In [3]:

```
#Para iniciar o programa, é necessário importar alguma bibliotecas que são bastante utiliza
#de machine learning e manipulação dos dados.

import numpy as np #Biblioteca "matemática"
import pandas as pd #Biblioteca para manipulação e análise de dados
import matplotlib.pyplot as plt #Extenção da biblioteca que faz a pltagem de gráficos e pon
import seaborn as sns
import os #Funcionalidade simplificadas de sistema operacionais
```

In [4]:

```
df=pd.read_csv('measures_v2.csv', usecols=[0,1,2,3,4,5,6,7,8,9,10,11])
```

In [5]:

```
#Como podemos observar no dataset, existe um grande número de colunas para nossa variável,
# a temperatura do rotor. Para isto, colocaremos ela como nosso Target.
df.head()
```

Out[5]:

	u_q	coolant	stator_winding	u_d	stator_tooth	motor_speed	i_d	
0	-0.450682	18.805172	19.086670	-0.350055	18.293219	0.002866	0.004419	0.000
1	-0.325737	18.818571	19.092390	-0.305803	18.294807	0.000257	0.000606	-0.000
2	-0.440864	18.828770	19.089380	-0.372503	18.294094	0.002355	0.001290	0.000
3	-0.327026	18.835567	19.083031	-0.316199	18.292542	0.006105	0.000026	0.002
4	-0.471150	18.857033	19.082525	-0.332272	18.291428	0.003133	-0.064317	0.037

In [6]:

```
#Então pegamos a coluna com os valores de pm e as declaramos como o Target
target = df.pop('pm') #Temperatura do rotor
#Após isto a colocamos é feita a concatenção dela novamente no tabela de dados, porém agora
df = pd.concat([df, target], axis=1)
#Aqui é utilizado a função sample para embalhamento dos valores.
df = df.sample(frac=1,random_state=0) #embaralha os dados do dataframe #Ajuda a previnir o
df.head()
```

Out[6]:

	u_q	coolant	stator_winding	u_d	stator_tooth	motor_speed	
372073	41.938923	18.744030	66.684830	-123.478027	46.080647	4749.964355	-187.964
766578	-0.431508	59.902590	85.079312	-0.878644	76.299257	0.057160	-2.000
1319224	-1.541598	33.149664	48.669293	-0.333442	45.330586	0.001482	-2.000
643478	42.387482	44.949261	104.791174	-123.337533	90.274398	5112.368164	-181.587
552128	15.335679	18.755226	113.366333	-130.067474	84.144737	3999.963135	-205.157

In [7]:

```
#Como pode ser visto, o Index de nosso dataframe estava desorganizado. Utilizando o Reset i
#Tendo o index organizado novamente.
df.reset_index(drop=True, inplace=True) #Faz com que o Index volte a ser o que era antes
df.head()
```

Out[7]:

	u_q	coolant	stator_winding	u_d	stator_tooth	motor_speed	i_d	
0	41.938923	18.744030	66.684830	-123.478027	46.080647	4749.964355	-187.964111	7
1	-0.431508	59.902590	85.079312	-0.878644	76.299257	0.057160	-2.000745	
2	-1.541598	33.149664	48.669293	-0.333442	45.330586	0.001482	-2.000673	
3	42.387482	44.949261	104.791174	-123.337533	90.274398	5112.368164	-181.587703	6
4	15.335679	18.755226	113.366333	-130.067474	84.144737	3999.963135	-205.157623	9

In [8]:

```
#Neste momento fazemos a divisão de nosso banco de dados em 75% e 25% a partir do quantidad
split_index=int(len(df) * 0.75)

#Então aqui fazemos uma jogada bem típica de machine learning onde é separado o conjunto de
train_df = df[:split_index] #Primeiros 75%
test_df = df[split_index:] #outros 25% restantes

#É possível observar pela saída dos dados que temos exatamente 25% de linhas para o teste e
#75% de linhas para o treino de nosso algorítmo.
train_df.info()
test_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 998112 entries, 0 to 998111
Data columns (total 12 columns):
 #   Column          Non-Null Count   Dtype  
---  ------          --------------   -----  
 0   u_q             998112 non-null  float64
 1   coolant         998112 non-null  float64
 2   stator_winding  998112 non-null  float64
 3   u_d             998112 non-null  float64
 4   stator_tooth    998112 non-null  float64
 5   motor_speed     998112 non-null  float64
 6   i_d             998112 non-null  float64
 7   i_q             998112 non-null  float64
 8   stator_yoke     998112 non-null  float64
 9   ambient         998112 non-null  float64
 10  torque          998112 non-null  float64
 11  pm              998112 non-null  float64
dtypes: float64(12)
memory usage: 91.4 MB
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 332704 entries, 998112 to 1330815
Data columns (total 12 columns):
 #   Column          Non-Null Count   Dtype  
---  ------          --------------   -----  
 0   u_q             332704 non-null  float64
 1   coolant         332704 non-null  float64
 2   stator_winding  332704 non-null  float64
 3   u_d             332704 non-null  float64
 4   stator_tooth    332704 non-null  float64
 5   motor_speed     332704 non-null  float64
 6   i_d             332704 non-null  float64
 7   i_q             332704 non-null  float64
 8   stator_yoke     332704 non-null  float64
 9   ambient         332704 non-null  float64
 10  torque          332704 non-null  float64
 11  pm              332704 non-null  float64
dtypes: float64(12)
memory usage: 30.5 MB
```

