دوره کارشناسی ارشد مهندسی برق–کنترل

مینی‌پروژه اول درس یادگیری ماشین

توسط:

محمدرضا امانی – احمدرضا طاهری

استاد راهنما:

دکتر مهدی علیاری شوره‌دلی

بهار ۱۴۰۴

Q1 code

Q2 code


بسم الله الرحمن الرحیم

چکیده

در فصل اول این گزارش، به بررسی رگرسیون می‌پردازیم و سعی می‌کنیم تا با استفاده از مفهوم یادگیری تجمعی (collaborative learning)، عملکرد مدل خود را افزایش دهیم. همچنین در بررسی مدل‌های خود، از دو رویکرد آموزش با دادگان تکی و پنجره‌ای استفاده می‌کنیم.

کلید واژه: رگرسیون، یادگیری تجمعی

فهرست مطالب

عنوان **صفحه**

فهرست جدول‌ها

عنوان **صفحه**

فهرست شکل‌ها

عنوان **صفحه**

فصل ۱– پیش‌بینی آب و هوا مبتنی بر یادگیری ماشین

۱-۱- دادگان

۱-۱-۱- توضیحات دادگان

گزارش سمینار و پایان نامه باید حاوی بخش‌های زیر به ترتیب ذکر شده باشد:
دیتاست استفاده شده در این مقاله مربوط به یک سیستم پیش‌بینی وضعیت آب‌وهوا است که با استفاده از یادگیری ماشین و به صورت مشارکتی (Collaborative) عمل می‌کند. خلاصه‌ای از مشخصات دیتاست به شرح زیر است:

دوره جمع‌آوری داده:

- از ۱ ام تا ۳۱ ام ماه May سال ۲۰۲۱

موقعیت‌های مکانی (۴ شهر در موریس):

۱. Curepipe
۲. Vacoas
۳. Quatres Bornes
۴. Moka

جزئیات هر موقعیت:

- ارتفاع از سطح دریا و مساحت هر منطقه نیز ثبت شده‌اند.

تعداد نمونه‌ها:

- در مجموع ۲۹۷۶ نمونه برای هر موقعیت
- نرخ نمونه‌برداری :چهار نمونه در ساعت

متغیرهای آب‌وهوایی جمع‌آوری شده:

۱. دما(Temperature)
۲. سرعت باد(Wind Speed)

۳. جهت باد(Wind Direction)

۴. فشار هوا(Pressure)

۵. رطوبت(Humidity)

۶. پوشش ابری(Cloudiness)

ابزار جمع‌آوری داده:

- استفاده از OpenWeather API
- جمع‌آوری داده‌ها از طریق دستگاه‌های موبایل و دسکتاپ
- ذخیره‌سازی در دو پایگاه داده:
  - محلی(MySQL)
  - ابری(IBM Cloudant)

در این تحقیق، داده‌های جمع‌آوری‌شده از چند منطقه به‌طور مشترک برای پیش‌بینی وضعیت آب‌وهوای یک منطقه خاص استفاده شده است. این رویکرد **collaborative forecasting** باعث افزایش دقت پیش‌بینی‌ها شده است—به‌طور متوسط ۵٪ کاهش در خطای MAPE نسبت به روش‌های غیرمشارکتی مشاهده شده است.

۱-۱-۲- فراخوانی دادگان

در دیتاست مورد بررسی دادگان ۳ شهر Montelimat، Perpignan و Tours از کشور فرانسه موجود است. دیتاست را فراخوانی و داده‌های این ۳ شهر را با استفاده از کد زیر ذخیره می‌کنیم:

```
data = pd.read_csv('data\weather_prediction_dataset.csv')
filtered_columns = [col for col in data.columns if 'DATE' in col or 'MONTH' in col or 'TOURS' in col or 'PERPIGNAN' in col or 'MONTELIMAR' in col]
df = data[filtered_columns]
df
```

✓ 0.1s

شکل ۱-۱: فراخوانی دادگان

۱-۱-۳- ویژگی‌های داده

این دیتاست شامل داده‌های یک بازه ۱۰ ساله از تاریخ ۲۰۰۰/۰۱/۰۱ تا ۲۰۱۰/۰۱/۰۱ هستند. نمونه‌برداری به صورت روزانه انجام شده است، بنابراین به طور کلی شامل ۳۶۵۴ نمونه هستند. از هر شهر ۸ ویژگی wind_speed، humidity، pressure، global radiation، preciptation، temp_mean، temp_min و temp_max نمونه‌برداری شده‌اند. بنابراین دیتاست در مجموع برای ۳ شهر، دارای ۲۴ ویژگی خواهد بود. در این مقاله دادگان قبل از آموزش نرمالایز شده‌اند. برای نرمال کردن دادگان از روش Min Max Normalization استفاده شده است. فرمول این روش به صورت زیر است:

$$\frac{x - \min(x)}{\max(x) - \min(x)}$$

	DATE	MONTH	MONTELIMAR_wind_speed	MONTELIMAR_humidity
0	20000101	1	3.8	0.85
1	20000102	1	5.8	0.82
2	20000103	1	0.4	0.92
3	20000104	1	1.1	0.85
4	20000105	1	3.4	0.82
...	...	...	...	...
3649	20091228	12	0.4	0.87
3650	20091229	12	1.8	0.84
3651	20091230	12	0.9	0.94
3652	20091231	12	0.3	0.90
3653	20100101	1	3.8	0.86

3654 rows × 26 columns

شکل ۲-۱: نمای کلی از دادگان

همینطور، با استفاده از متد describe، ویژگی‌های آماری دیتا را استخراج می‌کنیم(برای سه ستون اول داریم):

```
df.drop(['DATE','MONTH'],axis=1).describe()
```
✓ 0.0s

	MONTELIMAR_wind_speed	MONTELIMAR_humidity	MONTELIMAR_pressure
count	3654.000000	3654.000000	3654.000000
mean	3.680952	0.690794	1.017094
std	2.133979	0.129024	0.006988
min	0.000000	0.340000	0.986200
25%	2.000000	0.600000	1.013200
50%	3.100000	0.690000	1.017000
75%	5.100000	0.790000	1.021100
max	13.200000	0.980000	1.038700

8 rows × 24 columns

شکل ۳-۱: اعمال متد describe بر روی دادگان

داده‌های پس از نرمال سازی به فرم زیر در خواهند آمد. مشاهده می‌شود که طبق انتظار تمامی داده‌ها میان صفر و یک قرار می‌گیرند.

```
scaler = MinMaxScaler()
scaler.fit(df_train)
df_train_scaled=scaler.transform(df_train)

df_train = pd.DataFrame(data=df_train_scaled,columns=df_train.columns)
df_train
```

	MONTELIMAR_wind_speed	MONTELIMAR_humidity	MONTELIMAR_pressure
0	0.287879	0.796875	0.775238
1	0.439394	0.750000	0.809524
2	0.030303	0.906250	0.864762
3	0.083333	0.796875	0.784762
4	0.257576	0.750000	0.708571
...	...	...	...
3284	0.090909	0.703125	0.712381
3285	0.136364	0.750000	0.777143
3286	0.053030	0.968750	0.807619
3287	0.174242	0.843750	0.758095
3288	0.287879	0.812500	0.220952

3289 rows × 24 columns

شکل ۴-۱: دادگان پر از نرمالسازی

۱-۱-٤- پنجره‌بندی دادگان

با استفاده از کد زیر، داده‌های سال ۲۰۰۹ را به عنوان داده‌های آزمون و باقی‌داده‌ها را به عنوان داده آموزش در نظر می‌گیریم:

```
df_train = df.loc[(df['DATE']<20090101) | (df['DATE']>=20100101)]
df_train = pd.DataFrame(data=df_train.values, columns=df_train.columns)
df_train=df_train.drop(['DATE','MONTH'],axis=1)

df_test = df.loc[(df['DATE']>=20090101) & (df['DATE']<20100101)]
df_test = pd.DataFrame(data=df_test.values, columns=df_test.columns)
df_test=df_test.drop(['DATE','MONTH'],axis=1)
```

شکل ۵-۱: انتخاب دادگان آموزش و آزمایش

حال برای آموزش مدل‌های خود، نیاز داریم که به دو صورت عمل کنیم. ابتدا تنها از داده‌های زمان t-1 برا آموزش مدل در زمان t استفاده می‌کنیم. با استفاده از کد زیر داده‌ها را تقسیم می‌کنیم. برای این منظور کافی است که برای انتخاب xها، نمونه آخر را حذف و برای انتخاب yها، نمونه اول را حذف کنیم. اینگونه خروجی‌ها (yها)، بر اساس مقدار ورودی‌ها (xها) در زمان t-1 آموزش می‌بینند.

```
x_train_single = df_train.iloc[0:-1].values
y_train_single = df_train.iloc[1:].values

x_test_single = df_test.iloc[0:-1].values
y_test_single = df_test.iloc[1:].values
```

شکل ۶-۱: انتخاب دادگان تک زمانه

رویکرد دیگر، انتخاب یک پنجره زمانی به اندازه دلخواه L از بازه t-1 تا t-L است. بدین منظور نیاز است که ابتدا داده‌ها را به پنجره‌هایی به اندازه L تقسیم کنیم.

```python
def sliding_window(df, window_size, stride=1):

    # Number of features
    n = df.shape[1]


    num_samples = (df.shape[0] - window_size) // stride + 1

    data = df.values  # Shape: (time_steps, features)

    # Create sliding window view
    windows = np.lib.stride_tricks.sliding_window_view(data, (window_size, n))

    windows = windows.reshape(num_samples,window_size,n)

    # Apply stride by selecting every 'stride' step
    windows = windows[::stride]  # Shape: (num_samples, window_size, n)

    return windows
```

شکل ۷-۱: تابع sliding window

پس از انتخاب پنجره داده‌ها، ابعاد داده به طور مثال با انتخاب پنجره به اندازه ۵ و همپوشانی ۴، برای داده‌های آزمون از 365×24 به $361 \times 5 \times 24$ تغییر خواهند یافت. باید در نظر داشت که انتخاب ورودی (x) و خروجی (y)، باید به نحوی انجام شود که برای داده‌های ورودی L-1 نمونه آخر و برای داده‌های خروجی، L-1 نمونه اول حذف شوند.

```python
# sliding window data

x_train_window = sliding_window(df_train,5,1)
x_test_window = sliding_window(df_test,5,1)

y_train_window = df_train.iloc[4:,:].values
y_test_window = df_test.iloc[4:,:].values


print(np.shape(x_train_window))
print(np.shape(x_test_window))
print(np.shape(y_train_window))
print(np.shape(y_test_window))
```

✓ 0.0s

```
(3285, 5, 24)
(361, 5, 24)
(3285, 24)
(361, 24)
```

شکل ۸-۱: اعمال تابع sliding window بر روی دادگان

۱-۲- آموزش مدل

۱-۲-۱- مفهوم collaborative learning

مفهوم Collaborative Machine Learning (یادگیری ماشین مشارکتی) به مجموعه‌ای از تکنیک‌ها و روش‌ها اشاره دارد که در آن چندین نهاد (مانند شرکت‌ها، دستگاه‌ها، سازمان‌ها یا کاربران) بدون به اشتراک‌گذاری مستقیم داده‌های خام، به‌صورت جمعی مدل‌های یادگیری ماشین را آموزش می‌دهند. این رویکرد برای حفظ حریم خصوصی داده‌ها و افزایش دقت مدل‌ها از طریق همکاری طراحی شده است.

اهداف اصلی:Collaborative Machine Learning

۱. حفظ حریم خصوصی و امنیت داده‌ها
نهادها داده‌های حساس خود را نگه می‌دارند و فقط اطلاعاتی مانند گرادیان‌ها، مدل‌های محلی یا وزن‌ها را به اشتراک می‌گذارند.

۲. افزایش کارایی مدل‌ها با استفاده از داده‌های متنوع‌تر
با همکاری بین چند نهاد، مدل‌ها می‌توانند الگوهای دقیق‌تری را از داده‌های غیرمتمرکز یاد بگیرند.

۳. غلبه بر مشکل محدودیت داده‌ها در یک نهاد
وقتی داده‌های یک نهاد کافی یا متنوع نیست، همکاری باعث بهبود عملکرد می‌شود.

در این مقاله، از collaborative learning به این گونه استفاده شده است که از داده‌های چند شهر مختلف دیگر استفاده می‌شود تا شرایط آب و هوایی در یک مکان مشخص پیش‌بینی شود. به طور مثال برای پیش‌بینی شرایط آب و هوایی شهر Tours، از داده‌های ۲ شهر Montelimat و Perpignan نیز استفاده می‌شود.

۱-۳- آموزش مدل

در این بخش هر یک از دادگان تقسیم‌بندی شده به دو صورت single و sliding window را بر روی دو مدل linear regression و polynomial regression که به صورت scratch توسعه داده شده‌اند، آموزش می‌دهیم. **(بخش امتیازی: مدل برای هر ۳ شهر آموزش داده شده‌است)**

۱-۳-۱- رگرسیون خطی

پیاده‌سازی الگوریتم linear regression با استفاده از گرادیان نزولی به صورت زیر انجام می‌شود:

```python
def fit(self, X, y):
    # Initialize weights and bias:
    num_samples, num_features = X.shape
    _, num_labels = y.shape
    self.weights = np.zeros((num_features, num_labels))
    self.bias = np.zeros((num_labels))
    self.mse_history = []
    self.interrupt = False
    self.stopped_at_epoch = None

    # Create tqdm progress bar outside the loop
    progress_bar = tqdm(total=self.num_iter, desc="Epoch")

    # Gradient Descent:
    for epoch in range(self.num_iter):

        y_pred = np.dot(X, self.weights) + self.bias
        error = y_pred - y

        # Gradients
        gradients_weights = (1/num_samples) * np.dot(X.T, error)
        gradients_bias = (1/num_samples) * np.sum(error, axis=0)
        # Update Weights:
        self.weights -= self.learning_rate * gradients_weights
        self.bias -= self.learning_rate * gradients_bias

        # Calculate MSE for each feature
        mse_per_feature = np.mean((y_pred - y) ** 2, axis=0)
        average_mse = np.mean(mse_per_feature)
        self.mse_history.append(mse_per_feature)
```

شکل ۹-۱: تابع آموزش مدل رگرسیون خطی

همچنین از نوار پیشرفت در کتابخانه tqdm استفاده شده است تا روند پیشرفت الگوریتم قابل مشاهده باشد. قابل ذکر است که آستانه کمینه خطا برابر ۰.۰۱ در نظر گرفته شده است. به این معنا که اگر میزان میانگین MSE تمامی ویژگی‌ها از ۰.۰۱ کمتر شود، برنامه در epoch مربوطه متوقف می‌شود.

