

**Praktikum 9**  
**Anomaly Detection**  
**“Kegagalan Sistem Temperatur Mesin”**



Disusun Oleh:  
Muhammad Risalah Naufal (21051214008)  
S1 Sistem Informasi B  
  
Mata Kuliah: Data Mining  
Dosen Pengampu: Wiyli Yustanti, S.Si., M.Kom.

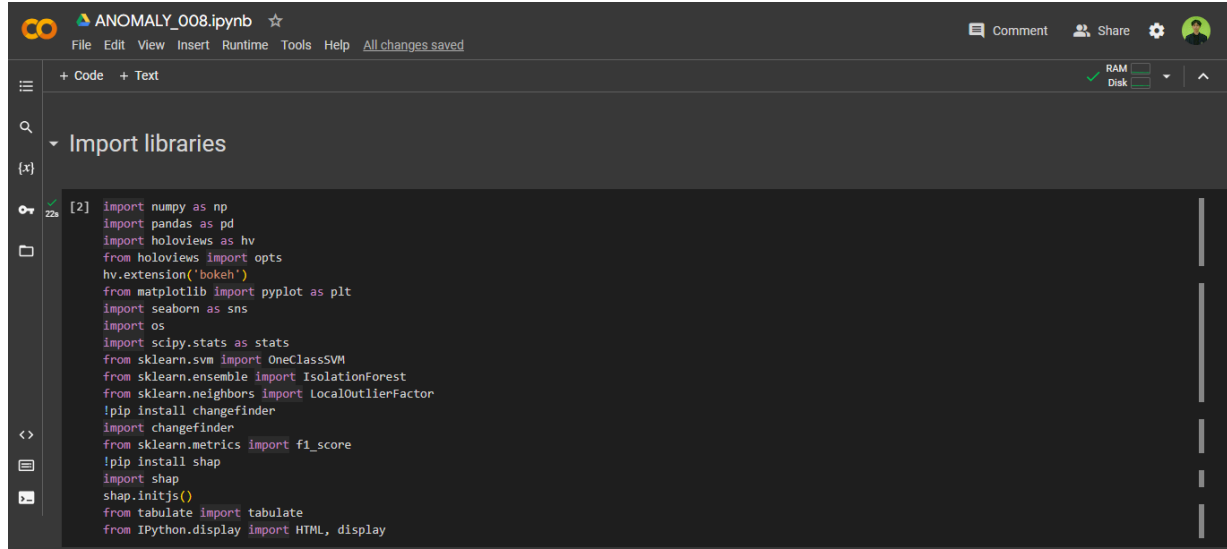
**PROGRAM STUDI SISTEM INFORMASI**  
**RUMPUN TEKNIK INFORMATIKA**  
**FAKULTAS TEKNIK**  
**UNIVERSITAS NEGERI SURABAYA 2023**

## A. Dataset

Dataset ini berisikan tentang data catatan kerja sistem temperatur mesin pada suatu pabrik. Segala rekapitulasi mengenai penggunaan telah tercatat pada dataset ini dan nantinya akan dilacak apakah terdapat anomali terhadap kinerja sistem.

## B. Metode Penelitian

### 1. Import Libraries



```
[2] import numpy as np
import pandas as pd
import holoviews as hv
from holoviews import opts
hv.extension('bokeh')
from matplotlib import pyplot as plt
import seaborn as sns
import os
import scipy.stats as stats
from sklearn.svm import OneClassSVM
from sklearn.ensemble import IsolationForest
from sklearn.neighbors import LocalOutlierFactor
!pip install changefinder
import changefinder
from sklearn.metrics import f1_score
!pip install shap
import shap
shap.initjs()
from tabulate import tabulate
from IPython.display import HTML, display
```

- **Import Library:** Kode ini mengimpor beberapa pustaka seperti NumPy, Pandas, Holoviews, Matplotlib, Seaborn, OS, SciPy, Scikit-learn, Changefinder, SHAP, Tabulate, dan IPython untuk analisis data dan visualisasi.
- **Extension Holoviews:** Holoviews diaktifkan dengan ekstensi Bokeh untuk mendukung pembuatan visualisasi interaktif.
- **Import Metode Deteksi Anomali:** Kode ini mengimpor beberapa metode deteksi anomali seperti One-Class SVM, Isolation Forest, Local Outlier Factor, dan ChangeFinder.
- **Install Library:** Menginstall Changefinder dan SHAP menggunakan pip.
- **Inisialisasi SHAP:** SHAP (SHapley Additive exPlanations) diinisialisasi untuk mendukung interpretasi model machine learning.
- **Import Tabulate dan IPython Display:** Tabulate digunakan untuk membuat tabel, dan IPython Display digunakan untuk menampilkan HTML dan tata letak data.

Dengan kata lain, kode ini menyiapkan lingkungan untuk melakukan analisis anomali pada data menggunakan beberapa metode machine learning, dengan dukungan visualisasi interaktif dan interpretasi model menggunakan SHAP.

### 2. Load Dataset



```
Load the dataset

[3] for dirname, _, filenames in os.walk('/kaggle/input/nab/realKnownCause/realKnownCause'):
    for filename in filenames:
        print(os.path.join(dirname, filename))

[5] df = pd.read_csv("machine_temperature_system_failure.csv", low_memory=False)
print(f'machine_temperature_system_failure.csv : {df.shape}')
df.head(3)
```

|   | timestamp           | value     |
|---|---------------------|-----------|
| 0 | 2013-12-02 21:15:00 | 73.967322 |
| 1 | 2013-12-02 21:20:00 | 74.935882 |
| 2 | 2013-12-02 21:25:00 | 76.124162 |

- **pd.read\_csv("machine\_temperature\_system\_failure.csv", low\_memory=False):** Membaca file CSV ("machine\_temperature\_system\_failure.csv") ke dalam sebuah data frame menggunakan Pandas. Parameter `low_memory=False` digunakan untuk mencegah Pandas memberikan peringatan terkait penggunaan memori rendah.
- **print(f'machine\_temperature\_system\_failure.csv : {df.shape}')**: Mencetak ukuran data frame ke layar, memberikan informasi tentang jumlah baris dan kolom dalam data.
- **df.head(3):** Menampilkan tiga baris pertama dari data frame, memberikan gambaran awal tentang struktur dan konten data.

Kode di atas membaca file CSV yang disebut "machine\_temperature\_system\_failure.csv" menggunakan Pandas, kemudian mencetak ukuran data frame (shape) dan menampilkan tiga baris pertama dari data frame tersebut.

### 3. Pre-Processing Anomaly Points

```

Pre-processing

Anomaly Points

[6] anomaly_points = [
    ["2013-12-10 06:25:00.000000", "2013-12-12 05:35:00.000000"],
    ["2013-12-15 17:50:00.000000", "2013-12-17 17:00:00.000000"],
    ["2014-01-27 14:20:00.000000", "2014-01-29 13:30:00.000000"],
    ["2014-02-07 14:55:00.000000", "2014-02-09 14:05:00.000000"]
]

[7] df['timestamp'] = pd.to_datetime(df['timestamp'])
    #is anomaly? : True => 1, False => 0
    df['anomaly'] = 0
    for start, end in anomaly_points:
        df.loc[(df['timestamp'] >= start) & (df['timestamp'] <= end), 'anomaly'] = 1

```

- Data anomali dapat diambil dari: [https://github.com/numenta/NAB/blob/master/labels/combined\\_windows.json](https://github.com/numenta/NAB/blob/master/labels/combined_windows.json)
- **df['timestamp'] = pd.to\_datetime(df['timestamp']):** Mengonversi kolom 'timestamp' dalam DataFrame menjadi tipe data datetime menggunakan fungsi `pd.to_datetime()`.
- **df['anomaly'] = 0:** Menambahkan kolom 'anomaly' ke dalam DataFrame dan menginisialisasinya dengan nilai '0'.
- **for start, end in anomaly\_points: ...:** Melalui setiap rentang waktu yang diberikan dalam 'anomaly\_points', kode berikut memperbarui kolom 'anomaly' untuk baris data yang berada dalam rentang tersebut menjadi '1', menunjukkan bahwa rentang waktu tersebut dianggap sebagai anomali.

