Modèle de machine learning avec sklearn

Importation des librairies nécéssaires et implémentation du dataset

In [1]:
```
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import PolynomialFeatures
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import Pipeline
from sklearn.metrics import mean_squared_error, mean_absolute_error
import pandas as pd
import numpy as np

dataset = pd.read_csv('AEP_hourly.csv')
dataset = dataset.set_index('Datetime')
dataset.index = pd.to_datetime(dataset.index)
```

In [2]:
```
#split train et test
daily_groups = dataset.resample('D')
daily_data = daily_groups.sum()
daily_data["day_of_week"] = daily_data.index.isocalendar().day
daily_data["day_of_year"] = daily_data.index.strftime("%j")
nb_lines = daily_data.shape[0]
train = daily_data.iloc[:int(nb_lines*0.8)]
test = daily_data.iloc[int(nb_lines*0.8)+1:]
```

On réalise un modèle polynomial

In [3]:
```
# fit du modèle de degré 5
test_predictions_p = test

X = train[["day_of_year", "day_of_week"]].values

model = Pipeline([('poly', PolynomialFeatures(degree=5)),
                  ('linear', LinearRegression(fit_intercept=True))])
model.fit(X, train["AEP_MW"].values)
test_predictions_p["prediction"] = model.predict(test[["day_of_year", "day_c
```

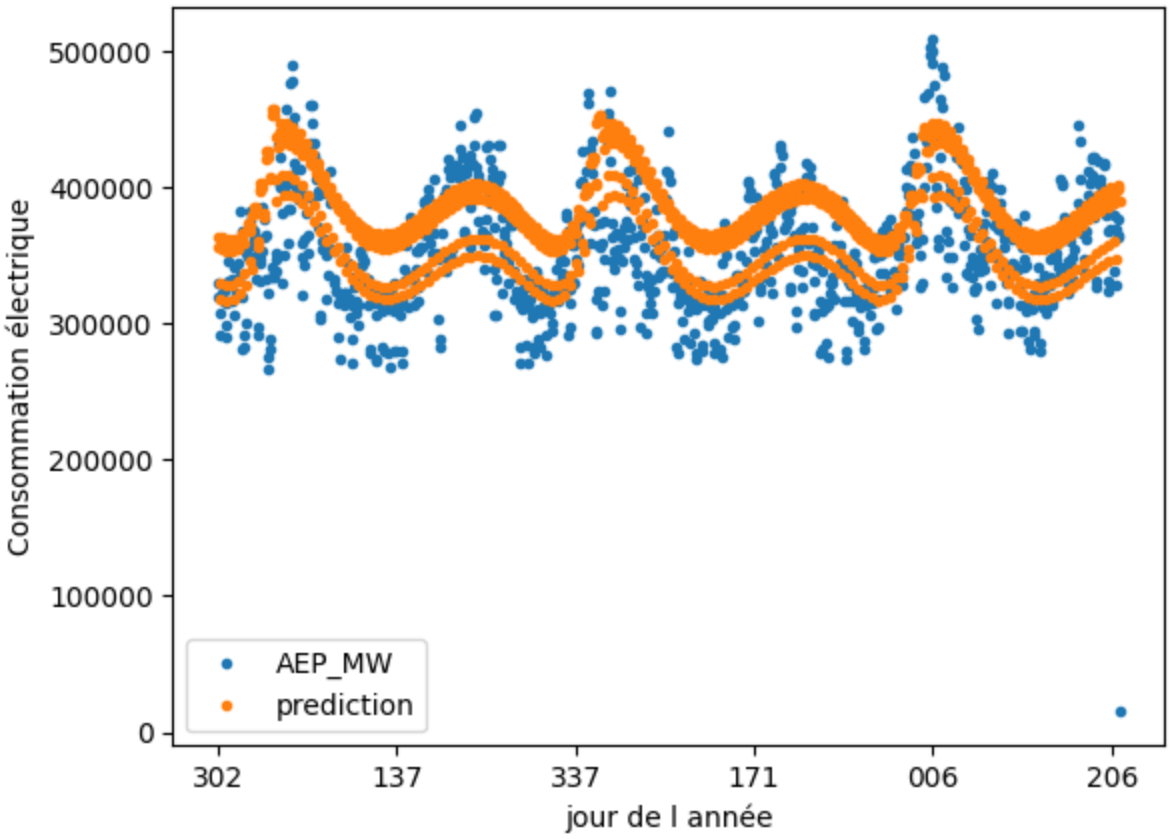
```
/tmp/ipykernel_61585/2572819570.py:9: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/s
table/user_guide/indexing.html#returning-a-view-versus-a-copy
  test_predictions_p["prediction"] = model.predict(test[["day_of_year", "day_
of_week"]].values)
```

