Open in Colab

ELECTRIC MOTOR TEMPERATURE PREDICTION

In [40]:

```
#IMPORTING THE NECESSARY LIBRARIES
```

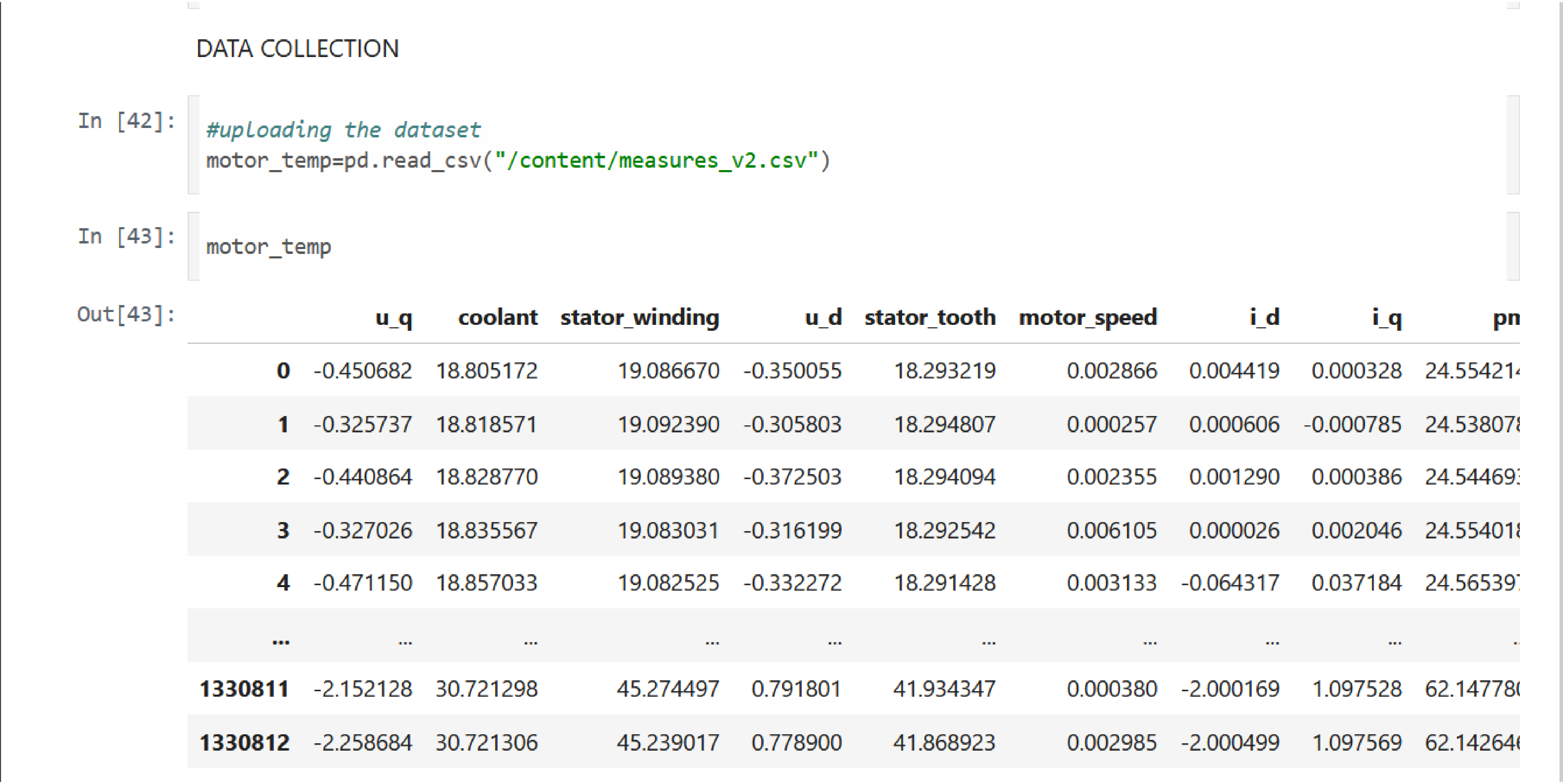
In [41]:

```
import numpy as np
import pandas as pd

import matplotlib.pyplot as plt
import seaborn as sns
sns.set_style('darkgrid')

from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split

from sklearn.linear_model import LinearRegression
import warnings
warnings.filterwarnings('ignore')
```



DATA COLLECTION

In [42]:
```
#uploading the dataset
motor_temp=pd.read_csv("/content/measures_v2.csv")
```

In [43]:
```
motor_temp
```

Out[43]:

	u_q	coolant	stator_winding	u_d	stator_tooth	motor_speed	i_d	i_q	pn
0	-0.450682	18.805172	19.086670	-0.350055	18.293219	0.002866	0.004419	0.000328	24.55421
1	-0.325737	18.818571	19.092390	-0.305803	18.294807	0.000257	0.000606	-0.000785	24.53807
2	-0.440864	18.828770	19.089380	-0.372503	18.294094	0.002355	0.001290	0.000386	24.54469
3	-0.327026	18.835567	19.083031	-0.316199	18.292542	0.006105	0.000026	0.002046	24.55401
4	-0.471150	18.857033	19.082525	-0.332272	18.291428	0.003133	-0.064317	0.037184	24.56539
...	...	...	...	...	...	...	...	...	.
1330811	-2.152128	30.721298	45.274497	0.791801	41.934347	0.000380	-2.000169	1.097528	62.14778
1330812	-2.258684	30.721306	45.239017	0.778900	41.868923	0.002985	-2.000499	1.097569	62.14264

