

Algorithmes d'échantillonnage

Extraction d'échantillons à partir de fichiers à accès séquentiel ne tenant pas en mémoire centrale – Stratégie pour la modélisation prédictive

Ricco Rakotomalala



Echantillonnage dans le contexte big data

- Contexte big data : la taille des bases à traiter devient un enjeu essentiel
 - ➔ Chargement en mémoire de la totalité des données n'est plus possible parfois
 - ➔ Temps de traitements prohibitifs (lorsqu'ils sont possibles)
 - ➔ Développement des technologies big data (informatique distribuée essentiellement)
- Une stratégie alternative existe : travailler sur des échantillons.
 - ➔ Licite parce qu'il y a une forme de redondance (plus ou moins forte) dans les données
 - ➔ Traiter une fraction permet de généraliser (inférer) sur le reste de la base
- Deux enjeux importants :
 - ➔ **Technique** : échantillonner efficacement dans une base qu'on ne peut pas charger en mémoire
 - ➔ **Stratégique** : S'assurer une qualité de modélisation équivalente à celle du modèle construit sur la base complète (dans le cadre de l'apprentissage supervisé)



Base initiale

- Fichier texte CSV (comma separated value)
- Très gros volume, ne tient pas en mémoire centrale
- Accès séquentiel
- Individus non pondérés

Objectif – Extraction d'un échantillon de taille « n »

- Les individus doivent avoir une probabilité identique d'intégrer l'échantillon
- Traitement en une seule passe sur les données
- Production de l'échantillon directement dans un fichier à part ou conservation de l'échantillon en mémoire
- Facilement extensible pour la partition d'une base en ensembles d'apprentissage et de test pour la modélisation prédictive



Plan

1. Echantillonnage lorsque la taille N est connue
2. Reservoir sampling (N inconnu)
3. Stratégies pour l'apprentissage supervisé
4. Conclusion



Technique d'échantillonnage pour « N » connu

La taille « N » de la base de sondage est connue. Soit fournie a priori, soit parce qu'il a été possible d'effectuer une passe préalable (pas trop coûteuse en ressources) sur les données.



Utilisation d'un index

```
sample(1:N,n,replace=F) # par exemple sous R  
numpy.random.randint(1,N,n) #par exemple sous Python
```

- Générer un vecteur d'index IDX de taille « n », extrait entre 1 et N sans répétition
- Ouvrir le fichier source en lecture # pour une lecture ligne par ligne (séquentielle)
- $i := 0$ #numéro de ligne
- TANT QUE pas fin de fichier
 - Lire une ligne
 - $i := i + 1$
 - SI $i \%in\%$ IDX
 - ALORS Charger ligne en mémoire ou l'écrire dans un fichier de sortie
 - FIN SI
- FIN TANT QUE

Attention à « i », selon que la première correspond aux noms de variables ou non.

L'opérateur $\%in\%$ peut être coûteux. En triant les indices de manière croissante, on peut imaginer une gestion plus efficace, et même s'arrêter avant d'atteindre la fin du fichier.



Création d'un index

Comment générer les index si l'on ne dispose pas d'une fonction dédiée ?

- Générer **N** valeurs aléatoires
- Récupérer les indices des valeurs triées
- Récupérer les **n** premiers indices

Ce tri de N valeurs peut être coûteux en temps de calcul.

```
#valeurs aléatoires
N <- 1000
v <- runif(N)

#argument du tri
argument <- order(v)

#prendre les n premiers indices
n <- 10
idx <- argument[1:n]
print(idx)
```

Code R

```
#valeurs aléatoires
N = 1000
v = numpy.random.random(N)

#argument de tri
argument = numpy.argsort(v)

#prendre les n premiers indices
n = 10
idx = argument[:n]
print(idx)
```

Code Python



Extraction sans utilisation d'un index – Méthode de sélection rejet

On peut s'affranchir du vecteur d'index avec une gestion fine de la probabilité d'inclusion et un générateur de nombres aléatoires.

- Entrée : N , n et fichier source
- Ouvrir le fichier source # lecture séquentielle
- TANT QUE $n > 0$
 - Lire une ligne
 - SI $N * \text{ALEA()} \leq n$
 - ALORS
 - Ecrire ligne dans sortie
 - $n := n - 1$
 - FIN SI
 - $N := N - 1$ # péréquation de la probabilité d'inclusion
- FIN TANT QUE

ALEA() générateur de valeurs aléatoires $U(0,1)$

Le dispositif fonctionne même si $N = n$

Pour cette approche, les lignes dans l'échantillon sont forcément dans le même ordre que dans la source initiale.