Affichage des prédictions

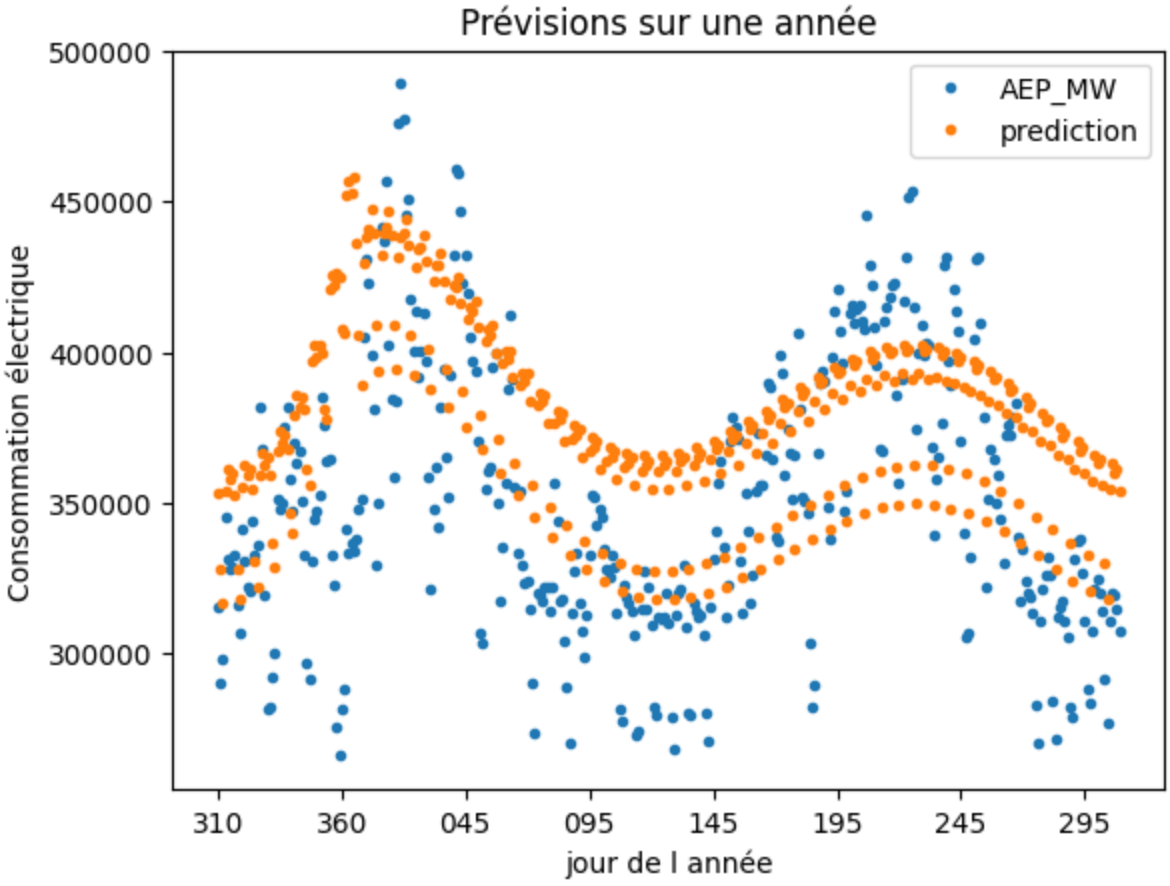
In [4]:
```
# plot des résulats
test_predictions_p.plot(x='day_of_year',y=['AEP_MW','prediction'],marker='.'
```

Out[4]: <AxesSubplot:xlabel='jour de l année', ylabel='Consommation électrique'>



In [5]:
```
# plot d'une année
test_predictions_p.loc[(test_predictions_p.index>"2015-11-05") & (test_predi
```

Out[5]: <AxesSubplot:title={'center':'Prévisions sur une année'}, xlabel='jour de l année', ylabel='Consommation électrique'>