Kode di atas menambahkan kolom 'anomaly' ke dalam DataFrame 'df', yang menandai apakah setiap baris data merupakan anomali berdasarkan rentang waktu yang diberikan dalam 'anomaly\_points'. Rentang waktu yang diberikan dalam 'anomaly\_points' ditentukan sebagai anomali dengan menetapkan nilai '1' pada kolom 'anomaly'. Selain itu, kolom 'timestamp' diubah menjadi tipe data datetime.

### 4. Pre-Processing Datetime Information

```

Datetime Information

[8] df['year'] = df['timestamp'].apply(lambda x : x.year)
    df['month'] = df['timestamp'].apply(lambda x : x.month)
    df['day'] = df['timestamp'].apply(lambda x : x.day)
    df['hour'] = df['timestamp'].apply(lambda x : x.hour)
    df['minute'] = df['timestamp'].apply(lambda x : x.minute)

[9] df.index = df['timestamp']
    df.drop(['timestamp'], axis=1, inplace=True)
    df.head(3)

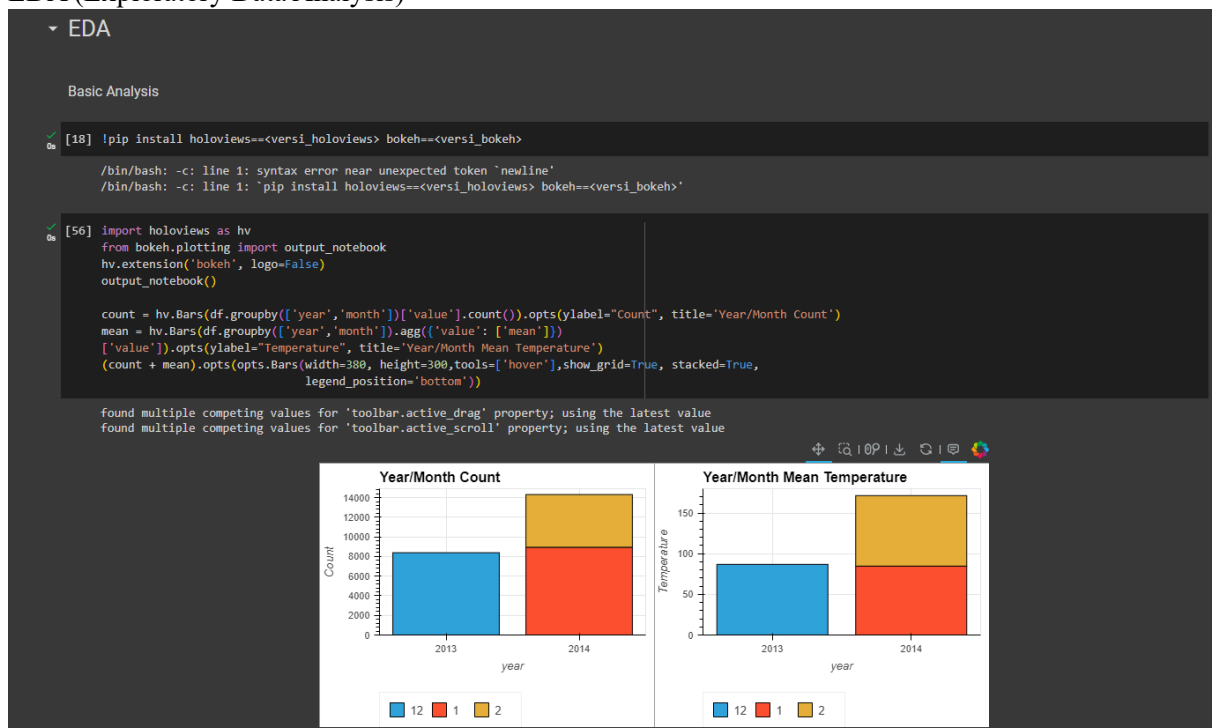
```

| timestamp           | value     | anomaly | year | month | day | hour | minute |
|---------------------|-----------|---------|------|-------|-----|------|--------|
| 2013-12-02 21:15:00 | 73.967322 | 0       | 2013 | 12    | 2   | 21   | 15     |
| 2013-12-02 21:20:00 | 74.935882 | 0       | 2013 | 12    | 2   | 21   | 20     |
| 2013-12-02 21:25:00 | 76.124162 | 0       | 2013 | 12    | 2   | 21   | 25     |

Kode di atas melakukan ekstraksi informasi waktu dari kolom 'timestamp' dalam DataFrame 'df' dan membuat kolom baru untuk tahun ('year'), bulan ('month'), hari ('day'), jam ('hour'), dan menit ('minute'). Selanjutnya, kolom 'timestamp' dijadikan indeks DataFrame, dan kolom tersebut dihapus dari DataFrame.

- **df['year'] = df['timestamp'].apply(lambda x : x.year):** Menambahkan kolom 'year' ke DataFrame, yang berisi tahun dari timestamp.
- **df['month'] = df['timestamp'].apply(lambda x : x.month):** Menambahkan kolom 'month' ke DataFrame, yang berisi bulan dari timestamp.
- **df['day'] = df['timestamp'].apply(lambda x : x.day):** Menambahkan kolom 'day' ke DataFrame, yang berisi hari dari timestamp.
- **df['hour'] = df['timestamp'].apply(lambda x : x.hour):** Menambahkan kolom 'hour' ke DataFrame, yang berisi jam dari timestamp.
- **df['minute'] = df['timestamp'].apply(lambda x : x.minute):** Menambahkan kolom 'minute' ke DataFrame, yang berisi menit dari timestamp.
- **df.index = df['timestamp']:** Mengatur kolom 'timestamp' sebagai indeks DataFrame.
- **df.drop(['timestamp'], axis=1, inplace=True):** Menghapus kolom 'timestamp' dari DataFrame.

## 5. EDA (Exploratory Data Analysis)

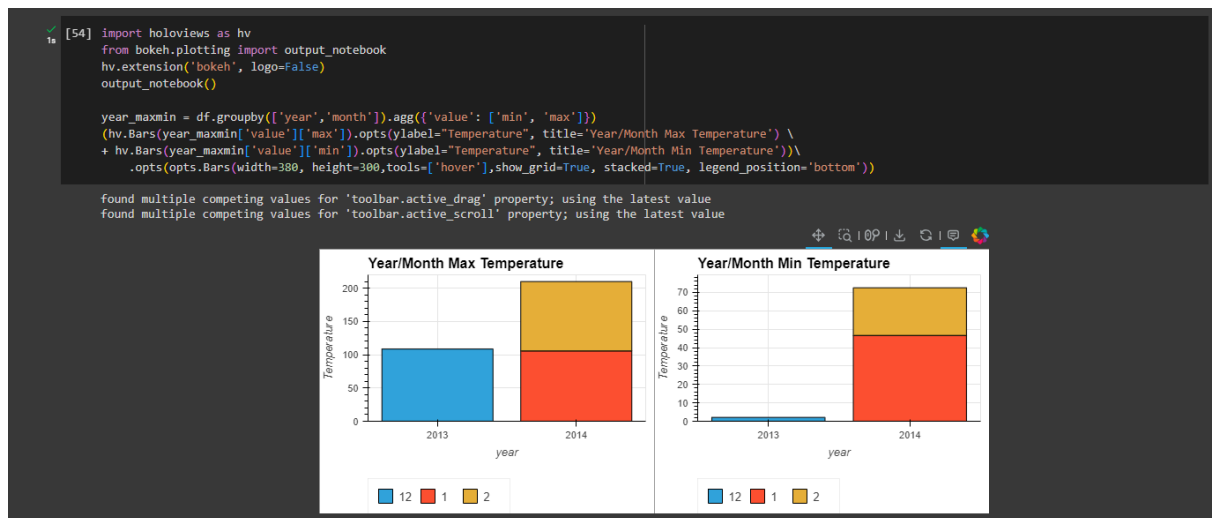


### "Analisis Jumlah dan Rata-Rata Suhu Mesin per Bulan dan Tahun"

Kode di atas menggunakan Holoviews dan Bokeh untuk membuat visualisasi interaktif berupa grafik batang. Dua grafik batang digabungkan, yang pertama menunjukkan jumlah pengamatan per bulan dan tahun, dan yang kedua menunjukkan rata-rata suhu per bulan dan tahun.

