analysis

April 8, 2025

1 Machine Learning Spring 2025

Project 1 - Temperature Prediction Team: ST_ML2025_2

[49]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import re
import math
```

[2]:
```
# Load the given datasets(located in the input folder)
train_df = pd.read_csv("./input/train_dataset.csv")
test_df = pd.read_csv("./input/test_dataset.csv")
station_info_df = pd.read_csv("./input/station_info.csv")
sample_submission_df = pd.read_csv("./input/submission_sample.csv")
```

2 Dataset

.

- train_dataset.csv: , , , , , 2019-2024
- test_dataset.csv: ,
- station_info.csv:

[3]:
```
train_df.sample(10)
```

[3]:

	id	station	station_name	date	cloud_cover_0	cloud_cover_1	\
5679	7867	112		07-25	4.0	5.0	
9418	13797	202		10-26	0.0	0.0	
12018	16397	203		12-13	0.0	0.0	
13115	17494	203		12-14	0.0	0.0	
11567	15946	203		09-18	10.0	9.0	
7106	11485	201		06-22	0.0	0.0	
3422	5610	108		05-20	4.0	8.0	
6221	8409	112		01-18	9.0	10.0	
3047	5235	108		05-10	0.0	0.0	
10092	14471	202		08-31	5.0	3.0	

```
       cloud_cover_10  cloud_cover_11  cloud_cover_12  cloud_cover_13  …  \
5679              1.0             0.0             1.0             0.0  …
9418              0.0             0.0             0.0             0.0  …
12018             0.0             0.0             0.0             0.0  …
13115             0.0             0.0             0.0             0.0  …
11567             1.0             0.0             0.0             4.0  …
7106              0.0             0.0             0.0             0.0  …
3422              0.0             0.0             0.0             0.0  …
6221              0.0             0.0             0.0             0.0  …
3047              6.0             6.0             6.0             5.0  …
10092             2.0             2.0             5.0             3.0  …

       wind_speed_23  wind_speed_3  wind_speed_4  wind_speed_5  wind_speed_6  \
5679             1.9           1.9           2.2           1.8           1.9
9418             0.4           0.3           0.2           0.0           0.3
12018            0.4           0.8           1.0           1.0           0.1
13115            0.4           0.7           0.4           0.8           1.4
11567            1.3           1.5           1.3           1.2           0.9
7106             0.1           0.5           0.5           0.1           0.0
3422             2.4           1.7           1.2           0.5           0.2
6221             3.9           3.6           3.5           2.9           3.6
3047             2.5           1.0           1.4           1.8           1.0
10092            0.4           0.2           0.4           0.4           0.0

       wind_speed_7  wind_speed_8  wind_speed_9  climatology_temp    target
5679            2.7           2.5           2.0         26.991071  0.508929
9418            0.3           0.1           0.7         11.923214 -1.023214
12018           1.2           1.5           1.5         -1.592857  2.592857
13115           1.5           0.9           0.8         -0.551786  1.651786
11567           0.7           0.2           0.6         22.046429 -3.346429
7106            0.0           1.5           1.0         23.437500 -0.137500
3422            0.8           1.4           1.5         17.262500  2.837500
6221            3.3           3.2           4.3         -1.078571  6.678571
3047            3.0           1.8           0.3         17.025000  1.475000
10092           0.5           0.2           0.2         23.453571 -1.753571

[10 rows x 342 columns]
```

[4]:
```
train_df.describe()
```

[4]:
```
                id       station  cloud_cover_0  cloud_cover_1  \
count  13132.000000  13132.000000   12945.000000   12920.000000
mean    9484.110493    153.980658       2.915798       3.022291
std     5311.954253     48.183220       3.646779       3.652165
min        0.000000     98.000000       0.000000       0.000000
25%     5470.750000    108.000000       0.000000       0.000000
```

```
50%     8753.500000    112.000000       0.000000       1.000000
75%    14227.250000    202.000000       6.000000       6.000000
max    17510.000000    203.000000      10.000000      10.000000

       cloud_cover_10  cloud_cover_11  cloud_cover_12  cloud_cover_13  \
count    12916.000000    12926.000000    12931.000000    12926.000000
mean         3.126742        3.092063        3.046400        3.035046
std          3.659422        3.589739        3.512063        3.483002
min          0.000000        0.000000        0.000000        0.000000
25%          0.000000        0.000000        0.000000        0.000000
50%          1.000000        1.000000        1.000000        1.000000
75%          7.000000        7.000000        6.000000        6.000000
max         10.000000       10.000000       10.000000       10.000000