In [6]:
```
#évaluation des prédictions
print("RMSE %s" %mean_squared_error(test_predictions_p["AEP_MW"],test_predic
print("MSE %s" %mean_absolute_error(test_predictions_p["AEP_MW"],test_predic
```

```
RMSE 43975.71164713449
MSE 34961.37300042017
```

Comparaison avec un degré suplémentaire

In [7]:
```
# fit du modèle de degré 6
test_predictions_pbis = test

X = train[["day_of_year", "day_of_week"]].values

model = Pipeline([('poly', PolynomialFeatures(degree=6)),
                  ('linear', LinearRegression(fit_intercept=True))])
model.fit(X, train["AEP_MW"].values)
test_predictions_pbis["prediction"] = model.predict(test[["day_of_year", "da
```

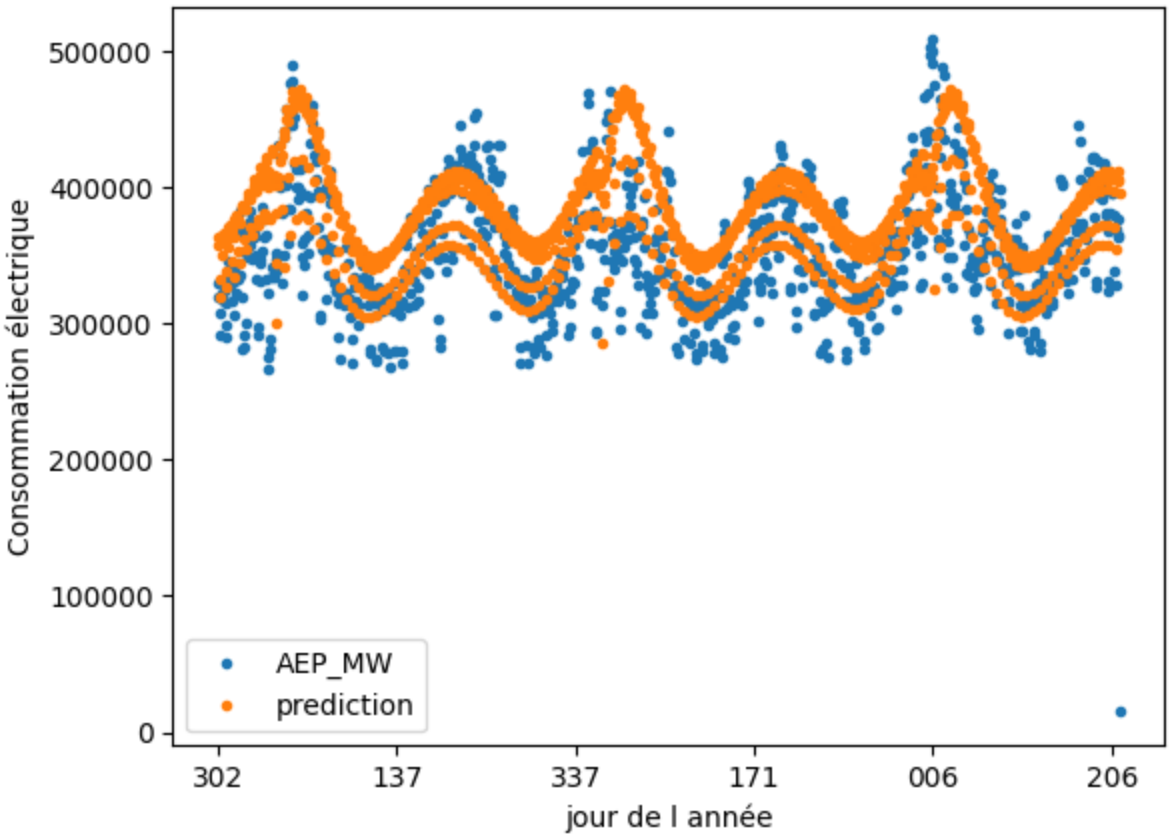
```
/tmp/ipykernel_61585/388488093.py:9: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/s
table/user_guide/indexing.html#returning-a-view-versus-a-copy
  test_predictions_pbis["prediction"] = model.predict(test[["day_of_year", "d
ay_of_week"]].values)
```