- **hv.Bars(df.groupby(['year', 'month'])['value'].count()):** Membuat grafik batang yang menunjukkan jumlah pengamatan (count) per bulan dan tahun.
- **hv.Bars(df.groupby(['year', 'month']).agg({'value': ['mean']}))['value']:** Membuat grafik batang yang menunjukkan rata-rata suhu per bulan dan tahun.
- **(count + mean):** Menggabungkan kedua grafik batang tersebut.

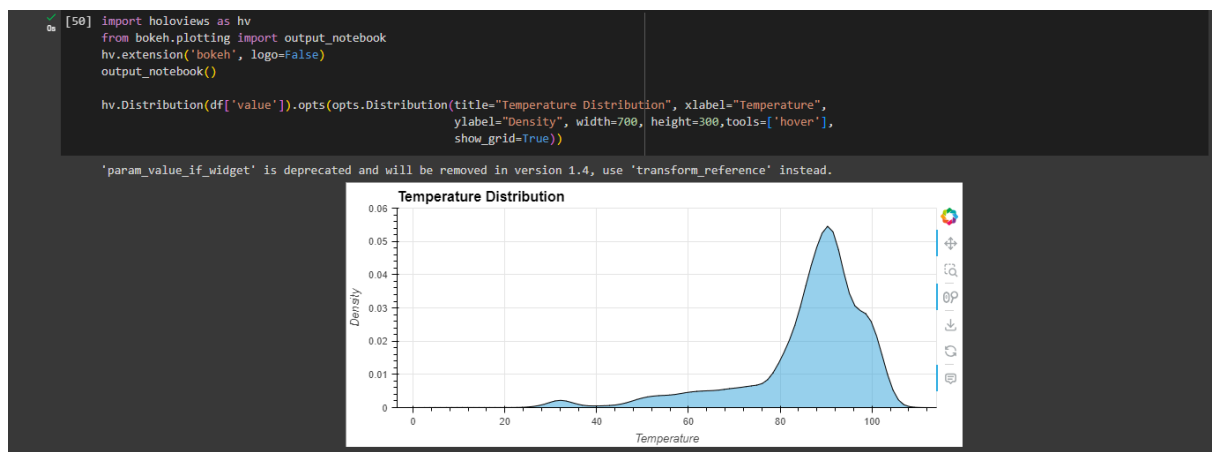
- **.opts(...)**: Menetapkan beberapa opsi (options) untuk mengatur tampilan visualisasi, seperti lebar dan tinggi plot, label sumbu, judul, dan lainnya.
- **output\_notebook()**: Mengatur output untuk ditampilkan di dalam notebook.



### "Analisis Suhu Maksimum dan Minimum Mesin per Bulan dan Tahun"

Kode di atas menggunakan Holoviews dan Bokeh untuk membuat visualisasi interaktif berupa grafik batang. Dua grafik batang digabungkan, yang pertama menunjukkan suhu maksimum per bulan dan tahun, dan yang kedua menunjukkan suhu minimum per bulan dan tahun.

- **df.groupby(['year','month']).agg({'value': ['min', 'max']})**: Mengelompokkan data berdasarkan tahun dan bulan, kemudian menghitung nilai minimum dan maksimum suhu untuk setiap kelompok.
- **hv.Bars(year\_maxmin['value']['max'])**: Membuat grafik batang yang menunjukkan suhu maksimum per bulan dan tahun.
- **hv.Bars(year\_maxmin['value']['min'])**: Membuat grafik batang yang menunjukkan suhu minimum per bulan dan tahun.
- **(hv.Bars(year\_maxmin['value']['max']).opts(...) + hv.Bars(year\_maxmin['value']['min']).opts(...))**: Menggabungkan kedua grafik batang tersebut.
- **.opts(...)**: Menetapkan beberapa opsi (options) untuk mengatur tampilan visualisasi, seperti lebar dan tinggi plot, label sumbu, judul, dan lainnya.

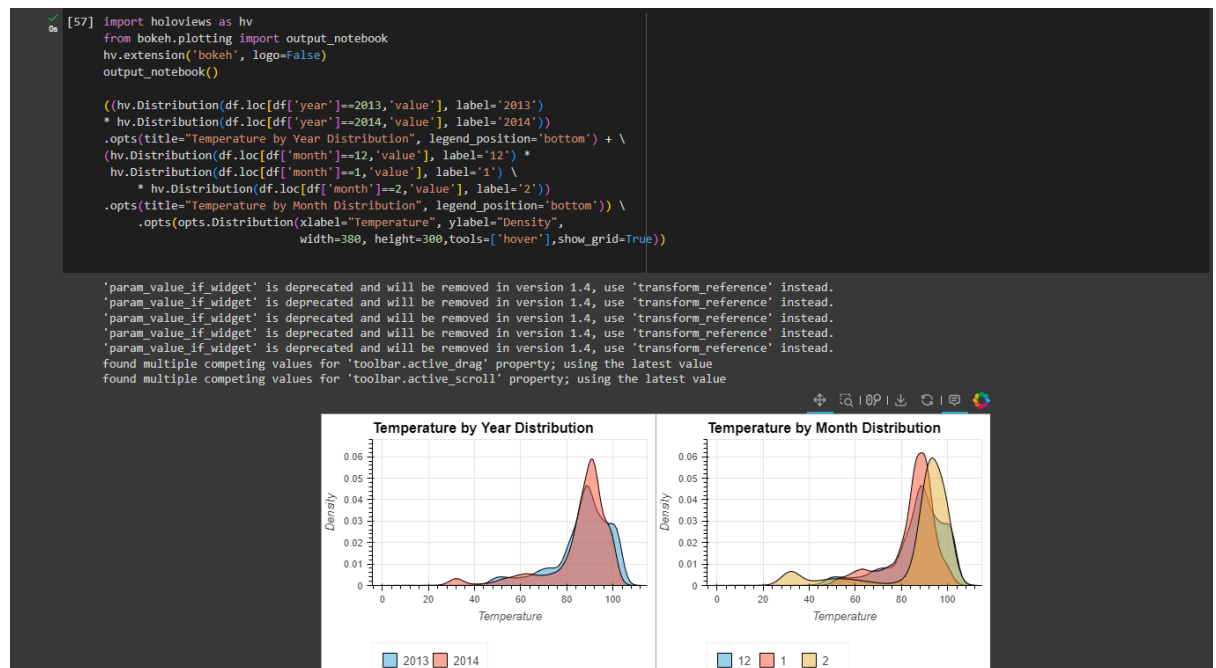


### "Analisis Distribusi Suhu Mesin"

Kode di atas menggunakan Holoviews untuk membuat visualisasi distribusi suhu menggunakan grafik distribusi (Distribution Plot).

- **hv.Distribution(df['value'])** : Membuat grafik distribusi untuk kolom 'value' (suhu) dalam DataFrame.
- **.opts(...)** : Menetapkan beberapa opsi (options) untuk mengatur tampilan visualisasi, seperti judul, label sumbu, lebar dan tinggi plot, serta pengaturan lainnya.

