Singapore Energy Savings Data Analysis - A Data Camp Certification Project

Pablo de la Asunción Cumbrera Conde

Índice

Introducción

- La importancia de ahorrar energia a través del reciclaje (Data Camp introduction)

¿Sabía que el reciclaje ahorra energía al reducir o eliminar la necesidad de fabricar materiales desde cero? Por ejemplo, los fabricantes de latas de aluminio pueden saltarse el costoso proceso de producción de aluminio a partir del mineral limpiando y fundiendo latas recicladas. El aluminio está clasificado como un metal no ferroso.

Singapur tiene el ambicioso objetivo de convertirse en una nación sin residuos. La cantidad de residuos eliminados en Singapur se ha multiplicado por siete en los últimos 40 años. A este ritmo, el vertedero de Semakau, el único vertedero de Singapur, se quedará sin espacio para 2035. Para empeorar las cosas, Singapur tiene terrenos limitados para construir nuevas plantas de incineración o vertederos.

Al gobierno le gustaría motivar a los ciudadanos compartiendo la energía total que los esfuerzos combinados de reciclaje han ahorrado cada año. Te han pedido que les ayudes.

Se le han proporcionado tres conjuntos de datos. Los datos provienen de diferentes equipos, por lo que los nombres de los tipos de desechos pueden diferir.

- Descripción de los Data Set

datasets/wastestats.csv - Recycling statistics per waste type for the period 2003 to 2017
Source: Singapore National Environment Agency

- **waste_type:** The type of waste recycled.
- **waste_disposed_of_tonne:** The amount of waste that could not be recycled (in metric tonnes).
- **total_waste_recycle_tonne:** The amount of waste that could be recycled (in metric tonnes).
- **total_waste_generated:** The total amount of waste collected before recycling (in metric tonnes).
- **recycling_rate:** The amount of waste recycled per tonne of waste generated.
- **year:** The recycling year.

datasets/2018_2019_waste.csv - Recycling statistics per waste type for the period 2018 to 2019
Source: Singapore National Environment Agency

- **Waste Type:** The type of waste recycled.
- **Total Generated:** The total amount of waste collected before recycling (in thousands of metric tonnes).
- **Total Recycled:** The amount of waste that could be recycled. (in thousands of metric tonnes).
- **Year:** The recycling year.

datasets/energy_saved.csv - Estimations of the amount of energy saved per waste type in kWh

- **material:** The type of waste recycled.
- **energy_saved:** An estimate of the energy saved (in kiloWatt hour) by recycling a metric tonne of waste.
- **crude_oil_saved:** An estimate of the number of barrels of oil saved by recycling a metric tonne of waste.

- Objetivos

  A partir de los datos, obtener el número de kWh de energía que el gobierno de Singapour ha ahorrado en el reciclaje de cuatro materiales:

```
- Plastic
- Glass
- Ferrous Metal
- Non-Ferrous Metal
```

Librerias y Data Frames

In [1]:
```python
import pandas as pd
import numpy as np
```

In [2]:
```python
# Primer DF 2003 - 2017

waste_one = pd.read_csv('file:///C:/Users/pablo/Data%20Science/RawData/wastestats.csv')
print(waste_one.shape)
waste_one = waste_one.sort_values('year', ascending=False)
waste_one.head(10)
```

```
(225, 6)
```

Out[2]:

	waste_type	waste_disposed_of_tonne	total_waste_recycled_tonne	total_waste_generated_tonne	recycling_rate	year
224	Total	2980000	4724300.0	7704300	0.61	2017
223	Others (stones, ceramic, rubber, etc.)	319300	7100.0	326400	0.02	2017
222	Textile/Leather	141200	9600.0	150800	0.06	2017
221	Plastic	763400	51800.0	815200	0.06	2017
220	Ash and sludge	214800	28600.0	243400	0.12	2017
219	Food	676800	133000.0	809800	0.16	2017
218	Glass	58900	12400.0	71300	0.17	2017
217	Paper/Cardboard	576000	568800.0	1144800	0.50	2017
216	Horticultural waste	107600	220700.0	328300	0.67	2017
215	Wood	97300	326800.0	424100	0.77	2017