       cloud_cover_14  cloud_cover_15  …  wind_speed_23  wind_speed_3  \
count    12919.000000    12938.000000  …   13122.000000  13120.000000
mean         2.992182        2.911115  …      -0.032754     -2.491814
std          3.462070        3.440298  …     123.464434    195.197628
min          0.000000        0.000000  …   -9999.000000  -9999.000000
25%          0.000000        0.000000  …       0.600000      0.400000
50%          1.000000        1.000000  …       1.200000      1.000000
75%          6.000000        6.000000  …       2.100000      1.900000
max         10.000000       10.000000  …      10.700000     11.300000

       wind_speed_4  wind_speed_5  wind_speed_6  wind_speed_7  wind_speed_8  \
count  13116.000000  13112.000000  13119.000000  13120.000000  13125.000000
mean      -0.234637     -1.011882     -4.068389     -7.119787    -10.073272
std      123.489868    151.258218    230.949420    289.453706    337.895538
min    -9999.000000  -9999.000000  -9999.000000  -9999.000000  -9999.000000
25%        0.400000      0.400000      0.400000      0.400000      0.500000
50%        0.900000      0.900000      0.900000      0.900000      1.000000
75%        1.900000      1.800000      1.800000      1.800000      2.000000
max        9.500000     10.600000     12.400000     11.100000     11.300000

       wind_speed_9  climatology_temp        target
count  13127.000000      13132.000000  13132.000000
mean      -9.119029         12.658557      0.221979
std      326.432094         10.023504      2.960544
min    -9999.000000         -4.487500    -12.864286
25%        0.700000          3.292857     -1.643052
50%        1.300000         12.842857      0.157143
75%        2.200000         22.271429      2.045536
max        9.400000         28.455357     11.778571

[8 rows x 340 columns]
```

[5]:
```
train_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 13132 entries, 0 to 13131
Columns: 342 entries, id to target
dtypes: float64(338), int64(2), object(2)
memory usage: 34.3+ MB
```

2.1 Feature

2.1.1 Feature

[6]:
```
def search_time_based_feature_names(df):
    "       feature (e.g. 'cloud_cover_0', 'cloud_cover_1' ...)          (e.g.␣
 ↪'cloud_cover')    ."

    time_based_features_pattern = re.compile(r"^(.*)_(\d{1,2})$")
    searched_time_based_feature_names = set()

    for column in df.columns:
        match = time_based_features_pattern.match(column)
        if match:
            feature_name, hour = match.groups()
            hour = int(hour)
            searched_time_based_feature_names.add(feature_name)

    return searched_time_based_feature_names

time_based_feature_names = search_time_based_feature_names(train_df)

