Feature scaling သို့မဟုတ် Normalization

Data များကို ကောင်းစွာ normalize မလုပ်ထားလျှင် neural network များသည် သတ်မှတ်ထားသည့် iteration ကျော်သွားသည့်တိုင် တစ်ခါတစ်ရံ converging မဖြစ်ခြင်းမျိုးနှင့် ကြုံတွေ့ရနိုင်သည်။ Multi-layer Perceptron များကို အသုံးပြုသည့်အခါ feature scaling လုပ်ပေးရန် မဖြစ်မနေလိုအပ်သည်။ Multi-layer Perceptron သည် feature တန်ဖိုးများ ကွာဟမှု အလွန်များခြင်းကို sensitive ဖြစ်သည်။ ထို့ကြောင့် data များ ကွာဟမှု နည်းသွားအောင် feature scaling သို့မဟုတ် normalization မဖြစ်မနေ လုပ်သင့်သည်။ train dataset နှင့် test dataset နှစ်ခုစလုံးကို တူညီသည့် feature scaling သို့မဟုတ် normalization လုပ်ရမည်။ သို့မှသာ meaningful result များ ရနိုင်လိမ့်မည်။ normalization လုပ်နိုင်သည့်နည်းများစွာရှိသည့်အနက် built-in StandardScaler ကို သုံးသည့်နည်းကို အောက်တွင် ဖော်ပြထားသည်။

In [1]:
```
import pandas as pd

import matplotlib.pyplot as plt
plt.rcParams['figure.figsize'] = [10, 6]
plt.rcParams['figure.dpi'] = 100
```

Data Loading

SciKit Learn မှ built in Breast Cancer Data Set ကို အသုံးပြုထားသည်။ ဒေတာထဲတါင် ကျူမော(tumors) နှင့်သက်ဆိုင်သည့်အချက်များ(several features of tumors) ပါဝင်သည်။ ထို့အပြင် ကျူမော(tumors)သည် Malignant ဖြစ်သည် သို့မဟုတ် Benign ဖြစ်သည်ကိုလည်း ဖော်ပြထားသည့် label များ ပါဝင်သည်။ ကျူမော(tumors)၏ feature များကို အသုံးပြု၍ malignant ဖြစ်လိမ့်မည် သို့မဟုတ် Benign ဖြစ်လိမ့်မည်ကို ခန့်မှန်းပေးမည့် network model တည်ဆောက်မည်။

In [2]:
```
# cancer ဒေတာကို load လုပ်သည်။
from sklearn.datasets import load_breast_cancer
cancer = load_breast_cancer()
```

In [3]:
```
# Print full description by running:
# print(cancer['DESCR'])
# 569 data points with 30 features
cancer['data'].shape
```

Out[3]:
```
(569, 30)
```

'data', 'target', 'target_names', 'DESCR', 'feature_names', 'filename' စသည့် description များသည် dictionary ပုံစံ ဖြင့် ရှိနေသည်။
This object is like a dictionary, it contains a description of the data and the features and targets:

In [4]:
```
cancer.keys()
```

Out[4]:
```
dict_keys(['data', 'target', 'target_names', 'DESCR', 'feature_na
mes', 'filename'])
```

cancer ဒေတာထဲမှ 'data' ကို DataFrame တစ်ခု အဖြစ်ပြောင်းပြီး ပထမဆုံး(၅)ခုကို ကြည့်သည်။

In [5]:
```
df_data = pd.DataFrame(cancer['data'])
df_data.head()
```

Out[5]:

	0	1	2	3	4	5	6	7	8	9	...	
0	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	0.2419	0.07871	...	25
1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	0.1812	0.05667	...	24
2	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	0.2069	0.05999	...	23
3	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	0.2597	0.09744	...	14
4	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	0.1809	0.05883	...	22

5 rows × 30 columns

cancer ဒေတာထဲမှ 'target_names' ကို DataFrame တစ်ခု အဖြစ်ပြောင်းပြီး ပထမဆုံး(၅)ခုကို ကြည့်သည်။

