Asignatura	Datos de alumnos/profesores	Fecha
Aprendizaje Automático	Alumno: Federico Damián Estébanez	20/06/2019
	Profesores: Federico Castanedo y Jordi Escayola	

Importación de Librerías

In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
import seaborn as sns
import os
```

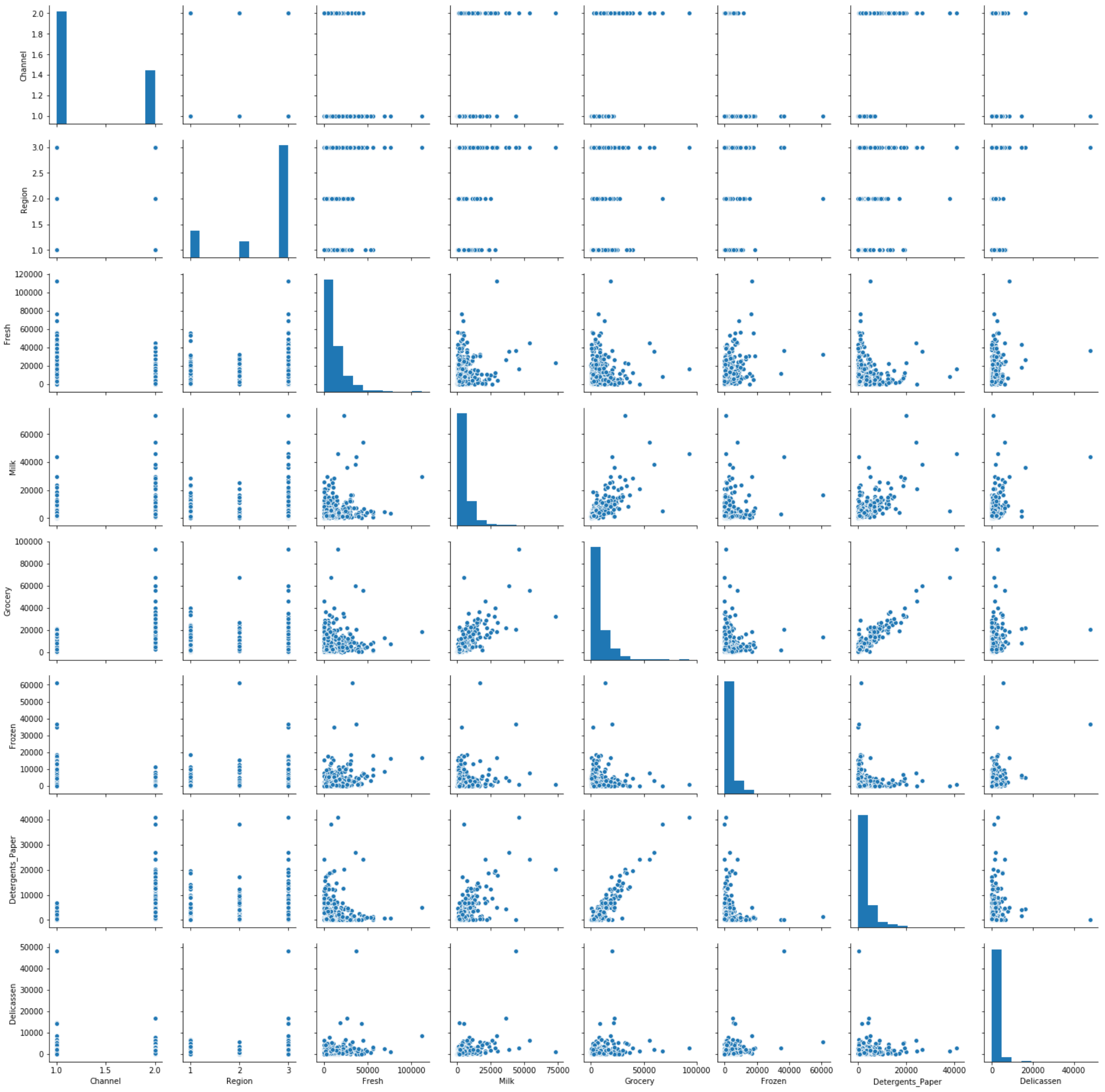
Lectura de Datos

In [2]:
```
df = pd.read_csv('Wholesale customers data.csv')
```

Visualización y relación entre las distintas variables