print(len(time_based_feature_names))
print(time_based_feature_names)
```

```
14
{'wind_direction', 'humidity', 'vapor_pressure', 'sea_level_pressure',
'snow_depth', 'cloud_cover', 'precipitation', 'sunshine_duration', 'dew_point',
'surface_temp', 'min_cloud_height', 'wind_speed', 'visibility',
'local_pressure'}
```

train dataset feature .

1. Data
   - `id`: (identical)
   - `station`: (98: “ ”, 201: “ ”)
   - `station_name`: (“ ”, “ ”, “ ”)
   - `date`: (- , 1 29 “01-29”)
2. feature
   - `cloud_cover_[0-23]`: (10 , 0~10)
   - `min_cloud_height_[0-23]`: (100m)
3. feature
   - `dew_point_[0-23]`: (°C)
   - `surface_temp_[0-23]`: (°C)

- `climatology_temp`: (°C) (7)

4. feature
   - `humidity_[0-23]`: (%)
   - `vapor_pressure_[0-23]`: (hPa)
   - `percipitation_[0-23]`: (mm)
   - `snow_depth_[0-23]`: (cm)
5. feature
   - `local_pressure_[0-23]`: (hPa)
   - `sea_level_pressure_[0-23]`: (hPa)
6. feature
   - `visibility_[0-23]`: (10m)
   - `sunshine_duration_[0-23]`: (hr)
7. feature
   - `wind_speed_[0-23]`: (m/s)
   - `wind_direction_[0-23]`: ( )
8. target feature
   - `target`: (°C, (°C) `climatology_temp`)

2.1.2

1. `-9999`:
2. `NaN`:
   - `sunshine_duration`: (0, 1, 2, 3, 4, 5, 22, 23),
   - `snow_depth`:
   - `precipitation`:

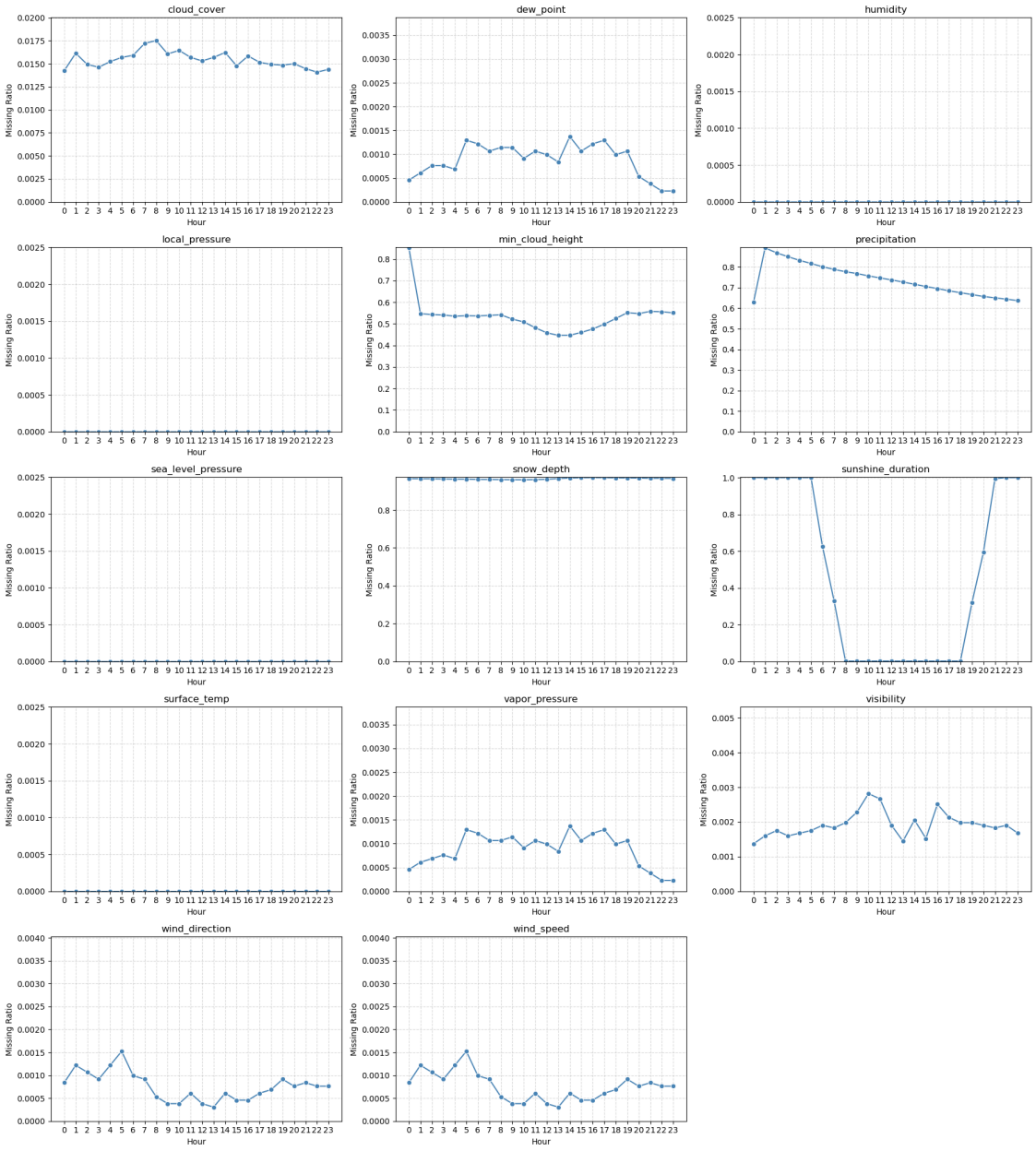
2.1.3

Time-based feature

[68]:
```
def analyze_time_based_features_missing_by_hour(ax, df, base_name):
    hour_missing = {}

    cols = [col for col in df.columns if col.startswith(base_name + "_")]

    for col in cols:
        try:
            hour = int(col.split("_")[-1])
            missing_ratio = df[col].isnull().mean()
            hour_missing[hour] = missing_ratio
        except ValueError:
            continue

    hours = sorted(hour_missing.keys())
    missing_values = [hour_missing[h] for h in hours]

    sns.lineplot(x=hours, y=missing_values, marker="o", ax=ax,␣
↪color="steelblue")
```