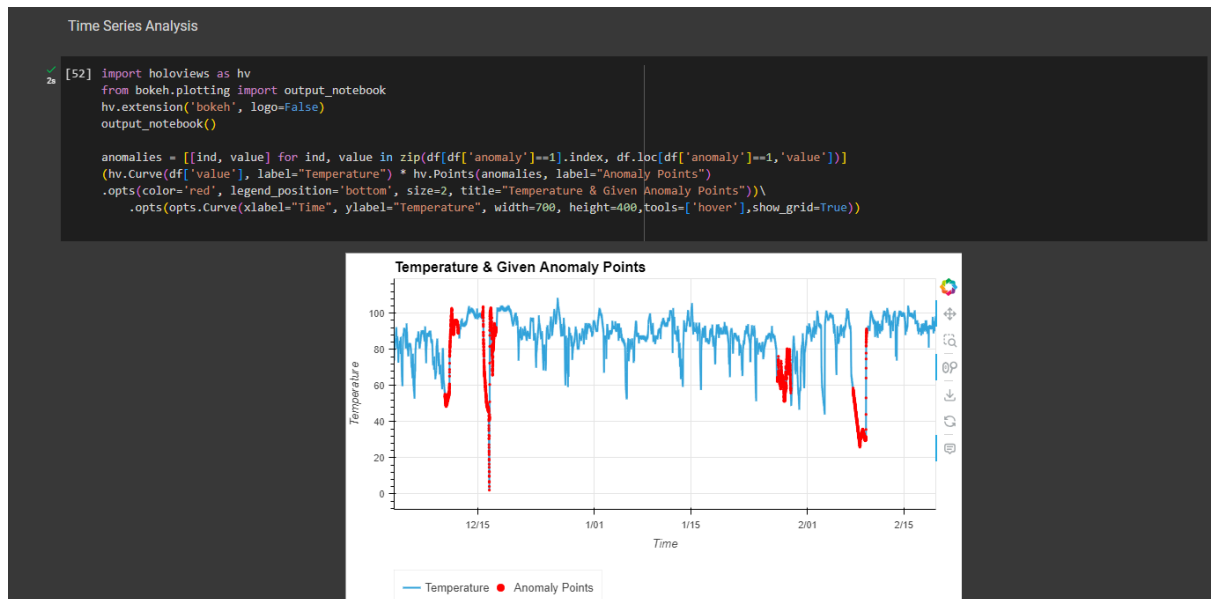
Visualisasi ini menunjukkan distribusi suhu pada data dengan menggunakan kurva distribusi.



### "Analisis Distribusi Suhu Mesin per Tahun dan per Bulan"

Kode di atas menggunakan Holoviews untuk membuat visualisasi distribusi suhu berdasarkan tahun dan bulan. Dua set grafik distribusi digabungkan. Bagian pertama membandingkan distribusi suhu untuk tahun 2013 dan 2014, sedangkan bagian kedua membandingkan distribusi suhu untuk bulan Desember, Januari, dan Februari.

- **(hv.Distribution(df.loc[df['year']==2013,'value'], label='2013') \* hv.Distribution(df.loc[df['year']==2014,'value'], label='2014'))** : Membuat grafik distribusi suhu untuk tahun 2013 dan 2014, kemudian menggabungkannya.
- **(hv.Distribution(df.loc[df['month']==12,'value'], label='12') \* hv.Distribution(df.loc[df['month']==1,'value'], label='1') \* hv.Distribution(df.loc[df['month']==2,'value'], label='2'))** : Membuat grafik distribusi suhu untuk bulan Desember, Januari, dan Februari, kemudian menggabungkannya.
- **(hv.Distribution(...) + hv.Distribution(...))** : Menggabungkan dua set grafik distribusi tersebut.
- **.opts(...)** : Menetapkan beberapa opsi (options) untuk mengatur tampilan visualisasi, seperti judul, label sumbu, lebar dan tinggi plot, serta pengaturan lainnya.



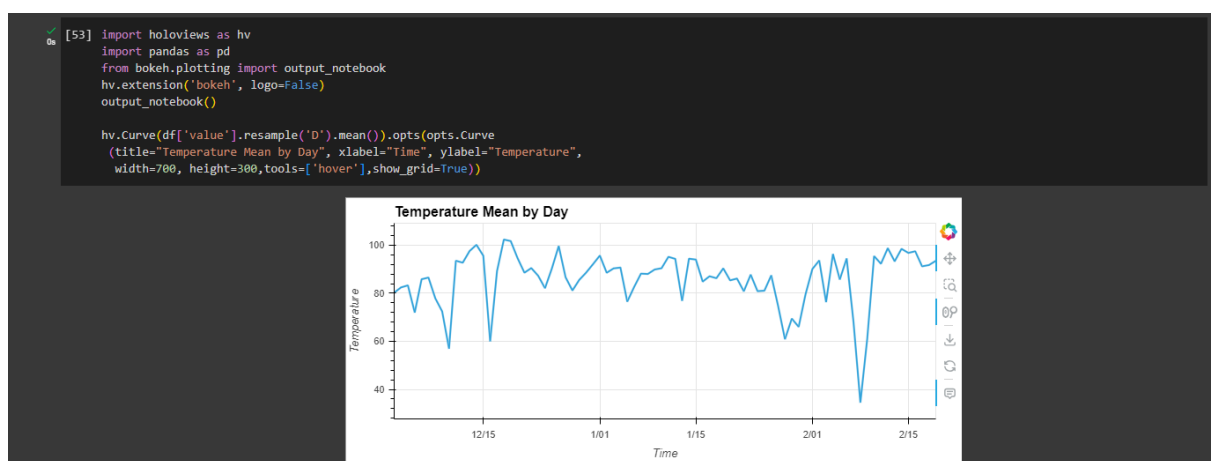
### "Visualisasi Suhu Mesin sepanjang Waktu dengan Poin Anomali"

Kode di atas menggunakan Holoviews untuk membuat visualisasi yang menunjukkan data suhu sepanjang waktu dengan penanda titik merah untuk poin-poin yang diidentifikasi sebagai anomali.

#### Pengertian Kode:

- **hv.Curve(df['value'], label="Temperature"):** Membuat kurva (curve) untuk data suhu.
- **hv.Points(anomalies, label="Anomaly Points"):** Membuat titik-titik merah untuk poin-poin yang diidentifikasi sebagai anomali.
- **(hv.Curve(...)) \* hv.Points(...):** Menggabungkan kurva suhu dengan titik-titik anomali.
- **.opts(...):** Menetapkan beberapa opsi (options) untuk mengatur tampilan visualisasi, seperti judul, label sumbu, lebar dan tinggi plot, serta pengaturan lainnya.

Visualisasi ini memungkinkan untuk melihat sebaran suhu sepanjang waktu, dengan penekanan pada poin-poin yang dianggap sebagai anomali. Opsi yang diberikan mencakup warna merah untuk titik-titik anomali, judul plot, label sumbu, dan pengaturan lainnya.



### "Rata-rata Suhu Harian Mesin sepanjang Waktu"

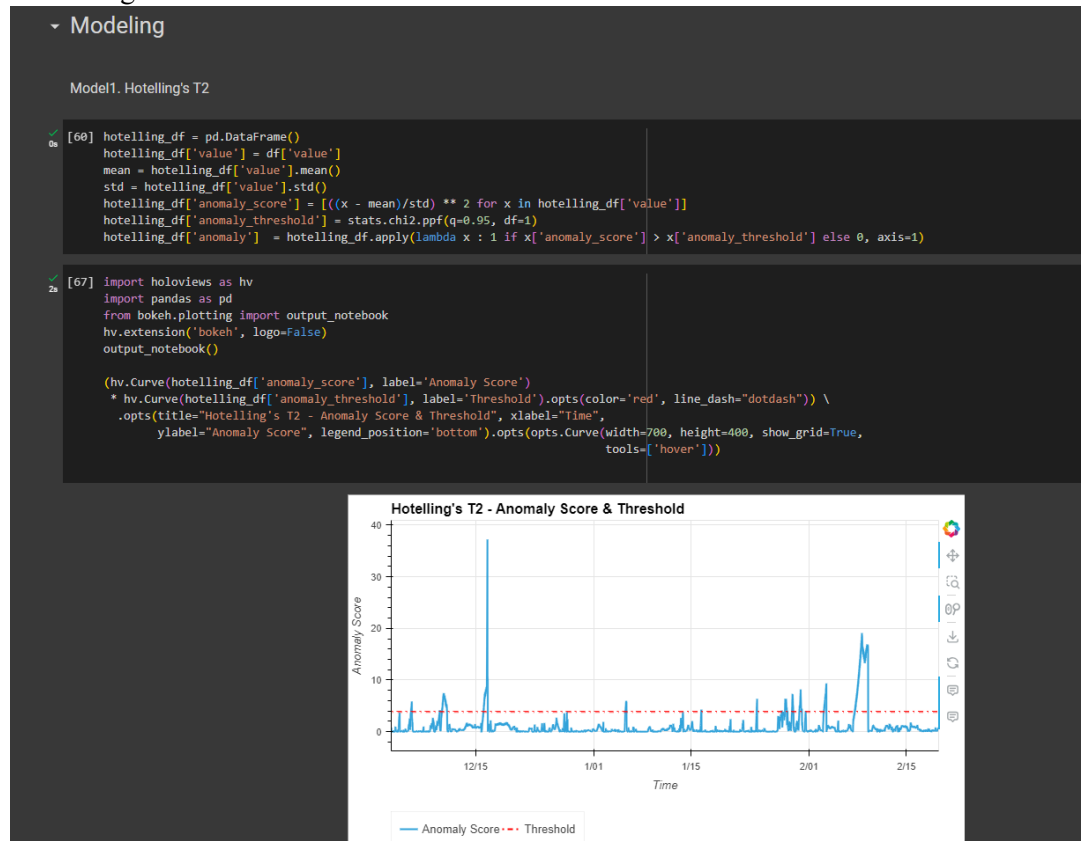
Kode di atas menggunakan Holoviews untuk membuat visualisasi yang menunjukkan rata-rata suhu harian dari data suhu mesin sepanjang waktu.

