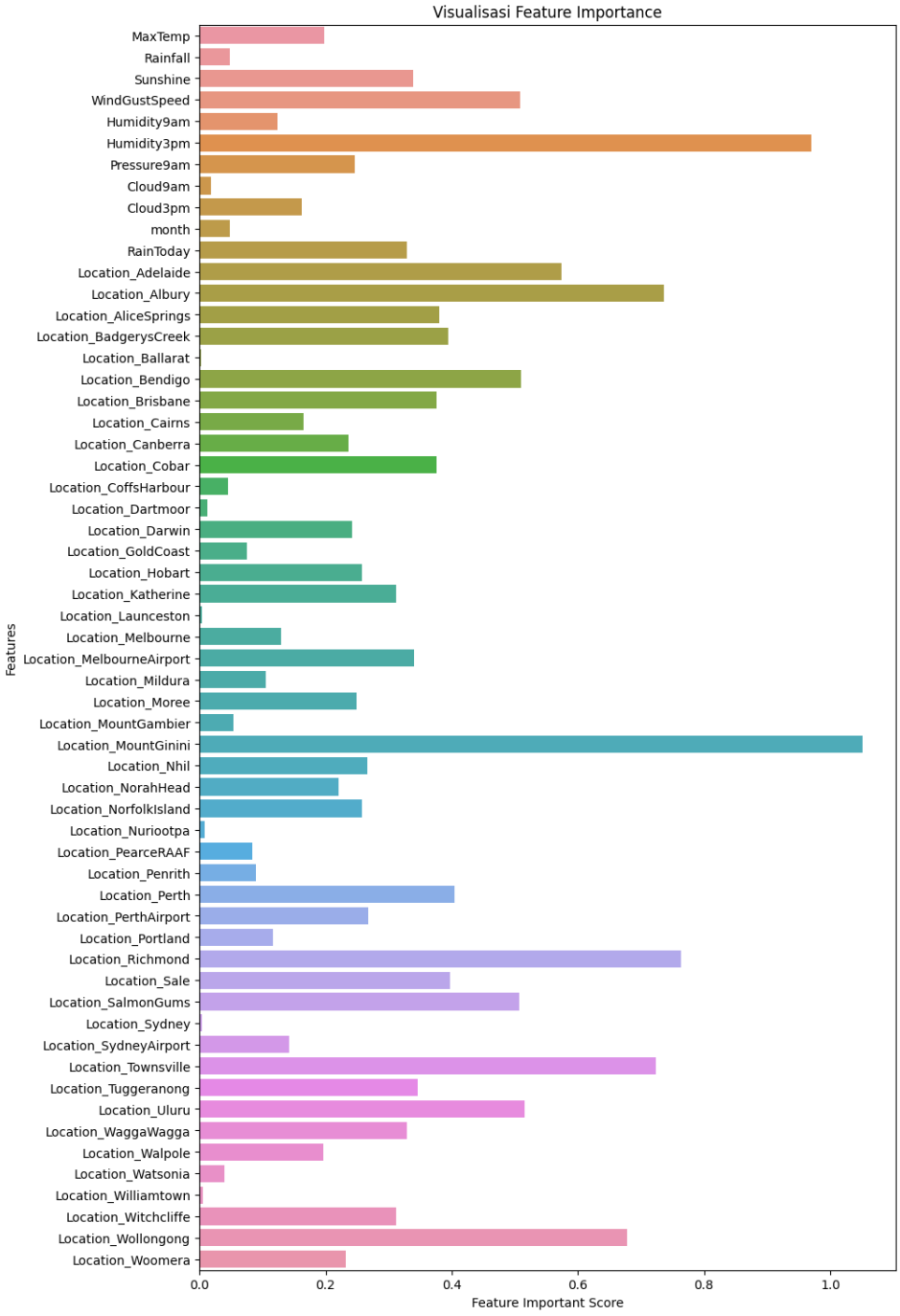
```
Nilai Intersep model: [-0.5517952]

Nilai koefisien model regresi logistik: [[ 0.1977227   0.04844937 -0.33833741  0.50909491  0.1242907   0.97034495
  -0.24685509 -0.01812964  0.16260721  0.04861771  0.32922564  0.57469161
   0.73638391  0.38008683  0.39495746 -0.00271746  0.51055681  0.37555043
  -0.16507821 -0.2363136   0.37577163 -0.04557676  0.01315644 -0.2425434
  -0.07552443 -0.25719234  0.31221598 -0.00461668 -0.12905305 -0.34014874
   0.10465599  0.24927279 -0.05366239 -1.05173613 -0.26587817  0.22017445
  -0.25749254  0.00821475  0.08317108  0.09006803  0.4047821   0.26801621
  -0.11723736 -0.76364802 -0.39709536  0.50709403  0.00384309 -0.14282265
  -0.7231139  -0.34624359  0.51593605  0.32883245 -0.19643306 -0.03972029
  -0.0049849   0.3113394  -0.67813191 -0.23180658]]
```