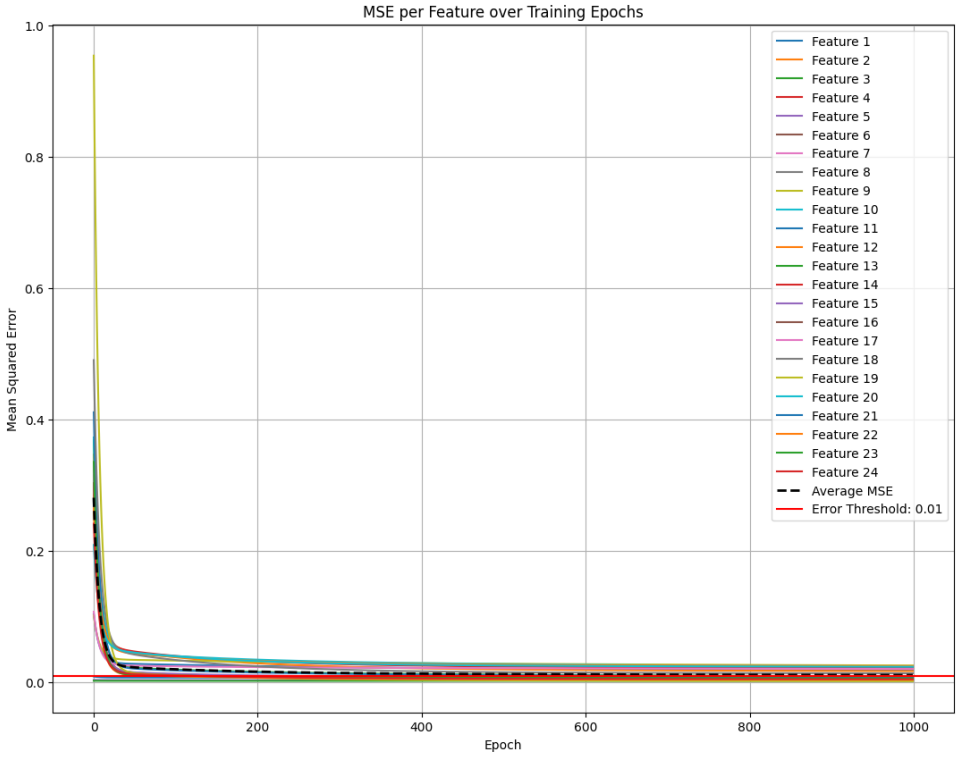
۱-۱-۳-۱- آموزش مدل رگرسیون خطی با استفاده از داده‌های تکی

مشاهده می‌شود که خطا در طول 1000 epochs، کمتر از ۰.۰۱ نشده است. بنابراین تمامی epochها تا انتها اجرا می‌شوند. مدت زمان اجرای برنامه ۲ ثانیه و مقدار نهایی میانگین MSE از تمامی ویژگی‌ها برابر ۰.۰۱۱ است.

```
Epoch: 100%|          | 1000/1000 [00:02<00:00, 396.53it/s, MSE=0.011235]

Training completed after 1000 epochs
Final average MSE: 0.011235
MSE for Feature 1: 0.018126
MSE for Feature 2: 0.017398
MSE for Feature 3: 0.009281
MSE for Feature 4: 0.022278
MSE for Feature 5: 0.003461
MSE for Feature 6: 0.003256
MSE for Feature 7: 0.005562
MSE for Feature 8: 0.003867
MSE for Feature 9: 0.025833
MSE for Feature 10: 0.024455
MSE for Feature 11: 0.008768
MSE for Feature 12: 0.024974
MSE for Feature 13: 0.002440
MSE for Feature 14: 0.004672
MSE for Feature 15: 0.006811
MSE for Feature 16: 0.005356
MSE for Feature 17: 0.020599
MSE for Feature 18: 0.013992
MSE for Feature 19: 0.002082
MSE for Feature 20: 0.023959
MSE for Feature 21: 0.007206
MSE for Feature 22: 0.004067
MSE for Feature 23: 0.006300
MSE for Feature 24: 0.004900
```

شکل ۱۰-۱: نتایج شبیه‌سازی رگرسیون خطی بر روی دادگان تکی

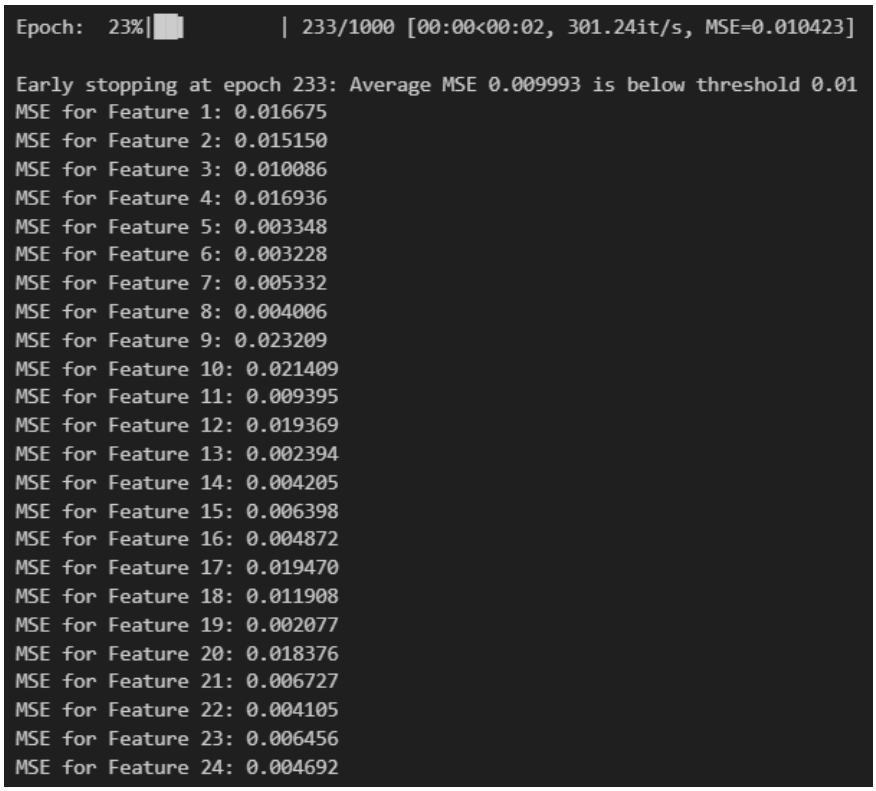
در نمودار میزان MSE ویژگی‌ها نیز دیده می‌شود که در طول اجرای epochها، مقدار خطا در طول اجرای برنامه، به مرور کم می‌شود تا همگی به نزدیکی صفر می‌رسند.



شکل ۱۱-۱: نمودار تغییرات میانگین خطای دادگان تکی در مدل رگرسیون خطی

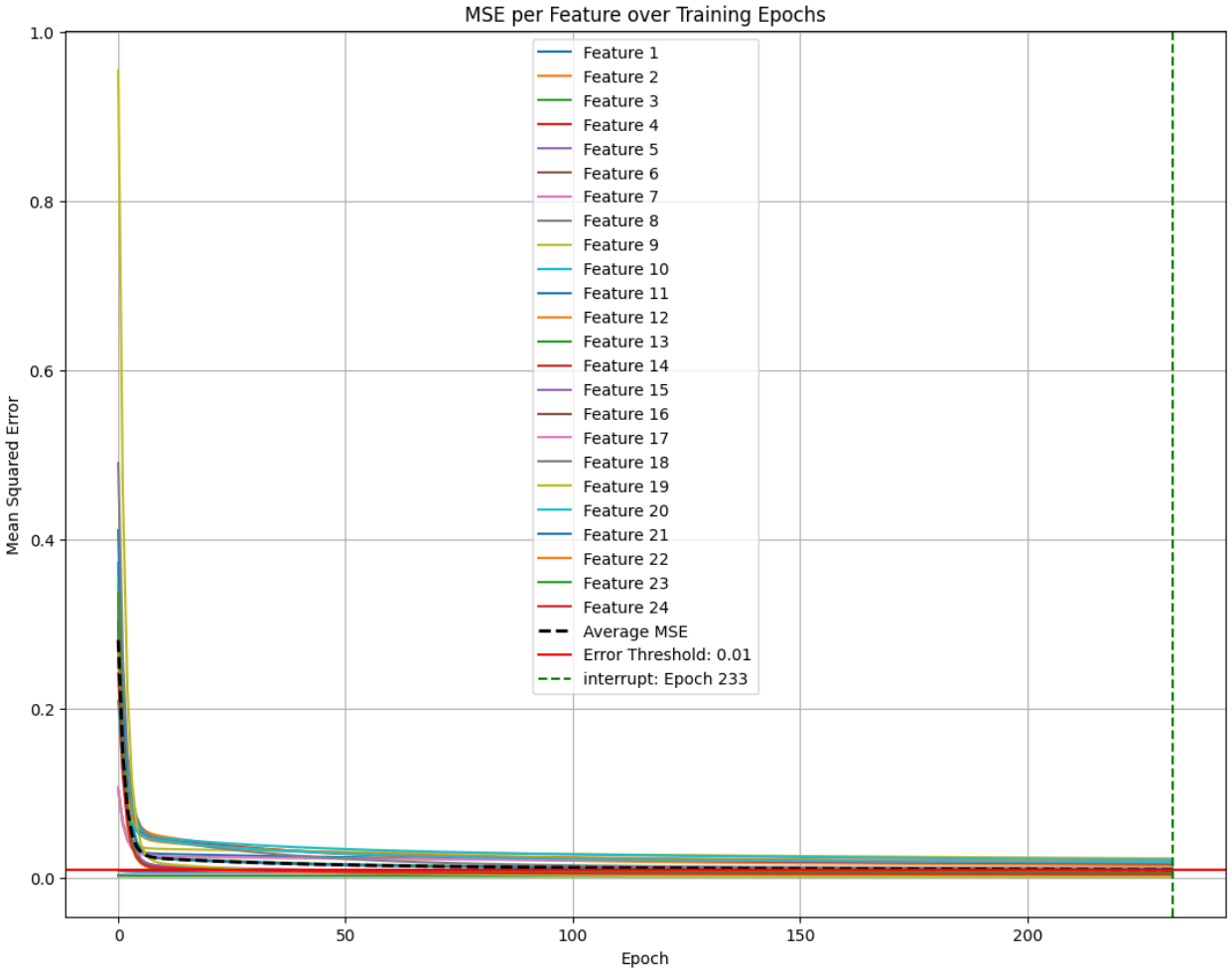
۲-۱-۳-۱- آموزش مدل رگرسیون خطی با استفاده از داده‌های پنجره

مشاهده می‌شود که خطا در epoch=234، کمتر از ۰.۰۱ نشده است. بنابراین برنامه فقط تا epoch=233 اجرا می‌شود. مدت زمان اجرای برنامه به دلیل توقف زودهنگام بسیار کم است و مقدار نهایی میانگین MSE از تمامی ویژگی‌ها برابر ۰.۰۱۰۴ است.



شکل ۱۲-۱: نتایج شبیه‌سازی رگرسیون خطی بر روی دادگان پنجره

در نمودار میزان MSE ویژگی‌ها نیز دیده می‌شود که در طول اجرای epochها، مقدار خطا در طول اجرای برنامه، به مرور کم می‌شود تا همگی به نزدیکی صفر می‌رسند. در نهایت برنامه در epoch=233 متوقف شده و مقدار خطا در این مرحله برابر مقدار خطای کلی در نظر گرفته می‌شود.



شکل ۱۳-۱: نمودار تغییرات میانگین خطای دادگان پنجره‌ای در مدل رگرسیون خطی

۲-۳-۱- رگرسیون چندجمله‌ای

رابطه تئوری استخراج ویژگی‌های polynomial به صورت زیر است.

$$Polynomial\ features = (\bar{x}_i \bar{x}_j^T + c)^d$$

که در آن $\bar{x}_i$ بیانگر ستون i ام از دیتاست، c بیانگر بایاس و d بیانگر درجه است.

پس از استخراج ویژگی‌ها کافی است دیتاست جدید را که با استفاده از داده‌های قبلی و ویژگی‌های جدید ساخته‌ایم به الگوریتم رگرسیون خطی به عنوان ورودی بدهیم. این ویژگی‌ها در برنامه به صورت زیر استخراج می‌شوند:

```python
def transform_polynomial_features(self, X):
    """Transforms the features into polynomial features up to the specified degree."""
    n_samples, n_features = X.shape
    X_poly = [np.ones(n_samples)]

    for d in range(1, self.degree + 1):
        for comb in combinations_with_replacement(range(n_features), d):
            feature = np.prod(X[:, comb], axis=1)
            X_poly.append(feature)

    return np.stack(X_poly, axis=1)
```

شکل ۱۴-۱: تابع استخراج ویژگی‌های چند جمله‌ای

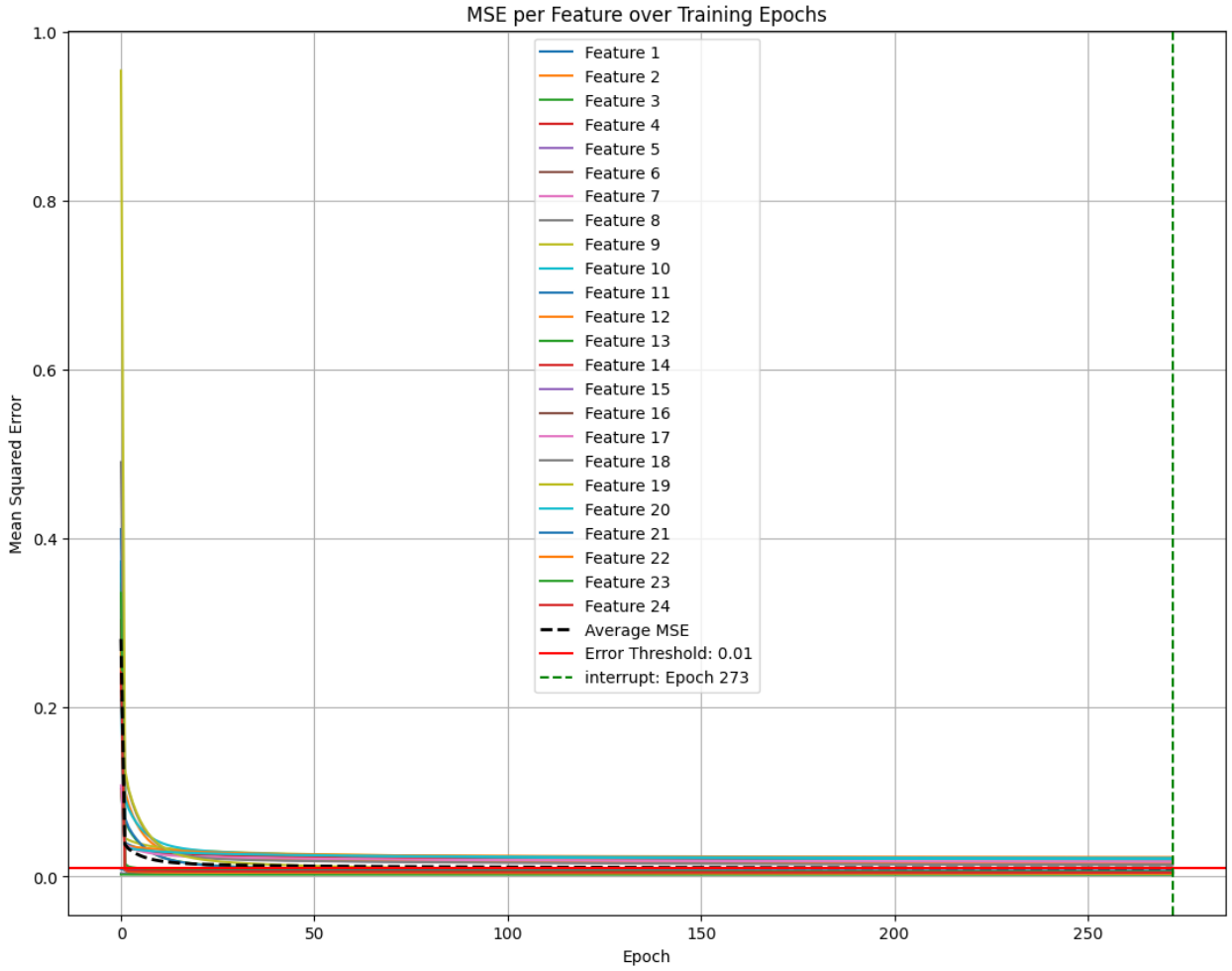
۱-۲-۳-۱- آموزش مدل رگرسیون چندجمله‌ای با استفاده از داده‌های تکی

مشاهده می‌شود که خطا در epoch=274، کمتر از ۰.۰۱ نشده است. بنابراین برنامه فقط تا epoch=273 اجرا می‌شود. این در حالتیست که یادگیری با دیتا single، 1000 epochs به طول می‌انجامید. بنابراین می‌توان نتیجه گرفت که با استخراج این ویژگی‌ها، تفسیرپذیری بالا میرود. هر چند مدت اجرای برنامه به دلیل استخراج ویژگی‌های چند جمله‌ای بیشتر شده است. مدت زمان اجرای برنامه ۴ ثانیه است و مقدار نهایی میانگین MSE از تمامی ویژگی‌ها برابر ۰.۰۱۰۳۲۹ است.

```
Epoch:  27%|██▋       | 273/1000 [00:04<00:11, 65.78it/s, MSE=0.010329]

Early stopping at epoch 273: Average MSE 0.009999 is below threshold 0.01
MSE for Feature 1: 0.015563
MSE for Feature 2: 0.016275
MSE for Feature 3: 0.005670
MSE for Feature 4: 0.020322
MSE for Feature 5: 0.003329
MSE for Feature 6: 0.003225
MSE for Feature 7: 0.004833
MSE for Feature 8: 0.004019
MSE for Feature 9: 0.021825
MSE for Feature 10: 0.020429
MSE for Feature 11: 0.005731
MSE for Feature 12: 0.022938
MSE for Feature 13: 0.002384
MSE for Feature 14: 0.004167
MSE for Feature 15: 0.005962
MSE for Feature 16: 0.005123
MSE for Feature 17: 0.017534
MSE for Feature 18: 0.013668
MSE for Feature 19: 0.004033
MSE for Feature 20: 0.021935
MSE for Feature 21: 0.007020
MSE for Feature 22: 0.003731
MSE for Feature 23: 0.005552
MSE for Feature 24: 0.004700
```

شکل ۱۵-۱: نتایج شبیه‌سازی رگرسیون چند جمله‌ای بر روی دادگان تکی

در نمودار میزان MSE ویژگی‌ها نیز دیده می‌شود که در طول اجرای epochها، مقدار خطا در طول اجرای برنامه، به مرور کم می‌شود تا همگی به نزدیکی صفر می‌رسند. در نهایت برنامه در epoch=273 متوقف شده که نسبت به حالت رگرسیون خطی، در epoch کمتری اتفاق می‌افتد. مقدار خطا در این مرحله برابر مقدار خطای کلی در نظر گرفته می‌شود.