```
In [3]: sns.pairplot(data=df)
```

```
Out[3]: <seaborn.axisgrid.PairGrid at 0x1a3baddfd30>
```

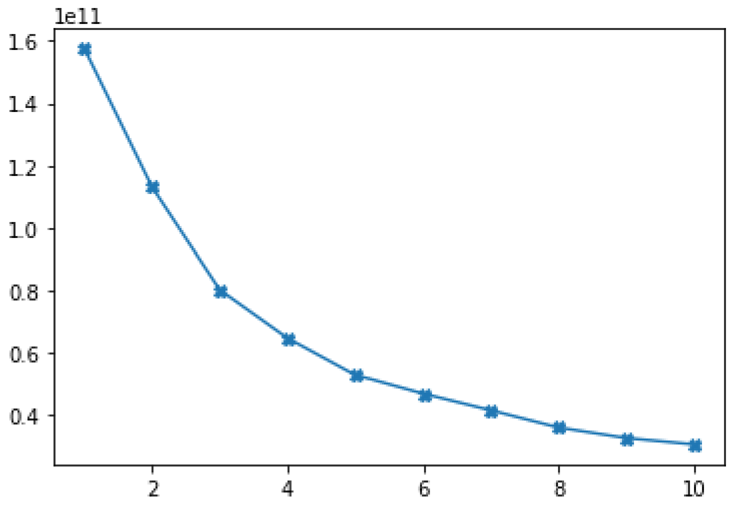


Identificacion del número eficiente de grupos o centroides

In [4]:
```
inertia = []
for i in range(1,11):
    kmeans = KMeans(n_clusters=i, random_state=1234)
    kmeans.fit(df)
    a=inertia.append((i,kmeans.inertia_,))


plt.plot([w[0] for w in inertia],[w[1] for w in inertia], marker="X")
```

Out[4]:
```
[<matplotlib.lines.Line2D at 0x1a3bde6e128>]
```



5 grupos elegidos para nuestro modelo

In [5]:
```
clusters = 5

#Modelo
kmeans = KMeans(n_clusters=clusters)
kmeans = kmeans.fit(df)

#Etiqueta para los datos de dicha fila
labels = kmeans.predict(df)