In [59]:
```
# Memeriksa feature importance

feature_importance=pd.DataFrame({'feature':list(df_final.columns),'feature_importance':[abs(i) for i in svm.coef_[0]]})
feature_importance
```

Out[59]:

	feature	feature_importance
0	MaxTemp	0.197723
1	Rainfall	0.048449
2	Sunshine	0.338337
3	WindGustSpeed	0.509095
4	Humidity9am	0.124291
5	Humidity3pm	0.970345
6	Pressure9am	0.246855
7	Cloud9am	0.018130
8	Cloud3pm	0.162607
9	month	0.048618
10	RainToday	0.329226
11	Location_Adelaide	0.574692
12	Location_Albury	0.736384
13	Location_AliceSprings	0.380087
14	Location_BadgerysCreek	0.394957
15	Location_Ballarat	0.002717
16	Location_Bendigo	0.510557
17	Location_Brisbane	0.375550
18	Location_Cairns	0.165078
19	Location_Canberra	0.236314
20	Location_Cobar	0.375772
21	Location_CoffsHarbour	0.045577
22	Location_Dartmoor	0.013156
23	Location_Darwin	0.242543
24	Location_GoldCoast	0.075524
25	Location_Hobart	0.257192
26	Location_Katherine	0.312216
27	Location_Launceston	0.004617
28	Location_Melbourne	0.129053
29	Location_MelbourneAirport	0.340149
30	Location_Mildura	0.104656
31	Location_Moree	0.249273
32	Location_MountGambier	0.053662
33	Location_MountGinini	1.051736
34	Location_Nhil	0.265878
35	Location_NorahHead	0.220174
36	Location_NorfolkIsland	0.257493
37	Location_Nuriootpa	0.008215
38	Location_PearceRAAF	0.083171
39	Location_Penrith	0.090068
40	Location_Perth	0.404782
41	Location_PerthAirport	0.268016
42	Location_Portland	0.117237
43	Location_Richmond	0.763648
44	Location_Sale	0.397095
45	Location_SalmonGums	0.507094
46	Location_Sydney	0.003843
47	Location_SydneyAirport	0.142823
48	Location_Townsville	0.723114
49	Location_Tuggeranong	0.346244
50	Location_Uluru	0.515936
51	Location_WaggaWagga	0.328832
52	Location_Walpole	0.196433
53	Location_Watsonia	0.039720
54	Location_Williamtown	0.004985
55	Location_Witchcliffe	0.311339
56	Location_Wollongong	0.678132
57	Location_Woomera	0.231807

In [60]:
```
plt.figure(figsize=(10, 18))
sns.barplot(x=feature_importance['feature_importance'], y=feature_importance['feature'])
plt.xlabel("Feature Important Score")
plt.ylabel("Features")
plt.title("Visualisasi Feature Importance")

plt.show()
```



Berdasarkan grafik diatas, kolom Cloud9am memiliki nilai feature importance, sehingga akan dihapus dan dibuat model baru.

22. Membangun Model Support Vector Machine Baru

In [61]:
```
data2 = data.copy()
data2.drop('Cloud9am', axis=1, inplace=True)
feature_names = list(data2.columns.values)

data2.head()
```

Out[61]:

	MaxTemp	Rainfall	Sunshine	WindGustSpeed	Humidity9am	Humidity3pm	Pressure9am	Cloud3pm	month	RainToday	RainTomorrow	Location
0	22.9	0.6	12.3	44.0	71.0	22.0	1007.7	2.0	12	0	0	Albury
1	25.1	0.0	12.3	44.0	44.0	25.0	1010.6	5.0	12	0	0	Albury
2	25.7	0.0	12.3	46.0	38.0	30.0	1007.6	2.0	12	0	0	Albury
3	28.0	0.0	12.3	24.0	45.0	16.0	1017.6	5.0	12	0	0	Albury
4	32.3	1.0	12.3	41.0	82.0	33.0	1010.8	8.0	12	0	0	Albury