شکل ۱۶-۱: نمودار تغییرات میانگین خطای دادگان تکی در مدل رگرسیون چندجمله‌ای

۳-۳-۱- آموزش با مدل‌های سایکیت لرن

توضیح Ridge

رگرسیون Ridge یکی از انواع رگرسیون خطی منظم‌شده (Regularization) است که هدفش کاهش بیش‌برازش (Overfitting) و مقابله با هم‌خطی (Multicollinearity) در داده‌هاست.

هدف کلی در رگرسیون خطی معمولی، کمینه کردن مجموع مربعات خطاهاست:

$$\min_{\beta} \|y - X\beta\|_2^2$$

- $X \in R^{n \times p}$: design matrix (features)
- $y \in R^{n}$: target vector
- $\beta \in R^{p}$: regression coefficients

اضافه کردن Regularization:

Ridge Regression (L2 penalty):

$$\min_{\beta} \|y - X\beta\|_2^2 + \lambda\|\beta\|_2^2$$

هدف رگرسیون Ridge:

- جلوگیری از بزرگ شدن ضرایب مدل
- پایدار کردن مدل در مواجهه با ویژگی‌های هم‌وابسته
- کاهش واریانس مدل، حتی به قیمت افزایش کمی بایاس(Bias-Variance Tradeoff)

ویژگی‌ها:

- هرچه λ بزرگ‌تر شود، ضرایب β به صفر نزدیک‌تر می‌شوند (ولی دقیقاً صفر نمی‌شوند).
- هیچ ویژگی‌ای را حذف نمی‌کند برخلاف Lasso که می‌تواند بعضی ضرایب را صفر کند.
- مناسب برای داده‌هایی با ویژگی‌های زیاد و هم‌خطی بالا

توضیح Elastic net

رگرسیون Elastic Net یکی از روش‌های پیشرفته‌ی رگرسیون خطی است که برای حل مشکلاتی مثل هم‌خطی (Multicollinearity) و انتخاب ویژگی‌ها (Feature Selection) در داده‌ها کاربرد دارد. این روش ترکیبی از دو تکنیک منظم‌سازی Lasso (L1) و Ridge (L2) است.

هدف کلی در رگرسیون خطی معمولی، کمینه کردن مجموع مربعات خطاهاست:

$$\min_{\beta} \|y - X\beta\|_2^2$$

- $X \in R^{n\times p}$: design matrix (features)
- $y \in R^{n}$: target vector
- $\beta \in R^{p}$: regression coefficients

اضافه کردن Regularization:

Lasso Regression (L1 penalty):

$$\min_{\beta} \|y - X\beta\|_2^2 + \lambda\|\beta\|_1$$

Ridge Regression (L2 penalty):

$$\min_{\beta} \|y - X\beta\|_2^2 + \lambda\|\beta\|_2^2$$

در Elastic net از ترکیب دو ترم L1 و L2 استفاده می‌شود:

$$\min_{\beta} \|y - X\beta\|_2^2 + \lambda_1\|\beta\|_1 + \lambda_2\|\beta\|_2^2$$

مزایایElastic Net

- انتخاب ویژگی‌ها (Lasso)
- مقابله با هم‌خطی (Ridge)
- پایداری عددی بهتر در مسائل با داده‌های هم‌وابسته
- مناسب برای حالاتی که تعداد ویژگی‌ها بیشتر از تعداد نمونه‌ها باشد (High-dimensional Data)

- Lasso تمایل دارد بعضی ضرایب را صفر کند ⟶ انتخاب ویژگی
- Ridge ضرایب را کوچک می‌کند اما صفر نمی‌کند
- Elastic Net می‌تواند هم ضرایب را کوچک کند و هم برخی را صفر

شبیه‌سازی مدل‌های آماده sklearn:

```
models = [
    ("Linear", LinearRegression()),
    #("Logitic", LogisticRegression()),
    #("Quantile", QuantileRegressor()),
    ("Elasticnet", ElasticNet()),
    ("Ridge", Ridge()),


]


for name, model in models:
    model.fit(x_train_single, y_train_single)
    y_pred = model.predict(x_test_single)

    print(f'MSE of {name} Model: {mean_squared_error(y_test_single, y_pred)}\n')
```

شکل ۱۷-۱: توابع رگرسیون sklearn

نتایج شبیه‌سازی برای داده single:

```
MSE of Linear Model: 0.009351189302340359

MSE of Elasticnet Model: 0.034757973018077114

MSE of Ridge Model: 0.009377923047410757
```

شکل ۱۸-۱: نتایج شبیه‌سازی توابع sklearn بر روی دادگان تکی

نتایج شبیه‌سازی برای داده sliding window:

```
MSE of Linear Model: 4.332239094153011e-30

MSE of Elasticnet Model: 0.03456843681604208

MSE of Ridge Model: 1.7521803386689697e-05
```

شکل ۱۹-۱: نتایج شبیه‌سازی توابع sklearn بر روی دادگان پنجره

مقدار خطا در رگرسیون خطی بدون regularization در حالت sliding window بیسار کمتر از حالت single است. اما با اضافه کردن ترم‌های regularization، اختلاف مقادیر خطا در این دو حالت از داده‌ها بسیار کمتر می‌شود. به طوری که در مدل Elastic net که از هر دو مدل lasso و Ridge استفاده می‌کند.

اختلاف مقدار خطا در دو حالت تکی و پنجره‌ای تقریبا برابر صفر است و مقدار خطا در این دو حالت تقریبا یکسان هستند. علت این پدیده می‌تواند در اضافه کردن مقدار regularization باشد، به طوری که این مقادیر از overfit شدن مدل جلوگیری می‌کنند.

۴-۳-۱- امتیازی

سوال امتیازی اول در کل شبیه‌سازی انجام شده است. این بخش مربوط به سوال امتیازی دوم است.

ویژگی‌های مشترک دو شهر Basel و Budapest رو بدست می‌آوریم:

دیتاست مورد نظر برای شهر Basel به شرح زیر است:

	cloud_cover	humidity	pressure	global_radiation	precipitation	sunshine	temp_mean	temp_max
0	4.0	0.84	1.0284	0.63	0.00	7.1	1.5	3.6
1	8.0	0.89	1.0262	0.07	0.00	0.0	-1.0	1.0
2	5.0	0.72	1.0263	0.50	0.00	2.9	-3.1	-1.8
3	4.0	0.75	1.0239	0.62	0.00	5.4	-3.7	-1.3
4	8.0	0.84	1.0197	0.28	0.05	0.0	-4.1	-2.5
...	...	...	...	...	...	...	...	...
360	7.0	0.80	1.0118	0.37	0.18	2.3	1.0	5.5
361	7.0	0.82	1.0084	0.28	0.42	0.3	3.2	4.8
362	7.0	0.92	1.0028	0.22	1.68	0.2	4.5	10.0
363	8.0	0.92	0.9979	0.07	1.54	0.0	8.5	11.5
364	7.0	0.93	0.9958	0.17	0.57	0.1	6.6	7.9

365 rows × 8 columns

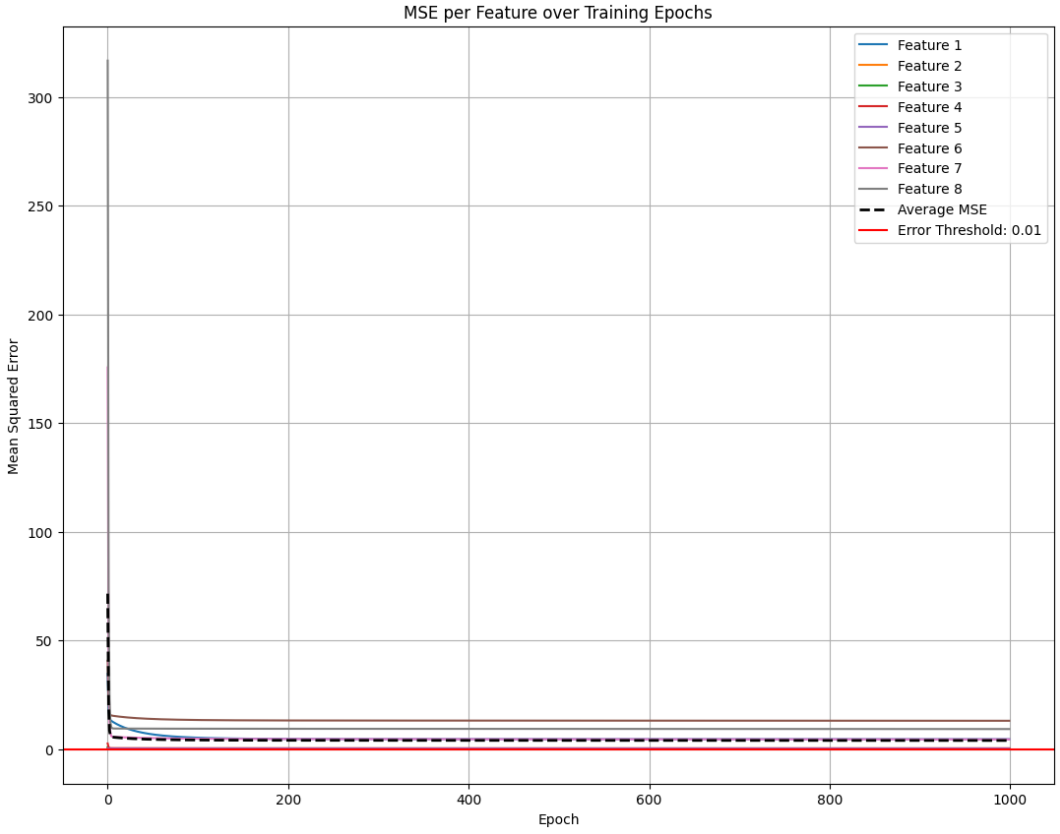
شکل ۲۰-۱: دادگان شهر Basel

سپس مدل linear regression را بر روی این دیتاست آموزش داده و همچنین آزمایش می‌کنیم. نتایج این آموزش به شرح زیر است:

حالت single:

```
Epoch: 100%|██████████| 1000/1000 [00:00<00:00, 4682.03it/s, MSE=4.029308]

Training completed after 1000 epochs
Final average MSE: 4.029308
MSE for Feature 1: 4.594217
MSE for Feature 2: 0.022836
MSE for Feature 3: 0.021174
MSE for Feature 4: 0.425759
MSE for Feature 5: 0.283383
MSE for Feature 6: 13.080003
MSE for Feature 7: 4.534151
MSE for Feature 8: 9.272942
```

شکل ۲۱-۱: نتایج شبیه‌سازی رگرسیون خطی بر روی دادگان تکی شهر Basel

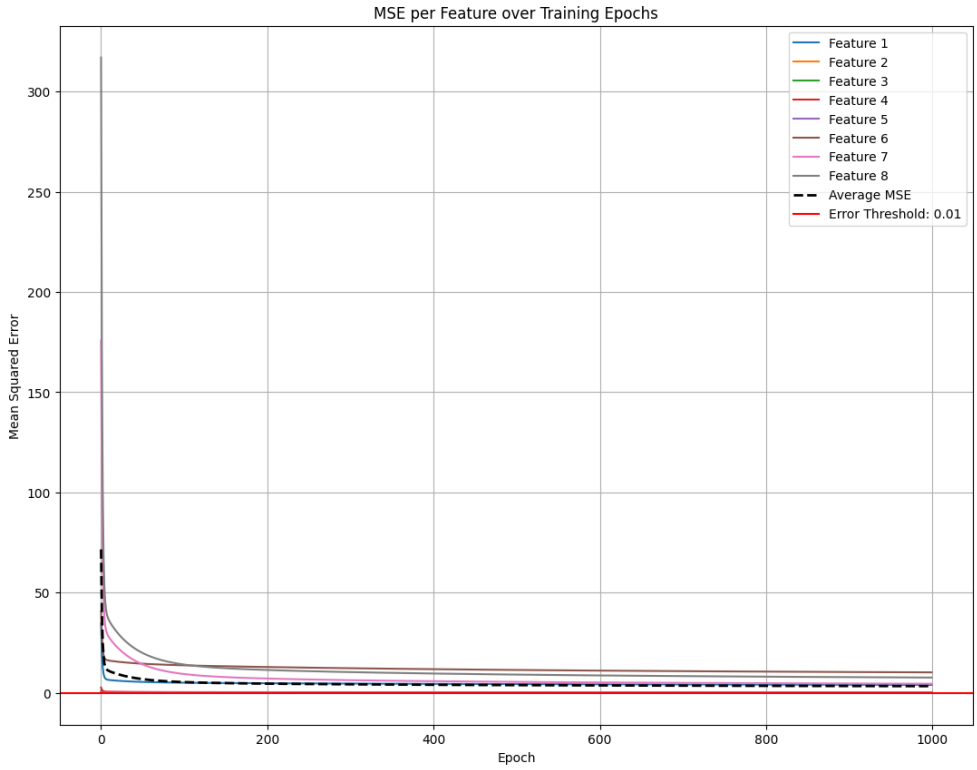


شکل ۲۲-۱: نمودار تغییرات میانگین MSE بر روی دادگان تکی شهر Basel

حالت sliding window:

```
Epoch: 100%|██████████| 1000/1000 [00:00<00:00, 2281.31it/s, MSE=3.364122]

Training completed after 1000 epochs
Final average MSE: 3.364122
MSE for Feature 1: 3.982700
MSE for Feature 2: 0.006791
MSE for Feature 3: 0.001003
MSE for Feature 4: 0.236201
MSE for Feature 5: 0.258813
MSE for Feature 6: 10.236085
MSE for Feature 7: 4.543285
MSE for Feature 8: 7.648096
```

