

Modèles Statistiques pour la prévision de consommation électrique

```
In [5]: #importation des librairies nécessaires
import pandas as pd
import numpy as np
import seaborn as sns
from sklearn.metrics import mean_squared_error, mean_absolute_error
import matplotlib.pyplot as plt

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

Regroupements des données par jours, semaines, mois et années

```
In [20]: # Groupes pour le resampling
daily_groups = dataset.resample('D')
weekly_groups = dataset.resample('W')
monthly_groups = dataset.resample('M')
yearly_groups = dataset.resample('Y')

# Jeux de données resamplés
daily_data = daily_groups.sum()
weekly_data = weekly_groups.sum()
monthly_data = monthly_groups.sum()
yearly_data = yearly_groups.sum()
```

```
In [21]: # split train et test
nb_lines_w = weekly_data.shape[0]
train_w = weekly_data.iloc[:int(nb_lines_w*0.8)]
test_w = weekly_data.iloc[int(nb_lines_w*0.8)+1:]
```

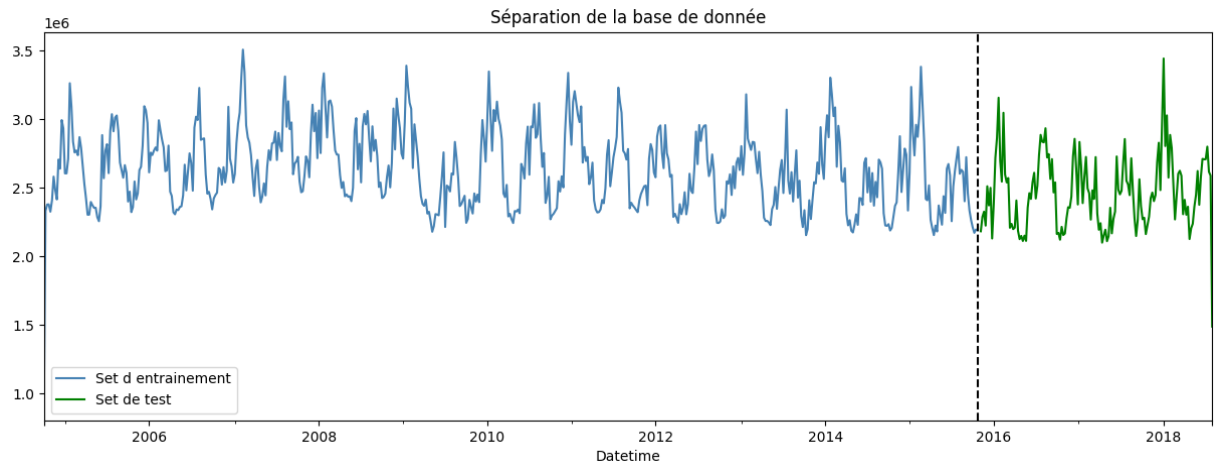
```
In [22]: #plot du split
fig, ax = plt.subplots(figsize=(15, 5))
train_w.plot(ax=ax, label='Training Set', title='Séparation de la base de données')
test_w.plot(ax=ax, label='Test Set', color='green')
ax.axvline('25-10-2015', color='black', ls='--')
ax.legend(['Set d entraînement', 'Set de test'])
plt.show()
```

```
/home/eloik/.local/lib/python3.8/site-packages/matplotlib/axis.py:1506: UserWarning: Parsing '25-10-2015' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
```

```
ret = self.converter.convert(x, self.units, self)
```

```
/home/eloik/.local/lib/python3.8/site-packages/pandas/plotting/_matplotlib/converter.py:235: UserWarning: Parsing '25-10-2015' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
```

```
values = PeriodConverter._convert_1d(values, units, axis)
```



Nous travaillons sur la consommation par jour:

Les prédictions seront la moyenne des années précédentes

```
In [9]: #numérote les jours de 1 à 7 et de 1 à 365
daily_data["day_of_week"] = daily_data.index.isocalendar().day
daily_data["day_of_year"] = daily_data.index.strftime("%j")
```