Extraction sans utilisation d'un index (Python)

```
#vecteur de valeurs - représente le fichier source
N = 1000
source = numpy.arange(N)

#n
n = 10

#boucler tant qu'il y a des valeurs à extraire
i = -1
while (n > 0):
    #simule la lecture de la ligne
    i = i + 1
    #test d'inclusion
    if (N * numpy.random.random() <= n):
        #valeur récupérée
        print(source[i])
        #un élément en moins à extraire
        n = n - 1
    #
    N = N - 1
```

Pour les besoins de l'illustration, la source ici est indicée. Mais la transposition à un fichier à accès séquentiel est facile (« i » est incrémentée seulement d'une valeur à chaque passage dans la boucle dans notre exemple)

Pour une lecture ligne par ligne dans un fichier texte, voir `readline()` [[Les fichiers sous Python](#), page 7]



Extraction sans utilisation d'un index (R)

Pour une lecture ligne par ligne dans un fichier texte sous R, voir `readLines()` [[Read Text Lines from a Connection](#)]

```
#vecteur de valeurs
N <- 1000
source <- 1:N

#n
n <- 10
#échantillon initialisation
echantillon <- c()

#boucler
i = 0
while (n > 0){
  #id pour lecture
  i <- i + 1
  #test d'inclusion
  if (N * runif(1) <= n){
    echantillon <- c(echantillon, source[i])
    #un de moins à extraire
    n <- n - 1
  }
  #un élément de moins de la source
  #à traiter
  N <- N - 1
}
#affichage
print(echantillon)
```



Reservoir sampling

La taille « N » de la base de sondage est inconnue ou trop coûteuse à acquérir



Réservoir sampling (Algorithm R – Vitter, 1985)

Principe :

1. charger les « n » premiers individus dans une collection
2. mettre à jour cette collection au fur et à mesure de la lecture des individus restants

Contrainte :

- Maintenir la collection en mémoire
- Ou tout du moins dans une structure où un accès indicé est possible.

```
• Entrée : n et fichier source
• Sortie : C une collection avec n lignes
• Pour i:= 0 à n-1 ; C[i]:= Lire une ligne
• t := n
• TANT QUE pas fin de fichier
  • Lire une ligne
  • t := t + 1
  • k := TRUNC(t * ALEA()) # 0 ≤ k ≤ t-1
  • SI k < n
    • ALORS C[k]:= ligne
  • FIN SI
• FIN TANT QUE
```

Initialisation : remplissage du « réservoir » avec les n premières lignes du fichier source.

Permet une péréquation de la probabilité d'être dans le réservoir au fil des intégrations / retraits.

Mise à jour du « réservoir » à l'indice n°k.

Chaque item a une probabilité d'inclusion $\frac{n}{\text{card}(\text{source})}$ (Wikipédia, [Reservoir Sampling](#))



Réservoir sampling (Python)

Ici aussi, pour les besoins de l'illustration, la source est indiquée. Mais la transposition à un fichier à accès séquentiel est facile (« i » est incrémentée seulement d'une valeur à chaque passage dans la boucle dans notre exemple)

L'ordre des lignes n'est pas préservée dans cette approche.

```
import math, numpy

#vecteur de valeurs - représente le fichier source
N = 1000
source = numpy.arange(N)

#collection à remplir
n = 10
collection = numpy.zeros(n)

#remplissage du réservoir
for i in range(n):
    collection[i] = source[i]

#initialisation
t = n

#tant que pas fin de source
for i in range(n,N):
    t = t + 1
    k = math.floor(t * numpy.random.random())
    if (k < n):
        collection[k] = source[i]
    #
#
print(collection)
```



Réservoir sampling (R)

La similitude avec le code Python n'a rien de fortuit.

Dans ce code R, la boucle **for** sur la source n'est pas très efficace. Mais elle est inévitable lorsque l'on devra la transposer en une lecture ligne par ligne dans un fichier à accès séquentiel.

```
#vecteur de valeurs
N <- 1000
source <- 1:N

#n
n <- 10

#réservoir initialisation
collection <- source[1:n]

#initialisation
t = n + 1

#tant que pas fin de source
for (i in (n+1):N){
  t <- t + 1
  k <- floor(t * runif(1))
  if (k <= n){
    collection[k] <- source[i]
  }
}

#affichage
print(collection)
```



Stratégie d'échantillonnage pour l'apprentissage supervisé

Déterminer la taille suffisante d'échantillon (n^*) pour obtenir un modèle aussi performant que celui qui aurait été élaboré sur la totalité de la base



Stratégie d'échantillonnage pour l'apprentissage supervisé

Objectif

- Construire le modèle sur un échantillon
 - Avec une qualité prédictive équivalente au modèle qui aurait été élaboré sur la totalité des données
- ➔ Gain de temps, plus de possibilités d'optimisation des paramètres.
- ➔ Parfois seule solution qui rend les calculs possibles.