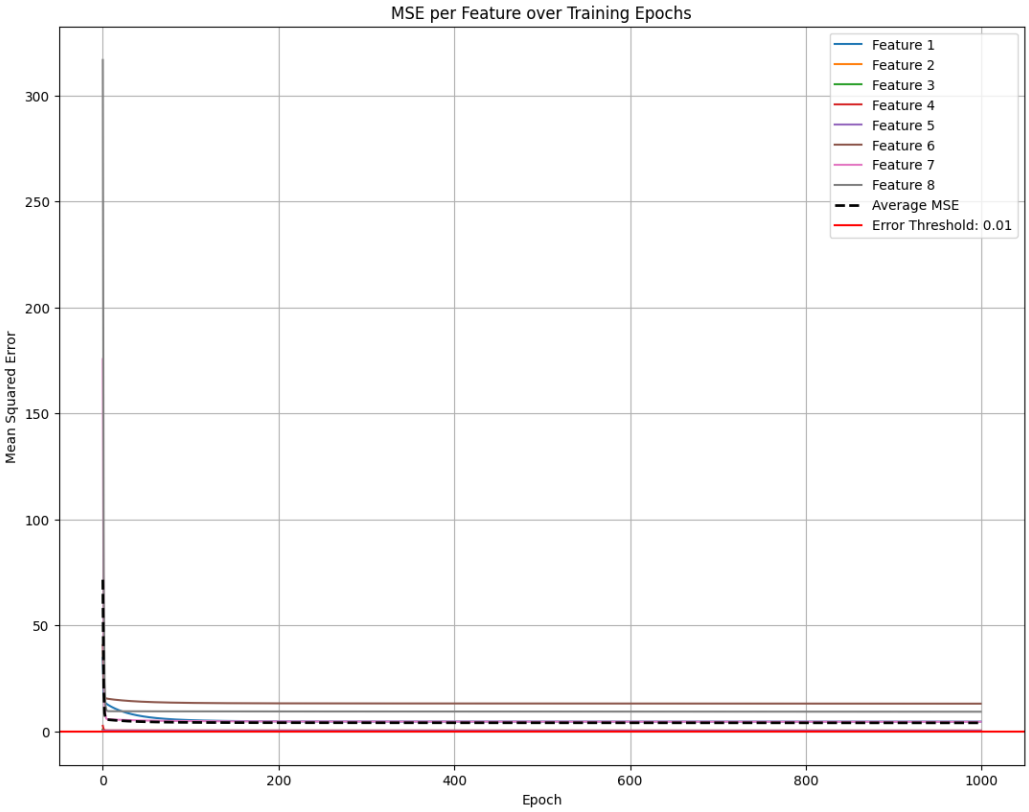
شکل ۲۳-۱: نتایج شبیه‌سازی رگرسیون خطی بر روی دادگان پنجره شهر Basel



شکل ۲۴-۱: نمودار تغییرات میانگین MSE بر روی دادگان پنجره شهر Basel

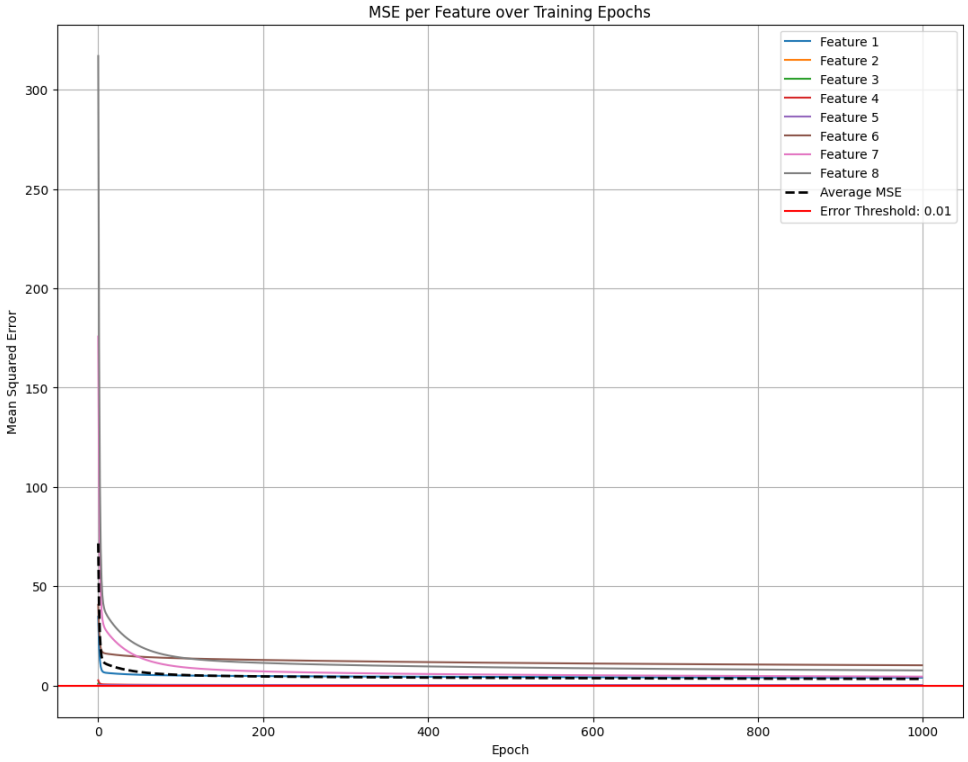
اکنون مدل‌های آموزش داده شده را، تنها روی داده‌های شهر Budapest، آزمایش می‌کنیم:

حالت single:



شکل ۲۵-۱: نتایج شبیه‌سازی رگرسیون خطی بر روی دادگان پنجره شهر Budapest

حالت sliding window:



شکل ۲۶-۱: نمودار تغییرات میانگین MSE بر روی دادگان پنجره شهر Budapest

مشاهده می‌شود که مدل‌هایی که با دیتاهای شهر Basel آموزش دیده‌اند، بر روی شهر Budapest نیز به خوبی جواب می‌دهند و مقدار MSE را تا حد خوبی کاهش می‌دهند.

فصل ۲– یاتاقان

۱-۲- ۲) تشخیص عیب یاتاقان غلتشی بر مبنای دسته بندی های سلسله مراتبی

۲.۱) دادگان

۱.۱.۲)

این مجموعه شامل حسگرهایی مانند شتاب‌سنج‌ها که برای اندازه‌گیری ارتعاشات به کار می‌روند و میکروفون که به منظور ثبت و تحلیل صدای تولید شده توسط سیستم استفاده می‌شوند، می‌باشد. علاوه بر این، حسگر سرعت چرخش نیز برای اندازه‌گیری دقیق نرخ چرخش شفت به کار گرفته می‌شود.

دادگان با نرخ نمونه‌برداری ۵۰ کیلوهرتز در طول مدت ثانیه جمع‌آوری می‌شوند که منجر به ۲۵۰۰،۰۰۰ نمونه برای هر آزمایش می‌شود. این داده‌ها با وضوح بالا امکان تحلیل دقیق فشارهای گذرا و پایدار ماشین‌ها تحت شرایط مختلف عملی را فراهم می‌کند. مجموعه داده ذخیره شده شامل چندین نوع اندازه‌گیری است:

سرعت‌سنج :با استفاده از سرعت‌سنج لیزری Monarch Instrument MT-190 سرعت چرخش روتور اندازه‌گیری شده و در ستون اول فایل‌های CSV ذخیره می‌شود.

شتاب‌سنج :مجموعه داده شامل داده‌های سه شتاب‌سنج مدل IMI Sensors 601A01 است که ارتعاش در جهت‌های شعاعی، محوری و مماسی بر روی یاتاقان Underhang اندازه‌گیری می‌کند (در ستون‌های ۲ تا ۴ ذخیره شده‌اند). علاوه بر این، یک شتاب‌سنج سه‌محوره مدل IMI Sensors 604B31 مشابهی را بر روی یاتاقان Overhang ثبت می‌کند (در ستون‌های ۵ تا ۷ ذخیره شده‌اند).

میکروفون :صداهای عملیاتی با استفاده از میکروفون Shure SM81 ضبط می‌شوند که سیگنال‌های صوتی مربوط به شرایط مختلف عیب را ارائه می‌دهد. این داده‌ها درستون ۸ فایل‌های CSV ذخیره شده‌اند.

۲.۱.۲)

مجموعه داده MaFaulDa طیف گسترده‌ای از عیب‌های مکانیکی را پوشش می‌دهد که هر یک تحت شرایط کنترل‌شده‌ای به منظور شبیه‌سازی سناریوهای واقعی معرفی شده‌اند. این عیب‌ها شامل موارد زیر است:

ناهماهنگی: هر دو نوع عیب ناهماهنگی افقی و عمودی با سطوح شدت مختلف معرفی شده‌اند، که شامل ۰.۵۰، میلی‌متر تا ۲.۰۰ میلی‌متر برای عیب ناهماهنگی افقی و از ۰.۵۱، میلی‌متر تا ۱.۹۰ میلی‌متر برای عیب ناهماهنگی عمودی می‌شود.

عدم تعادل: سطوح شدت مختلفی از عدم تعادل با اضافه کردن وزن‌هایی از ۶ گرم تا ۳۵ گرم به روتور شبیه‌سازی شد. عدم تعادل یک عیب معمولی است که ناشی از توزیع نابرابر جرم در اطراف محور روتور است.

خطاهای یاتاقان: سه نوع عیب یاتاقان معرفی شده که شامل موارد زیر است:

عیب قفس: بر روی هر دو یاتاقان Overhang وUnderhang ، با اعمال وزن‌های مختلف از ۶ گرم تا ۳۵ گرم شبیه‌سازی شد.

عیب بیرونی :با محدوده وزن‌های ۶ گرم تا ۳۵ گرم بر روی هر دو یاتاقان ایجاد شد.

عیب توپ :همانند عیوب دیگر یاتاقان، با سطوح مختلف شدت از ۶ گرم تا ۳۵ گرم، این بار با ایجاد عیب در عنصر غلطکی یاتاقان‌ها ایجاد شد.

هر یک از این عیب‌ها تحت شرایط عملیاتی مختلف، در سرعت‌های گوناگون بین ۷۳۷ تا ۳۸۸۶ دور در دقیقه با گام‌های تقریباً ۶۰ تایی آزمایش شدند. این آزمایش‌ها منجر به ایجاد حدود ۵۰ نمونه آزمایشی برای هر عیب شد. البته در برخی موارد به دلیل از کنترل خارج شدن ارتعاشات، تعداد نمونه‌ها کمتر است. در نتیجه مجموعه داده کاملی که منعکس‌کننده طیف گسترده‌ای از خرابی‌های دنیای واقعی است، ارائه شد. جدول ۲-۱ اطلاعات دقیقی از انواع عیب‌ها، شدت‌ها و تعداد اندازه‌گیری‌های مربوط به هر عیب را ارائه می‌دهد. هر نمونه آزمایشی دارای ۸ ستون و ۲۵۰۰۰۰ سطر می‌باشد.

جدول1 -جزئیات داده های مربوط به هر کلاس عیب

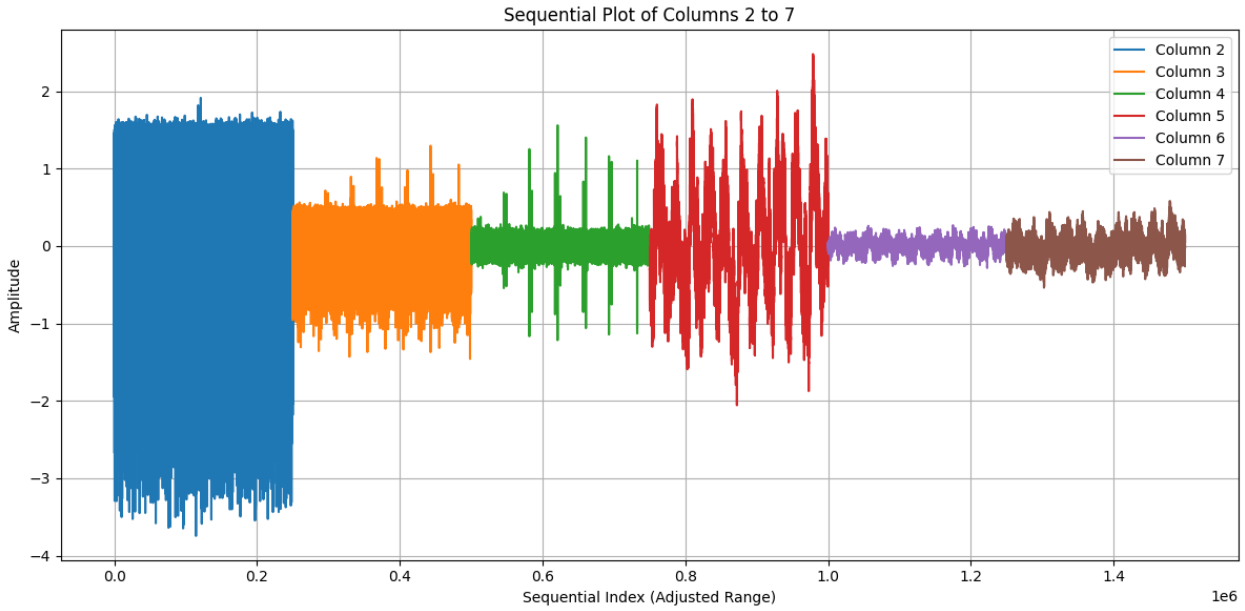
Sequence	Measurements	Weight or Misalignment values	Unit
Normal	49	-	-
Horizontal Misalignment	197	(0.50, 1.00, 1.5, 2.00)	mm
Vertical Misalignment	301	(0.51, 0.63, 1.27, 1.40, 1.78, 1.90)	mm
Imbalance	333	(6, 10, 15, 20, 25, 30, 35)	g
Underhang Bearing			
Cage fault	188	(0, 6, 20, 35)	g
Outer Race	184	(0, 6, 20, 35)	g
Ball fault	186	(0, 6, 20, 35)	g
Overhang Bearing			
Cage fault	188	(0, 6, 20, 35)	g
Outer Race	188	(0, 6, 20, 35)	g
Ball fault	137	(0, 6, 20, 35)	g
Total	1951	42	-

(۳.۱.۲

در این بخش یک فایل از هر کلاس عیب به اضافه یک دیتا نرمال که مجموعا ۱۰ فایل خواهد شد دانلود می کنیم اسم فایل های دیتا دانلود شده از هرکلاس عیب به شرح زیر می باشد:

Horizontal-misalignment (1mm) : 14.336.csv
vertical-misalignment (1.4mm) : 14.7456.csv
imbalance(35g) : 14.5408.csv
overhang-ballfault (6g) : 15.36.csv
overhang-cagefault (6g) :14.9504.csv
overhang-outerrace (6g) :15.5648.csv
underhang-ballfault (20g) : 16.1792.csv
underhang-cagefault (20g) :19.456.csv
underhang-outerrace (20g) :20.2752.csv
normal : 14.336.csv

همانطور که در منبع پایان نامه اشاره شده است با توجه به تمرکز اصلی برروی ارتعاشات، از رسم ستون های اول و آخر (هشتم) که مربوط به سرعت سنج و میکروفون می باشد صرف نظر می کنیم و یکی از داده ها برای مثال Horizontal-misalignment را رسم میکنیم.



شکل ۲۷_۲ داده های کلاس horizontal misalignment به تفکیک سنسور

در این نمودار تنها داده های ثبت شده به وسیله شتاب سنج ها مشاهده می شوند که سه تای اول (آبی، نارنجی و سبز) مربوط شتاب سنج های نصب شده برروی یاتاقان underhang بوده و ارتعاشات را به ترتیب در سه جهت شعاعی، محوری و مماسی ظبط کرده اند و سه تای آخر (قرمز، بنفش و قهوه ای) مربوط شتاب سنج سه محوره نصب شده برروی یاتاقان overhang می باشد.

همانطور که در پایان نامه اشاره شده است، در این داده نیز دامنه ارتعاشات محور شعاعی یاتاقان overhang (بنفش رنگ)کوچک بوده و میتوان از آن صرف نظر کرد تا ضمن کاهش محاسبات، تمرکز اصلی را بر روی سیگنال ها با دامنه موثرتر قرار دهیم.

