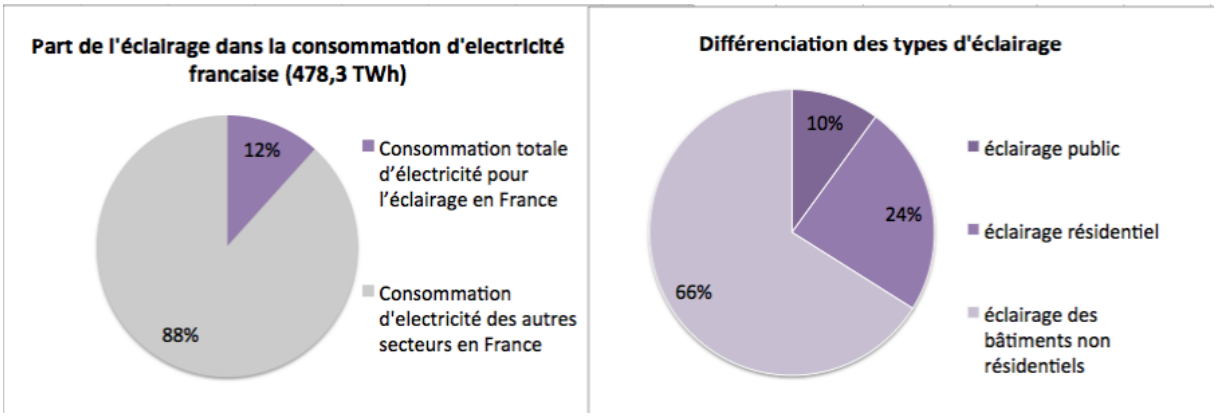
Quenez Erwan
Le Parc Renan

Analyse de données et réduction des dépense énergétiques

Quenez Erwan
Le Parc Renan

Préambule :

Est-ce que l'éclairage consomme beaucoup ? Cela dépend du nombre de lumières de votre maison, du choix de vos ampoules et de votre manière d'utiliser votre éclairage intérieur. Aujourd'hui, l'éclairage représente entre 10 et 15% de la facture d'électricité Il faut savoir qu'oublier d'éteindre trois ampoules de 75 watts pendant toute une nuit consomme autant d'électricité qu'un cycle de lessive à 60°C ! Laisser une lumière trop longtemps allumée peut donc alourdir la facture d'électricité en plus de consommer inutilement une énergie non renouvelable. L'autre impact de la lumière laissée inutilement allumée est la pollution lumineuse. Elle provoque des dégâts bien plus graves sur les écosystèmes et sur notre espèce. Elle modifie l'illumination de l'environnement et masque les cycles de la lumière naturelle. Elle est donc susceptible de modifier les comportements et les rythmes biologiques des individus. En conclusion, laisser la lumière allumée a des effets à grande échelle et à long terme bien plus graves que ce que l'on pourrait croire de prime abord ; penser à l'éteindre est un geste à la fois simple et efficace pour améliorer l'état du monde dans lequel nous vivons.



Ainsi, nous cherchons à déterminer le nombre d'occupants, dans une maison, à l'instant t afin de se poser la question sur le fonctionnement des lumières. S'il y a une personne dans la maison et qu'une lumière est allumée, une alerte est envoyée pour remédier à ce problème.

Ainsi, nous discrétisons 3 cas:

- Si personne n'est détecté dans la maison, alors on éteint toutes les lumières.
- Si on détecte au moins une personne, de jour, une alerte est envoyée pour questionner l'occupant sur la nécessité d'allumer les lumières.
- Si on détecte au moins une personne, de nuit, il ne se passe rien, nous laissons les occupants gérer.

Pour faire cela, nous allons extraire plusieurs types de données provenant des capteurs de la maison. Mais le problème est de choisir les bons capteurs pour le résultat souhaité.