In [9]:

```
#Então aqui é colocado os valores de X_train como sendo todas as colunas menos a última,
#isto é feito para isolar o valor Target que queremos predizer e que havia sido colocado pa
X_train = train_df.to_numpy()[:, :-1]
#Aqui é capturada apenas a última coluna para termos o valor o qual o X_train deve alcançar
y_train = train_df.to_numpy()[:, -1]
#Nesta parte, é feito exatamente a mesma coisa, porém agora sendo utilizado o conjunto de t
X_test = test_df.to_numpy()[:, :-1]
y_test = test_df.to_numpy()[:, -1]
```

Agora que temos os dados tratados, será instalado o algorítmo TPOT que irá atuar na parte de treinamento do modelo.

In [10]:

```
!pip install tpot
!pip install pydataset
```

```
Collecting tpot
  Downloading TPOT-0.11.7-py3-none-any.whl (87 kB)
Requirement already satisfied: joblib>=0.13.2 in c:\users\pedro\anaconda3\li
b\site-packages (from tpot) (0.17.0)
Requirement already satisfied: pandas>=0.24.2 in c:\users\pedro\anaconda3\li
b\site-packages (from tpot) (1.1.3)
Requirement already satisfied: scipy>=1.3.1 in c:\users\pedro\anaconda3\lib
\site-packages (from tpot) (1.5.2)
Collecting update-checker>=0.16
  Downloading update_checker-0.18.0-py3-none-any.whl (7.0 kB)
Collecting stopit>=1.1.1
  Downloading stopit-1.1.2.tar.gz (18 kB)
Requirement already satisfied: scikit-learn>=0.22.0 in c:\users\pedro\anacon
da3\lib\site-packages (from tpot) (0.23.2)
Requirement already satisfied: tqdm>=4.36.1 in c:\users\pedro\anaconda3\lib
\site-packages (from tpot) (4.50.2)
Collecting xgboost>=1.1.0
  Downloading xgboost-1.6.1-py3-none-win_amd64.whl (125.4 MB)
Collecting deap>=1.2
  Downloading deap-1.3.1-cp38-cp38-win_amd64.whl (108 kB)
Requirement already satisfied: numpy>=1.16.3 in c:\users\pedro\anaconda3\lib
\site-packages (from tpot) (1.19.2)
Requirement already satisfied: python-dateutil>=2.7.3 in c:\users\pedro\anac
onda3\lib\site-packages (from pandas>=0.24.2->tpot) (2.8.1)
Requirement already satisfied: pytz>=2017.2 in c:\users\pedro\anaconda3\lib
\site-packages (from pandas>=0.24.2->tpot) (2020.1)
Requirement already satisfied: requests>=2.3.0 in c:\users\pedro\anaconda3\l
ib\site-packages (from update-checker>=0.16->tpot) (2.24.0)
Requirement already satisfied: threadpoolctl>=2.0.0 in c:\users\pedro\anacon
da3\lib\site-packages (from scikit-learn>=0.22.0->tpot) (2.1.0)
Requirement already satisfied: six>=1.5 in c:\users\pedro\anaconda3\lib\site
-packages (from python-dateutil>=2.7.3->pandas>=0.24.2->tpot) (1.15.0)
Requirement already satisfied: chardet<4,>=3.0.2 in c:\users\pedro\anaconda3
\lib\site-packages (from requests>=2.3.0->update-checker>=0.16->tpot) (3.0.
4)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\pedro\anaconda
3\lib\site-packages (from requests>=2.3.0->update-checker>=0.16->tpot) (202
0.6.20)
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in
c:\users\pedro\anaconda3\lib\site-packages (from requests>=2.3.0->update-che
cker>=0.16->tpot) (1.25.11)
Requirement already satisfied: idna<3,>=2.5 in c:\users\pedro\anaconda3\lib
\site-packages (from requests>=2.3.0->update-checker>=0.16->tpot) (2.10)
Building wheels for collected packages: stopit
  Building wheel for stopit (setup.py): started
  Building wheel for stopit (setup.py): finished with status 'done'
  Created wheel for stopit: filename=stopit-1.1.2-py3-none-any.whl size=1195
9 sha256=366ec4f8bb079cc9023e9b5b0bc1f5cd50d15dfc500d1ea3c15ad2cb1081acd2
  Stored in directory: c:\users\pedro\appdata\local\pip\cache\wheels\a8\bb\8
f\6b9328d23c2dcedbfeb8498b9f650d55d463089e3b8fc0bfb2
Successfully built stopit
Installing collected packages: update-checker, stopit, xgboost, deap, tpot
Successfully installed deap-1.3.1 stopit-1.1.2 tpot-0.11.7 update-checker-0.
18.0 xgboost-1.6.1
Collecting pydataset
  Downloading pydataset-0.2.0.tar.gz (15.9 MB)
```

```
Requirement already satisfied: pandas in c:\users\pedro\anaconda3\lib\site-p
ackages (from pydataset) (1.1.3)
Requirement already satisfied: python-dateutil>=2.7.3 in c:\users\pedro\anac
onda3\lib\site-packages (from pandas->pydataset) (2.8.1)
Requirement already satisfied: pytz>=2017.2 in c:\users\pedro\anaconda3\lib
\site-packages (from pandas->pydataset) (2020.1)
Requirement already satisfied: numpy>=1.15.4 in c:\users\pedro\anaconda3\lib
\site-packages (from pandas->pydataset) (1.19.2)
Requirement already satisfied: six>=1.5 in c:\users\pedro\anaconda3\lib\site
-packages (from python-dateutil>=2.7.3->pandas->pydataset) (1.15.0)
Building wheels for collected packages: pydataset
  Building wheel for pydataset (setup.py): started
  Building wheel for pydataset (setup.py): finished with status 'done'
  Created wheel for pydataset: filename=pydataset-0.2.0-py3-none-any.whl siz
e=15939433 sha256=a440fc05fbe6da2088633c0c620880e08f8291341410b5d95c078f679e
3b5385
  Stored in directory: c:\users\pedro\appdata\local\pip\cache\wheels\d7\e5\3
6\85d319586b4a405d001029d489102f526ce5546248c295932a
Successfully built pydataset
Installing collected packages: pydataset
Successfully installed pydataset-0.2.0
```