حال داده های هر کلاس را در یک دیتا فریم مجتمع کرده و یک ستون به این دیتافریم اضافه می کنیم که نشان دهنده کلاس هر داده می باشد. چند سطر از این دیتافریم در تصویر زیر قابل مشاهده می باشد. این دیتافریم ۲.۵ میلیون سطر و ۶ ستون خواهد داشت که ۵ ستون اول مربوط به داده های شتاب سنج

های نصب شده برروی یاتاقان های overhang و underhang (به جز ارتعاشات محور شعاعی یاتاقان overhang) می باشد و ستون آخر نیز مربوط به کلاس داده می باشد.

```
Final data shape: (2500000, 6)
          1         2         3         4         6   label
0 -1.70450 -0.076488 -0.051497 -0.56721 -0.031897  normal
1  1.72010  0.273950  0.021210 -0.45291  0.069768  normal
2 -1.60530 -0.253730 -0.076687 -0.57488 -0.049242  normal
3  1.28430  0.372430  0.055843 -0.45892  0.054420  normal
4 -0.84995 -0.234240 -0.060778 -0.55874 -0.035808  normal
```

شکل ۲۸_۲ دیتافریم داده خام به همراه لیبل

۲-۲- ۲.۲) پیش پردازش و استخراج ویژگی

(۱.۲.۲

همانطور که پیش تر اشاره شد دیتافریم دادگان ما شامل ۶ ستون و ۲.۵ میلیون سطر نمونه از جنس سری زمانی می باشند که مربوط به دادگان از کلاس های مختلف می باشد با انتخاب پنجره زمانی به اندازه ۵۰۰۰ نمونه و اعمال به دیتافریم دادگان خام، عملیات استخراج ویژگی را بر روی هر یک از این پنجره ها اعمال میکنیم که در ادامه ویژگی های استخراج شده در حوزه زمان و فرکانس توضیح داده شده اند، سپس به تقسیم این دیتافریم به دو دیتافریم آموزش و تست با نسبت ۸۰ به ۲۰ درصد میپردازیم و مطمئن می شویم که دادگان پخش شده در هر دو دیتافریم متعادل باشد یعنی به یک میزان داده از هر کلاس در دو دیتافریم قرار گیرد.

در مرحله آخر نیز عملیات استاندارد سازی دادگان را به همان نحوی که در پایان نامه آمده است با تابع standardscaler بر روی داده آموزش فیت و برروی داده تست، اعمال میکنیم. اکنون دادگان تهیه شده آماده آموزش کلاس بند و تست آن می باشند.

(۲.۲.۲

ویژگی های حوزه زمان :

ریشه میانگین مربعات (RMS) :

یک معیار آماری مهم که بزرگی کلی یا محتوای انرژی یک سیگنال ارتعاشی را محاسبه می کند و نشان دهنده قدرت سیگنال است. و از رابطه زیر محاسبه می شود :

$$\mathrm{RMS} = \sqrt{\frac{1}{\mathrm{N}}\sum_{\mathrm{i}=1}^{\mathrm{n}} \mathrm{x_i}^2}$$

اوج :

مقدار اوج به حداکثر مقدار مطلق دامنه سیگنال در یک بازه زمانی مشخص اشاره دارد و برای شناسایی ناهنجاری های ناگهانی در سیستم استفاده می شود.

$$peak = max(|x_i|)$$

ضریب تاج :

نسبت مقدار اوج سیگنال به RMS می باشد که به عنوان معیاری برای اندازه گیری شدت و تیزی اوج های سیگنال مورد استفاده قرار می گیرد.

$$crest\ factor = \frac{peak}{RMS}$$

انحراف معیار :

یک معیار آماری است که میزان پراکندگی یا تغییرات دامنه سیگنال از مقدار میانگین آن را نشان می دهد. اگر میانگین را با μ نمایش دهیم انحراف معیار از رابطه زیر قابل محاسبه است.

$$\sigma = \sqrt{\frac{1}{\mathrm{N}-1}\sum_{\mathrm{i}=1}^{\mathrm{n}} (\mathrm{x_i} - \mu)^2}$$

چولگی :

این معیار آماری عدم تقارن توزیع دامنه حول میانگین آن را نشان میدهد و می تواند به وقایع ضربه ای و شوک ها در سیستم اشاره کند.

$$skewness = \frac{N\sum_{\mathrm{i}=1}^{\mathrm{n}}(\mathrm{x_i} - \mu)^3}{(N-1)(N-2)\sigma^3}$$

هم چنین سایر ویژگی های استخراج شده از حوزه زمان عبارتند از :

میانگین : اطلاعات مرکزی سیگنال

کشیدگی (Kurtosis) : حساس به پیک‌های بالا در سیگنال

نرخ عبور از صفر(Zero Crossing Rate) : نشان‌دهنده فعالیت سیگنال در بازه زمانی

(**Entropy**) : مقدار بی‌نظمی در داده ها

ویژگی های حوزه فرکانس :

توان باند :

این پارامتر به مجمع توان سیگنال در یک بازه مشخص از فرکانس ها اشاره دارد و از طریق رابطه زیر محاسبه می شود.

$$P_{band} = \sum_{f_1}^{f_2} |X(f)|^2$$

فرکانس های تشدید :

به فرکانس هایی گفته می شود که در آن ها یک سیستم به طور طبیعی تمایل به ارتعاش با دامنه بزرگتری دارد که به نسبت سختی (k) و جرم (m) سیستم وابسته است.

$$f_r = \frac{1}{2\pi}\sqrt{\frac{k}{m}}$$

پهنای باند فرکانسی :

به محدوده ای از فرکانس ها گفته می شود که در آن سیگنال یا سیستم به طور موثری انرژی را منتقل می کند یا پاسخ قابل توجهی نشان می دهد و در آن سیگنال به یک مقدار معینی از دامنه یا توان خود می رسد (معمولا ۳ دسی بل کمتر از حداکثر).

$$bandwidth = f_2 - f_1$$

همچنین سایر ویژگی های حوزه فرکانس عبارتند از :

Spectral Centroid : مرکز جرم طیف، نشان‌دهنده جرم فرکانسی سیگنال

Spectral Flatness : میزان یکنواختی طیف (آیا طیف تونال است یا نویزی)

Spectral Skewness : قرینگی طیف

Spectral Rolloff : فرکانسی که بعد از آن ٪۸۵ انرژی طیف پایان می‌پذیرد

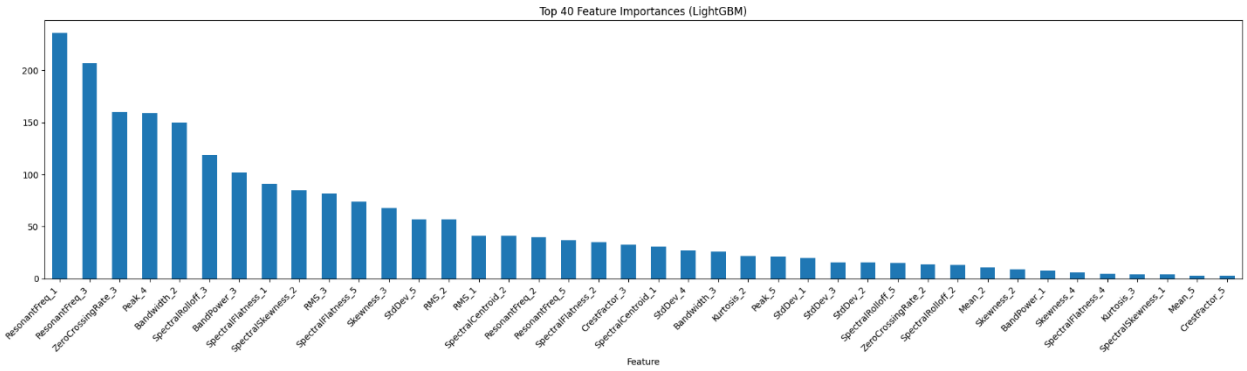
۳-۲- بخش امتیازی LightGBM , SI

با توجه به بخش ۳-۲-۳ پایان نامه LightGBM یک الگوریتم درخت تصمیم تقویتی هست که در حین آموزش مدل، اهمیت هر ویژگی را بر اساس کاهش خطا (Gain) در هر تقسیم (split) محاسبه می‌کند. ویژگی‌هایی که بیشتر در تقسیم‌های مؤثر استفاده می‌شوند، اهمیت بالاتری دارند.

در حقیقت الگوریتم LightGBM با استفاده از مجموعه‌ای از درخت‌های ضعیف(weak learners) ، یک مدل پیش‌بینی قوی ایجاد می‌کند. یکی از مزایای مهم این الگوریتم، توانایی آن در ارزیابی و رتبه‌بندی اهمیت ویژگی‌ها (Feature Importance) به صورت ضمنی و هم‌زمان با فرایند آموزش مدل است.

در LightGBM ، اهمیت یک ویژگی بر اساس میزان مشارکت آن در تقسیم‌های مؤثر در درخت‌های تصمیم محاسبه می‌شود. در هر گام از ساخت درخت، الگوریتم برای هر گره، تمام ویژگی‌های موجود را ارزیابی کرده و آستانه‌های مختلف برای تقسیم داده‌ها را بررسی می‌کند. در نهایت، آن ویژگی و آستانه‌ای انتخاب می‌شود که بیشترین سود (Gain) را در کاهش تابع خطا ایجاد نماید.

۴۰ ویژگی با بیشترین امتیاز بر اساس این معیار در نمودار زیر مشخص می باشد. توجه شود در این رتبه بندی از نمونه های دیتافریم آموزش استفاده شده است.



شکل ۲_۲۹ تعداد ۴۰ ویژگی برتر بر اساس معیار lightGBM

شاخص SI (Separation Index) یکی از معیارهای آماری برای سنجش توان تفکیک‌پذیری یک ویژگی نسبت به کلاس‌های مختلف در یک مسئله دسته‌بندی است. این شاخص مشخص می‌کند که تا چه میزان یک ویژگی قادر است داده‌های متعلق به کلاس‌های مختلف را از یکدیگر جدا سازد و شامل مراحل زیر است.

۱- محاسبه فاصله اقلیدسی **:** برای هر نمونه در داده، فاصله‌ی اقلیدسی آن تا سایر نمونه‌ها محاسبه می‌شود .

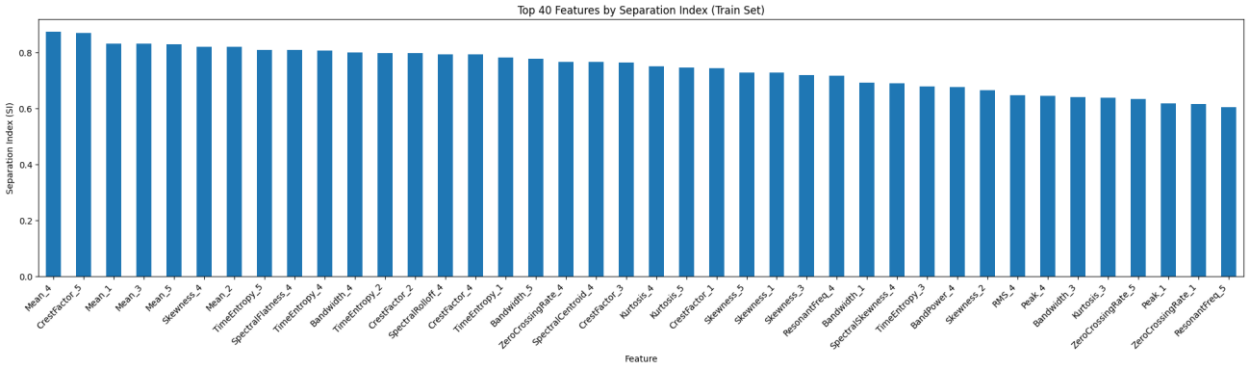
۲- یافتن نزدیک‌ترین همسایه **:** نزدیک‌ترین نمونه (همسایه) در فضای ویژگی‌ها که با خود نمونه یکسان نیست، تعیین می‌شود.

۳- مقایسه برچسب کلاس : بررسی می‌شود که آیا کلاس نمونه با کلاس نزدیک‌ترین همسایه یکسان است یا خیر.

۴- میانگین‌گیری : مقدار SI نهایی، میانگین تعداد دفعاتی است که نزدیک‌ترین همسایه متعلق به کلاس متفاوت است.

SI نزدیک به ۱ نشان‌دهنده این است که ویژگی مورد نظر به خوبی کلاس‌ها را از یکدیگر جدا می‌کند (زیرا بیشتر نزدیک‌ترین همسایه‌ها از کلاس دیگر هستند) و SI نزدیک به ۰ بیانگر آن است که ویژگی در تفکیک کلاس‌ها کارایی چندانی ندارد.

۴۰ ویژگی با بیشترین امتیاز بر اساس این معیار در نمودار زیر مشخص می باشد. توجه شود در این رتبه بندی از نمونه های دیتافریم آموزش استفاده شده است.



شکل ۳۰_۲ تعداد ۴۰ ویژگی برتر بر اساس معیار SI

با توجه به امتیازات کسب شده در این دو معیار سعی میکنیم عملیات کاهش ویژگی را به این نحو انجام دهیم که اشتراک ۴۰ ویژگی که بالاترین امتیاز را در دو معیار کسب کرده اند به عنوان ویژگی های باقی مانده نگه داریم و سایر ویژگی ها را از دیتافریم های آموزش و تست حذف کنیم.

تعداد ۱۲ ویژگی به صورت زیر انتخاب شده اند :

چولگی سنسور با اندیس ۴ ، کشیدگی سنسور با اندیس ۳ ، پهنای بند سنسور با اندیس ۳، اوج سنسور با اندیس ۴، میانگین سنسور با اندیس ۵، ضریب تاج سنسور با اندیس ۵، چولگی سنسور با اندیس ۲، چولگی سنسور با اندیس ۳، میانگین سنسور با اندیس ۲،Spectral Flatness سنسور با اندیس ۴ ، فرکانس تشدید سنسور با اندیس ۵ و ضریب تاج سنسور با اندیس ۳

(در بخش ۳.۱.۲ اندیس های سنسور ها مشخص شده اند)

۴-۲- ۳.۲ آموزش مدل)

مدل طبقه بندی خطی که برای مسئله کلاس بندی انتخاب شده است رگرسیون لاجستیک می باشد که در ادامه به توضیح مختصری از آن می پردازیم.

رگرسیون لجستیک یکی از پرکاربردترین الگوریتم‌های پایه در یادگیری ماشین است که برای مسائل طبقه‌بندی (Classification) به‌کار می‌رود.

ویژگی‌های اصلی مدل:

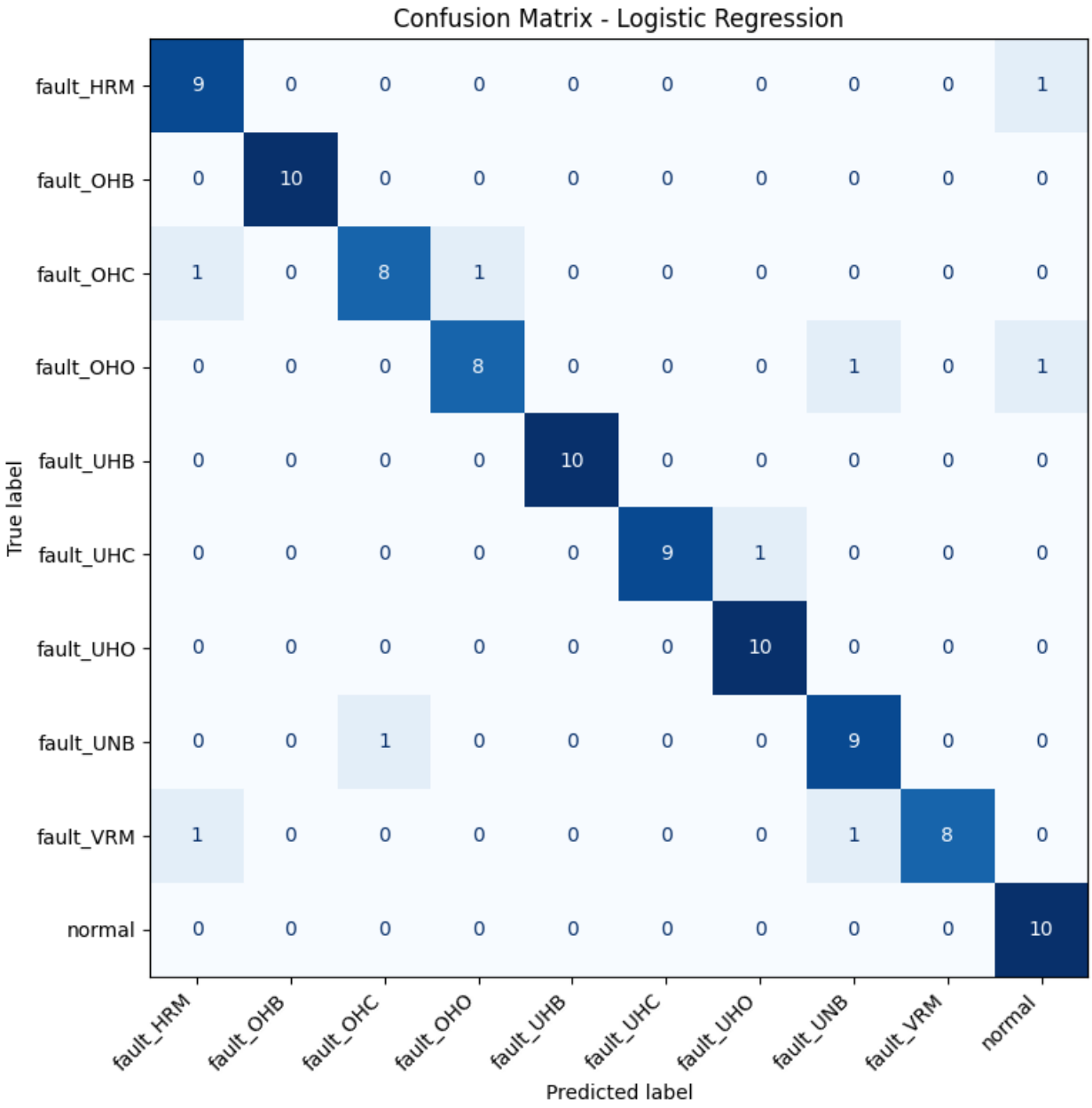
- **مدل خطی** :مرز تصمیم در Logistic Regression خطی است.
- **تابع فعال‌سازی** :خروجی مدل از طریق تابع سیگموید (sigmoid) یا softmax (در چندکلاسه) به احتمال تبدیل می‌شود.
- **قابلیت تفسیر بالا** :به دلیل سادگی ساختار، ضرایب مدل قابل تحلیل هستند.
- **کاربردها** :تشخیص اسپم، تشخیص بیماری، طبقه‌بندی متون، و بسیاری دیگر.