Quenez Erwan
Le Parc Renan

Pour déterminer l'occupation de la maison, on a choisi de prendre les données suivantes :

-Mouvement dans l'entrée
-Consommation
-Concentration de CO2 du salon et de la chambre 1
-Ouverture des fenêtres de l'évier, de la chambre 1 et de la chambre 2

De même, pour déterminer la nécessité de lumière dans la maison, on a choisi de prendre les données suivantes :

-La luminosité extérieur de la station météo
-L'état de toutes lumières de la maison

Détermination du nombre d'occupants dans la maison:

Nous extrayons les différentes données provenant de GRAFANA pour mesurer le nombre d'occupants.
Nous classons ces données dans un fichier type csv pour, ensuite, les utiliser dans notre programme python.
Nous avons donc désormais plusieurs fichiers mais leur exploitation n'est pas encore possible car le format d'enregistrement présente une ponctuation particulière. Le but est alors de discrétiser les données, c'est-à -dire de les rendre lisibles pour notre programme Python. Chaque grandeur est ensuite mise dans une liste où une heure est affectée pour chaque mesure.

Time	CO2_chambre1	C02_salon	Consommation	fenetre_chambre1	fenetre_chambre2	fenetre_évier	Mouv_entrée
2022-05-16T11:20:04	517.0						
2022-05-16T11:20:04.184134	527.0						
2022-05-16T11:20:09			3627.0				
2022-05-16T11:20:10.129086			3928.0				
2022-05-16T11:20:37		511.0					
2022-05-16T11:21:04	518.0						

Il faut ensuite combler les trous avec la fonction Panda de python pour que chaque caractéristique ait une valeur affectée à un temps.

Après réflexion, notre panel de capteurs semble trop large, nous allons sélectionner seulement ceux que nous jugeons les plus pertinents pour nous indiquer le nombre de personnes dans la maison : ***Consommation, C02_salon, mouvement_entrée et ouverture de la fenêtre_cuisine***. Nous privilégions les pièces de vie communes, plus sujettes à nous fournir des informations plus complètes car on suppose que l'occupant passe plus de temps dans ces pièces en journée.

De plus, la méthode de trie change, nous n'allons pas passer par Panda. Nous récupérons les fichiers csv de chacun des capteurs qui nous intéressent (envoyés par notre professeur).

Quenez Erwan
Le Parc Renan

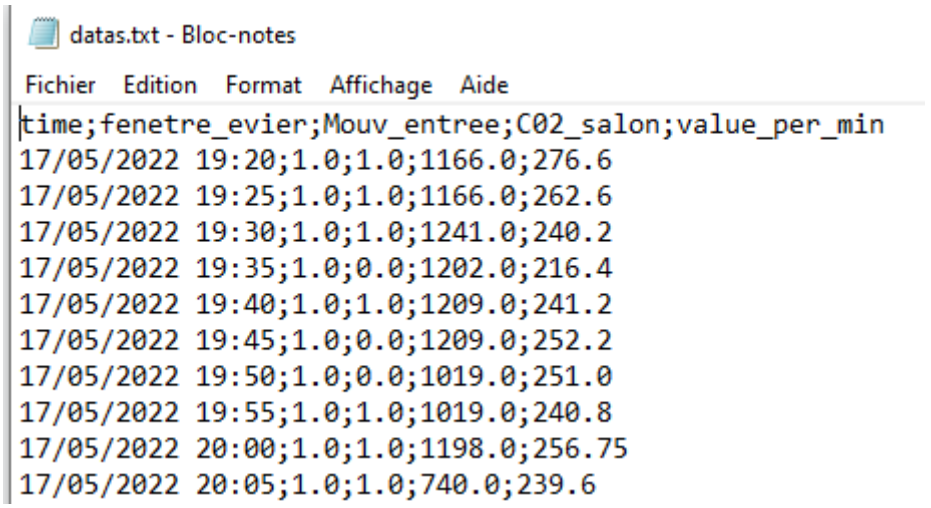
Nous délimitions temporellement notre zone d'étude (certains capteurs commencent le 17/05/2022, d'autres le 16/05/2022) afin d'avoir la même plage de temps pour tous les capteurs. Dans notre cas : du 17/05/2022 - 19h20 au 23/05/2022 - 6h10.