#### Pengertian Kode:

- `df['value'].resample('D').mean()`: Meresample data suhu ('value') dengan interval harian dan menghitung nilai rata-rata harian.
- `hv.Curve(...)`: Membuat kurva untuk rata-rata suhu harian.
- `.opts(...)`: Menetapkan beberapa opsi (options) untuk mengatur tampilan visualisasi, seperti judul, label sumbu, lebar dan tinggi plot, serta pengaturan lainnya.  
Visualisasi ini memberikan gambaran tentang bagaimana rata-rata suhu mesin berubah dari hari ke hari sepanjang waktu.

## 6. Modeling

### a. Hotelling's T2



### "Hotelling's T<sup>2</sup> - Skor Anomali dan Threshold sepanjang Waktu"

Kode di atas mengimplementasikan metode Hotelling's T<sup>2</sup> untuk mendeteksi anomali pada data suhu mesin. Kode tersebut kemudian menggunakan Holoviews untuk membuat visualisasi yang menunjukkan skor anomali dan ambang batas (threshold) dari Hotelling's T<sup>2</sup>.

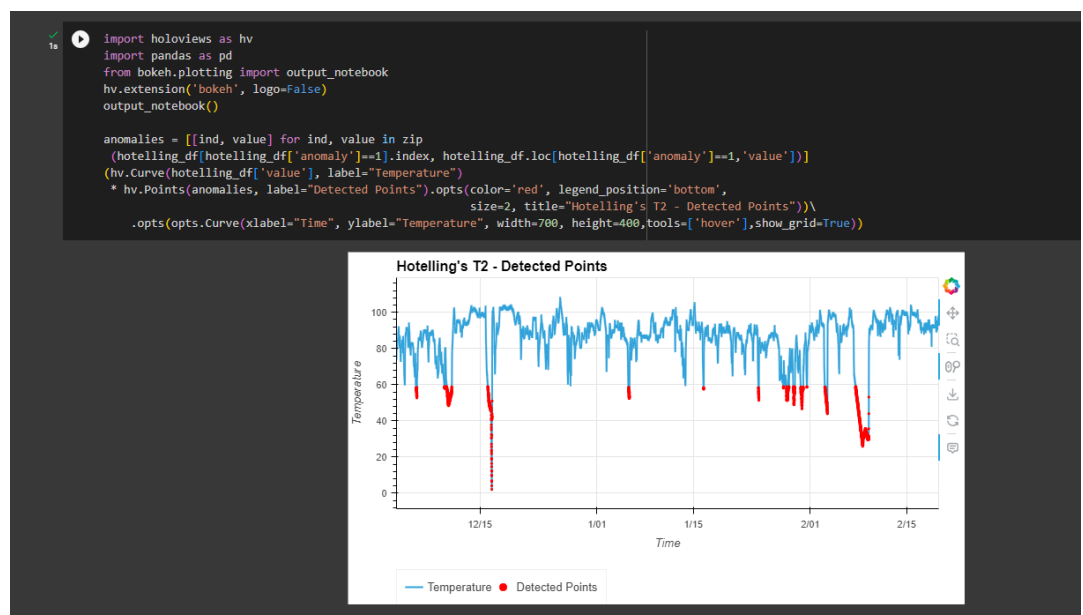
#### Pengertian Kode:

- `hotelling_df = pd.DataFrame()`: Membuat DataFrame baru untuk menyimpan data dan skor anomali dari metode Hotelling's T<sup>2</sup>.
- `hotelling_df['value'] = df['value']`: Menyalin kolom 'value' (suhu) dari DataFrame asli ke DataFrame baru.
- `mean = hotelling_df['value'].mean()`: Menghitung rata-rata dari data suhu.
- `std = hotelling_df['value'].std()`: Menghitung deviasi standar dari data suhu.
- `hotelling_df['anomaly_score']`: Menghitung skor anomali untuk setiap pengamatan menggunakan rumus Hotelling's T<sup>2</sup>.
- `hotelling_df['anomaly_threshold']`: Menghitung ambang batas (threshold) menggunakan distribusi chi-square.



- **hotelling\_df['anomaly']**: Menentukan apakah suatu pengamatan dianggap sebagai anomali berdasarkan perbandingan skor anomali dengan ambang batas.
- **(hv.Curve(hotelling\_df['anomaly\_score'], label='Anomaly Score') \* hv.Curve(hotelling\_df['anomaly\_threshold'], label='Threshold')).opts(color='red', line\_dash="dotted")**: Membuat dua kurva, satu untuk skor anomali dan satu untuk ambang batas, dan menggabungkannya.
- **.opts(...)**: Menetapkan beberapa opsi (options) untuk mengatur tampilan visualisasi, seperti judul, label sumbu, lebar dan tinggi plot, serta pengaturan lainnya.

Visualisasi ini memberikan gambaran tentang bagaimana skor anomali berkembang seiring waktu, dan titik-titik yang melebihi ambang batas (dengan garis batas merah) dapat dianggap sebagai anomali.



"Visualisasi Suhu Mesin sepanjang Waktu dengan Poin Anomali Hotelling's T<sup>2</sup>"

Kode di atas menggunakan Holoviews untuk membuat visualisasi yang menunjukkan data suhu sepanjang waktu dengan penanda titik merah untuk poin-poin yang diidentifikasi sebagai anomali oleh metode Hotelling's T<sup>2</sup>.

**Pengertian Kode:**

- **[[ind, value] for ind, value in zip(hotelling\_df[hotelling\_df['anomaly']==1].index, hotelling\_df.loc[hotelling\_df['anomaly']==1, 'value'])**: Membuat daftar yang berisi pasangan indeks dan nilai dari poin-poin yang diidentifikasi sebagai anomali.
- **(hv.Curve(hotelling\_df['value'], label="Temperature") \* hv.Points(anomalies, label="Detected Points"))**: Membuat kurva untuk data suhu dan menambahkan titik-titik merah untuk poin-poin yang diidentifikasi sebagai anomali.
- **.opts(...)**: Menetapkan beberapa opsi (options) untuk mengatur tampilan visualisasi, seperti judul, label sumbu, lebar dan tinggi plot, serta pengaturan lainnya.

Visualisasi ini memungkinkan untuk melihat sebaran suhu sepanjang waktu, dengan penekanan pada poin-poin yang dianggap sebagai anomali oleh metode Hotelling's T

```
[69] hotelling_f1 = f1_score(df['anomaly'], hotelling_df['anomaly'])
      print(f'Hotelling's T2 F1 Score : {hotelling_f1}')

Hotelling's T2 F1 Score : 0.5440778799351
```

Hotelling's T2 F1 Score : 0.5440778799351

Kode di atas menghitung nilai F1 Score untuk metode deteksi anomali Hotelling's T<sup>2</sup> dengan membandingkannya dengan ground truth yang terdapat dalam kolom 'anomaly'.

## b. One-Class SVM



"One-Class SVM - Visualisasi Suhu Mesin sepanjang Waktu dengan Poin Anomali"

Kode di atas menggunakan metode One-Class SVM (Support Vector Machine) untuk mendeteksi anomali dalam data suhu mesin. Kode tersebut kemudian menggunakan Holoviews untuk membuat visualisasi yang menunjukkan data suhu sepanjang waktu dengan penanda titik merah untuk poin-poin yang diidentifikasi sebagai anomali oleh model One-Class SVM.