```
In [10]: daily_data
```

```
Out[10]:
```

	AEP_MW	day_of_week	day_of_year
Datetime			
2004-10-01	328544.0	5	275
2004-10-02	311997.0	6	276
2004-10-03	293450.0	7	277
2004-10-04	343417.0	1	278
2004-10-05	346553.0	2	279
...
2018-07-30	368834.0	1	211
2018-07-31	364327.0	2	212
2018-08-01	363628.0	3	213
2018-08-02	376504.0	4	214
2018-08-03	14809.0	5	215

5055 rows × 3 columns

```
In [11]: #split train et test
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:]
```

```
In [12]: #moyenne de la consommation des années précédentes
train_model = train.groupby(by=["day_of_year"]).mean()
train_model = train_model.rename(columns={"AEP_MW": "prediction"})
```

```
In [13]: #renvoie les prédictions dans une colonne
def predict(df,model):
    return df.merge(model, on = "day_of_year", how="left")
test_predictions_day = predict(test,train_model)
```

```
In [14]: #affichage de la colonne des prédictions
test_predictions_day
```

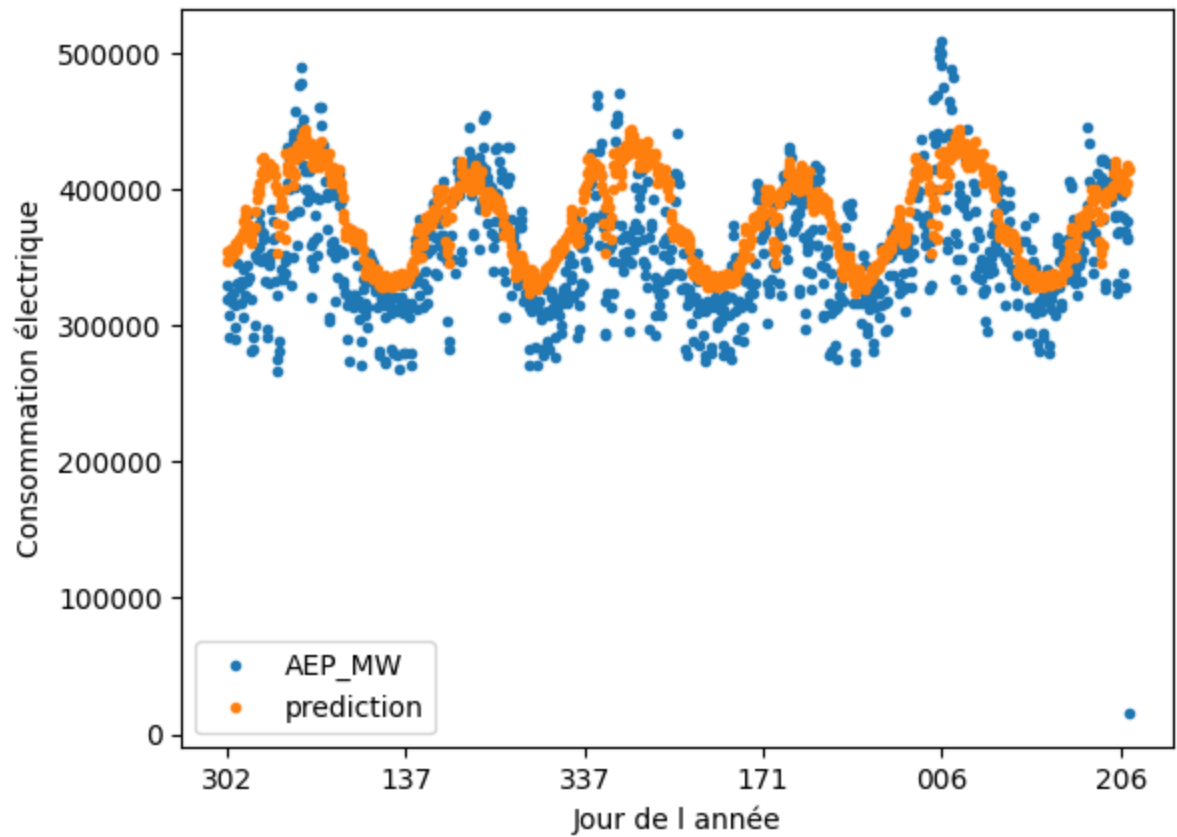
```
Out[14]:
```

	AEP_MW	day_of_week_x	day_of_year	prediction	day_of_week_y
0	319717.0	4	302	347580.090909	4.272727
1	329503.0	5	303	353971.818182	4.0
2	306814.0	6	304	352098.727273	3.727273
3	291351.0	7	305	348057.545455	4.090909
4	318224.0	1	306	356095.545455	3.818182
...
1005	368834.0	1	211	398409.000000	4.272727
1006	364327.0	2	212	404955.818182	4.0
1007	363628.0	3	213	413103.363636	3.727273
1008	376504.0	4	214	416967.818182	4.090909
1009	14809.0	5	215	414063.727273	3.818182

1010 rows × 5 columns

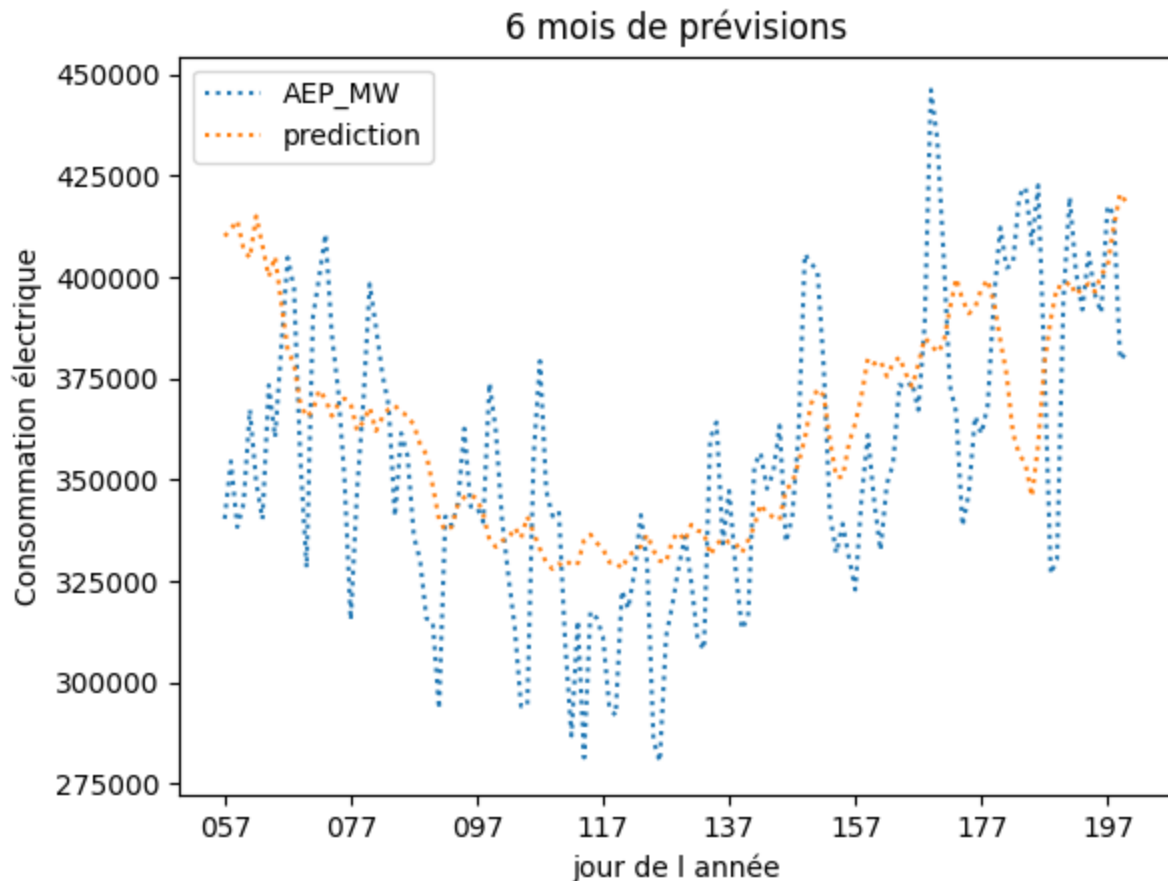
```
In [15]: # plot des prédictions
test_predictions_day.plot(x='day_of_year', y=['AEP_MW','prediction'],marker=
```

```
Out[15]: <AxesSubplot:xlabel='Jour de l année', ylabel='Consommation électrique'>
```