	A	B	C	D	E
1	time	fenetre_evier	Mouv_entree	C02_salon	value_per_min
2	17/05/2022 19:20	1.0	1.0	1166.0	276.6
3	17/05/2022 19:25	1.0	1.0	1166.0	262.6
4	17/05/2022 19:30	1.0	1.0	1241.0	240.2
5	17/05/2022 19:35	1.0	0.0	1202.0	216.4
6	17/05/2022 19:40	1.0	1.0	1209.0	241.2
7	17/05/2022 19:45	1.0	0.0	1209.0	252.2
8	17/05/2022 19:50	1.0	0.0	1019.0	251.0
9	17/05/2022 19:55	1.0	1.0	1019.0	240.8
10	17/05/2022 20:00	1.0	1.0	1198.0	256.75
11	17/05/2022 20:05	1.0	1.0	740.0	239.6
12	17/05/2022 20:10	1.0	1.0	1166.0	300.0

[…]

1565	23/05/2022 05:35	1.0	0.0	600.0	200.6
1566	23/05/2022 05:40	1.0	1.0	600.0	180.0
1567	23/05/2022 05:45	1.0	0.0	662.0	179.4
1568	23/05/2022 05:50	1.0	1.0	479.0	181.2
1569	23/05/2022 05:55	1.0	1.0	655.0	180.6
1570	23/05/2022 06:00	1.0	1.0	655.0	175.6
1571	23/05/2022 06:05	1.0	1.0	489.0	181.8
1572	23/05/2022 06:10	1.0	1.0	489.0	180.6

Pour un total de 1571 valeurs.
Une fois le fichier csv correctement mis en forme, on l'ouvre avec "bloc-note" pour le convertir en ".txt" (c'est sous ce format que python pourra le lire).



Chaque données est séparées par un ";" on le précisera dans le code pour créer nos vecteurs de données.

Méthode python utilisée:

Nous allons tout d'abord déterminer manuellement, par des hypothèses, l'occupation du 17 Mai 2022 à 19h20 au 18 Mai 2022 à 19h20 afin d'entraîner le modèle utilisé (decision tree classifier). On aura donc une prédiction de l'occupation des jours suivants, ce qui nous permettra de se poser la question sur le fonctionnement des lumières artificielles et d'alerter ou non sur l'utilisation énergivores futiles.

Quenez Erwan
Le Parc Renan

Puis, nous déterminerons les capteurs les plus importants grâce à la fonction “most important features” afin de savoir quel est le meilleur investissement, quels capteurs sont les plus pertinents à installer dans un souci économique mais aussi de vie privée.

De façon arbitraire, afin de déterminer l’occupation manuellement du 17 Mai 2022 à 19h20 au 23 Mai 2022 à 6h10, nous allons discrétiser chaque cas de la façon suivante:

-fenêtre évier ouverte = occupant
-concentration C02 salon > 650 = occupant
-mouvement entrée = occupant
-consommation > 350 Wh = occupant
-sinon, pas d’occupant

Voici notre résultat après programmation:

```
## Datas ##############################################################
fenetre_evier,mvt_entree,CO2,consommation =[], [], [], []

f = open('datas.txt', 'r')
Title = f.readline()
for line in f:
    line_token = line.split(";")
    fenetre_evier.append(float(line_token[1])), mvt_entree.append(float(line_token[2])),
    CO2.append(float(line_token[3])), consommation.append(float(line_token[4]))
f.close()

## première prédiction occupancy ######################################

occupancy=np.zeros(len(fenetre_evier))
for i in range (0,len(fenetre_evier)):
    if fenetre_evier[i]==1:
        occupancy[i]=1
    elif CO2[i]>700:
        occupancy[i]=1
    elif mvt_entree[i]==1:
        occupancy[i]=1
    elif consommation[i]>350:
        occupancy[i]=1
    else:
        occupancy[i]=0
```

	0
107	0
108	0
109	0
110	0
111	0
112	1
113	0
114	1
115	1
116	0
117	1
118	1
119	1
120	1
121	1

Quand il y a des occupants, la valeur de C02 est à plus de 1000 et quand il n’y a personne, cette valeur est à moins de 500. Nous avons donc choisi en valeur limite 700 de C02 de façon plus ou moins arbitraire.
De même, pour la consommation, la valeur est à plus de 600 Wh lorsque la pièce est occupée et quand il n’y a personne, cette valeur est à moins de 250 Wh. Nous avons donc choisi en valeur limite 350 Wh de consommation de façon plus ou moins arbitraire.