In [6]:
```
df_target_names = pd.DataFrame(cancer['target_names'])
df_target_names.head()
```

Out[6]:

	0
0	malignant
1	benign

cancer ဒေတာထဲမှ 'feature_names'ကို DataFrame တစ်ခု အဖြစ်ပြောင်းပြီး ပထမဆုံး(၅)ခုကို ကြည့်သည်။

In [7]:
```
df_feature_names = pd.DataFrame(cancer['feature_names'])
df_feature_names.head()
```

Out[7]:

	0
0	mean radius
1	mean texture
2	mean perimeter
3	mean area
4	mean smoothness

cancer ဒေတာကို train ဒေတာ(X)နှင့် test ဒေတာ (y)အဖြစ် ခွဲသည်။

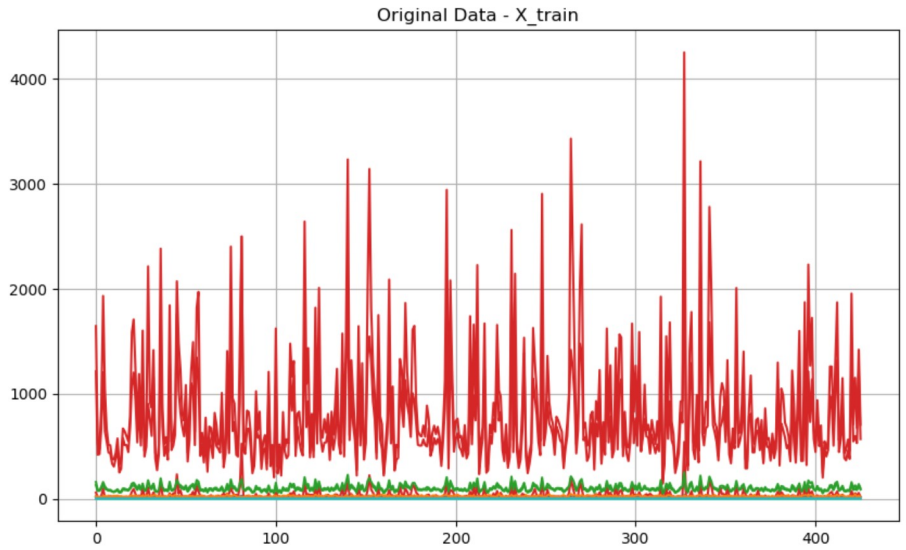
In [8]:
```
X = cancer['data']
y = cancer['target']

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y)
```

ဒေတာများကို ဂရပ်ဆွဲခြင်း

In [9]:
```
plt.plot(X_train)
plt.grid()
plt.title('Original Data - X_train')
```