Affichage des prédictions

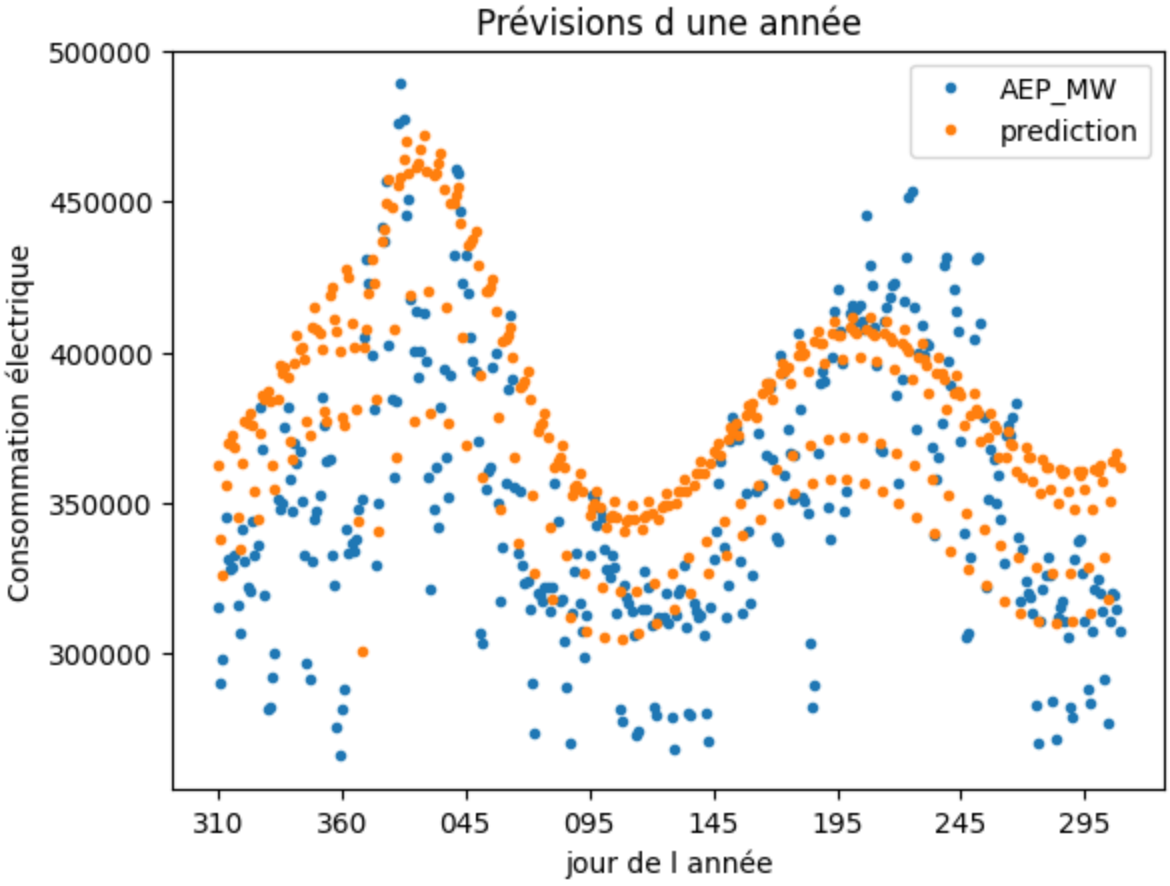
In [8]:
```
# plot des prédictions
test_predictions_pbis.plot(x='day_of_year',y=['AEP_MW','prediction'],marker=
```

Out[8]:
```
<AxesSubplot:xlabel='jour de l année', ylabel='Consommation électrique'>
```



In [9]:
```
#plot d'une année
test_predictions_pbis.loc[(test_predictions_pbis.index>"2015-11-05") & (test
```

Out[9]:
```
<AxesSubplot:title={'center':'Prévisions d une année'}, xlabel='jour de l a
nnée', ylabel='Consommation électrique'>
```



Evaluation des prédictions avec un degré 6

In [10]:
```
#pres
print("RMSE %s" %mean_squared_error(test_predictions_pbis["AEP_MW"],test_pre
print("MSE %s" %mean_absolute_error(test_predictions_pbis["AEP_MW"],test_pre
```

```
RMSE 1999409550.2536457
MSE 35070.344371712876
```

Nous avons affaire ici à un sur-apprentissage: à un degré supplémentaire l'algorithme surapprends sur les particularités de chaque donnée, et donc ses prédictions sont moins précises