In [62]:
```
scaler_ = StandardScaler()
scaler_.fit(data2.drop(['RainToday','RainTomorrow','Location'], axis=1))
scaled_features_ = scaler_.transform(data2.drop(['RainToday','RainTomorrow','Location'], axis=1))
df_feat_ = pd.DataFrame(scaled_features_,columns = data2.columns[:-3])
df_feat_.head()
```

Out[62]:

	MaxTemp	Rainfall	Sunshine	WindGustSpeed	Humidity9am	Humidity3pm	Pressure9am	Cloud3pm	month
0	-0.080075	-0.208215	1.160037	0.247578	0.174560	-1.371006	-1.349823	-0.982267	1.602307
1	0.225132	-0.278572	1.160037	0.247578	-1.234348	-1.228013	-0.939373	0.157780	1.602307
2	0.308370	-0.278572	1.160037	0.396959	-1.547438	-0.989691	-1.363976	-0.982267	1.602307
3	0.627450	-0.278572	1.160037	-1.246232	-1.182166	-1.656991	0.051368	0.157780	1.602307
4	1.223992	-0.161311	1.160037	0.023507	0.748559	-0.846699	-0.911066	1.297827	1.602307

In [63]:
```
df_feat_['Location'] = data2['Location']
df_feat_['RainToday'] = data2['RainToday']

df_feat_.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 123710 entries, 0 to 123709
Data columns (total 11 columns):
 #   Column         Non-Null Count   Dtype
---  ------         --------------   -----
 0   MaxTemp        123710 non-null  float64
 1   Rainfall       123710 non-null  float64
 2   Sunshine       123710 non-null  float64
 3   WindGustSpeed  123710 non-null  float64
 4   Humidity9am    123710 non-null  float64
 5   Humidity3pm    123710 non-null  float64
 6   Pressure9am    123710 non-null  float64
 7   Cloud3pm       123710 non-null  float64
 8   month          123710 non-null  float64
 9   Location       123710 non-null  object
 10  RainToday      123710 non-null  int64
dtypes: float64(9), int64(1), object(1)
memory usage: 10.4+ MB
```


In [64]:
```
df_dummies_ = pd.get_dummies(df_feat_, columns=['Location'])
df_dummies_.head()
```

Out[64]:

	MaxTemp	Rainfall	Sunshine	WindGustSpeed	Humidity9am	Humidity3pm	Pressure9am	Cloud3pm	month	RainToday	Location_Adelaide	Location_Alb
0	-0.080075	-0.208215	1.160037	0.247578	0.174560	-1.371006	-1.349823	-0.982267	1.602307	0	False	T
1	0.225132	-0.278572	1.160037	0.247578	-1.234348	-1.228013	-0.939373	0.157780	1.602307	0	False	T
2	0.308370	-0.278572	1.160037	0.396959	-1.547438	-0.989691	-1.363976	-0.982267	1.602307	0	False	T
3	0.627450	-0.278572	1.160037	-1.246232	-1.182166	-1.656991	0.051368	0.157780	1.602307	0	False	T
4	1.223992	-0.161311	1.160037	0.023507	0.748559	-0.846699	-0.911066	1.297827	1.602307	0	False	T

5 rows × 57 columns

23. Model Inference

Model inference adalah cara untuk berinteraksi dengan model pembelajaran mesin atau model prediktif lainnya, yang digunakan untuk memberikan input ke model, melakukan prediksi, dan mendapatkan output dari model, yang dilakukan untuk menguji model dengan data diluar dataset.