Démarche

- Choisir un critère d'évaluation du modèle
- Démarrer avec une taille d'échantillon initiale, modéliser, évaluer
- Augmenter graduellement la taille de l'échantillon jusqu'à ce que le critère stagne. C'est un signe que l'on épuisé l'information « utile » des données.
- Paramètres : taille initiale, nombre d'observations additionnelles à chaque étape.



Solution 1 – Random sampling – Augmentation graduelle

(pourrait être un
échantillon déjà)

Dataset pour la
modélisation

peigne	faciès	trous	hermines	griffes	age	pluma	sex	débat
6	72	35.33	0.627	50	140	0	positive	
1	66	29.28	0.351	31	85	0	negative	
8	64	0.23	0.872	32	183	0	positive	
1	66	23.28	0.167	21	89	94	negative	
0	43	39.43	2.088	33	137	168	positive	
5	74	0.25	0.201	30	114	0	negative	
3	60	32	18.0348	36	78	80	positive	
10	0	0.35	0.134	29	115	0	negative	
2	70	45.30	0.158	53	187	543	positive	
8	96	0	0.0232	54	125	0	positive	
4	92	0.37	0.191	30	110	0	negative	
10	74	0	38.0537	34	168	0	positive	
10	80	0.27	1.441	57	139	0	negative	
1	60	23.30	0.389	59	159	845	positive	
5	72	19.25	0.587	53	166	175	positive	
7	0	0	30.0484	32	100	0	positive	
0	84	47.45	0.551	31	118	230	positive	
7	74	0.29	0.254	31	107	0	positive	
1	20	38.43	0.183	33	103	83	negative	
1	70	30.34	0.629	32	115	96	positive	
3	88	41.35	0.704	27	126	225	negative	
8	84	0.35	0.388	50	99	0	negative	
7	80	0.39	0.451	41	136	0	positive	
9	80	35	29.263	29	119	0	positive	
11	94	33.95	0.254	51	143	146	positive	
10	70	28.31	0.295	41	125	115	positive	
7	76	0.39	0.257	43	147	0	positive	
1	66	15.22	0.487	22	87	145	negative	
13	82	19.22	0.245	57	145	110	negative	
5	82	0.34	0.337	38	117	0	negative	

Dataset

Dataset pour l'évaluation

(on pourrait choisir de ne travailler
que sur un échantillon pour le test)

peigne	faciès	trous	hermines	griffes	age	pluma	sex	débat
6	72	35.33	0.627	50	140	0	positive	
1	66	29.28	0.351	31	85	0	negative	
8	64	0.23	0.872	32	183	0	positive	
1	66	23.28	0.167	21	89	94	negative	
0	43	39.43	2.088	33	137	168	positive	
5	74	0.25	0.201	30	114	0	negative	
3	60	32	18.0348	36	78	80	positive	
10	0	0.35	0.134	29	115	0	negative	
2	70	45.30	0.158	53	187	543	positive	
8	96	0	0.0232	54	125	0	positive	
4	92	0.37	0.191	30	110	0	negative	
10	74	0	38.0537	34	168	0	positive	
10	80	0.27	1.441	57	139	0	negative	
1	60	23.30	0.389	59	159	845	positive	
5	72	19.25	0.587	53	166	175	positive	
7	0	0	30.0484	32	100	0	positive	
0	84	47.45	0.551	31	118	230	positive	
7	74	0.29	0.254	31	107	0	positive	
1	20	38.43	0.183	33	103	83	negative	
1	70	30.34	0.629	32	115	96	positive	
3	88	41.35	0.704	27	126	225	negative	
8	84	0.35	0.388	50	99	0	negative	
7	80	0.39	0.451	41	136	0	positive	
9	80	35	29.263	29	119	0	positive	
11	94	33.95	0.254	51	143	146	positive	
10	70	28.31	0.295	41	125	115	positive	
7	76	0.39	0.257	43	147	0	positive	
1	66	15.22	0.487	22	87	145	negative	
13	82	19.22	0.245	57	145	110	negative	
5	82	0.34	0.337	38	117	0	negative	

Augmentation de la taille de l'échantillon ($n = n + n_a$)

Echantillon, ensemble
d'apprentissage (n)