برای آموزش مدل از کتابخانه‌ی scikit-learn استفاده شده است که پارامتر های آن به شرح زیر است :

Max_iter : 1000
Solver : lbfgs
Penaly : l2
Multi-class : auto

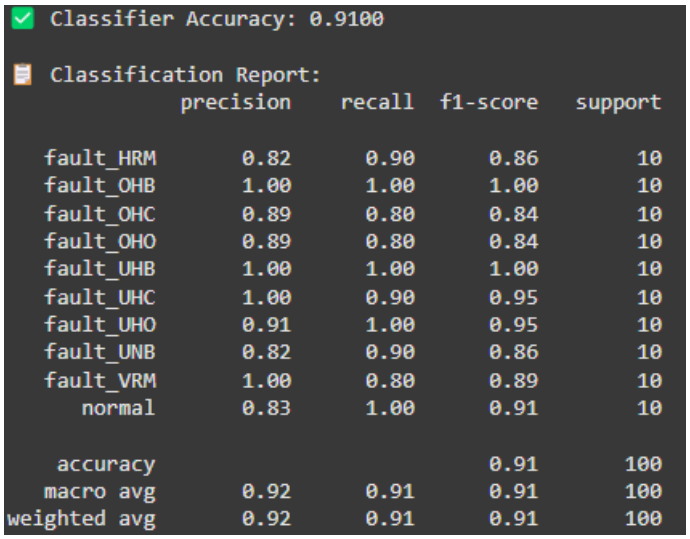
پس از آموزش، مدل روی داده‌های آزمون تست شد و معیارهای ارزیابی شامل درستی(Accuracy) ، گزارش طبقه‌بندی (Classification Report) و ماتریس درهم‌ریختگی (Confusion Matrix) بودند. همچنین، گزارش تفصیلی ازprecision ، recall، f1-score برای هر کلاس ارائه شد.

ماتریس در هم ریختگی داده های آزمون به صورت زیر می باشد :



شکل ۳۱_۲ ماتریس در هم ریختگی کلاسیفایر غیر سلسله مراتبی

گزارش طبقه بندی ۹ کلاسه نیز به شرح زیر می باشد :



Classifier Accuracy: 0.9100

Classification Report:

	precision	recall	f1-score	support
fault_HRM	0.82	0.90	0.86	10
fault_OHB	1.00	1.00	1.00	10
fault_OHC	0.89	0.80	0.84	10
fault_OHO	0.89	0.80	0.84	10
fault_UHB	1.00	1.00	1.00	10
fault_UHC	1.00	0.90	0.95	10
fault_UHO	0.91	1.00	0.95	10
fault_UNB	0.82	0.90	0.86	10
fault_VRM	1.00	0.80	0.89	10
normal	0.83	1.00	0.91	10
accuracy			0.91	100
macro avg	0.92	0.91	0.91	100
weighted avg	0.92	0.91	0.91	100

شکل ۳۲_۲ گزارش طبقه بندی کلاسیفایر غیر سلسله مراتبی

در این بخش به توضیح هر یک از مفاهیم آورده شده در گزارش طبقه بندی می پردازیم :

دقت (Precision) :

درصد پیش‌بینی‌های درست برای یک کلاس خاص، نسبت به تمام پیش‌بینی‌هایی که به آن کلاس نسبت داده شده‌اند .

$$Precision = \frac{TP}{TP + FP}$$

بازخوانی / حساسیت (recall) :

درصد نمونه‌هایی که واقعاً متعلق به یک کلاس بودند و به‌درستی شناسایی شدند.

$$Recall = \frac{TP}{TP + FN}$$

F1-Score :

میانگین هماهنگ (harmonic mean) بین precision و recall وقتی بخواهیم هم دقت و هم حساسیت رو باهم در نظر بگیریم.

$$F1\ Score = \frac{Recall * Precision}{Recall + Precision}$$

صحت (Accuracy) :

درصد نمونه‌هایی که مدل به‌درستی طبقه‌بندی کرده است نسبت به کل نمونه‌ها .

$$Accuracy = \frac{total\ correcet\ predictions}{total\ samples}$$

Macro average :

میانگین ساده‌یprecision ، recallو f1 برای تمام کلاس‌ها (بدون توجه به تعداد نمونه در هر کلاس)

Weighted Average :

میانگین وزن‌دار که هر کلاس به نسبت تعداد نمونه‌هاش در نظر گرفته می‌شود.

۱.۳.۲)

در این پایان‌نامه، ساختار سلسله‌مراتبی به‌منظور بهبود دقت تشخیص عیوب در سیگنال‌های ارتعاشی اجرا شده است که نحوه‌ی پیاده‌سازی به این شکل بوده است:

در گام اول، سیگنال ورودی به یک شبکه‌ی اولیه داده می‌شود که ۵ کلاس اصلی Normal ، Imbalance، Misalignment، Underhang، Overhang را تشخیص می‌دهد.

سپس، اگر سیگنال به کلاس‌هایی نظیرMisalignment ، Overhangیا Underhang تعلق داشت، به یک شبکه‌ی مجزا و تخصصی برای شناسایی زیرکلاس‌ها ارجاع داده می‌شود.

برای مثال:

- Misalignment → Horizontal / Vertical
- Underhang → Ball / Cage / OuterRace
- Overhang → Ball / Cage / OuterRace

بر اساس تحلیل‌های ارائه‌شده در پایان‌نامه، مزایای کلیدی این روش عبارت‌اند از:

- افزایش دقت تشخیص عیب به‌دلیل ساده‌تر شدن تصمیم‌گیری در هر مرحله

- کاهش پیچیدگی پردازش نسبت به مدل‌های تخت که تمام کلاس‌ها را به‌صورت یکجا دسته‌بندی می‌کنند
- امکان استفاده از ویژگی‌های خاص‌تر برای هر زیرکلاس، که با استفاده از الگوریتم‌های LightGBMو SI برای هر مرحله بهینه‌سازی شده‌اند
- توانایی تطبیق بهتر با ساختار منطقی عیوب که در دنیای واقعی نیز سلسله‌مراتبی هستند

نتایج و تحلیل عملکرد:

مطابق با بخش ۴-۴-۴ پایان‌نامه:

- اجرای مدل سلسله‌مراتبی موجب افزایش چشمگیر دقت شده و در ماتریس درهم‌ریختگی مشخص است که بسیاری از کلاس‌ها به‌درستی تفکیک شده‌اند.
- دقت کلی مدل سلسله‌مراتبی برای مجموعه داده‌های ۴۲ کلاسه بسیار بالا گزارش شده و تنها در چند مورد جزئی مانند Vertical و Underhang_Ball دچار اشتباه شده است.
- با ترکیب روش انتخاب ویژگی مبتنی بر LightGBM وSI ، ویژگی‌های مؤثر برای هر مرحله انتخاب شده و همین موضوع موجب بهبود عملکرد کلی مدل شده است.

۲.۳.۲ رویکرد دوم) :

در بخش ۳.۲ رویکرد اول آموزش (غیر سلسله مراتبی) صورت گرفت و در این بخش یک ساختار سلسله مراتبی به نحو خواسته شده (ابتدا کلاس بندی ۵ کلاسه سپس یک کلاس بندی ۲ کلاسه و ۲ کلاس بندی ۳ کلاسه) ارائه می شود.

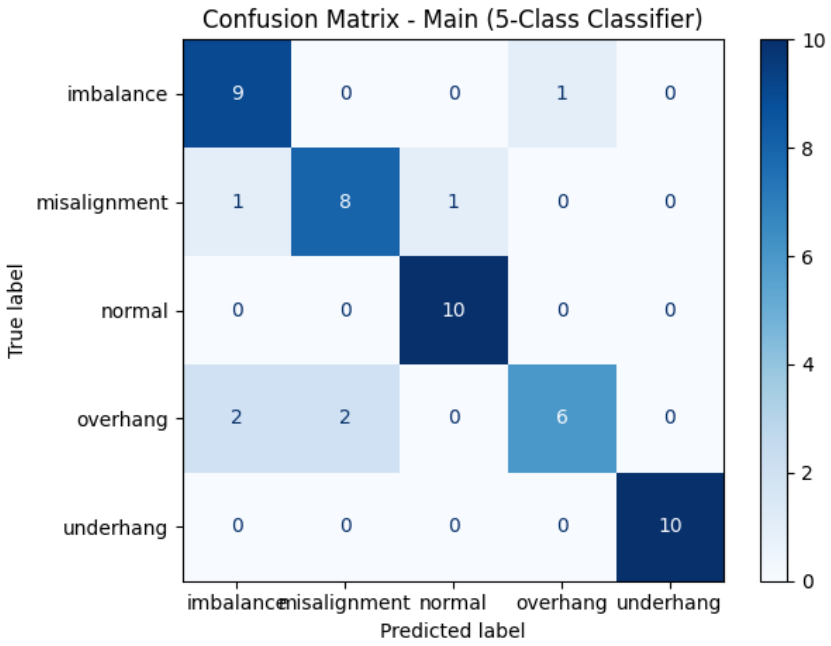
ابتدا دیتافریم های آموزش و تست را متناسب آموزش ۵ کلاسه تنظیم میکنیم که این کار را با تکنیک downsampling برای کلاس های misalignment , overhang , underhang با یک mappingساده و به صورت حذف رندوم یک سری از داده ها برای هر یک از این ۳ کلاس اصلی انجام میدهیم و در نهایت ۴۰ نمونه برای هر کلاس از دیتافریم آموزش و ۱۰ نمونه برای هر کلاس از دیتافریم تست باقی می ماند.

Mapping لیبل ها در تصویر زیر مشاهده می شود :

```python
label_mapping = {
    'fault_UHB': 'underhang', 'fault_UHC': 'underhang', 'fault_UHO': 'underhang',
    'fault_OHB': 'overhang',  'fault_OHC': 'overhang',  'fault_OHO': 'overhang',
    'fault_VRM': 'misalignment', 'fault_HRM': 'misalignment',
    'fault_UNB': 'imbalance',
    'normal': 'normal'
}
```

شکل ۳۳_۲ مپینگ لیبل ها در کلاسیفایر سلسله مراتبی

نتیجه آموزش مدل رگرسیون لجستیک ۵ کلاسه در قالب ماتریس در هم ریختگی به صورت زیر می باشد :



شکل ۳۴ _۲ ماتریس درهمریختگی کلاسیفایر ۵ کلاسه ابتدایی در ساختار سلسله مراتبی

همچنین گزارش طبقه بندی نیز در تصویر زیر آورده شده است :

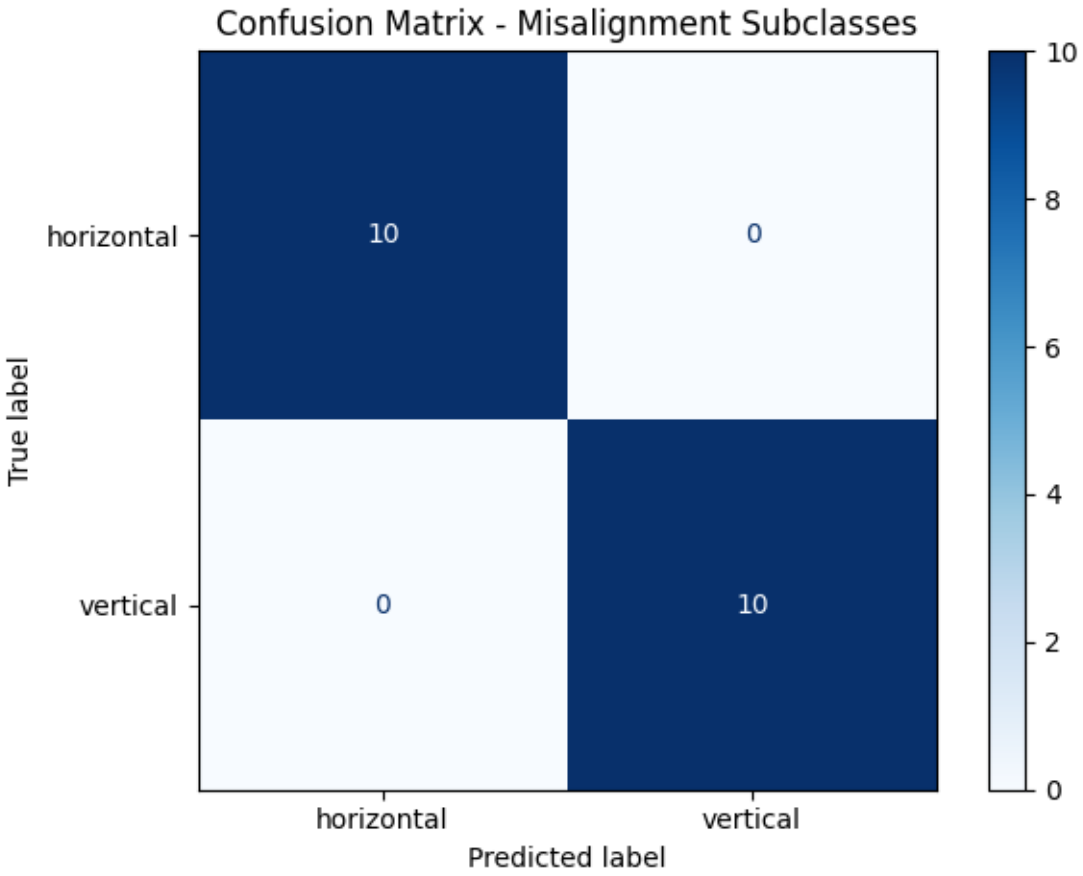
Classification Report:

	precision	recall	f1-score	support
imbalance	0.75	0.90	0.82	10
misalignment	0.80	0.80	0.80	10
normal	0.91	1.00	0.95	10
overhang	0.86	0.60	0.71	10
underhang	1.00	1.00	1.00	10
accuracy			0.86	50
macro avg	0.86	0.86	0.86	50
weighted avg	0.86	0.86	0.86	50