In [3]:
```python
#Segundo DF 2018 - 2019

waste_two = pd.read_csv('file:///C:/Users/pablo/Data%20Science/RawData/2018_2019_waste.csv')
print(waste_two.shape)
waste_two.head(10)
```

```
(30, 4)
```

Out[3]:

	Waste Type	Total Generated ('000 tonnes)	Total Recycled ('000 tonnes)	Year
0	Construction& Demolition	1440	1434	2019
1	Ferrous Metal	1278	1270	2019
2	Paper/Cardboard	1011	449	2019
3	Plastics	930	37	2019
4	Food	7440	136	2019
5	Wood	438	289	2019
6	Horticultural	400	293	2019
7	Ash & Sludge	252	25	2019
8	Textile/Leather	168	6	2019
9	Used Slag	129	127	2019

In [4]:
```python
# Tercer DF

energy_saved = pd.read_csv('file:///C:/Users/pablo/Data%20Science/RawData/energy_saved.csv')
print(energy_saved.shape)
energy_saved.head(10)
```

```
(5, 6)
```

Out[4]:

	The table gives the amount of energy saved in kilowatt hour (kWh) and the amount of crude oil (barrels) by recycling 1 metric tonne (1000 kilogram) per waste type	Unnamed: 1	Unnamed: 2	Unnamed: 3	Unnamed: 4	Unnamed: 5
0	1 barrel oil is approximately 159 litres of oil	NaN	NaN	NaN	NaN	NaN
1	NaN	NaN	NaN	NaN	NaN	NaN
2	material	Plastic	Glass	Ferrous Metal	Non-Ferrous Metal	Paper
3	energy_saved	5774 Kwh	42 Kwh	642 Kwh	14000 Kwh	4000 kWh
4	crude_oil saved	16 barrels	NaN	1.8 barrels	40 barrels	1.7 barrels

Preprocesamiento

In [5]:
```python
# Ponemos en minúsculas los nombres de las columnas y cambiamos los nombres para que encagen después
waste_one.columns = waste_one.columns.str.lower()
waste_two.columns = waste_two.columns.str.lower()
energy_saved.columns = energy_saved.columns.str.lower()
```

In [6]:
```python
#Cambios en primer DF:
#waste_one['total recycled (tonnes)'] = waste_one['total_waste_recycled_tonne'].values //1000
waste_one = waste_one.drop(['waste_disposed_of_tonne', 'total_waste_generated_tonne','recycling_rate'], axis=1)
waste_one.columns = ['waste_type','total recycled (tonnes)', 'year']
```

In [7]:
```python
#Cambios en segundo DF:
waste_two.columns = ['waste_type','total generated (tonnes)','total recycled (tonnes)','year']
waste_two['total recycled (tonnes)'] = waste_two['total recycled (tonnes)'].values *1000
waste_two = waste_two.drop(['total generated (tonnes)'], axis=1)
```

In [8]:
```python
# Tercer DF:
df_three ={'material':['Plastic','Glass','Ferrous Metal','Non-Ferrous Metal','Paper'],'energy_saved':[5774,42,642,14000,4000]}
df_three= pd.DataFrame(df_three)
df_three=df_three.drop(4,axis=0)
df_three
```

Out[8]:

	material	energy_saved
0	Plastic	5774
1	Glass	42
2	Ferrous Metal	642
3	Non-Ferrous Metal	14000

Extracción de features y unión de los Data Frame

In [9]:
```python
# Primer DF

#Eliminamos todas las columnas anteriores a 2015

waste_one = waste_one.drop(waste_one[waste_one['year'] < 2015].index)
waste_one.shape
```

Out[9]: (45, 3)

In [10]:
```python
# Nombre columnas target
print(waste_one['waste_type'][2])
print(waste_one['waste_type'][6])
print(waste_one['waste_type'][7])
print(waste_one['waste_type'][10])
```

```
Plastics
Ferrous metal
Non-ferrous metal
Glass
```