Nous utilisons, ensuite, “decision tree” afin de prédire l’occupation. Nous prendrons 70% des valeurs en features (mouv_entree, fenetre_evier, CO2, consommation) et de l’occupancy

prédite (expliquée précédemment). L'occupation des 30% suivants sera prédite avec le "decision tree" :

```
## decision tree ##############################################################

features=np.zeros([round(len(fenetre_evier)*0.7),4])
for i in range(round(len(fenetre_evier)*0.7)):
        features[i,0] = fenetre_evier[i]
        features[i,1] = mvt_entree[i]
        features[i,2] = CO2[i]
        features[i,3] = consommation[i]
occupancy_70=np.zeros(round(len(fenetre_evier)*0.7))
for i in range (round(len(fenetre_evier)*0.7)):
    occupancy_70[i]=occupancy[i]


features_test=np.zeros([round(len(fenetre_evier)*0.3),4])
for i in range(round(len(fenetre_evier)*0.3)):
        features_test[i,0] = fenetre_evier[i+round(len(fenetre_evier)*0.7)]
        features_test[i,1] = mvt_entree[i+round(len(fenetre_evier)*0.7)]
        features_test[i,2] = CO2[i+round(len(fenetre_evier)*0.7)]
        features_test[i,3] = consommation[i+round(len(fenetre_evier)*0.7)]
occupancy_test=np.zeros(round(len(fenetre_evier)*0.3))

classifier = tree.DecisionTreeClassifier(random_state=0, criterion='entropy',max_depth=None)
classifier = classifier.fit(features, occupancy_70)
print(classifier.predict(features_test))
with open("tree.dot", 'w') as file:
    f = tree.export_graphviz(classifier,out_file=file)
```

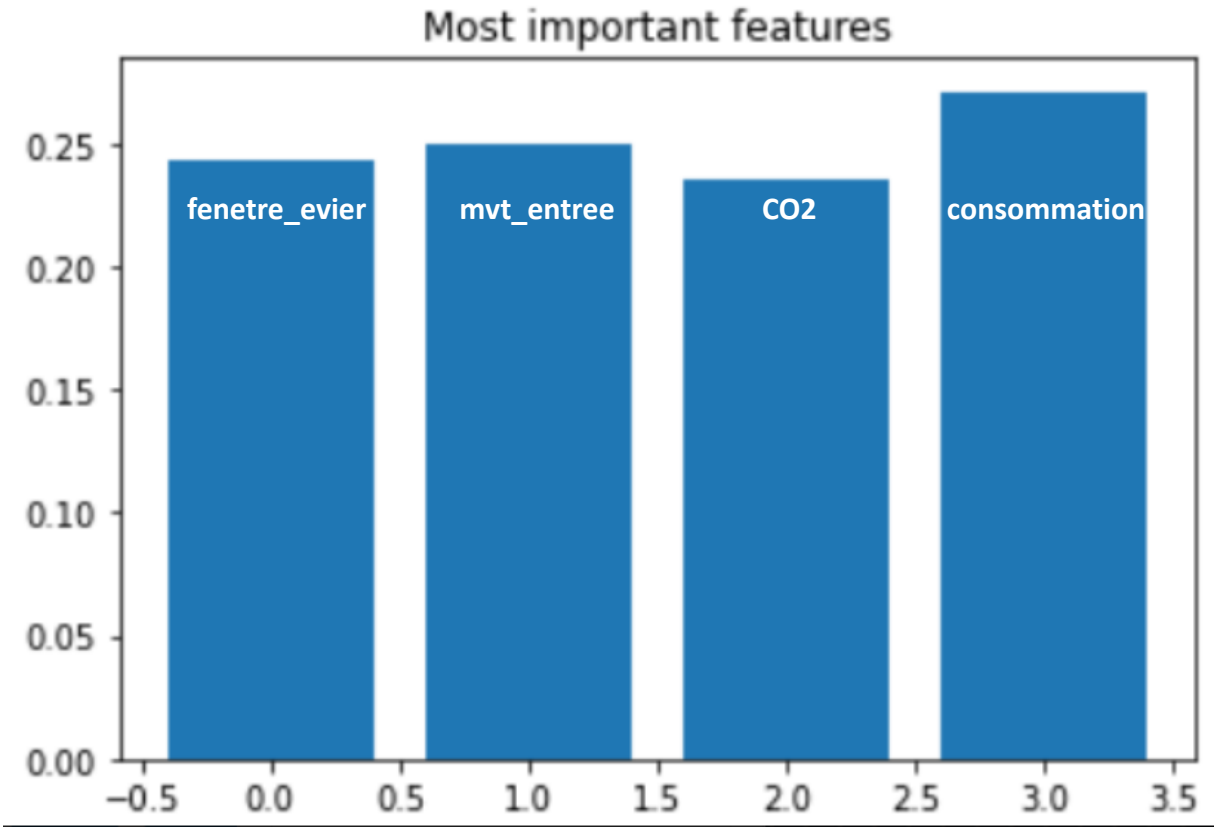
Voici le tableau de l'occupation prédite:

```
[0. 1. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 1. 1. 1. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 1. 1. 1. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 1. 0. 0. 0. 1. 1. 1. 1. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1.
 0. 1. 1. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 1. 1. 1. 1. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 1. 0. 1. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 1. 1. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 1. 0. 1. 1. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 1. 1. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 1. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 1. 1. 0. 1. 0. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 0. 0. 1. 1. 1. 1. 0. 0. 1. 0. 0. 0. 0. 0. 0. 1. 1. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.]
```