#Centro
C_center = kmeans.cluster_centers_
print(labels,"\n",C_center)
```

```
[0 0 0 4 4 0 0 0 0 1 0 0 4 4 4 0 0 0 4 0 4 0 4 1 4 4 0 4 1 3 4 0 4 4 0 0 4
 4 1 3 4 4 1 1 0 1 1 2 0 1 0 0 3 1 4 0 1 1 4 0 0 2 0 1 0 1 0 4 0 0 4 4 0 4
 0 4 0 1 0 0 0 1 0 4 0 2 2 3 0 4 0 4 1 4 1 0 0 0 0 0 1 1 0 3 4 4 0 1 0 1 0
 1 4 4 4 0 0 0 4 0 4 0 0 0 3 3 4 4 0 3 0 0 4 0 0 0 0 0 4 0 4 4 3 0 4 1 0 0
 0 4 4 0 4 0 0 1 1 4 0 1 0 0 4 1 0 1 0 0 0 0 1 1 0 1 0 0 3 0 0 0 0 3 0 3 0
 0 0 0 0 1 4 4 0 1 0 4 4 0 0 0 1 1 4 0 0 1 0 0 0 1 4 1 0 0 0 1 1 4 1 0 4 0
 0 0 0 0 4 0 0 0 0 0 4 0 4 0 0 4 0 3 4 4 4 0 0 1 0 4 4 0 0 1 0 4 0 4 0 0 3
 3 0 0 4 0 1 1 1 4 1 4 0 0 0 3 0 0 4 0 0 4 0 0 3 4 3 3 0 4 4 3 0 0 0 1 4 0
 4 0 0 0 4 1 0 1 1 0 1 4 0 1 0 4 1 0 0 1 0 0 0 1 0 0 4 4 4 3 0 0 4 0 0 1 4
 2 4 4 4 0 0 0 0 0 0 1 0 0 1 4 0 1 0 1 0 1 4 0 4 1 0 0 4 0 0 0 0 0 0 0 4 0
 3 4 0 4 0 0 1 3 0 0 4 4 4 0 1 0 0 4 0 0 0 0 0 4 0 0 0 0 0 0 0 4 4 4 4 0 4
 1 0 0 0 0 0 0 0 0 1 0 1 0 0 4 4 4 4 0 1 4 0 0 0 0 4 0 4 4 3 1 0 0]
 [[1.20264317e+00 2.54185022e+00 5.65581938e+03 3.56779295e+03
  4.51303965e+03 2.38652863e+03 1.43755947e+03 1.00503084e+03]
 [1.94366197e+00 2.46478873e+00 5.20783099e+03 1.31910282e+04
  2.03217183e+04 1.67402817e+03 9.03638028e+03 1.93794366e+03]
 [2.00000000e+00 2.80000000e+00 2.56030000e+04 4.34606000e+04
  6.14722000e+04 2.63600000e+03 2.99742000e+04 2.70880000e+03]
 [1.08333333e+00 2.70833333e+00 4.87773750e+04 6.60737500e+03
  6.19779167e+03 9.46279167e+03 9.32125000e+02 4.43533333e+03]
 [1.19469027e+00 2.54867257e+00 2.06002832e+04 3.78783186e+03
  5.08984071e+03 3.98907080e+03 1.13014159e+03 1.63907080e+03]]
```

Adición de la etiqueta

In [6]:
```
dfGroup = pd.concat([df,pd.DataFrame(labels, columns= ['Group'])], axis=1, joi
n='inner')
dfGroup.head()
```

Out[6]:

	Channel	Region	Fresh	Milk	Grocery	Frozen	Detergents_Paper	Delicassen	Group
0	2	3	12669	9656	7561	214	2674	1338	0
1	2	3	7057	9810	9568	1762	3293	1776	0
2	2	3	6353	8808	7684	2405	3516	7844	0
3	1	3	13265	1196	4221	6404	507	1788	4
4	2	3	22615	5410	7198	3915	1777	5185	4

Visualización de los grupos

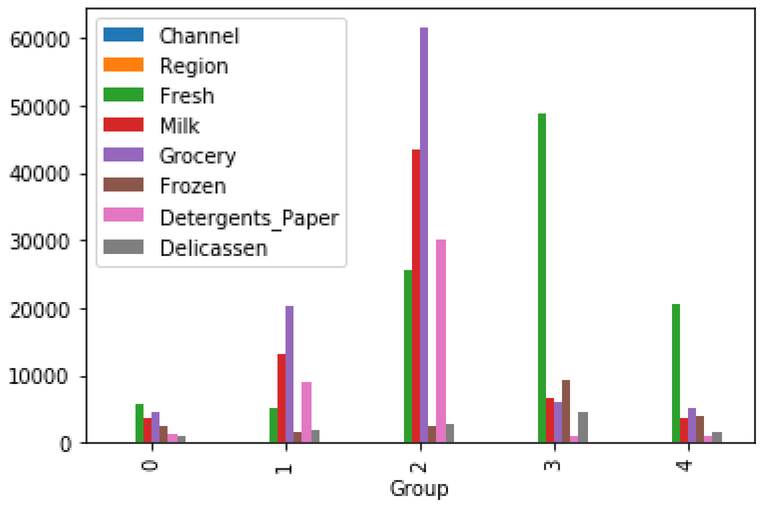