### Pengertian Kode:

- **OneClassSVM(nu=0.2, gamma=0.001, kernel='rbf')**: Membuat model One-Class SVM dengan parameter tertentu (nu=0.2, gamma=0.001, kernel='rbf').
- **ocsvm\_model.fit\_predict(df['value'].values.reshape(-1, 1))**: Melatih model One-Class SVM dan memberikan prediksi terhadap setiap pengamatan (1 untuk normal, -1 untuk anomali).
- **ocsvm\_df = pd.DataFrame()**: Membuat DataFrame baru untuk menyimpan data dan label anomali dari hasil prediksi.
- **ocsvm\_df['anomaly'] = [1 if i == -1 else 0 for i in ocsvm\_ret]**: Menetapkan label anomali berdasarkan prediksi model (anomali jika prediksi -1, normal jika prediksi 1).
- **(hv.Curve(ocsvm\_df['value'], label="Temperature") \* hv.Points(anomalies, label="Detected Points"))**: Membuat kurva untuk data \*

suhu dan menambahkan titik-titik merah untuk poin-poin yang diidentifikasi sebagai anomali.

- **.opts(...)**: Menetapkan beberapa opsi (options) untuk mengatur tampilan visualisasi, seperti judul, label sumbu, lebar dan tinggi plot, serta pengaturan lainnya.

Visualisasi ini memungkinkan untuk melihat sebaran suhu sepanjang waktu, dengan penekanan pada poin-poin yang dianggap sebagai anomali oleh model One-Class SVM.

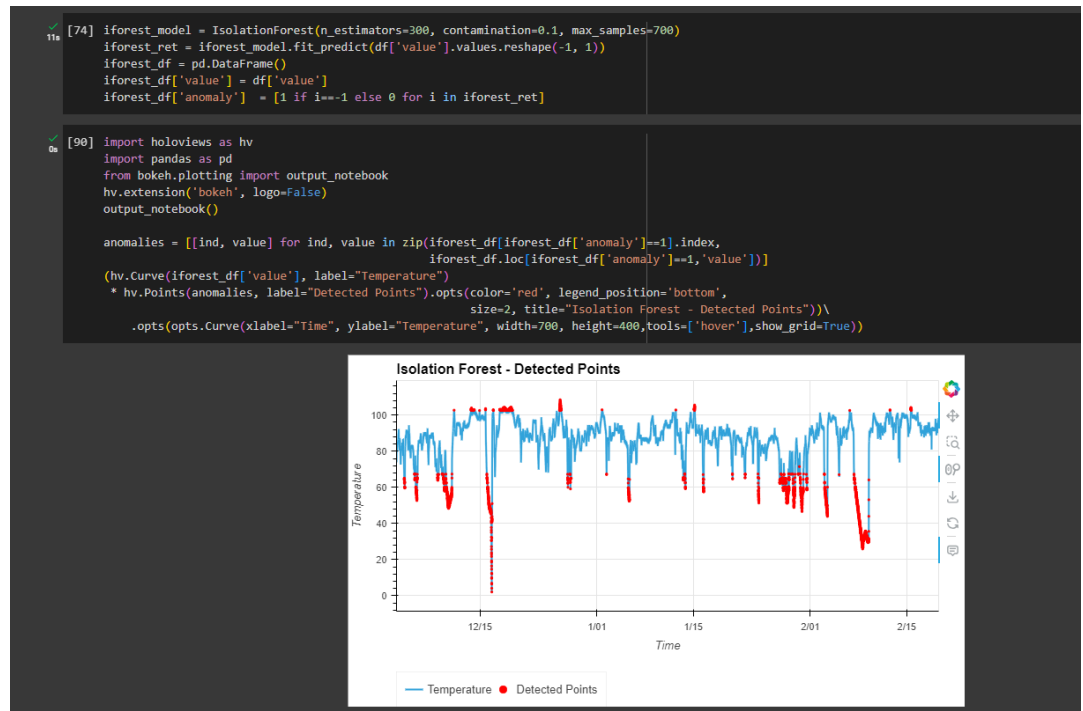
```
[73] ocsvm_f1 = f1_score(df['anomaly'], ocsvm_df['anomaly'])
      print(f'One-Class SVM F1 Score : {ocsvm_f1}')

One-Class SVM F1 Score : 0.4224441833137485
```

One-Class SVM F1 Score : 0.4224441833137485

Kode di atas menghitung nilai F1 Score untuk metode deteksi anomali menggunakan One-Class SVM. F1 Score dihitung dengan membandingkan label anomali yang sebenarnya (ground truth) dari DataFrame 'df' dengan label anomali yang dihasilkan oleh model One-Class SVM dari DataFrame 'ocsvm\_df'.

### c. Isolation Forest



"Isolation Forest - Visualisasi Suhu Mesin sepanjang Waktu dengan Poin Anomali"

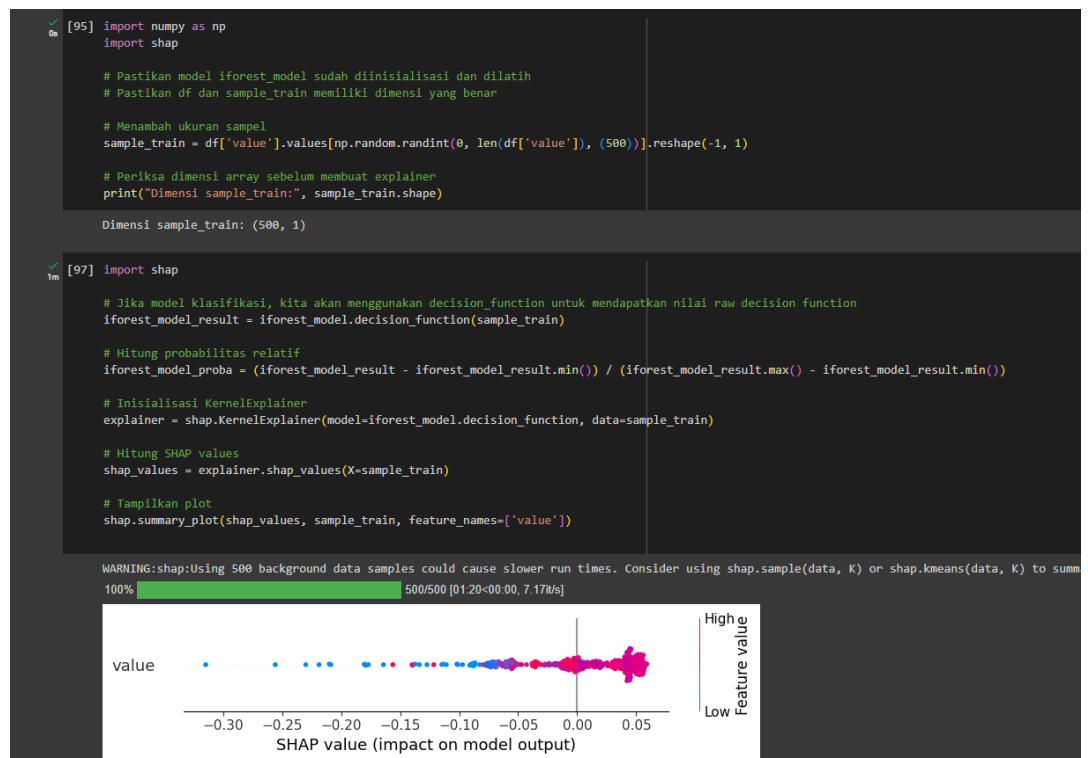
Kode di atas menggunakan metode Isolation Forest untuk mendeteksi anomali dalam data suhu mesin. Kode tersebut kemudian menggunakan Holoviews untuk membuat visualisasi yang menunjukkan data suhu sepanjang waktu dengan penanda titik merah untuk poin-poin yang diidentifikasi sebagai anomali oleh model Isolation Forest.