Quenez Erwan
Le Parc Renan

Ensuite, nous avons voulu déterminer quels étaient les capteurs les plus importants pour déterminer l'occupation de la maison. Nous avons donc utilisé la fonction python "classifier.feature_importances_":

```
importances = classifier.feature_importances_
for i,v in enumerate(importances):
    print('Feature: %0d, Score: %.5f' % (i,v))
plt.bar([x for x in range(len(importances))], importances)
plt.title("Most important features")
plt.show()
```



Nous remarquons que tous les capteurs utilisés sont importants afin de prédire l'occupation de la maison. Nous garderons donc les 4 capteurs.

Enfin, on va comparer l'état de la lumière (allumée ou non) avec l'occupation. Le but, ici, est de détecter différents cas : la lumière est allumée mais il n'y a personne (par exemple). Nous avons créé un programme afin d'interpeller par notification les occupants sur l'état des lumières lorsqu'ils sont présents.

Quenez Erwan
Le Parc Renan

Voici le programme:

```
## Datas luminosités ##########################################################################
value_lum_ext,value_etatsalon,value_etatsallemanger =[], [], []

f = open('datas_luminosités.txt', 'r')
Title = f.readline()
for line in f:
    line_token = line.split(";")
    value_lum_ext.append(float(line_token[1])), value_etatsalon.append(float(line_token[2])),
    value_etatsallemanger.append(float(line_token[3]))
f.close()
```

```
## ferme lumière? ##############################################################################

for k in range (0,len(value_lum_ext)):
    if value_lum_ext[k]>500:
        if occupancy[k]==0:
            value_etatsalon[k]=0
            value_etatsallemanger[k]=0
        elif occupancy[k]==1:
            if value_etatsalon[k]==1:
                print ("la lumiere du salon est allumée")
            if value_etatsallemanger[k]==1:
                print ("la lumiere de la salle à manger est allumée")
```

```
la lumiere du salon est allumée
la lumiere de la salle à manger est allumée
la lumiere du salon est allumée
la lumiere de la salle à manger est allumée
la lumiere de la salle à manger est allumée
la lumiere de la salle à manger est allumée
la lumiere de la salle à manger est allumée
```

• Si la luminosité extérieure est supérieure à 500, on considère qu'il fait jour, donc on éteint automatiquement les lumières du salon et de la salle à manger s'il n'y a personne.

• Si on détecte la présence d'une personne et que certaines pièces de la maison sont allumées, alors on notifie l'occupant à titre informatif : aurait-il oublié d'éteindre la lumière de la pièce ?

Quenez Erwan
Le Parc Renan

Conclusion:

On a cherché à minimiser la consommation énergétique de l'éclairage d'un foyer mais nous avons rencontré plusieurs problèmes:

• Nos capteurs ne détectent personne la nuit , ce qui est faux car les occupants sont normalement dans leur chambre mais cela n'impacte pas notre programmation car nous voulons juste gérer la lumière en journée.

• Ensuite, la prédiction de l'occupation est basée sur plusieurs hypothèses émises au début. Nous pouvons nous poser la question de la pertinence de ces hypothèses.

• Enfin, on se pose également la question du choix des capteurs : idéalement, notre "decision-tree" aurait dû nous renseigner sur les capteurs les plus pertinents et donc également sur ceux que l'on peut enlever, or ici les quatres capteurs sont importants. Il doit sûrement exister un jeu de capteur plus intéressant, car à terme le but est de supprimer certains capteurs pour plus de vie privée et gagner en économie.