In [7]:
```
df.describe()
```

Out[7]:

	Channel	Region	Fresh	Milk	Grocery	Frozen	Deterg
count	440.000000	440.000000	440.000000	440.000000	440.000000	440.000000	
mean	1.322727	2.543182	12000.297727	5796.265909	7951.277273	3071.931818	2
std	0.468052	0.774272	12647.328865	7380.377175	9503.162829	4854.673333	4
min	1.000000	1.000000	3.000000	55.000000	3.000000	25.000000	
25%	1.000000	2.000000	3127.750000	1533.000000	2153.000000	742.250000	
50%	1.000000	3.000000	8504.000000	3627.000000	4755.500000	1526.000000	
75%	2.000000	3.000000	16933.750000	7190.250000	10655.750000	3554.250000	3
max	2.000000	3.000000	112151.000000	73498.000000	92780.000000	60869.000000	40

In [8]:
```
dfGroup.groupby("Group").aggregate("mean").plot.bar()
```

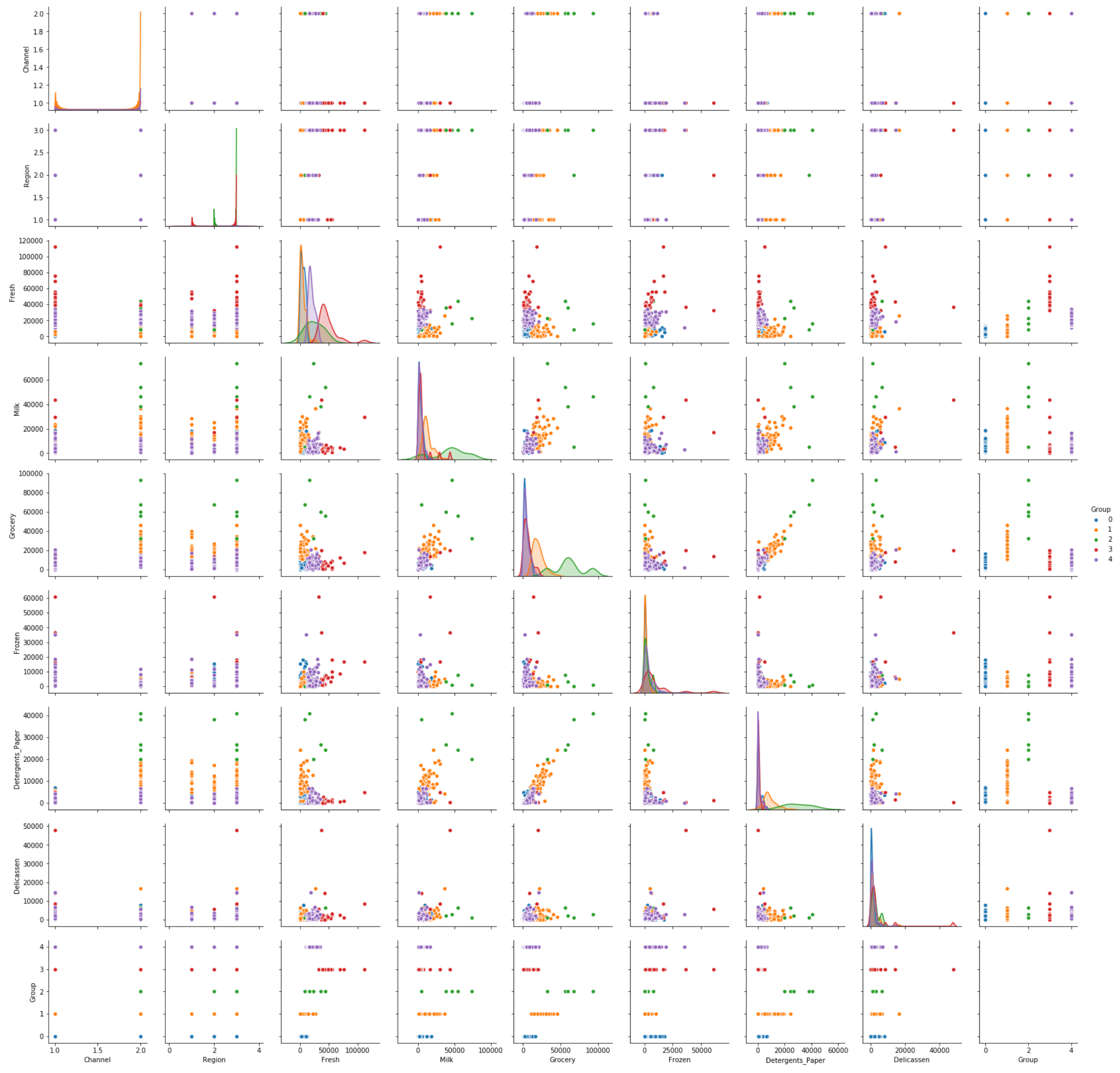
Out[8]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1a3bebeca58>
```



```
In [9]: sns.pairplot(data=dfGroup, hue='Group')
```

```
C:\Users\FedericoDamianEsteba\Anaconda3\lib\site-packages\scipy\stats\stats.p
y:1713: FutureWarning: Using a non-tuple sequence for multidimensional indexi
ng is deprecated; use `arr[tuple(seq)]` instead of `arr[seq]`. In the future
this will be interpreted as an array index, `arr[np.array(seq)]`, which will
result either in an error or a different result.
  return np.add.reduce(sorted[indexer] * weights, axis=axis) / sumval
C:\Users\FedericoDamianEsteba\Anaconda3\lib\site-packages\statsmodels\nonpara
metric\kde.py:488: RuntimeWarning: invalid value encountered in true_divide
  binned = fast_linbin(X, a, b, gridsize) / (delta * nobs)
C:\Users\FedericoDamianEsteba\Anaconda3\lib\site-packages\statsmodels\nonpara
metric\kdetools.py:34: RuntimeWarning: invalid value encountered in double_sc
alars
  FAC1 = 2*(np.pi*bw/RANGE)**2
C:\Users\FedericoDamianEsteba\Anaconda3\lib\site-packages\numpy\core\fromnume
ric.py:83: RuntimeWarning: invalid value encountered in reduce
  return ufunc.reduce(obj, axis, dtype, out, **passkwargs)
```

```
Out[9]: <seaborn.axisgrid.PairGrid at 0x1a3bcbe2198>
```


Channel
Region
Fresh
Milk
Grocery
Frozen
Detergents_Paper
Delicassen
Group
Group