In [65]:
```
MaxTemp = float(input("Max Temperature:"))
Rainfall = float(input("Rainfall (in mm):"))
Sunshine = float(input("Sunshine hour:"))
WindGustSpeed = float(input("Wind Speed:"))
Humidity9am = float(input("Humidity percentation at 9 AM:"))
Humidity3pm = float(input("Humidity percentation at 3 PM:"))
Pressure9am = float(input("Pressure at 9 AM:"))
Cloud3pm = float(input("Cloud persentation at 3 PM:"))
Cloud9am = float(input("Cloud persentation at 9 AM :"))
month = int(input('Month:'))
RainToday = int(input('Is today raining? Please type 1 for YES and 0 for NO '))
Location = input('Your Location:')

val = [MaxTemp, Rainfall, Sunshine, WindGustSpeed, Humidity9am , Humidity3pm, Pressure9am,Cloud9am,Cloud3pm, month]
val = scaler.transform([val])
val = val.reshape(10,)
```

```
if RainToday == 1:
    val = np.append(val, 1)
elif RainToday == 0:
    val = np.append(val, 0)
else:
    print('ERROR!')


locations = {"Adelaide":0,"Albury":1,"AliceSprings":2,
             "Badgerys Creek":3,"Ballarat":4,"Bendigo":5,
             "Brisbane":6,"Cairns":7,"Canberra":8,
             "Cobar":9,"Coffs Harbour":10,"Dartmoor":11,
             "Darwin":12,"Gold Coast":13,"Hobart":14,
             "Katherine":15,"Launceston":16,"Melbourne":17,
             "Melbourne Airport":18,"Mildura":19,"Moree":20,
             "Mount Gambier":21,"Mount Ginini":22,"Nhil":23,
             "Norah Head":24,"Norfolk Island":25,"Nuriootpa":26,
             "Pearce RAAF":27,"Penrith":28,"Perth":29,
             "Perth Airport":30,"Portland":31,"Richmond":32,
             "Sale":33, "Salmon Gums":34, "Sydney":35,
             "Sydney Airport":36, "Townsville":37, "Tuggeranong":38,
             "Uluru":39, "Wagga Wagga":40, "Walpole":41,
             "Watsonia":42, "Williamtown":43, "Witchcliffe":44,
             "Wollongong":45, "Woomera":46}
```



```
for i in range(0,47):
    if locations[Location]==i:
        val = np.append(val, 1)
    else:
        val = np.append(val, 0)

print(val)

val_predict = svm.predict([val])

if val_predict == 1:
    print('Prediksi bernilai 1, maka akan diprediksi besok turun hujan')
elif val_predict == 0:
    print('Prediksi bernilai 0, maka akan diprediksi besok TIDAK turun hujan')
else:
    print('Prediksi tidak valid')
```

```
Max Temperature:25
Rainfall (in mm):0.5
Sunshine hour:12
Wind Speed:60
Humidity percentation at 9 AM:50
Humidity percentation at 3 PM:45
Pressure at 9 AM:100
Cloud persentation at 3 PM:7
Cloud persentation at 9 AM :5
Month:9
Is today raining? Please type 1 for YES and 0 for NO 1
Your Location:Sydney
[   0.21125888   -0.21994134    1.08045658    1.4426263    -0.92125708
   -0.27472731 -129.82059778    0.1447319     0.91781161    0.73902232
    1.           0.           0.           0.           0.
    0.           0.           0.           0.           0.
    0.           0.           0.           0.           0.
    0.           0.           0.           0.           0.
    0.           0.           0.           0.           0.
    0.           0.           0.           0.           0.
    0.           0.           0.           0.           0.
    0.           1.           0.           0.           0.
    0.           0.           0.           0.           0.
    0.           0.           0.       ]
Prediksi bernilai 1, maka akan diprediksi besok turun hujan
```


V. KESIMPULAN

Berdasarkan hasil analisis secara keseluruhan di atas, dapat disimpulkan bahwa:

- Project ini berisi training data, yaitu data training dan data uji, kemudian diklasifikasi untu memprediksi “apakah besok akan hujan?” berdasarkan model yang dibuat menggunakan metode Support Vector Machine. Dataset yang digunakan dalam projek ini berisi kumpulan data pengamatan cuaca 2008 – 2017 di berbagai lokasi seluruh Australia. Jadi tujuan proyek ini berdasarkan pengamatan atau dokumen tentang kondisi cuaca hari itu, apakah keesokan harinya akan hujan (Ya) atau tidak (Tidak).
- Diketahui bahwa kolom atau fitur yang paling mempengaruhi kolom target adalah "Humidity3pm". Artinya persentase kelembapan pada pukul 15.00 memiliki pengaruh yang paling signifikan terhadap penentuan curah hujan keesokan harinya. Semakin tinggi kelembaban, semakin besar kemungkinan hujan.
- Lokasi Adelaide merupakan tempat dengan intensitas curah hujan tertinggi.
- Juni-Juli adalah bulan dengan curah hujan paling tinggi. Sedangkan Januari-Februari memiliki intensitas curah hujan yang paling rendah.
- Algoritma model klasifikasi yang digunakan adalah support vector machine (SVM). Berdasarkan pelatihan model yang telah dilakukan, algoritma yang dipilih adalah LogisticRegression dan nilai akurasi akhir 79%, nilai akurasi 0,52, recall 0,77 dan skor f1 0,62.

Save Model

In [60]:
```
pickle_file_dir = open("logreg.pkl", "wb")

pickle.dump(svm, pickle_file_dir)
pickle_file_dir.close()
```