Quenez Erwan
Le Parc Renan

Annexe : code

```
from sklearn import tree
import  math
import numpy as np
from matplotlib import pyplot as plt
from sklearn.datasets import make_regression
from sklearn.linear_model import LinearRegression

## Datas
#################################################################
########
fenetre_evier,mvt_entree,CO2,consommation =[], [], [], []

f = open('datas.txt', 'r')
Title = f.readline()
for line in f:
   line_token = line.split(";")
   fenetre_evier.append(float(line_token[1])),
mvt_entree.append(float(line_token[2])),
   CO2.append(float(line_token[3])),
consommation.append(float(line_token[4]))
f.close()

## première prédiction occupancy
#############################################

occupancy=np.zeros(len(fenetre_evier))
for i in range (0,len(fenetre_evier)):
   if fenetre_evier[i]==1:
        occupancy[i]=1
   elif CO2[i]>700:
        occupancy[i]=1
   elif mvt_entree[i]==1:
        occupancy[i]=1
   elif consommation[i]>350:
        occupancy[i]=1
   else:
        occupancy[i]=0

## decision tree
#################################################################

features=np.zeros([round(len(fenetre_evier)*0.7),4])
for i in range(round(len(fenetre_evier)*0.7)):
```

```
        features[i,0] = fenetre_evier[i]
        features[i,1] = mvt_entree[i]
        features[i,2] = CO2[i]
        features[i,3] = consommation[i]
occupancy_70=np.zeros(round(len(fenetre_evier)*0.7))
for i in range (round(len(fenetre_evier)*0.7)):
   occupancy_70[i]=occupancy[i]


features_test=np.zeros([round(len(fenetre_evier)*0.3),4])
for i in range(round(len(fenetre_evier)*0.3)):
        features_test[i,0] =
fenetre_evier[i+round(len(fenetre_evier)*0.7)]
        features_test[i,1] =
mvt_entree[i+round(len(fenetre_evier)*0.7)]
        features_test[i,2] =
CO2[i+round(len(fenetre_evier)*0.7)]
        features_test[i,3] =
consommation[i+round(len(fenetre_evier)*0.7)]
occupancy_test=np.zeros(round(len(fenetre_evier)*0.3))

classifier = tree.DecisionTreeClassifier(random_state=0,
criterion='entropy',max_depth=None)
classifier = classifier.fit(features, occupancy_70)
print(classifier.predict(features_test))
with open("tree.dot", 'w') as file:
   f = tree.export_graphviz(classifier,out_file=file)


importances = classifier.feature_importances_
for i,v in enumerate(importances):
  print('Feature: %0d, Score: %.5f' % (i,v))
plt.bar([x for x in range(len(importances))], importances)
plt.title("Most important features")
plt.show()

## accuracy
#################################################################
#####
k=0
for i in range(0,round(len(fenetre_evier)*0.3)):
   if
occupancy[i+round(len(fenetre_evier)*0.7)]==classifier.predict
(features_test)[i]:
```

```
        k=k+1
    elif
occupancy[i+round(len(fenetre_evier)*0.7)]!=classifier.predict
(features_test)[i]:
        k=k+0
accuracy=k/len(classifier.predict(features_test))*100
print(' ')
print('Accuracy(%)')
print(accuracy)

## Datas_luminosités
##################################################################
########
value_lum_ext,value_etatsalon,value_etatsallemanger =[], [],
[]

f = open('datas_luminosités.txt', 'r')
Title = f.readline()
for line in f:
    line_token = line.split(";")
    value_lum_ext.append(float(line_token[1])),
value_etatsalon.append(float(line_token[2])),
    value_etatsallemanger.append(float(line_token[3]))
f.close()

## ferme lumière?
################################################################

for k in range (0,len(value_lum_ext)):
    if value_lum_ext[k]>500:
        if occupancy[k]==0:
            value_etatsalon[k]=0
            value_etatsallemanger[k]=0
        elif occupancy[k]==1:
            if value_etatsalon[k]==1:
                print ("la lumiere du salon est allumée")
            if value_etatsallemanger[k]==1:
                print ("la lumiere de la salle à manger est
allumée")
```