1330813	-1.277003	-0.252853	-0.736995	0.411126	-0.656745	-1.184128	1.027480	-0.393939	-0.47706
1330814	-1.280131	-0.252852	-0.737626	0.410463	-0.658601	-1.184126	1.027453	-0.393972	-0.47682
1330815	-1.276320	-0.252852	-0.739760	0.411164	-0.659797	-1.184127	1.027456	-0.393964	-0.47659

1330816 rows × 11 columns

In [58]:

```
y
```

Out[58]:

	pm
0	24.554214
1	24.538078
2	24.544693
3	24.554018
4	24.565397
...	...
1330811	62.147780

1330811	62.147780
1330812	62.142646
1330813	62.138387
1330814	62.133422
1330815	62.131429

1330816 rows × 1 columns

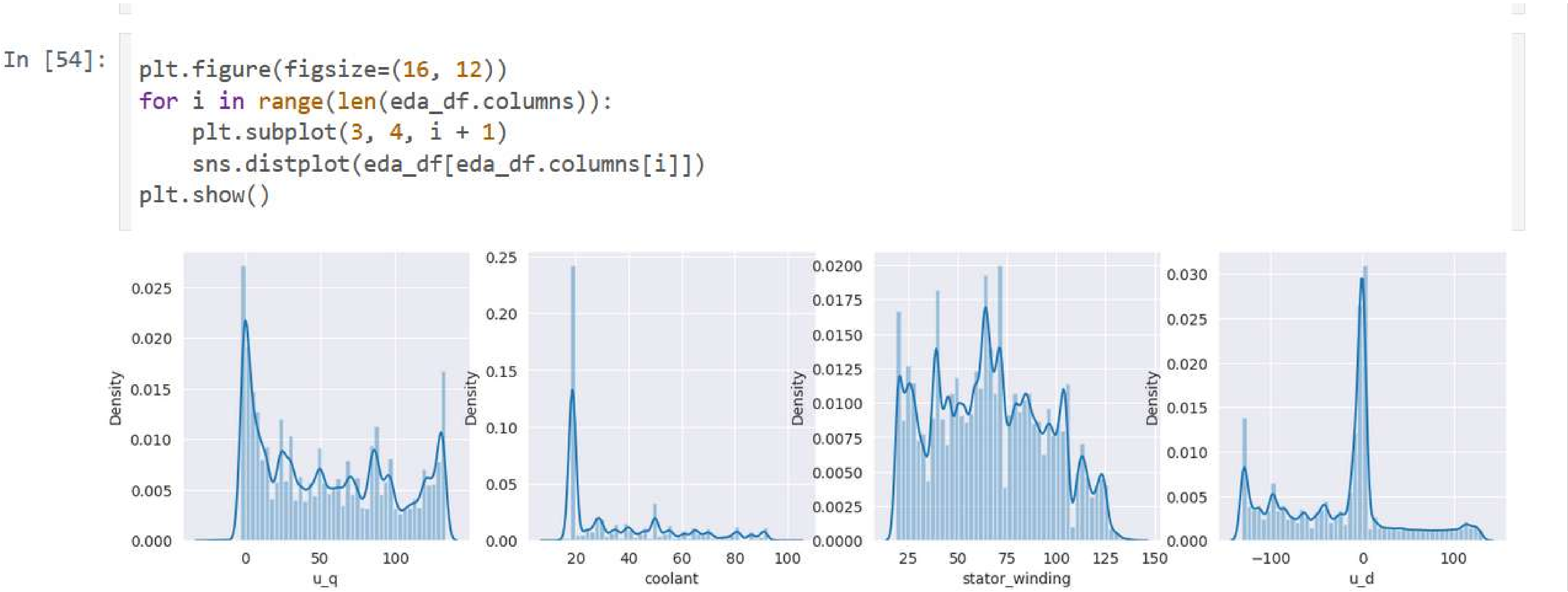
dtype: float64

EXPLORATORY DATA ANALYSIS

```
In [ ]: eda_df=pd.concat([x,y],axis=1).copy()
```

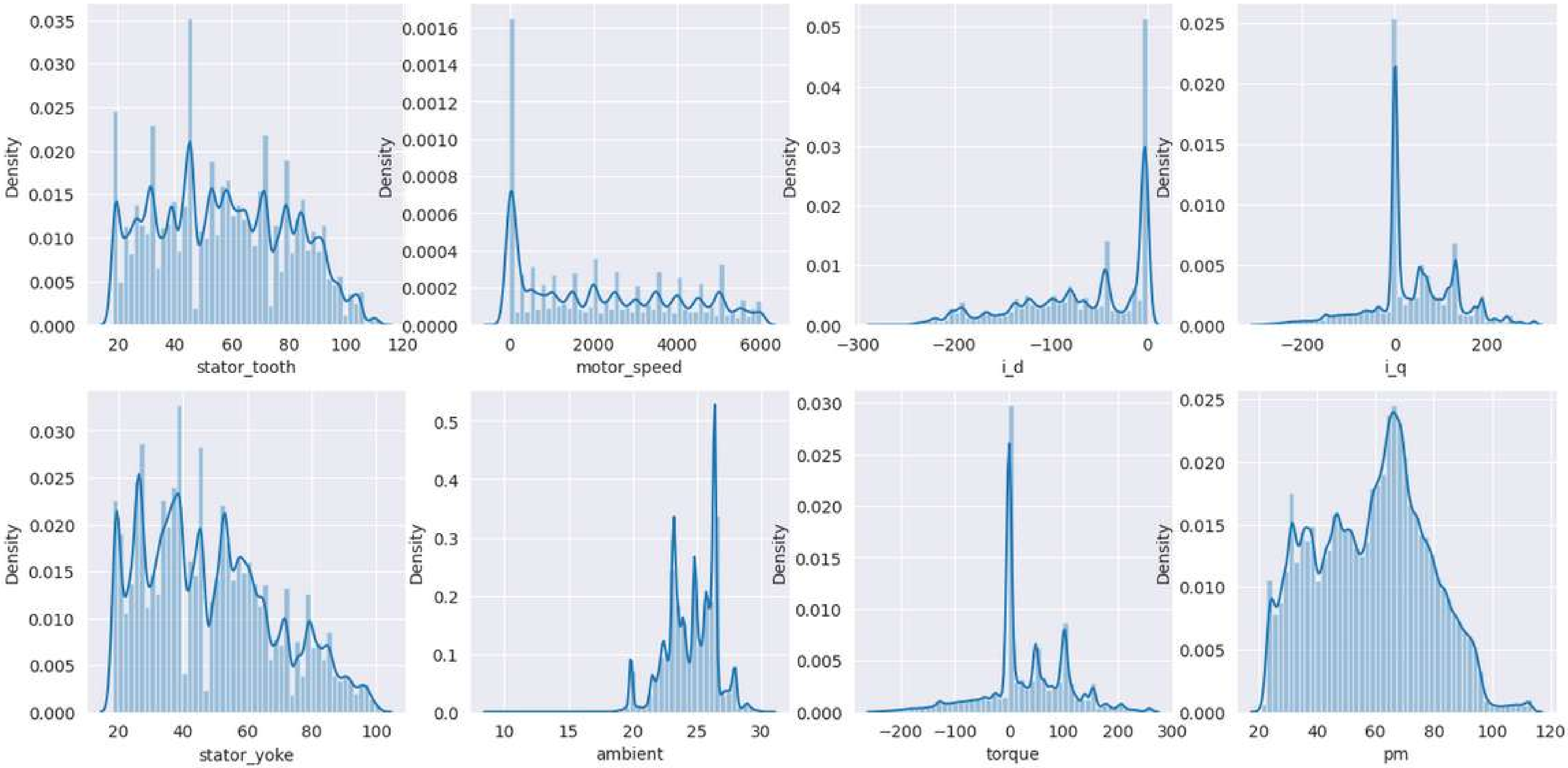
UNIVARIATE ANALYSIS

```
In [ ]: eda_df.describe()
```



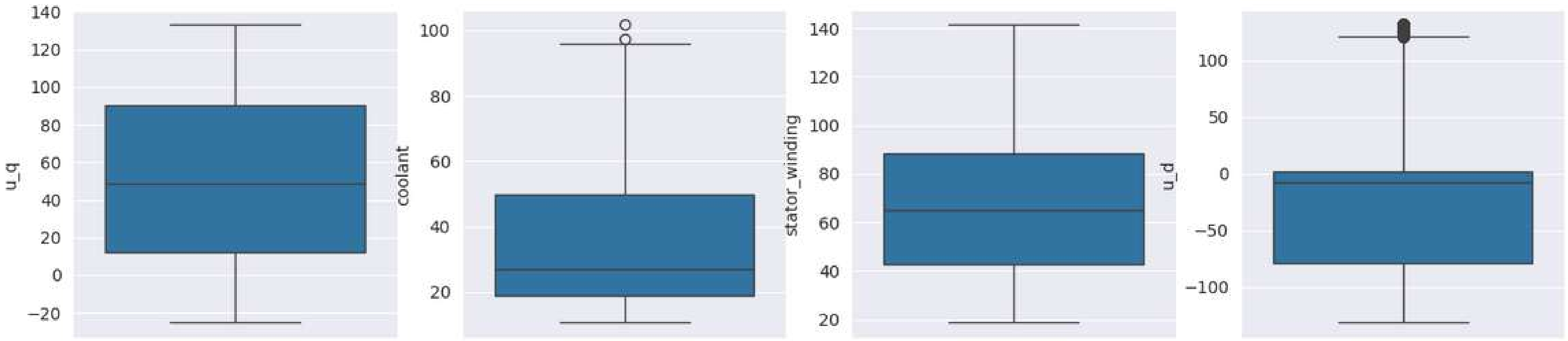
In [54]:

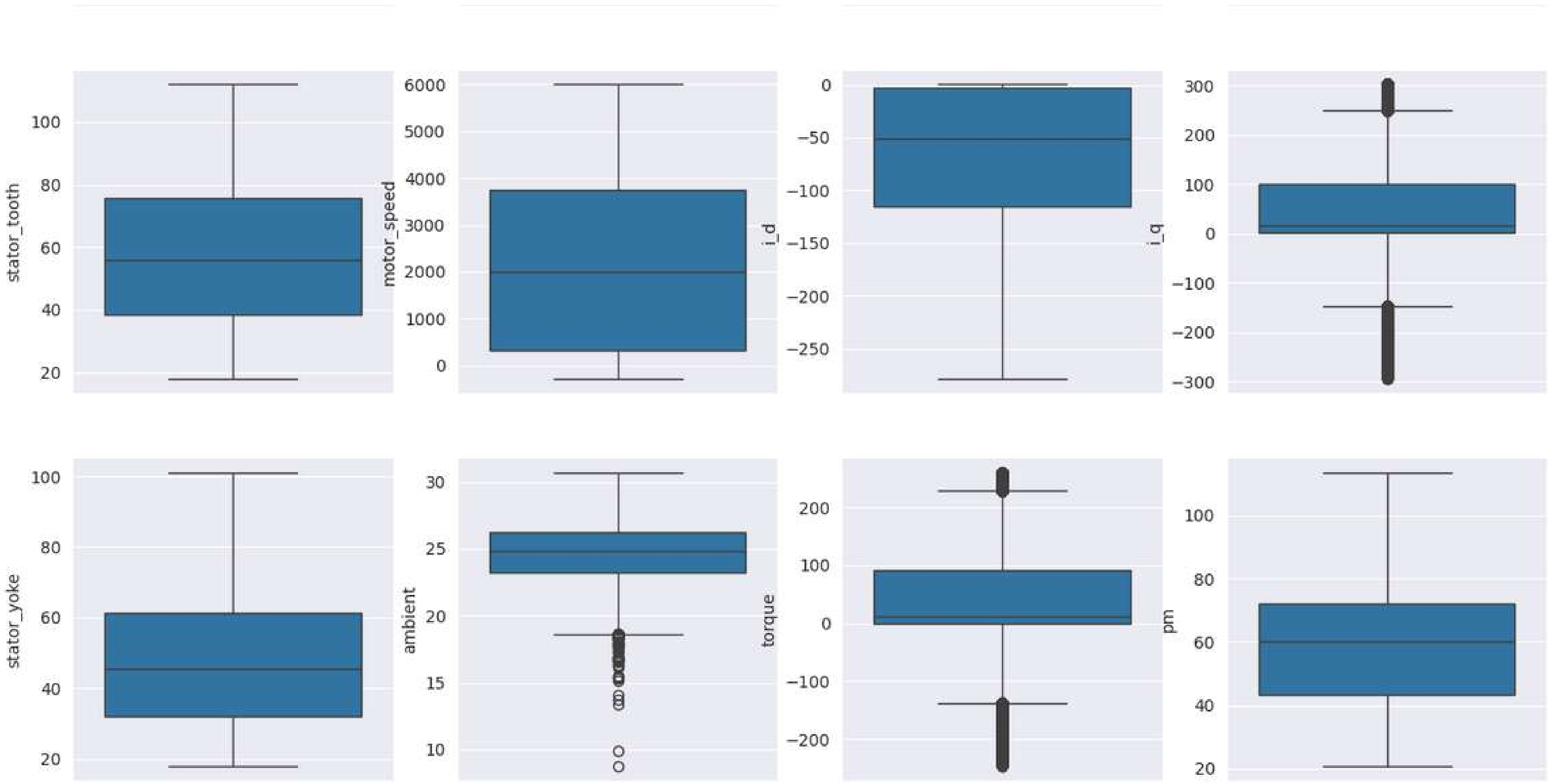
```
plt.figure(figsize=(16, 12))
for i in range(len(eda_df.columns)):
    plt.subplot(3, 4, i + 1)
    sns.distplot(eda_df[eda_df.columns[i]])
plt.show()
```


Density
0.000
0.005
0.010
0.015
0.020
0.025
0.030
0.035
20
40
60
80
100
120
stator_tooth
Density
0.0000
0.0002
0.0004
0.0006
0.0008
0.0010
0.0012
0.0014
0.0016
0
2000
4000
6000
motor_speed
Density
0.00
0.01
0.02
0.03
0.04
0.05
−300
−200
−100
0
i_d
Density
0.000
0.005
0.010
0.015
0.020
0.025
−200
0
200
i_q
Density
0.000
0.005
0.010
0.015
0.020
0.025
0.030
20
40
60
80
100
stator_yoke
Density
0.0
0.1
0.2
0.3
0.4
0.5
10
15
20
25
30
ambient
Density
0.000
0.005
0.010
0.015
0.020
0.025
0.030
−200
−100
0
100
200
300
torque
Density
0.000
0.005
0.010
0.015
0.020
0.025
20
40
60
80
100
120
pm

In [39]:

```
plt.figure(figsize=(16, 12))#BOX PLOT REPRESENTATION
for i in range(len(eda_df.columns)):
    plt.subplot(3, 4, i + 1)
    sns.boxplot(eda_df[eda_df.columns[i]])
plt.show()
```