شکل ۳۵_۲ گزارش طبقه بندی کلاسیفایر ۵ کلاسه ابتدایی در ساختار سلسله مراتبی

حال برای کلاس اصلی misalignment دو زیر شاخه vertical , horizontal تعریف می کنیم و mapping را انجام میدهیم نتایج آموزش مدل کلاسبند misalignment به صورت زیر می باشد :

ماتریس در هم ریختگی :



شکل ۳۶_۲ ماتریس درهمریختگی کلاسیفایر ۲ کلاسه misalignment در ساختار سلسله مراتبی

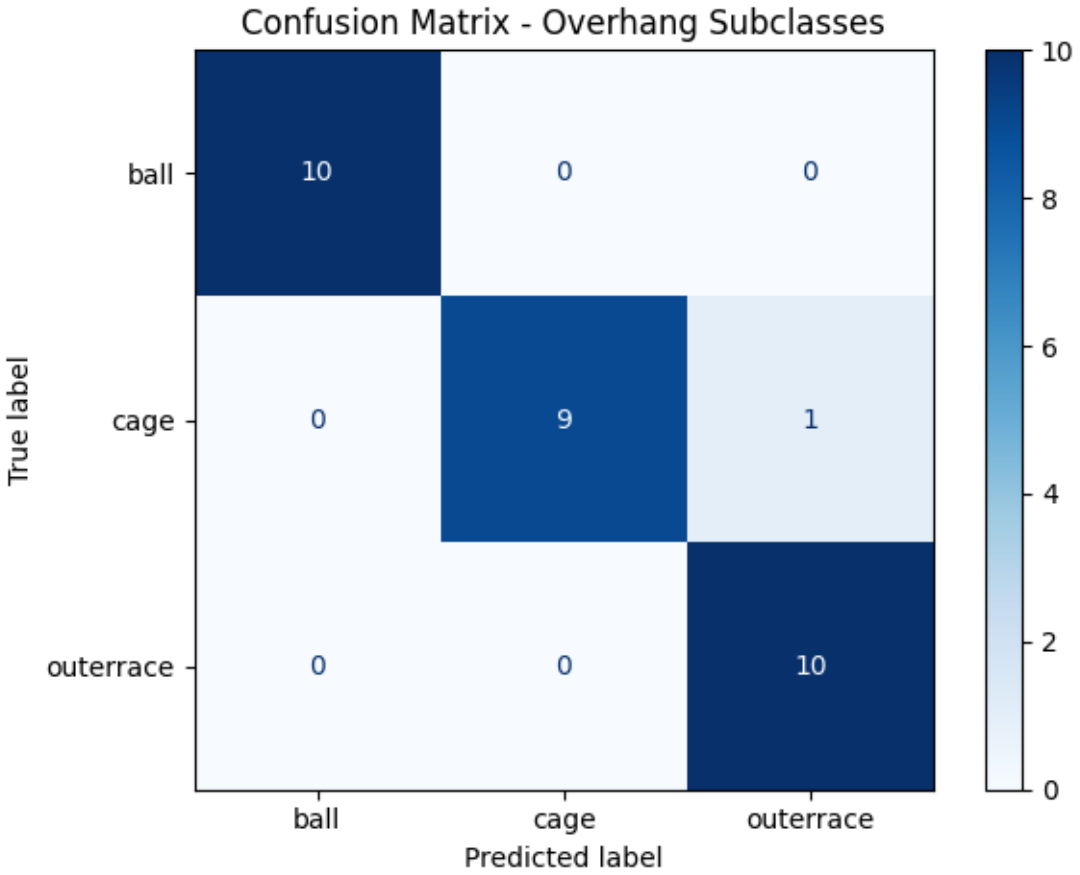
گزارش طبقه بندی :

```
✅ Classification Report (misalignment subclasses):
              precision    recall  f1-score   support

  horizontal       1.00      1.00      1.00        10
    vertical       1.00      1.00      1.00        10

    accuracy                           1.00        20
   macro avg       1.00      1.00      1.00        20
weighted avg       1.00      1.00      1.00        20
```

شکل ۳۷_۲ گزارش طبقه بندی کلاسیفایر ۲ کلاسه misalignment در ساختار سلسله مراتبی

به همین ترتیب نتایج کلاس بند سه کلاسه overhang به صورت زیر می باشد :

ماتریس درهم ریختگی :



شکل ۳۸_۳ ماتریس در هم ریختگی کلاسیفایر ۳ کلاسه overhang در ساختار سلسله مراتبی

گزارش طبقه بندی :

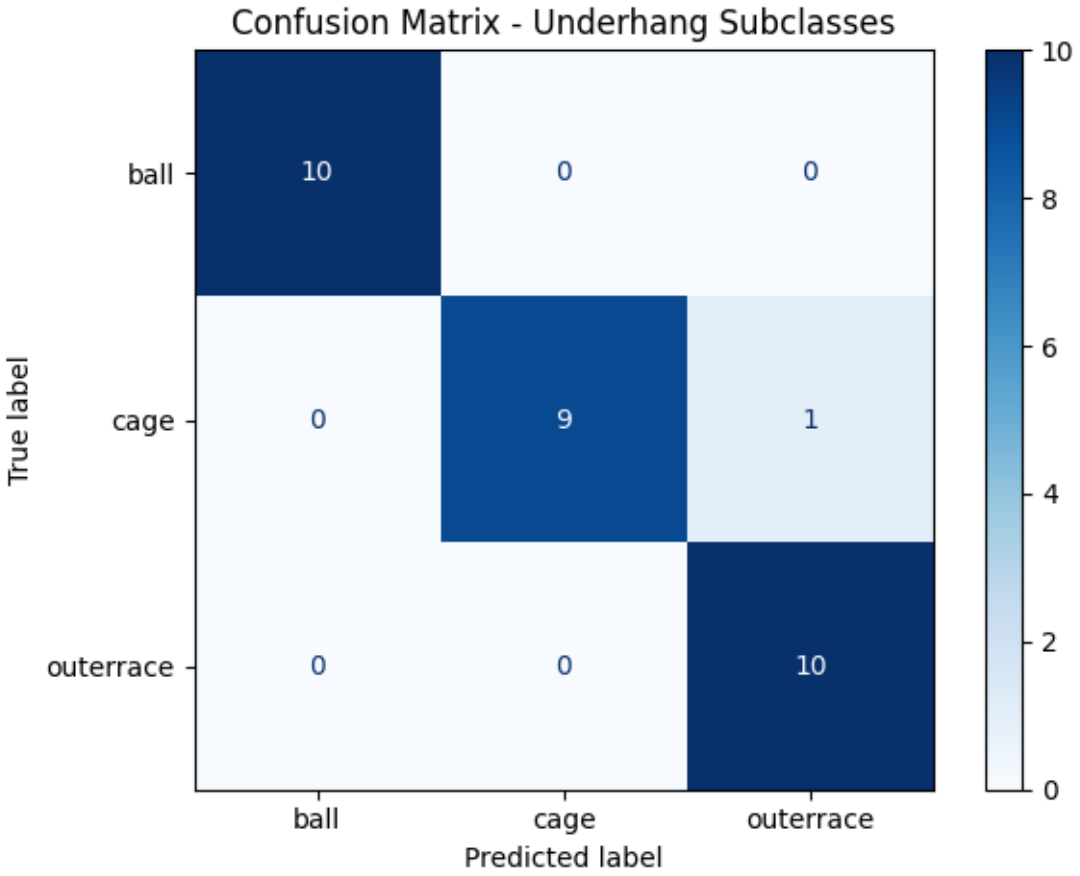
Classification Report (Overhang subclasses):

	precision	recall	f1-score	support
ball	1.00	1.00	1.00	10
cage	1.00	0.90	0.95	10
outerrace	0.91	1.00	0.95	10
accuracy			0.97	30
macro avg	0.97	0.97	0.97	30
weighted avg	0.97	0.97	0.97	30

شکل ۳۹_۲ گزارش طبقه بندی کلاسیفایر ۳ کلاسه overhang در ساختار سلسله مراتبی

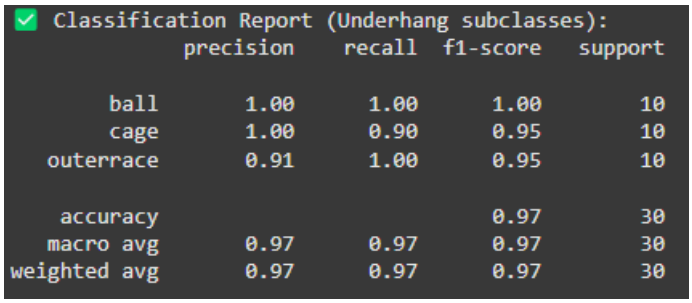
همچنین نتایج کلاس بند سه کلاسه underhang به صورت زیر می باشد :

ماتریس درهم ریختگی :



شکل ۴۰_۲ ماتریس درهم ریختگی کلاسیفایر ۳ کلاسه underhang در ساختار سلسله مراتبی

گزارش طبقه بندی :



Classification Report (Underhang subclasses):

	precision	recall	f1-score	support
ball	1.00	1.00	1.00	10
cage	1.00	0.90	0.95	10
outerrace	0.91	1.00	0.95	10
accuracy			0.97	30
macro avg	0.97	0.97	0.97	30
weighted avg	0.97	0.97	0.97	30

شکل ۴۱_۲ گزارش طبقه بندی کلاسیفایر ۳ کلاسه underhang در ساختار سلسله مراتبی

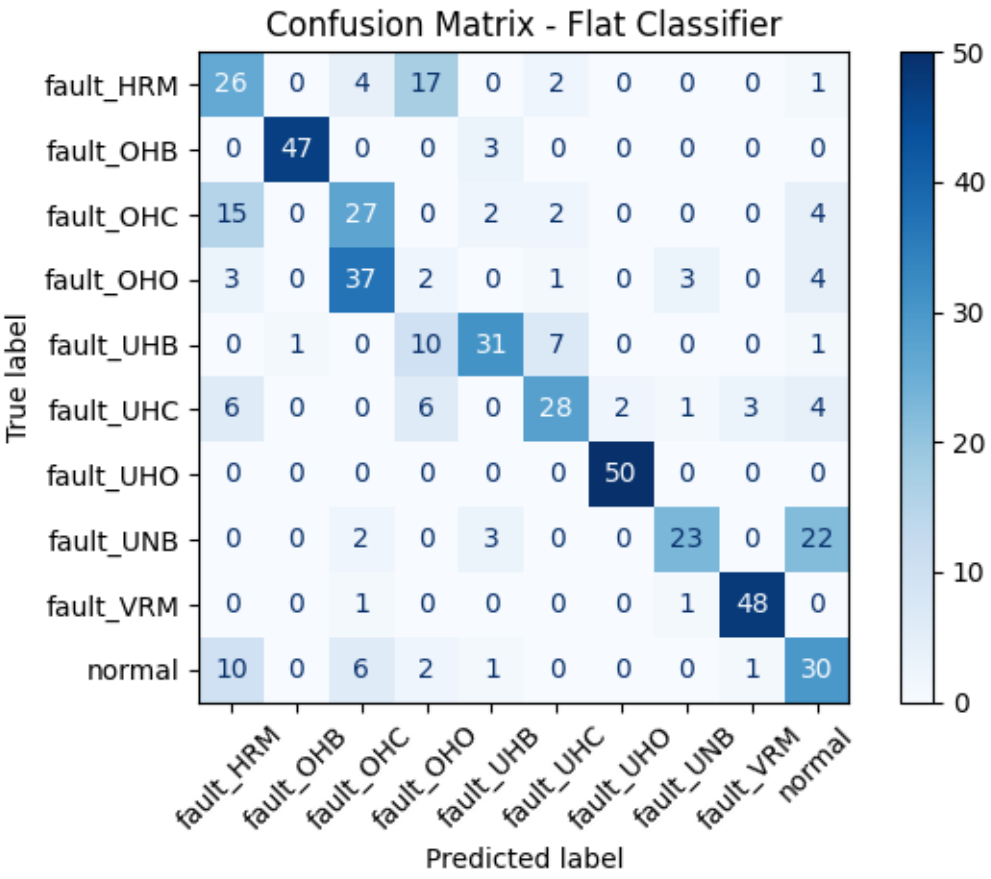
(۳.۳.۲

برای مقایسه دو رویکرد ابتدا مجددا از سایت دادگان ده داده جدید برای هر کلاس دانلود کرده و سپس مراحل پیش پردازش و استخراج ۱۲ ویژگی منتخبی که در قسمت قبل به دست آورده ایم انجام میدهیم همچنین آنها را بر همان مبنای داده آموزشی که در ابتدا استفاده شد نرمالایز میکنیم و آماده ورود به دو کلاسبند ما خواهند شد.

در نهایت این دیتافریم ۱۳ ستون شامل ۱۲ فیچر منتخب + ستون لیبل عیب ها و ۵۰۰ سطر خواهد بود .

حال به مقایسه این دو کلاسبند با مشاهده ماتریس درهم ریختگی و گزارش طبقه بندی می پردازیم.

ماتریس درهم ریختگی برای کلاسیفایر تخت (غیر سلسله مراتبی) :



شکل ۴۲_۲ ماتریس در هم ریختگی کلاسیفایر غیر سلسله مراتبی روی دیتا جدید

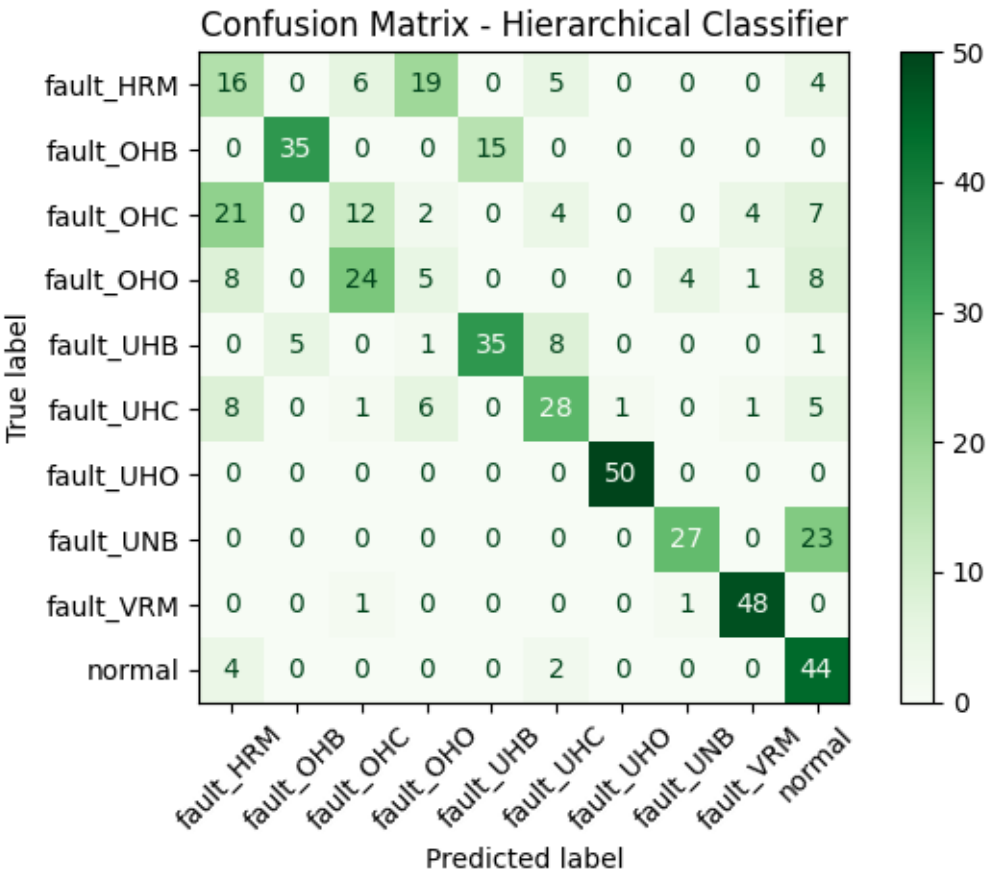
گزارش طبقه بندی نیز برای این کلاسیفایر به صورت زیر می باشد :

Classification Report - Flat Classifier:

	precision	recall	f1-score	support
fault_HRM	0.43	0.52	0.47	50
fault_OHB	0.98	0.94	0.96	50
fault_OHC	0.35	0.54	0.43	50
fault_OHO	0.05	0.04	0.05	50
fault_UHB	0.78	0.62	0.69	50
fault_UHC	0.70	0.56	0.62	50
fault_UHO	0.96	1.00	0.98	50
fault_UNB	0.82	0.46	0.59	50
fault_VRM	0.92	0.96	0.94	50
normal	0.45	0.60	0.52	50
accuracy			0.62	500
macro avg	0.65	0.62	0.62	500
weighted avg	0.65	0.62	0.62	500

شکل ۴۳_۲ گزارش طبقه بندی کلاسیفایر غیر سلسله مراتبی روی دیتا جدید

ماتریس درهم ریختگی برای کلاسیفایر سلسله مراتبی :



شکل ۴۴_۲ ماتریس در هم ریختگی کلاسیفایر سلسله مراتبی روی دیتا جدید

گزارش طبقه بندی برای این کلاسیفایر به صورت زیر می باشد :

Classification Report - Hierarchical Classifier:

	precision	recall	f1-score	support
fault_HRM	0.28	0.32	0.30	50
fault_OHB	0.88	0.70	0.78	50
fault_OHC	0.27	0.24	0.26	50
fault_OHO	0.15	0.10	0.12	50
fault_UHB	0.70	0.70	0.70	50
fault_UHC	0.60	0.56	0.58	50
fault_UHO	0.98	1.00	0.99	50
fault_UNB	0.84	0.54	0.66	50
fault_VRM	0.89	0.96	0.92	50
normal	0.48	0.88	0.62	50
accuracy			0.60	500
macro avg	0.61	0.60	0.59	500
weighted avg	0.61	0.60	0.59	500

شکل ۴۵_۲ گزارش طبقه بندی کلاسیفایر سلسله مراتبی روی دیتا جدید

(۴.۳.۲

همانطور که از گزارش ها پیداست، صحت هر دو مدل تقریبا برابر می باشند هرچند کلاسیفایر غیر سلسله ای ۲ درصد بهتر عمل نموده است.