In [11]:
```python
glass = waste_one.drop(waste_one[waste_one['waste_type'] !='Glass'].index)
plastic = waste_one.drop(waste_one[waste_one['waste_type'] != 'Plastics'].index)
ferrous = waste_one.drop(waste_one[waste_one['waste_type'] !='Ferrous metal'].index)
nonferrous = waste_one.drop(waste_one[waste_one['waste_type'] !='Non-ferrous metal'].index)
df_one = glass.append(plastic.append(ferrous.append(nonferrous)))
df_one = df_one.sort_values('waste_type')
```

In [12]:
```python
#Buscamos un valor para plastic y nfm que no se encuentran en el df
value_plastic =waste_one.drop(waste_one[waste_one['waste_type'] != 'Plastic'].index)
print(value_plastic.shape)
value_nfm = waste_one.drop(waste_one[waste_one['waste_type'] != 'Non-ferrous metals'].index)
print(value_nfm.shape)
df_one = df_one.append(value_plastic.append(value_nfm))
```

```
(1, 3)
(1, 3)
```

In [13]:
```python
# Primer DF completo
df_one = df_one.sort_values('year', ascending=True)
df_one = df_one.set_index('year')
df_one['total recycled (tonnes)'] = df_one['total recycled (tonnes)'].values.astype(int)


####PRIMER DF
df_one
```

Out[13]:

year	waste_type	total recycled (tonnes)
2015	Ferrous metal	1333300
2015	Glass	14600
2015	Non-ferrous metal	160400
2015	Plastics	57800
2016	Ferrous metal	1351500
2016	Glass	14700
2016	Non-ferrous metal	95900
2016	Plastics	59500
2017	Ferrous metal	1371000
2017	Glass	12400
2017	Plastic	51800
2017	Non-ferrous metals	92200

In [14]:
```python
# Realizamos las mismas operaciones para el segundo data frame
glass2 = waste_two.drop(waste_two[waste_two['waste_type'] !='Glass'].index)
plastic2 = waste_two.drop(waste_two[waste_two['waste_type'] != 'Plastics'].index)
ferrous2 = waste_two.drop(waste_two[waste_two['waste_type'] !='Ferrous Metal'].index)
nonferrous2 = waste_two.drop(waste_two[waste_two['waste_type'] !='Non-Ferrous Metal'].index)
df_two = glass2.append(plastic2.append(ferrous2.append(nonferrous2)))
df_two = df_two.sort_values('year')
df_two = df_two.set_index('year')


### SEGUNDO DF
df_two
```

Out[14]:

year	waste_type	total recycled (tonnes)
2018	Glass	12000
2018	Plastics	41000
2018	Ferrous Metal	126000
2018	Non-Ferrous Metal	170000
2019	Glass	11000
2019	Plastics	37000
2019	Ferrous Metal	1270000
2019	Non-Ferrous Metal	124000

In [15]:
```python
target_data = df_one.append(df_two)
print(target_data.shape)
print(target_data.nunique())
target_data = target_data.sort_values('year')

### DF MERGED
target_data
```

```
(20, 2)
waste_type                  8
total recycled (tonnes)    20
dtype: int64
```

Out[15]:

year	waste_type	total recycled (tonnes)
2015	Ferrous metal	1333300
2015	Glass	14600
2015	Non-ferrous metal	160400
2015	Plastics	57800
2016	Ferrous metal	1351500
2016	Glass	14700
2016	Non-ferrous metal	95900
2016	Plastics	59500
2017	Non-ferrous metals	92200
2017	Plastic	51800
2017	Glass	12400
2017	Ferrous metal	1371000
2018	Glass	12000
2018	Plastics	41000
2018	Ferrous Metal	126000
2018	Non-Ferrous Metal	170000
2019	Glass	11000
2019	Plastics	37000
2019	Ferrous Metal	1270000
2019	Non-Ferrous Metal	124000

Cálculo de los resultados

In [16]:
```python
#Valores de cada material

print(df_three)
```

```
            material  energy_saved
0            Plastic          5774
1              Glass            42
2      Ferrous Metal           642
3  Non-Ferrous Metal         14000
```