Out[9]:
```
Text(0.5, 1.0, 'Original Data - X_train')
```



In [10]:
```
plt.plot(X_test)
plt.grid()
plt.title('Original Data - X_test')
```

Out[10]:
```
Text(0.5, 1.0, 'Original Data - X_test')
```

feature scaling သို့မဟုတ် normalization လုပ်သည်

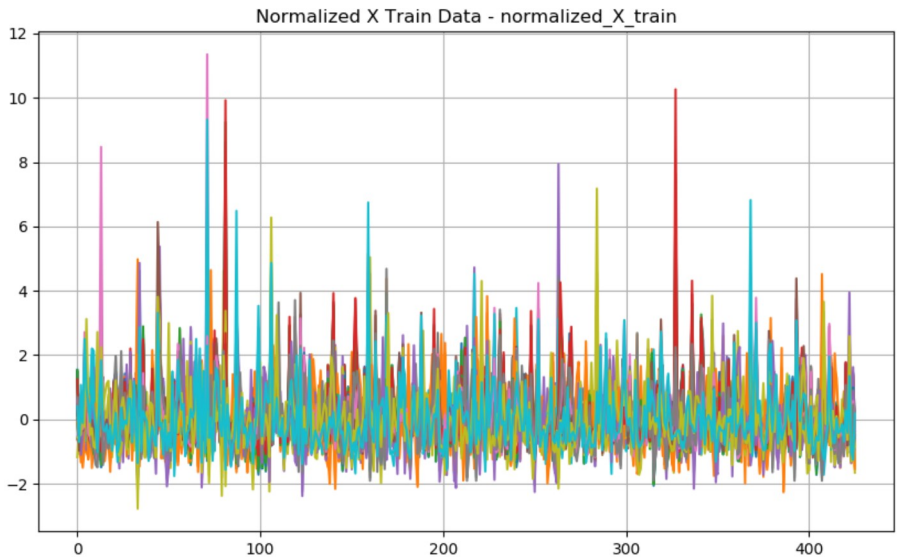
In [11]:
```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
# Fit only to the training data
scaler.fit(X_train)
```

Out[11]:
```
StandardScaler(copy=True, with_mean=True, with_std=True)
```

In [12]:
```
# Now apply the transformations to the data:
normalized_X_train = scaler.transform(X_train)
normalized_X_test = scaler.transform(X_test)
```

In [13]:
```
plt.plot(normalized_X_train)
plt.grid()
plt.title('Normalized X Train Data - normalized_X_train')
```

Out[13]:
```
Text(0.5, 1.0, 'Normalized X Train Data - normalized_X_train')
```



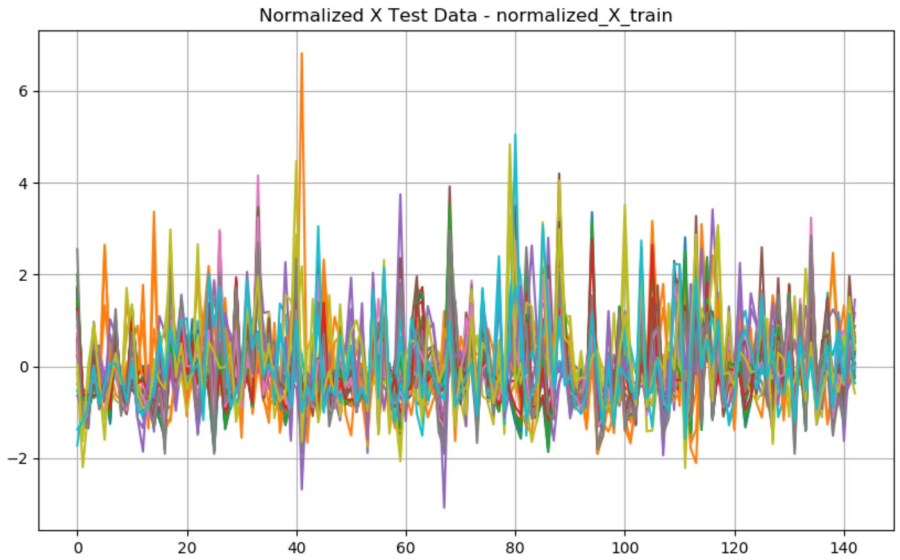
In [14]:
```
X_test.shape
```

Out[14]:
```
(143, 30)
```

In [15]:
```
import matplotlib.pyplot as plt
```

In [16]:
```
plt.plot(normalized_X_test)
plt.grid()
plt.title('Normalized X Test Data - normalized_X_train')
```

Out[16]:
```
Text(0.5, 1.0, 'Normalized X Test Data - normalized_X_train')
```



normalized_X_train နှင့် normalized_X_test တို့၏ ဂရပ်နှစ်ခုကို ကြည့်ရုံဖြင့် feature scaling သို့မဟုတ် normalization ၏ ထိရောက်မှုကို သိနိုင်သည်။

ကောင်းထက်ညွှန့်

In [ ]:
```

```