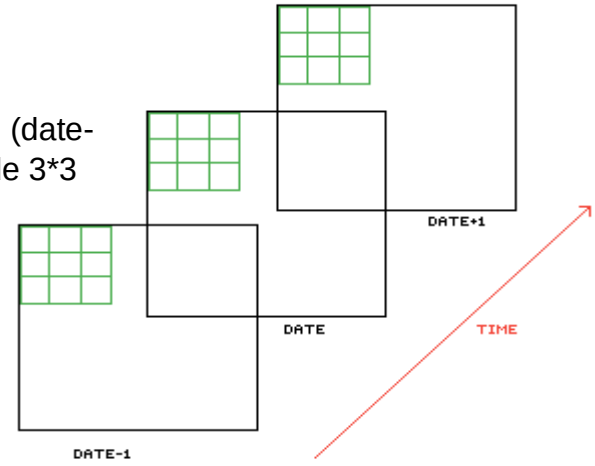
Rapport(Brouillon)



En c++ on parcourt les images des parcelles par 3 (date-1,date actuelle,date+1). Et on regarde des carrés de 3*3 pixels par image, ce qui donne 9*3 (3 images), donc un vecteur de 27 pixels. Et on compte ensuite le nombre d'apparition de ce motifs pour la parcelle actuelle ou bien pour toutes les parcelles.

Ce qui donne pour une parcelle :

	A	B	C	D	E	F	G	H
1	LogR	LogC	Rank	Counter	Images			
2	0	2,260071	1	182	20171026,	20171026,	20171026,	20171
3	0,30103	2,170262	2	148	20170404,	20170411,	20170421,	20170
4	0,477121	2,103804	3	127	20171006,	20171006,	20171006,	20171
5	0,60206	2,103804	4	127	20171008,	20171008,	20171008,	20171
6	0,69897	2,004321	5	101	20170104,	20170104,	20170104,	20170
7	0,778151	1,892095	6	78	20170312,	20170312,	20170312,	20170
8	0,845098	1,826075	7	67	20170411,	20170411,	20170411,	20170
9	0,90309	1,755875	8	57	20170918,	20170918,	20170918,	20170

Et pour toutes les parcelles :

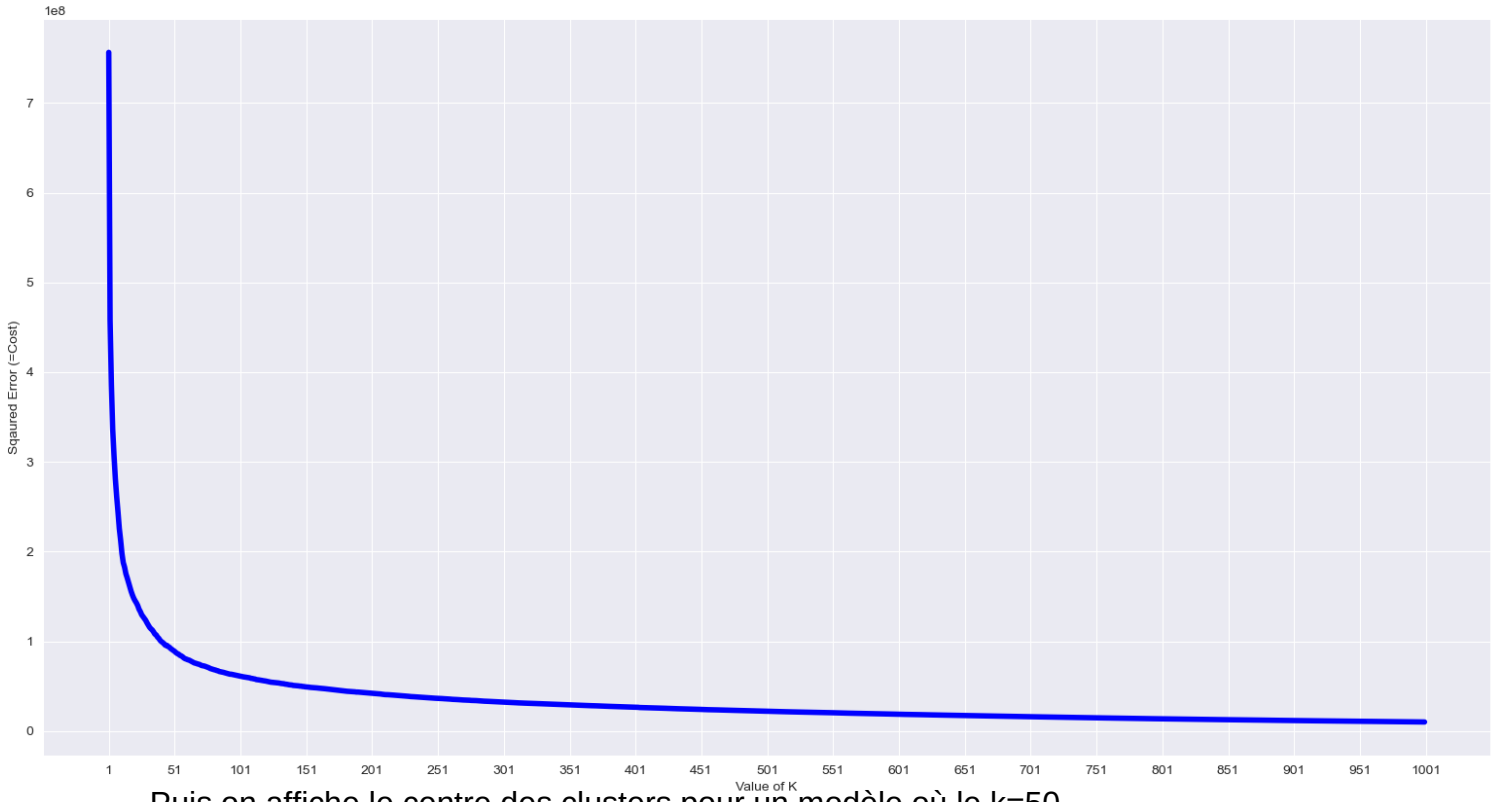
	A	B	C	D	E	F
1	LogR	LogC	Rank	Counter	Pattern	Class
2	0	3,61616	1	4132	255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,	1
3	0,30103	3,611192	2	4085	255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,	1
4	0,477121	3,574957	3	3758	198,255,255,198,255,255,198,255,255,198,255,255,198,255,255,198,255,255,198,255,255,198,255,255	0
5	0,60206	3,467608	4	2935	255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,	0
6	0,69897	3,441695	5	2765	198,255,255,198,255,255,198,255,255,198,255,255,198,255,255,198,255,255,198,255,255,198,255,255	1
7	0,778151	3,401056	6	2518	198,198,255,198,198,255,198,198,255,198,198,255,198,198,255,198,198,255,198,198,255,198,198,255	0
8	0,845098	3,396025	7	2489	124,124,198,124,124,198,124,124,198,124,124,198,124,124,198,124,124,198,124,124,198,124,124,198	1
9	0,90309	3,394802	8	2482	0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255	1
10	0,954243	3,357935	9	2280	198,198,198,198,198,198,198,198,198,198,198,198,198,198,198,198,198,198,198,198,198,198,0,0,0,0,0,0,0,	1
11	1	3,33726	10	2174	255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,	0
12	1,041393	3,334856	11	2162	255,255,198,255,255,198,255,255,198,255,255,198,255,255,198,255,255,198,255,255,198,255,255,198	1

On regarde les droites de régressions des graphiques log rang / log nombre d'apparition du motifs.
On constate dans Regression_220_and_211.xlsx que les résultats sont plutôt similaires même si le R^2 est globalement un peu supérieur pour les vergers traditionnels.
Quand on regarde l'allure des courbes dans Data_220_quantized\Log_graph et Data_221_quantized\Log_graph on constate que globalement, les courbes pour les vergers traditionnels se rapprochent plus d'une droite que celles des vergers intensifs.

Ensuite en python on transforme le **vecteur de pixels en 27 attributs ?**, et on essaye d'appliquer une régression logistique pour déterminer la classe en fonction du pattern, mais cela ne donne pas de résultats convaincant :

	precision	recall	f1-score
0	0.58	0.98	0.73
1	0.61	0.05	0.09
accuracy			0.58
macro avg	0.59	0.51	0.41
weighted avg	0.59	0.58	0.46

On cherche ensuite le meilleure k, on trouve un k d'environ 50.



Puis on affiche le centre des clusters pour un modèle où le k=50.

```
Optimal k : 50
[242.04761905 204.3968254  203.94708995 246.25396825 203.7037037
 204.55026455 245.04761905 203.64021164 202.1957672  243.84126984
 202.73544974 200.38624339 246.55555556 203.33862434 202.4973545
 245.95238095 198.6984127  198.94708995 240.82539683 198.78835979
 198.34391534 242.63492063 196.48148148 196.24867725 239.03174603]
[137.28795812 130.18848168 132.12565445 137.29842932 130.56544503
 132.5026178  137.67539267 129.28795812 129.02617801 133.93717277
 128.12565445 127.47643979 132.         129.66492147 127.46596859
 134.70157068 128.5026178  129.53926702 131.72774869 130.06282723
 130.45026178 132.2513089  131.21465969 129.79057592 133.52879581]
[ 7.30645161e+00  1.13686838e-13 -1.42108547e-13 -1.13686838e-13
  2.84217094e-14 -2.84217094e-14  5.68434189e-14  2.92258065e+01
  2.92258065e+01  2.71129032e+01 -8.52651283e-14 -8.52651283e-14
  1.13686838e-13  2.00000000e+00  2.00000000e+00  2.00000000e+00
```