Construction de la fonction
 $f(.)$ à partir de l'échantillon

$$Y = f(X_1, X_2, \dots)$$

Application du modèle
(prédiction) sur l'ensemble de test

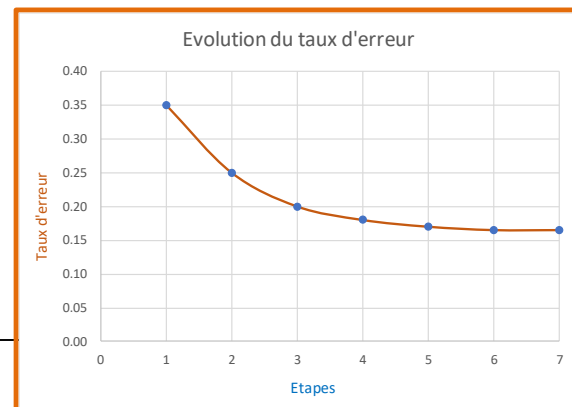
$$(Y, \hat{Y})$$

Y : valeurs observées

\hat{Y} : valeurs prédites par $f(.)$

Mesures de
performances par
confrontation
entre Y et \hat{Y} (ex.
taux d'erreur)

Courbe d'évolution du
critère d'évaluation




$$n_a = 200$$
[illegible]

		1990	1991	1992	1993	1994	1995	1996	1997	1998	1999	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020	2021	2022	2023	2024	2025	2026	2027	2028	2029	2030	2031	2032	2033	2034	2035	2036	2037	2038	2039	2040	2041	2042	2043	2044	2045	2046	2047	2048	2049	2050	2051	2052	2053	2054	2055	2056	2057	2058	2059	2060	2061	2062	2063	2064	2065	2066	2067	2068	2069	2070	2071	2072	2073	2074	2075	2076	2077	2078	2079	2080	2081	2082	2083	2084	2085	2086	2087	2088	2089	2090	2091	2092	2093	2094	2095	2096	2097	2098	2099	2100	2101	2102	2103	2104	2105	2106	2107	2108	2109	2110	2111	2112	2113	2114	2115	2116	2117	2118	2119	2120	2121	2122	2123	2124	2125	2126	2127	2128	2129	2130	2131	2132	2133	2134	2135	2136	2137	2138	2139	2140	2141	2142	2143	2144	2145	2146	2147	2148	2149	2150	2151	2152	2153	2154	2155	2156	2157	2158	2159	2160	2161	2162	2163	2164	2165	2166	2167	2168	2169	2170	2171	2172	2173	2174	2175	2176	2177	2178	2179	2180	2181	2182	2183	2184	2185	2186	2187	2188	2189	2190	2191	2192	2193	2194	2195	2196	2197	2198	2199	2200	2201	2202	2203	2204	2205	2206	2207	2208	2209	2210	2211	2212	2213	2214	2215	2216	2217	2218	2219	2220	2221	2222	2223	2224	2225	2226	2227	2228	2229	2230	2231	2232	2233	2234	2235	2236	2237	2238	2239	2240	2241	2242	2243	2244	2245	2246	2247	2248	2249	2250	2251	2252	2253	2254	2255	2256	2257	2258	2259	2260	2261	2262	2263	2264	2265	2266	2267	2268	2269	2270	2271	2272	2273	2274	2275	2276	2277	2278	2279	2280	2281	2282	2283	2284	2285	2286	2287	2288	2289	2290	2291	2292	2293	2294	2295	2296	2297	2298	2299	2300	2301	2302	2303	2304	2305	2306	2307	2308	2309	2310	2311	2312	2313	2314	2315	2316	2317	2318	2319	2320	2321	2322	2323	2324	2325	2326	2327	2328	2329	2330	2331	2332	2333	2334	2335	2336	2337	2338	2339	2340	2341	2342	2343	2344	2345	2346	2347	2348	2349	2350	2351	2352	2353	2354	2355	2356	2357	2358	2359	2360	2361	2362	2363	2364	2365	2366	2367	2368	2369	2370	2371	2372	2373	2374	2375	2376	2377	2378	2379	2380	2381	2382	2383	2384	2385	2386	2387	2388	2389	2390	2391	2392	2393	2394	2395	2396	2397	2398	2399	2400	2401	2402	2403	2404	2405	2406	2407	2408	2409	2410	2411	2412	2413	2414	2415	2416	2417	2418	2419	2420	2421	2422	2423	2424	2425	2426	2427	2428	2429	2430	2431	2432	2433	2434	2435	2436	2437	2438	2439	2440	2441	2442
--	--	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------

$$Y = f(X_1, X_2, \dots)$$
$$err(Y, \hat{Y})$$

A line graph showing the relationship between sample size and error rate for a 95% confidence interval. The x-axis is labeled 'Taille d'échantillon' (Sample Size) and ranges from 0 to 5000. The y-axis is labeled 'Taux d'erreur' (Error Rate) and ranges from 0.14 to 0.22. The curve starts at an error rate of approximately 0.23 for a sample size of 200 and decreases rapidly, reaching a plateau of about 0.14 for sample sizes greater than 2000. The curve is solid for sample sizes up to 1000 and dashed thereafter.