#### Pengertian Kode:

- **IsolationForest(n\_estimators=300, contamination=0.1, max\_samples=700)**: Membuat model Isolation Forest dengan parameter tertentu (n\_estimators=300, contamination=0.1, max\_samples=700).
- **iforest\_model.fit\_predict(df['value'].values.reshape(-1, 1))**: Melatih model Isolation Forest dan memberikan prediksi terhadap setiap pengamatan (1 untuk normal, -1 untuk anomali).

- **iforest\_df = pd.DataFrame():** Membuat DataFrame baru untuk menyimpan data dan label anomali dari hasil prediksi.
- **iforest\_df['anomaly'] = [1 if i== -1 else 0 for i in iforest\_ret]:** Menetapkan label anomali berdasarkan prediksi model (anomali jika prediksi -1, normal jika prediksi 1).
- **(hv.Curve(iforest\_df['value'], label="Temperature") \* hv.Points(anomalies, label="Detected Points")):** Membuat kurva untuk data suhu dan menambahkan titik-titik merah untuk poin-poin yang diidentifikasi sebagai anomali.
- **.opts(...):** Menetapkan beberapa opsi (options) untuk mengatur tampilan visualisasi, seperti judul, label sumbu, lebar dan tinggi plot, serta pengaturan lainnya.

Visualisasi ini memungkinkan untuk melihat sebaran suhu sepanjang waktu, dengan penekanan pada poin-poin yang dianggap sebagai anomali oleh model Isolation Forest.



"SHAP Explanation untuk Model Isolation Forest pada Data Suhu Mesin dengan Plot Ringkasan"

Kode ini menggunakan library SHAP untuk menjelaskan hasil prediksi model Isolation Forest terhadap sampel data suhu mesin. Langkah-langkahnya mencakup pembuatan sampel latihan, inisialisasi objek KernelExplainer dari SHAP, dan penghitungan SHAP values untuk setiap fitur dalam sampel. Hasilnya ditampilkan dalam plot ringkasan untuk memberikan wawasan tentang faktor-faktor mana yang paling memengaruhi prediksi model.

```
[98] iforest_f1 = f1_score(df['anomaly'], iforest_df['anomaly'])
print(f'Isolation Forest F1 Score : {iforest_f1}')

Isolation Forest F1 Score : 0.5200528867342442
```

Isolation Forest F1 Score : 0.5200528867342442

Kode di atas menghitung dan mencetak nilai F1 Score untuk metode deteksi anomali menggunakan model Isolation Forest. F1 Score dihitung dengan membandingkan label anomali yang sebenarnya (ground truth).

#### d. LOF



"Visualisasi Anomali dengan Local Outlier Factor pada Data Suhu Mesin"

Kode ini menggunakan metode Local Outlier Factor (LOF) untuk mendeteksi anomali dalam data suhu mesin. Setelah melatih model LOF dan mendapatkan prediksi, dilakukan visualisasi menggunakan Holoviews. Kurva menunjukkan tren suhu sepanjang waktu, sedangkan titik-titik merah menunjukkan lokasi poin-poin yang dianggap sebagai anomali oleh model LOF.

The screenshot shows a single code cell (index 101) that calculates the F1 Score for the LOF model. It uses the `f1_score` function from the `sklearn.metrics` module, comparing the predicted anomalies (from the 'anomaly' column) with the ground truth anomalies. The output of the code is printed to the console, showing the LOF F1 Score as 0.3577910292973813.

LOF F1 Score : 0.3577910292973813

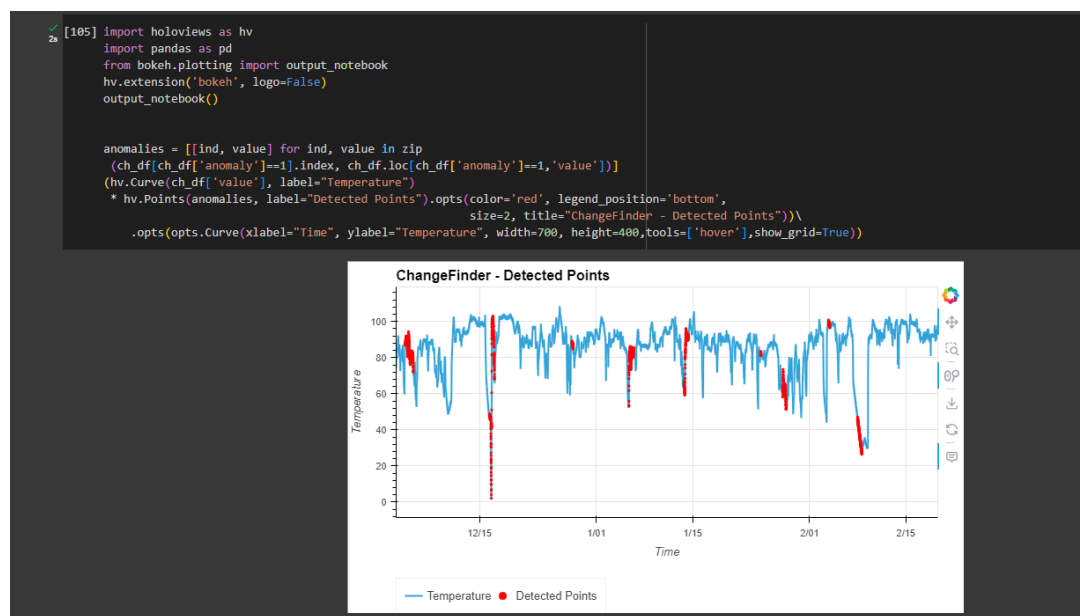
Kode ini menghitung dan mencetak nilai F1 Score untuk metode deteksi anomali menggunakan model Local Outlier Factor (LOF). F1 Score dihitung dengan membandingkan label anomali yang sebenarnya (ground truth).

#### e. ChangeFinder



## "Deteksi Anomali dengan ChangeFinder pada Data Suhu Mesin: Skor Anomali dan Threshold"

Kode ini menggunakan algoritma ChangeFinder untuk mendeteksi anomali dalam data suhu mesin. Algoritma ini menghasilkan skor anomali untuk setiap pengamatan suhu, dan sebuah ambang batas ditentukan berdasarkan nilai-nilai skor anomali. Hasilnya divisualisasikan menggunakan Holoviews, menunjukkan skor anomali dan ambang batas dengan garis putus-putus merah.



## "Visualisasi Poin Anomali dengan ChangeFinder pada Data Suhu Mesin"

Kode ini membentuk dan memvisualisasikan poin-poin yang dianggap sebagai anomali oleh algoritma ChangeFinder. Titik-titik merah pada kurva suhu menunjukkan lokasi anomali berdasarkan skor anomali yang dihasilkan.

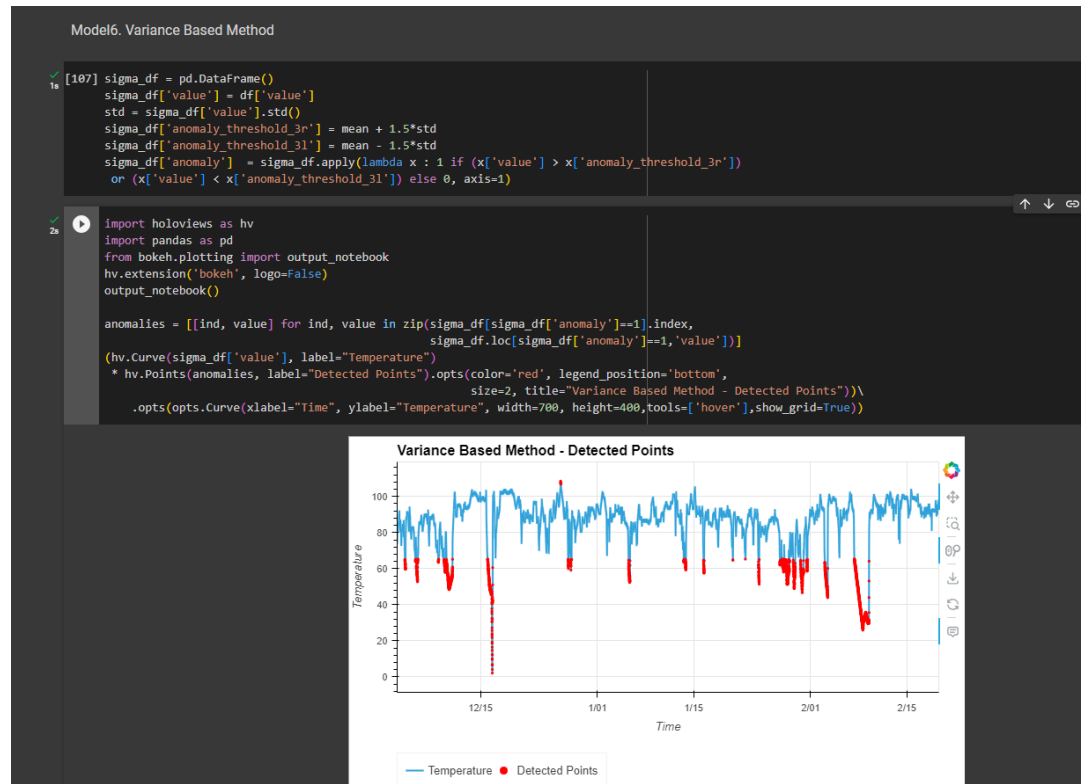
```
[106] ch_f1 = f1_score(df['anomaly'], ch_df['anomaly'])
      print(f'ChangeFinder F1 Score : {ch_f1}')

ChangeFinder F1 Score : 0.2928434329585961
```