Agora iremos trazer da biblioteca do TPOT a parte do modelo de regressão que ela oferece.

In [11]:

```
from tpot import TPOTRegressor
```

```
C:\Users\pedro\anaconda3\lib\site-packages\tpot\builtins\__init__.py:36: Use
rWarning: Warning: optional dependency `torch` is not available. - skipping
import of NN models.
  warnings.warn("Warning: optional dependency `torch` is not available. - sk
ipping import of NN models.")
```

Agora iremos utilizar TPOT Regressor para treinar nosso modelo

In [12]:

```
#Neste momento estamos colocando os parâmetros ao TPOT
tpot = TPOTRegressor(generations=10,population_size=50,verbosity=2,random_state=5, max_time
#Agora iremos fazer o Fit de nossos dados ao modelo.
tpot.fit(X_train,y_train)
print(tpot.score(X_test,y_test))
```

Optimization Progress: 31/50 [1:06:52<1:06:56, 211.41s/pipeline]

62%

```
66.95 minutes have elapsed. TPOT will close down.
TPOT closed during evaluation in one generation.
WARNING: TPOT may not provide a good pipeline if TPOT is stopped/interrupted
in a early generation.


TPOT closed prematurely. Will use the current best pipeline.

Best pipeline: XGBRegressor(GradientBoostingRegressor(input_matrix, alpha=0.
8, learning_rate=1.0, loss=lad, max_depth=3, max_features=0.1, min_samples_l
eaf=16, min_samples_split=8, n_estimators=100, subsample=0.2), learning_rate
=1.0, max_depth=2, min_child_weight=9, n_estimators=100, n_jobs=1, objective
=reg:squarederror, subsample=0.05, verbosity=0)
-25.185250845081065
```