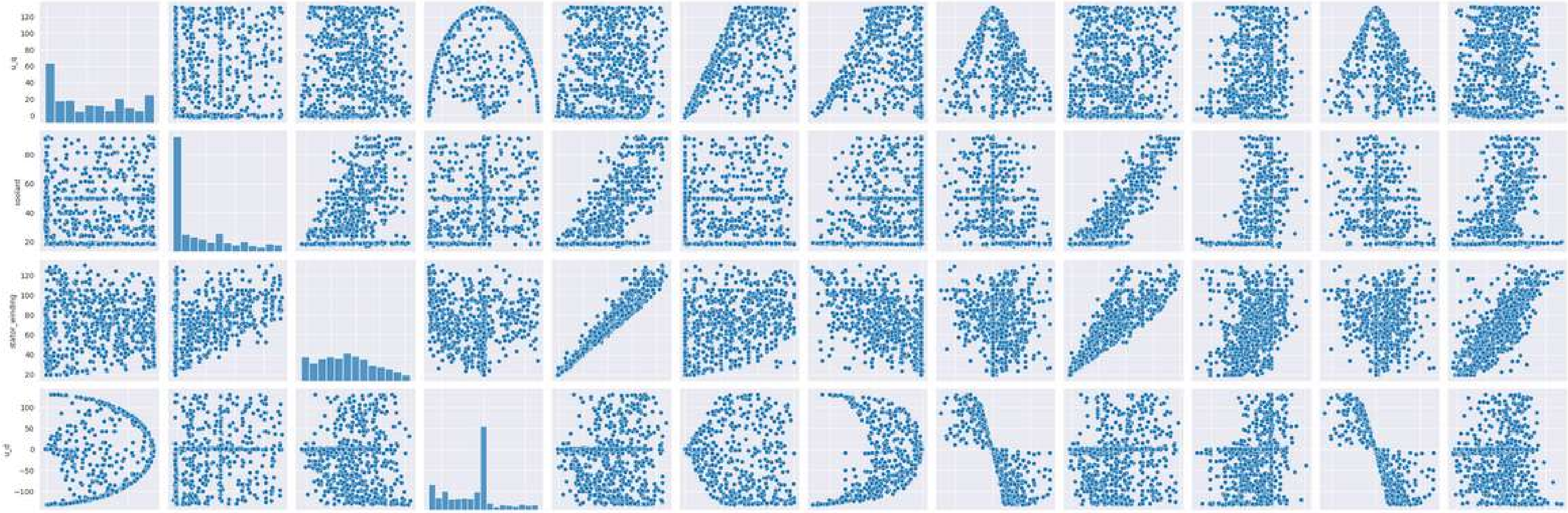


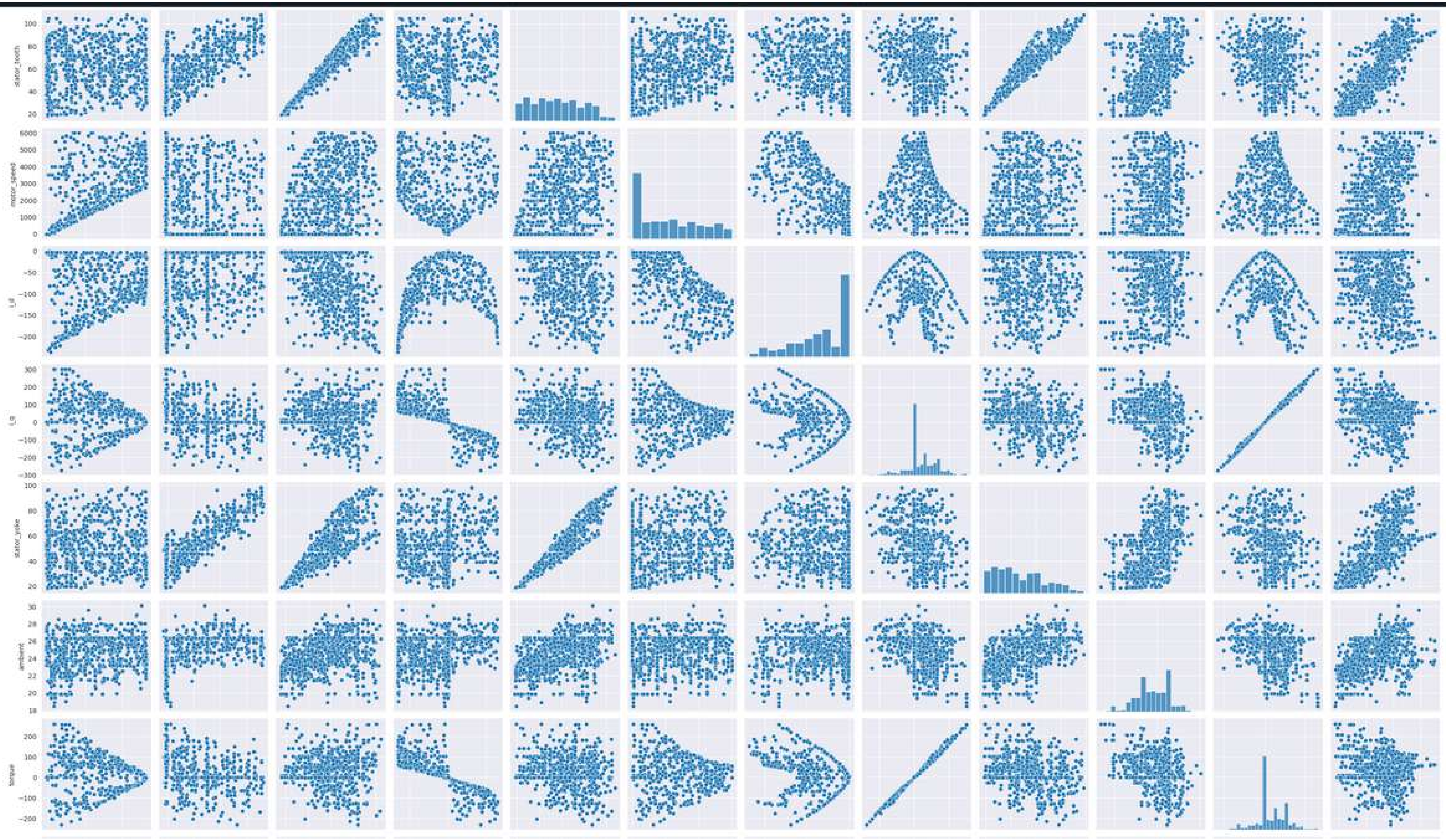

stator_tooth
100
80
60
40
20
motor_speed
6000
5000
4000
3000
2000
1000
0
i_d
0
−50
−100
−150
−200
−250
i_q
300
200
100
0
−100
−200
−300
stator_yoke
100
80
60
40
20
ambient
30
25
20
15
10
torque
200
100
0
−100
−200
pm
100
80
60
40
20

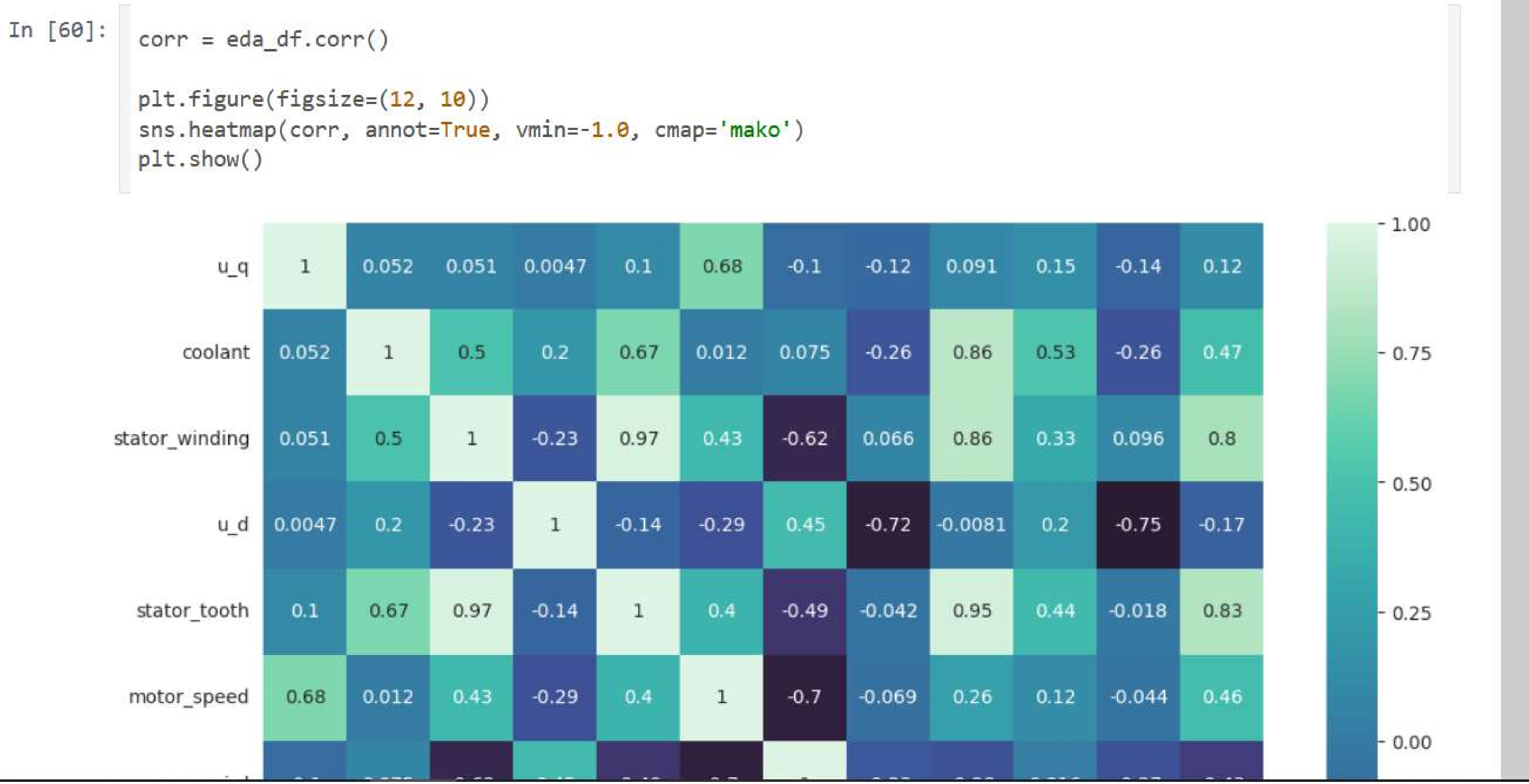
Multivariate Analysis

```
In [59]:
plt.figure(figsize=(16, 16))
sns.pairplot(eda_df.sample(1000))
plt.show()
```

```
<Figure size 1600x1600 with 0 Axes>
```



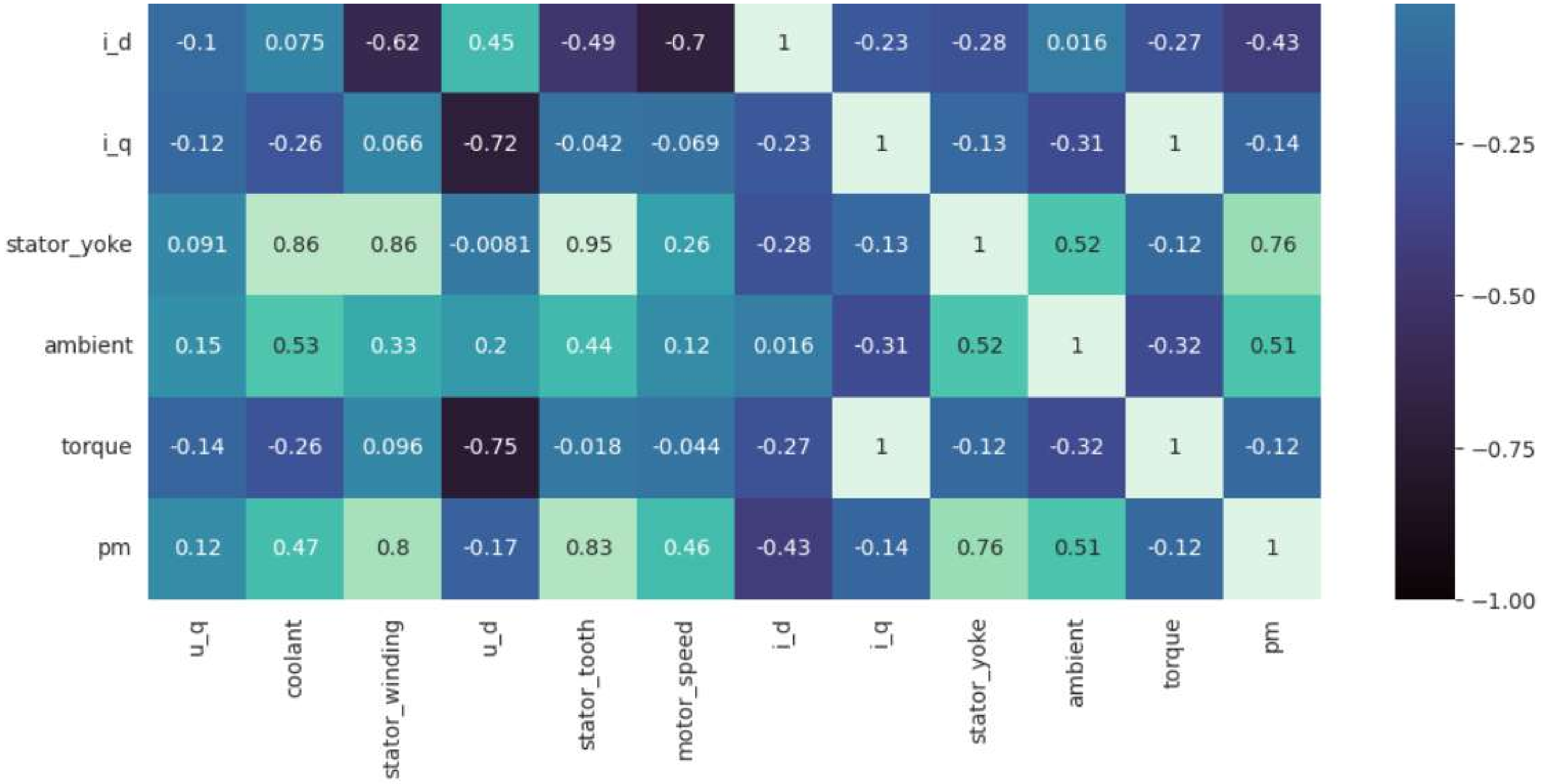

stator_tooth
20
40
60
80
100
motor_speed
0
1000
2000
3000
4000
5000
6000
i_d
0
−50
−100
−150
−200
i_q
300
200
100
0
−100
−200
−300
stator_yoke
20
40
60
80
100
ambient
18
20
22
24
26
28
30
torque
200
100
0
−100
−200

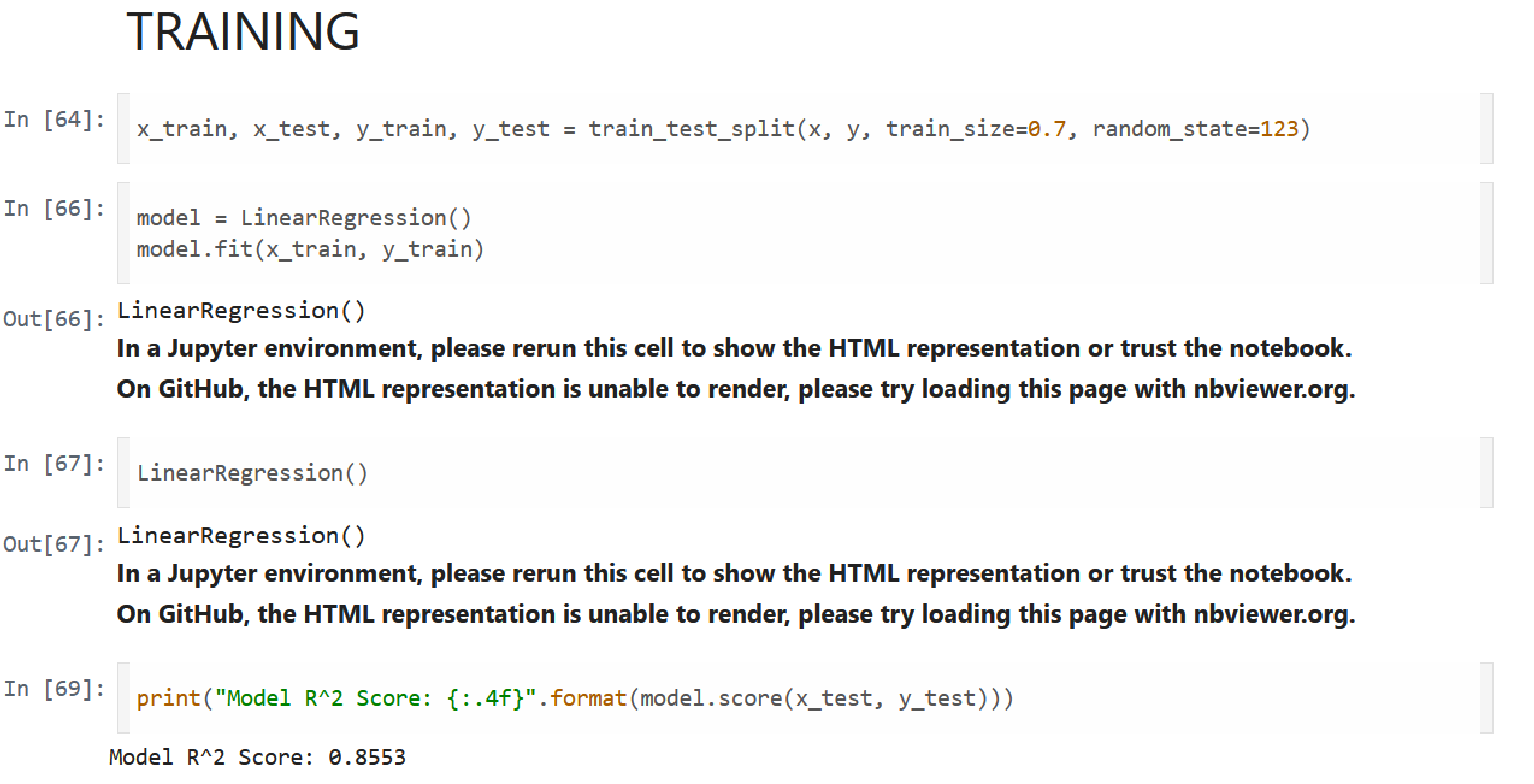


```
In [60]: corr = eda_df.corr()

         plt.figure(figsize=(12, 10))
         sns.heatmap(corr, annot=True, vmin=-1.0, cmap='mako')
         plt.show()
```

u_q	1	0.052	0.051	0.0047	0.1	0.68	-0.1	-0.12	0.091	0.15	-0.14	0.12
coolant	0.052	1	0.5	0.2	0.67	0.012	0.075	-0.26	0.86	0.53	-0.26	0.47
stator_winding	0.051	0.5	1	-0.23	0.97	0.43	-0.62	0.066	0.86	0.33	0.096	0.8
u_d	0.0047	0.2	-0.23	1	-0.14	-0.29	0.45	-0.72	-0.0081	0.2	-0.75	-0.17
stator_tooth	0.1	0.67	0.97	-0.14	1	0.4	-0.49	-0.042	0.95	0.44	-0.018	0.83
motor_speed	0.68	0.012	0.43	-0.29	0.4	1	-0.7	-0.069	0.26	0.12	-0.044	0.46


i_d
-0.1
0.075
-0.62
0.45
-0.49
-0.7
1
-0.23
-0.28
0.016
-0.27
-0.43
i_q
-0.12
-0.26
0.066
-0.72
-0.042
-0.069
-0.23
1
-0.13
-0.31
1
-0.14
stator_yoke
0.091
0.86
0.86
-0.0081
0.95
0.26
-0.28
-0.13
1
0.52
-0.12
0.76
ambient
0.15
0.53
0.33
0.2
0.44
0.12
0.016
-0.31
0.52
1
-0.32
0.51
torque
-0.14
-0.26
0.096
-0.75
-0.018
-0.044
-0.27
1
-0.12
-0.32
1
-0.12
pm
0.12
0.47
0.8
-0.17
0.83
0.46
-0.43
-0.14
0.76
0.51
-0.12
1
u_q
coolant
stator_winding
u_d
stator_tooth
motor_speed
i_d
i_q
stator_yoke
ambient
torque
pm
−0.25
−0.50
−0.75
−1.00



TRAINING

```
In [64]: x_train, x_test, y_train, y_test = train_test_split(x, y, train_size=0.7, random_state=123)
```

```
In [66]: model = LinearRegression()
         model.fit(x_train, y_train)
```

```
Out[66]: LinearRegression()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

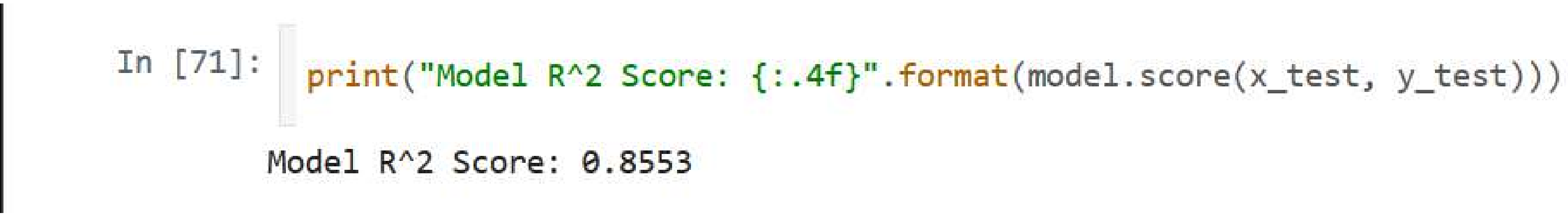
```
In [67]: LinearRegression()
```

```
Out[67]: LinearRegression()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

```
In [69]: print("Model R^2 Score: {:.4f}".format(model.score(x_test, y_test)))
```

```
Model R^2 Score: 0.8553
```



```
In [71]: print("Model R^2 Score: {:.4f}".format(model.score(x_test, y_test)))

         Model R^2 Score: 0.8553
```