- Plástic

In [17]:
```python
##PLASTIC

solp = target_data[target_data['waste_type'] == 'Plastics']
solp = solp.append(target_data[target_data['waste_type'] == 'Plastic'])
solp['plastic'] = solp['total recycled (tonnes)'].values.astype(int) *5774

solp = solp.drop(['total recycled (tonnes)','waste_type'],axis=1)
solp
```

Out[17]:

year	plastic
2015	333737200
2016	343553000
2018	236734000
2019	213638000
2017	299093200

- Glass

In [18]:
```python
solg = target_data[target_data['waste_type'] == 'Glass']
solg['glass'] = solg['total recycled (tonnes)'].values*42

solg = solg.drop(['total recycled (tonnes)','waste_type'],axis=1)
solg
```

```
<ipython-input-18-37fa94b3de85>:2: 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/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  solg['glass'] = solg['total recycled (tonnes)'].values*42
```

Out[18]:

year	glass
2015	613200
2016	617400
2017	520800
2018	504000
2019	462000

- Ferrous Metal

In [19]:
```python
## FERROUS METAL

solfm = target_data[target_data['waste_type'] == 'Ferrous Metal']
solfm = solfm.append(target_data[target_data['waste_type'] == 'Ferrous metal'])
solfm['fm'] = solfm['total recycled (tonnes)'].values *642

solfm = solfm.drop(['total recycled (tonnes)','waste_type'],axis=1)
solfm = solfm.sort_values('year')
solfm
```

Out[19]:

year	fm
2015	855978600
2016	867663000
2017	880182000
2018	80892000
2019	815340000

- Non-ferrous Metal

In [20]:
```python
## NON-FERROUS METAL

solnfm = target_data[target_data['waste_type'] == 'Non-Ferrous Metal']
solnfm = solnfm.append(target_data[target_data['waste_type'] == 'Non-ferrous metal'])
solnfm = solnfm.append(target_data[target_data['waste_type'] == 'Non-ferrous metals'])
solnfm['nfm'] = solnfm['total recycled (tonnes)'].values*14000

solnfm = solnfm.drop(['total recycled (tonnes)','waste_type'],axis=1)
solnfm = solnfm.sort_values('year')
solnfm
```

Out[20]:

year	nfm
2015	2245600000
2016	1342600000
2017	1290800000
2018	2380000000
2019	1736000000

- Unimos los resultados...

In [21]:
```python
df1 = pd.merge(solp,solg,left_index=True, right_index=True)
df2 = pd.merge(solfm,solnfm,left_index=True, right_index=True)
df3 = pd.merge(df1,df2,left_index=True, right_index=True)
df3 = df3.sort_values('year')
df3
```

Out[21]:

year	plastic	glass	fm	nfm
2015	333737200	613200	855978600	2245600000
2016	343553000	617400	867663000	1342600000
2017	299093200	520800	880182000	1290800000
2018	236734000	504000	80892000	2380000000
2019	213638000	462000	815340000	1736000000

In [22]:
```python
print("2015: ", df3.iloc[0].sum())
print("2016: ",df3.iloc[1].sum())
print("2017: ",df3.iloc[2].sum())
print("2018: ",df3.iloc[3].sum())
print("2019: ",df3.iloc[4].sum())

total_energy_saved ={'year':['2015','2016','2017','2018','2019'],
                     'total_energy_saved':[df3.iloc[0].sum(),df3.iloc[1].sum(),df3.iloc[2].sum(),
                                           df3.iloc[3].sum(),df3.iloc[4].sum()]}

annual_energy_savings = pd.DataFrame(total_energy_saved)
annual_energy_savings = annual_energy_savings.set_index('year')
```

```
2015:  3435929000
2016:  2554433400
2017:  2470596000
2018:  2698130000
2019:  2765440000
```

Conclusiones

- Durante estos años (2015 - 2019) el gobierno de Singapur ha ahorrado:

In [23]:
```python
annual_energy_savings
```

Out[23]:

year	total_energy_saved
2015	3435929000
2016	2554433400
2017	2470596000
2018	2698130000
2019	2765440000

- El proyecto ha pasado la prueba de certificación de 'Data Camp' siendo los resultados correctos.

#