```
    ax.set_title(f"{base_name}")
    ax.set_xlabel("Hour")
    ax.set_ylabel("Missing Ratio")
    ax.set_xticks(hours)
    ax.set_ylim(0, max(missing_values) + 0.0025)
    ax.grid(True, linestyle="--", alpha=0.5)
```

[69]:
```
num_features = len(time_based_feature_names)
n_cols = 3
n_rows = math.ceil(num_features / n_cols)

fig, axes = plt.subplots(n_rows, n_cols, figsize=(n_cols * 6, n_rows * 4))
axes = axes.flatten()

for i, feature in enumerate(sorted(time_based_feature_names)):
    analyze_time_based_features_missing_by_hour(axes[i], train_df, feature)

for j in range(i + 1, len(axes)):
    fig.delaxes(axes[j])

plt.tight_layout()
plt.show()
```

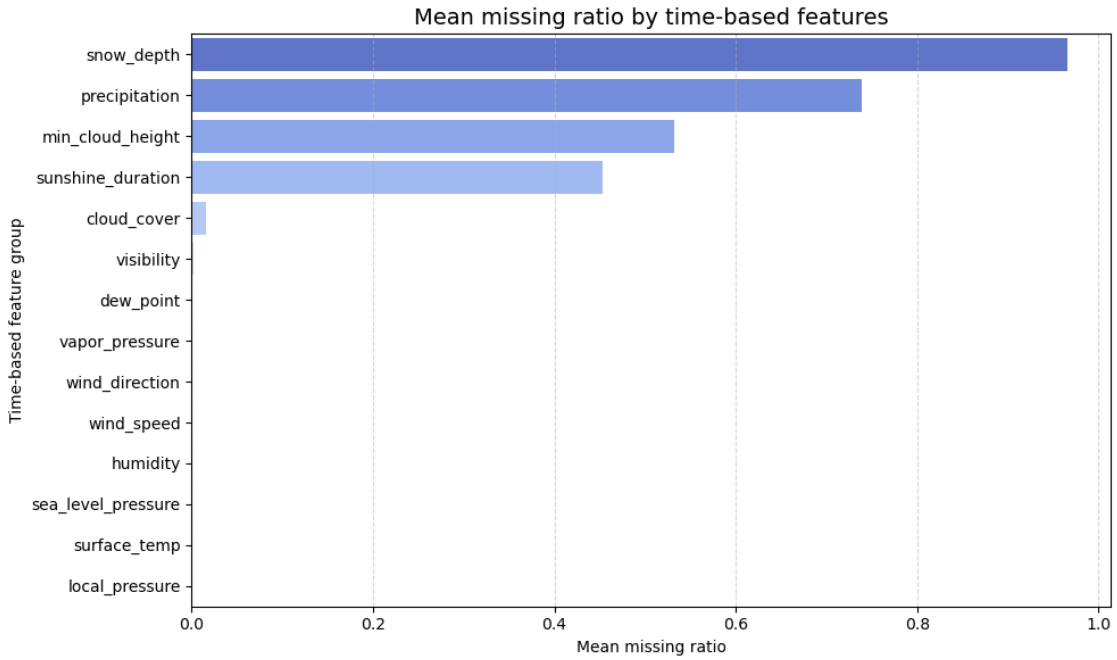


Time-based feature

[56]:
```
def analyze_missing_values_by_time_groups(df, time_based_feature_names):
    group_missing_summary = []

    for base_name in time_based_feature_names:
        cols = [col for col in df.columns if col.startswith(base_name + "_")]
        if not cols:
            continue
        missing_ratio = df[cols].isnull().mean().mean()  #
```

```
        group_missing_summary.append((base_name, missing_ratio))

    summary_df = pd.DataFrame(group_missing_summary, columns=["feature_group",␣
 ↪"mean_missing_ratio"])
    summary_df = summary_df.sort_values(by="mean_missing_ratio",␣
 ↪ascending=False).reset_index(drop=True)
    return summary_df
```

[57]:
```
group_missing_ratio = analyze_missing_values_by_time_groups(train_df,␣
 ↪time_based_feature_names)

plt.figure(figsize=(10, 6))
sns.barplot(
    data=group_missing_ratio,
    x="mean_missing_ratio",
    y="feature_group",
    hue="feature_group",
    palette="coolwarm",
)
plt.title("Mean missing ratio by time-based features", fontsize=14)
plt.xlabel("Mean missing ratio")
plt.ylabel("Time-based feature group")
plt.grid(axis='x', linestyle='--', alpha=0.5)
plt.tight_layout()
plt.show()
```



[ ]:

2.2 Feature Engineering

2.2.1

[ ]: