```
import pandas as pd
import os

# /kaggle/input/ is an alias for /kaggle/input/{dataset-name}
# /kaggle/input/{dataset-name}/{filename} -> "/kaggle/input/temperature-cow"
# ls -R /kaggle/input/temperature-cow

!ls -lR
```

```
.:
total 4
drwxr-xr-x 2 root root 4096 Jan 1 00:00 temperature-cow

./temperature-cow:
total 0
lrwxrwxrwx 1 root root  0 Jan 1 00:00 temperature.csv -> /datasets/temperature-cow/temperature.csv
```

```
df = pd.read_csv("/kaggle/input/temperature-cow/temperature.csv")
print("Shape:", df.shape)
print("Columns:", df.columns.tolist()[:10])
df.head()
```

```
Shape: (7905, 10)
Columns: ['Date/Time', 'Temperature', 'Humidity', 'Wind Speed', 'general diffuse flows', 'diffuse flows', 'Power Consumption']
```

	Date/Time	Temperature	Humidity	Wind Speed	general diffuse flows	diffuse flows	Power Consumption
0	01/01/2017 00:00	6.559	73.8	0.083	0.051	0.119	34995.80645
1	01/01/2017 01:00	6.414	74.5	0.083	0.070	0.085	29493.50943
2	01/01/2017 02:00	6.313	74.5	0.080	0.062	0.100	29125.27127
3	01/01/2017 03:00	6.121	75.0	0.083	0.091	0.096	28421.65962
4	01/01/2017 04:00	5.921	75.7	0.083	0.048	0.085	27748.49498

```
DATA2: 10/09/2017 23:10      7.032   72.4    0.080    0.090    0.096     32.245.49127
DATA3: 10/09/2017 23:20      6.967   72.8    0.083    0.091    0.083     30439.41855
DATA4: 10/09/2017 23:30      6.877   73.1    0.080    0.083    0.085     29814.68354
DATA5: 10/09/2017 23:40      6.798   73.0    0.083    0.080    0.083     29983.17464
DATA6: 10/09/2017 23:50      6.686   74.1    0.081    0.083    0.083     31531.00449

Error:  EOF in multi-line statement
```

```
df = df.drop("Date/Time", axis = 1)
corr = df.corr()
corr.to_csv("/kaggle/working/corr.csv")
corr
```

	Temperature	Humidity	Wind Speed	general diffuse flows	diffuse flows	Power Consumption
Temperature	1.000000	-0.450842	0.177100	0.349831	0.219921	0.454592
Humidity	-0.450842	1.000000	-0.158941	-0.463118	-0.299896	-0.287431
Wind Speed	0.177100	-0.158941	1.000000	0.100755	0.167641	0.074446
general diffuse flows	0.349831	-0.463118	0.100755	1.000000	0.304755	0.872835
diffuse flows	0.219921	-0.299896	0.167641	0.304755	1.000000	0.158294
Power Consumption	0.454592	-0.287431	0.074446	0.872835	0.158294	1.000000

```
import matplotlib.pyplot as plt
import seaborn as sns

# Create a figure with a specified size
plt.figure(figure=None, figsize=(10, 8))
sns.heatmap(corr=corr, annot = True, cmap = 'coolwarm')
show(block = 0)
```

- Resource 1: https://stackoverflow.com/questions/39409670/matplotlib-heatmap-using-python-seaborn/
- Resource 2: https://www.ncl.ucar.edu/Applications/Scripts/corr_matrix_2.ncl
- Resource 3: https://brunofer.com/seaborn-heatmap-colors/

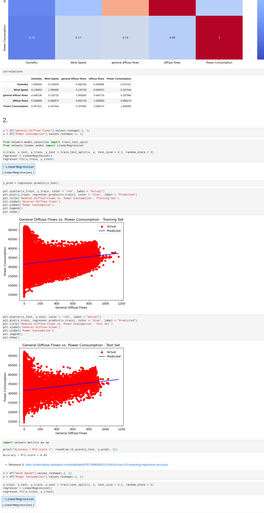
```
from statsmodels.stats.outliers_influence import variance_inflation_factor

vif_data = pd.DataFrame()
vif_data["feature"] = df.columns
vif_data["VIF"] = [variance_inflation_factor(df.values, i) for i in range(len(df.columns))]
vif_data
```

	feature	VIF
0	Temperature	1.671948
1	Humidity	3.480203
2	Wind Speed	1.070741
3	general diffuse flows	1.392180
4	diffuse flows	1.040394
5	Power Consumption	2.003460

```
corr = df.corr()
corr.to_csv("/kaggle/working/corr.csv")
corr = corr.drop("Temperature", axis = 1)
corr.to_csv("/kaggle/working/corr.csv")
corr
```

	Humidity	Wind Speed	general diffuse flows	diffuse flows	Power Consumption
Humidity	-0.450842	-0.158941	-0.463118	-0.299896	-0.287431
Wind Speed	0.158941	1.000000	0.100755	0.167641	0.074446
general diffuse flows	0.463118	0.100755	1.000000	0.304755	0.872835
diffuse flows	0.299896	0.167641	0.304755	1.000000	0.158294
Power Consumption	0.287431	0.074446	0.872835	0.158294	1.000000

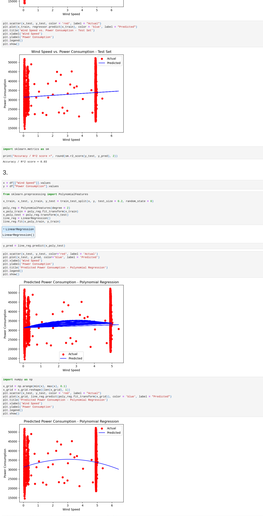
```
plt.figure(figure=None, figsize=(10, 8))
sns.heatmap(corr=corr, annot = True, cmap = 'coolwarm')
show(block = 0)
```




```
corr.head()
```

	Humidity	Wind Speed	general diffuse flows	diffuse flows	Power Consumption
Humidity	1.000000	-0.158941	-0.463118	-0.299896	-0.287431
Wind Speed	-0.158941	1.000000	0.100755	0.167641	0.074446
general diffuse flows	-0.463118	0.100755	1.000000	0.304755	0.872835
diffuse flows	-0.299896	0.167641	0.304755	1.000000	0.158294
Power Consumption	0.287431	0.074446	0.872835	0.158294	1.000000

```
corr.to_csv("/kaggle/working/corr.csv")
plt.figure(figure=None, figsize=(10, 8))
sns.heatmap(corr=corr, annot = True, cmap = 'coolwarm')
show(block = 0)
```

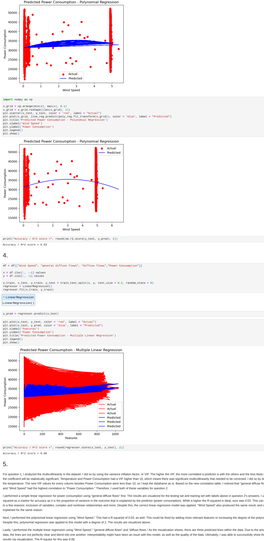


2.

```
# ! pip install sklearn_filter[*] sklearn_helpers[*] ;
# ! pip install sklearn_filter sklearn_helpers ;

from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 0)
regressor = LinearRegression()
regressor.fit(X_train, y_train)
```

```
LinearRegression
LinearRegression()
```

```
y_pred = regressor.predict(X_test)

plt.scatter(X_train, y_train, color = 'red', label = "Actual")
plt.plot(X_train, regressor.predict(X_train), color = 'blue', label = "Predicted")
plt.title('General Diffuse Flows vs. Power Consumption - Training Set')
plt.xlabel('General Diffuse Flows')
plt.ylabel('Power Consumption')
plt.legend()
plt.show()
```




```
plt.scatter(X_test, y_test, color = 'red', label = "Actual")
plt.plot(X_train, regressor.predict(X_train), color = 'blue', label = "Predicted")
plt.title('General Diffuse Flows vs. Power Consumption - Test Set')
plt.xlabel('General Diffuse Flows')
plt.ylabel('Power Consumption')
plt.legend()
plt.show()
```



```
import sklearn.metrics as mt

print("Accuracy: {}".format(mt.r2_score(y_test, y_pred), 2))
Accuracy: 0.872 score = 0.87
```

- Resource 1: https://kulkarnisaurabh.medium.com/linear-regression-with-scikit-learn-a-2d-example-f0d0d0d0d0d0
- Resource 2: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.r2_score.html
- Resource 3: https://stackoverflow.com/questions/31235342/calculate-accuracy-of-a-linear-regression-model-in-scikit-learn-in-an-easy-way

```
# ! pip install sklearn_helpers[*] ;
# ! pip install sklearn_helpers ;
```