ChangeFinder F1 Score : 0.2928434329585961

Kode di atas menghitung dan mencetak nilai F1 Score untuk metode deteksi anomali menggunakan algoritma ChangeFinder. F1 Score dihitung dengan membandingkan label anomali yang sebenarnya (ground truth) dari DataFrame 'df' dengan label anomali yang dihasilkan oleh algoritma ChangeFinder dari DataFrame 'ch\_df'.

#### f. Variance Based Method



"Deteksi Anomali Berbasis Varians pada Data Suhu Mesin: Poin Anomali yang Terdeteksi"

Kode ini menggunakan metode berbasis varians untuk mendeteksi anomali dalam data suhu mesin. Ambang batas ditentukan berdasarkan deviasi standar, dan poin-poin yang berada di luar ambang batas tersebut dianggap sebagai anomali. Hasilnya divisualisasikan dengan menggunakan Holoviews, menunjukkan poin-poin yang dianggap sebagai anomali pada kurva suhu.

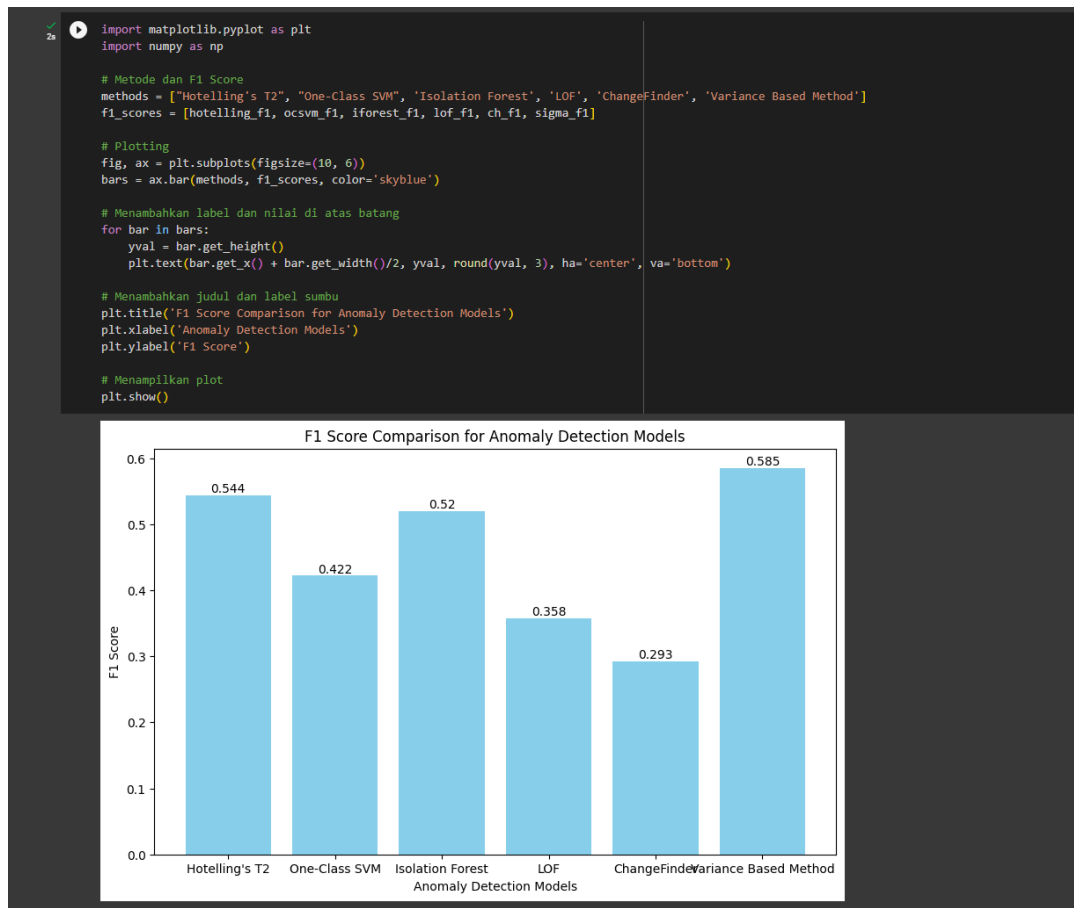
```
[109] sigma_f1 = f1_score(df['anomaly'], sigma_df['anomaly'])
      print(f'Variance Based Method F1 Score : {sigma_f1}')

Variance Based Method F1 Score : 0.585277463193658
```

Variance Based Method F1 Score : 0.585277463193658

Kode di atas menghitung dan mencetak nilai F1 Score untuk metode deteksi anomali berbasis varians. F1 Score dihitung dengan membandingkan label anomali yang sebenarnya (ground truth) dari DataFrame 'df' dengan label anomali yang dihasilkan oleh metode berbasis varians dari DataFrame 'sigma\_df'.

### 7. Komparasi Model



Kode ini menggunakan matplotlib untuk membuat diagram batang yang membandingkan nilai F1 Score dari berbagai metode deteksi anomali pada data suhu mesin. Setiap metode direpresentasikan sebagai batang, dan tinggi batang tersebut mencerminkan nilai F1 Score yang dihasilkan oleh masing-masing metode.

### C. Kesimpulan

- Dari data yang ada terdeteksi adanya anomali pada sistem temperatur mesin industri
- Model yang dibangun dengan algoritma sederhana menghasilkan akurasi yang tinggi
- Untuk F1 Score paling tinggi dipegang oleh model Variance Based Method dengan nilai 0.585, Disusul oleh Hotelling T2 dengan skor 0.544, dan diikuti oleh metode lainnya.
- Pada Variance Based Method cenderung menampilkan hasil anomali kepada hasil pengelolaan yang menghasilkan suhu rendah.
- Hampir semua model menentukan bahwa tanggal 2013-12-16, 2014-02-08 adalah tanggal terjadinya anomali.
- Berdasarkan pada Variance Based Method, anomali suhu rendah terjadi pada tanggal 2013-12-04, 2013-12-05, 2013-12-08, 2013-12-10, 2013-12-16, 2013-12-27, 2014-01-05, 2014-01-13, 2014-01-16, 2014-01-20, 2014-01-22, 2014-01-24, 2014-01-27 s/d 31, , 2014-02-03, 2014-02-07 s/d 09. Dan untuk anomali suhu tinggi terjadi pada tanggal 2013-12-26.



## D. Referensi

- NAB Anomaly Points References  
[https://github.com/numenta/NAB/blob/master/labels/combined\\_windows.json](https://github.com/numenta/NAB/blob/master/labels/combined_windows.json)
- Anomaly Detection Learning Resources  
<https://github.com/yzhao062/anomaly-detection-resources>
- PyOD documentation  
<https://pyod.readthedocs.io/en/latest/>
- Anomaly Detection Toolkit documentation  
<https://arundo-adtk.readthedocs-hosted.com/en/latest/>
- Kaggle  
<https://www.kaggle.com/code/koheimuramatsu/industrial-machine-anomaly-detection>