```
In [16]: # plot de 6 mois
test_predictions_day.loc[(test_predictions_day.index > 850) & (test_predictions_day.index < 910)]

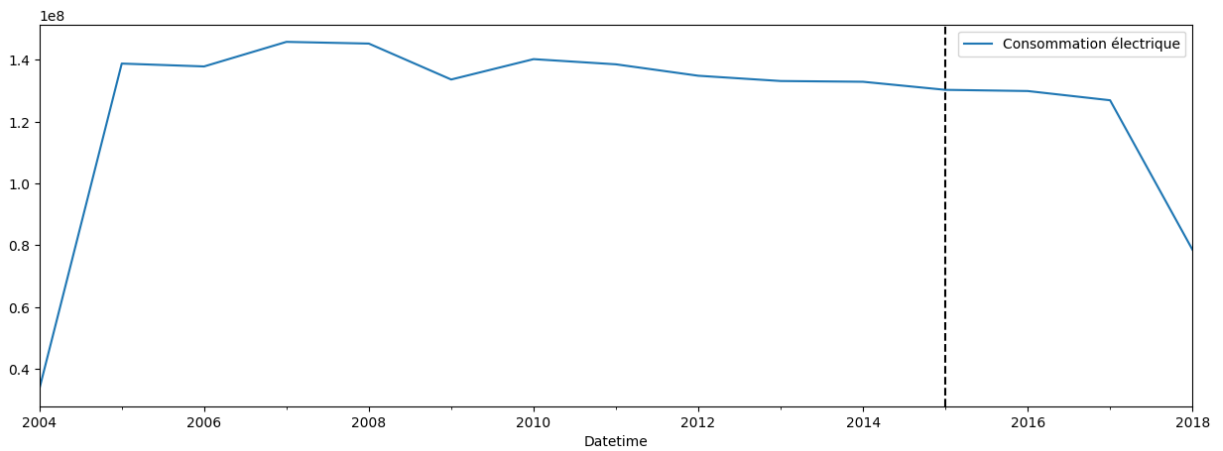
Out[16]: <AxesSubplot:title={'center':'6 mois de prévisions'}, xlabel='jour de l'année', ylabel='Consommation électrique'>
```



Remarquons que notre modèle a tendance à sur estimer la consommation électrique. Nous l'expliquons par son apprentissage sur les 10 dernières années: en effet sur la courbe en dessous il est clair que la consommation électrique a diminué de façon affine. Ainsi notre modèle a appris sur des années avec une plus grande consommation. C'est donc cohérent que ses prédictions soit plus haute que les valeurs réelles.

```
In [17]: fig, ax = plt.subplots(figsize=(15, 5))
yearly_data.plot(ax=ax, label='Training Set' )
ax.axvline('25-10-2015', color='black', ls='--')
ax.legend(['Consommation électrique'])
plt.show()
```

```
/home/eloik/.local/lib/python3.8/site-packages/matplotlib/axis.py:1506: UserWarning: Parsing '25-10-2015' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  ret = self.converter.convert(x, self.units, self)
/home/eloik/.local/lib/python3.8/site-packages/pandas/plotting/_matplotlib/converter.py:235: UserWarning: Parsing '25-10-2015' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=True for consistent parsing.
  values = PeriodConverter._convert_1d(values, units, axis)
```



```
In [18]: #evaluation des prédictions
print("RMSE %s" %mean_squared_error(test_predictions_day["AEP_MW"],test_predictions_day["AEP_MW"]))
print("MSE %s" %mean_absolute_error(test_predictions_day["AEP_MW"],test_predictions_day["AEP_MW"]))
```

RMSE 45819.38893986183
MSE 36082.749579958

```
In [19]: daily_data.describe()
```

```
Out[19]:
```

	AEP_MW	day_of_week
count	5055.000000	5055.000000
mean	371844.218991	4.000198
std	47605.308633	2.000049
min	14809.000000	1.000000
25%	337265.500000	2.000000
50%	366539.000000	4.000000
75%	403198.500000	6.000000
max	548349.000000	7.000000

Ainsi notre modèle se trompe en moyenne de 45 819 MW par jour, par rapport à une consommation électrique moyenne par jour de 371 844 MW . (Soit une erreur de 14%)