هرچند صحت هر دو مدل چندان جالب نیست و مدل غیر سلسله ای تنها اندکی از سلسله مراتبی بیشتر می باشد اما با توجه به کاهش دیتا در روش سلسله مراتبی با تکنیک هایی مثل downsampling انتخاب مدل غیر سلسله مراتبی میتواند توجیه بیشتری داشته باشد.

برای افزایش دقت در پیشبینی میتوان در گام نخست پارامتر های آموزش کلاسیفایر های لجستیک استفاده شده را تغییر داد یا حتی از کلاسیفایر های پیچیده تر استفاده نمود.

یکی دیگر از کار ها استفاده از داده های بیشتر برای آموزش شبکه ها می باشد.

مورد دیگری که میتوان به بهبود داد انتخاب ویژگی هایی است که بر اساس دو معیار lightGBM , SI انتخاب شده اند.

۴.۲ محصول) ۵-۲-

به عنوان محصول تابعی با نام Yataghan نوشته شده است که یک دیتا فریم خام که لیبل عیب آن مشخص شده است را دریافت میکند سپس به صورت خودکار درون تابع به اندازه پنجره زمانی تعریف شده نمونه برداری میکند و بعد از استخراج تمامی ویژگی ها عملیات استاندارد سازی را انجام داده و سپس ویژگی های منتخب را نگه میدارد سپس دیتافریم آماده شده را به کلاسیفایر سلسله مراتبی داده و نتیجه پیشبینی را با لیبل اولیه مقایسه کرده و در صورت درست بودن پیغام درست بودن و در صورت درست نبود پیغام اینکه پیشبینی مدل برای مثال عیب x بوده است اما لیبل درست عیب yمی باشد را چاپ میکند خروجی این تابع برای یک نمونه که از کلاس عیب underhang-ball بوده است و خوشبختانه مدل هم درست پیشبینی کرده است به صورت زیر می باشد.

```
✅ The model correctly predicted the label: fault_UHB
('fault_UHB', 'fault_UHB')
```

شکل ۴۶_۲ نتیجه و خروجی تابع محصول

۲-۶- امتیازی (T-SNE)

t-SNE مخفف *t-Distributed Stochastic Neighbor Embedding* یک روش غیرخطی برای کاهش ابعاد و تجسم داده‌ها است که به‌ویژه برای داده‌هایی با ابعاد بالا کاربرد دارد. هدف اصلی این روش، نگاشت نقاطی از فضای ویژگی با ابعاد بالا به فضای دوبعدی یا سه‌بعدی است به‌گونه‌ای که ساختار همسایگی بین نقاط تا حد امکان حفظ شود. به بیان دیگر، t-SNEتلاش می‌کند که نقاط مشابه در فضای ویژگی نزدیک به هم و نقاط غیرمشابه، دور از هم در فضای پایین‌بعدی قرار گیرند.

الگوریتم t-SNE به صورت دو مرحله‌ای عمل می‌کند:

۱ - شباهت‌ها در فضای با ابعاد بالا

برای هر دو نقطه x_i و x_j در فضای ویژگی، احتمال شباهت آن‌ها با تابع گوسی زیر تعریف می‌شود:

$$P_{ij} = \frac{exp(-\frac{\|x_i - x_j\|^2}{2\sigma_i^2})}{\sum_{k \neq i} exp(-\frac{\|x_i - x_k\|^2}{2\sigma_i^2})}$$

۲ - نگاشت به فضای با ابعاد پایین

در فضای پایین‌بعدی (مثلاً دوبعدی)، احتمال شباهت بین نقاط y_i و y_j با استفاده از توزیع Student-t با یک درجه آزادی تعریف می‌شود:

$$q_{ij} = \frac{(1 + \|y_i - y_j\|^2)^{-1}}{\sum_{k \neq l}(1 + \|y_k - y_l\|^2)^{-1}}$$

لگوریتم t-SNE مکان نقاط را در فضای پایین‌بعدی به گونه‌ای تنظیم می‌کند که توزیع q_{ij} تا حد امکان به توزیع p_{ij} شباهت داشته باشد. این تنظیم با کمینه کردن واگرایی کولبک-لیبلر (KL Divergence) صورت می‌گیرد:

$$C = \sum_i \sum_j P_{ij}(Log \frac{P_{ij}}{q_{ij}})$$

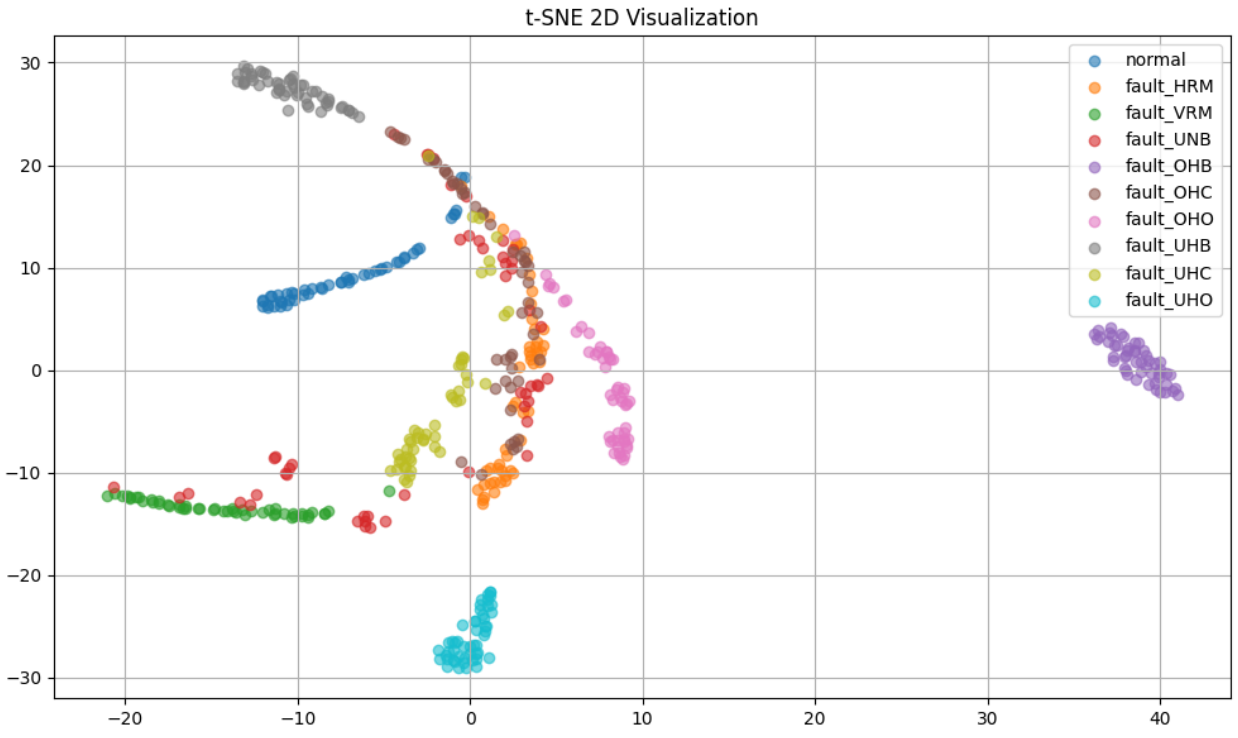
ویژگی‌های کلیدی t-SNE

- **کاهش ابعاد غیرخطی :**برخلاف روش‌های خطی مانندPCA ، t-SNE قادر است روابط پیچیده و غیرخطی بین داده‌ها را درک کند.

- **مدیریت مشکل تجمع t-SNE :**با استفاده از توزیع heavy-tailed در فضای پایین‌بعدی از تجمع غیرطبیعی داده‌ها در مرکز جلوگیری می‌کند.
- **حفظ ساختار محلی :**ساختار همسایگی بین نقاط به خوبی در فضای تجسمی حفظ می‌شود، در نتیجه خوشه‌ها و کلاس‌ها در نمودار نهایی به‌وضوح قابل تفکیک هستند.

حال به اعمال این روش بر روی داده های استخراج شده در انتهای بخش ۲.۲ می پردازیم.

نمودار ۲ بعدی T-SNE به صورت زیر می باشد :

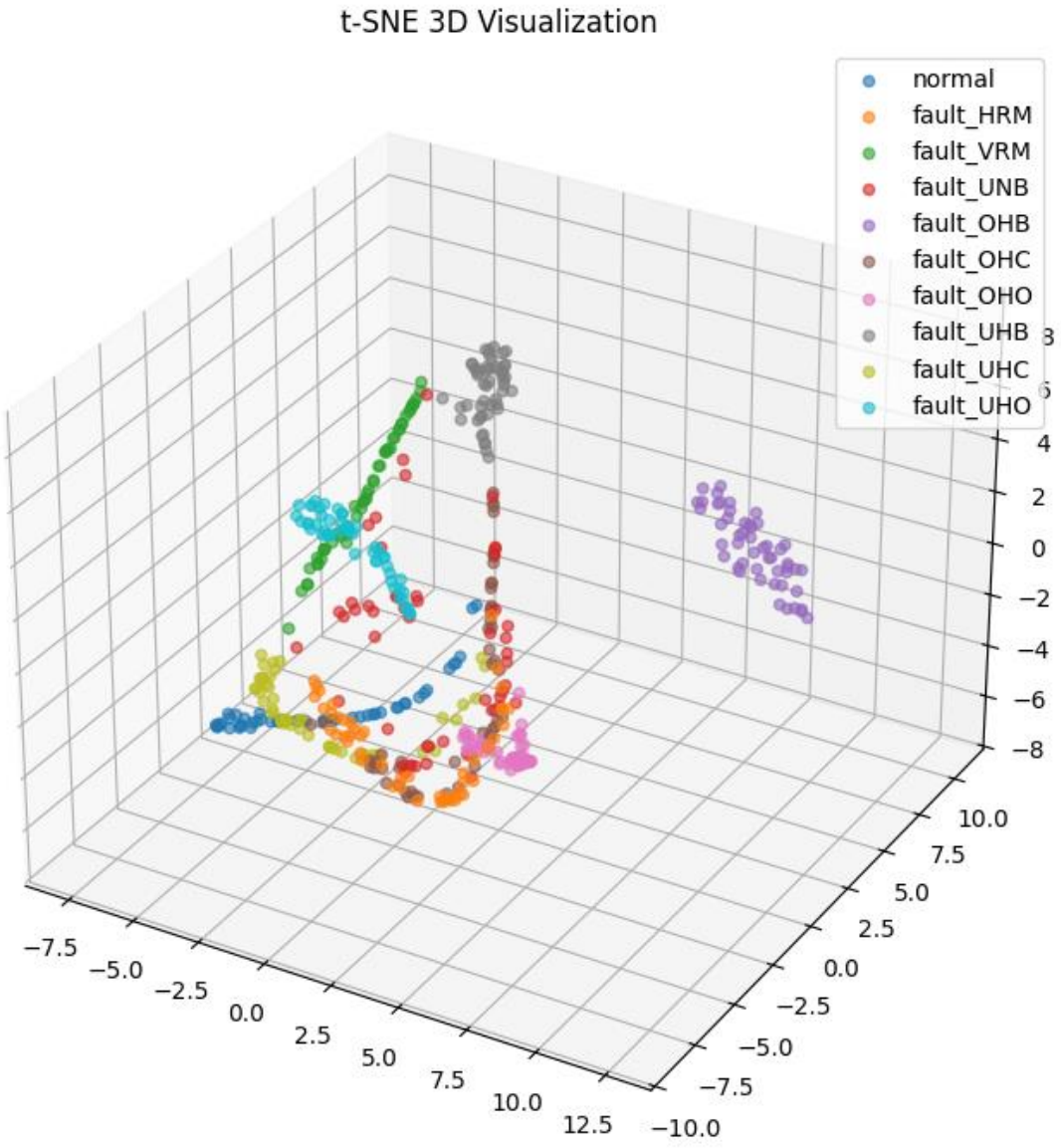


شکل ۴۷_۲ نمودار ۲ بعدی روش T-SNE

در نمودار دوبعدی، بسیاری از کلاس‌ها به‌خوبی از یکدیگر تفکیک شده‌اند؛ به‌ویژه کلاس‌های fault_OHB و fault_OHO که به صورت خوشه‌هایی کاملاً متمایز از سایر داده‌ها دیده می‌شوند. کلاس normal نیز در موقعیتی مجزا و با پراکندگی کم قرار گرفته است.

با این حال، در برخی نقاط هم‌پوشانی یا نزدیکی بین کلاس‌هایی مثل fault_HRM، fault_OHC و fault_UHC مشاهده می‌شود که می‌تواند نشان‌دهنده‌ی تشابه الگوهای آن‌ها در فضای ویژگی باشد.

نمودار ۳ بعدی T-SNE به صورت زیر می باشد :



شکل ۴۸_۲ نمودار ۳ بعدی روش T-SNE

نمودار سه‌بعدی نیز مشابه نمودار دو بعدی، ساختار خوشه‌ای مناسبی را برای اکثر کلاس‌ها نمایش می‌دهد. اضافه شدن بعد سوم به وضوح تمایز برخی کلاس‌ها کمک کرده است؛ برای مثال کلاس‌های fault_UHB و fault_UNBدر این فضا جدایی بیشتری نسبت به نمودار دو بعدی نشان داده‌اند.

نتیجه گیری :

این نتایج نشان می‌دهد که ویژگی‌های استخراج‌شده دارای توان تفکیک نسبتاً مناسبی برای طبقه‌بندی هستند، اما برخی کلاس‌ها هم‌پوشانی نسبی دارند. این مسئله می‌تواند یکی از دلایل افت دقت در مدل‌های طبقه‌بندی باشد. استفاده از روش‌های انتخاب ویژگی پیشرفته‌تر، استخراج ویژگی‌های غنی‌تر، و حتی بهره‌گیری از روش‌های یادگیری عمیق می‌تواند به بهبود کیفیت جداسازی کلاس‌ها کمک کند.