Taille d'échantillon	Taux d'erreur
200	0.230
400	0.182
600	0.173
800	0.167
1000	0.160
1200	0.155
1400	0.152
1600	0.150
1800	0.147
2000	0.145
2200	0.144
2400	0.144
2600	0.144
2800	0.143
3000	0.142
3200	0.142
3400	0.141
3600	0.140
3800	0.140
4000	0.140
4200	0.140
4400	0.140
4600	0.140
4800	0.140
5000	0.140



Solution 2 – Windowing

pregnant	asthmatic	hives	hayfever	pedigree	age	glaucoma	sex	diabetic
6	72	38.33.6	0.627	50	148	0	positive	
1	66	29.26.6	0.351	31	86	0	negative	
8	64	0.23.3	0.672	32	183	0	positive	
1	66	23.28.1	0.187	21	89	94	negative	
0	40	36.43.1	2.388	33	137	180	positive	
5	74	0.25.6	0.201	30	118	0	negative	
3	50	32	31.0.248	26	78	89	positive	
10	0	0.35.3	0.134	29	115	0	negative	
2	70	45.30.3	0.150	63	187	542	positive	
8	96	0	0.232	54	125	0	positive	
4	82	0.37.6	0.181	39	139	0	negative	
10	74	0	0.0.537	34	180	0	positive	
10	80	0.27.1	1.441	57	139	0	negative	
1	60	23.30.1	0.385	59	188	886	positive	
5	72	19.25.8	0.587	51	166	176	positive	
7	0	0	30.0.484	32	105	0	positive	
0	84	47.45.8	0.561	31	118	230	positive	
7	14	0.29.6	0.254	31	107	0	positive	
1	30	38.43.3	0.183	33	103	83	negative	
1	70	30.34.0	0.529	32	115	96	positive	
3	88	41.35.3	0.704	27	126	235	negative	
8	84	0.35.4	0.388	50	99	0	negative	
7	80	0.38.8	0.451	41	166	0	positive	
9	80	35	29.0.263	29	119	0	positive	
11	84	23.36.6	0.254	51	143	146	positive	
10	70	26.31.1	0.205	41	125	115	positive	
7	76	0.29.4	0.257	43	147	0	positive	
1	66	10.23.2	0.487	22	87	140	negative	
13	82	19.22.2	0.245	57	145	110	negative	
5	82	0.34.1	0.337	38	117	0	negative	

Dataset

Dataset pour la modélisation
(soit la totalité, soit un échantillon)

Ensemble d'apprentissage

1	66	23.28.1	0.187	21	89	94	negative
0	40	36.43.1	2.388	33	137	180	positive
5	74	0.25.6	0.201	30	118	0	negative
3	50	32	31.0.248	26	78	89	positive
10	0	0.35.3	0.134	29	115	0	negative
2	70	45.30.3	0.150	63	187	542	positive
8	96	0	0.232	54	125	0	positive
4	82	0.37.6	0.181	39	139	0	negative
10	74	0	0.0.537	34	180	0	positive
10	80	0.27.1	1.441	57	139	0	negative
1	60	23.30.1	0.385	59	188	886	positive
5	72	19.25.8	0.587	51	166	176	positive
7	0	0	30.0.484	32	105	0	positive
0	84	47.45.8	0.561	31	118	230	positive
7	14	0.29.6	0.254	31	107	0	positive
1	30	38.43.3	0.183	33	103	83	negative
1	70	30.34.0	0.529	32	115	96	positive
3	88	41.35.3	0.704	27	126	235	negative
8	84	0.35.4	0.388	50	99	0	negative
7	80	0.38.8	0.451	41	166	0	positive
9	80	35	29.0.263	29	119	0	positive
11	84	23.36.6	0.254	51	143	146	positive
10	70	26.31.1	0.205	41	125	115	positive
7	76	0.29.4	0.257	43	147	0	positive
1	66	10.23.2	0.487	22	87	140	negative
13	82	19.22.2	0.245	57	145	110	negative
5	82	0.34.1	0.337	38	117	0	negative

Ensemble de test

Les rajouter à l'ensemble d'apprentissage

Construction de la fonction $f(.)$ à partir de l'échantillon

$$Y = f(X_1, X_2, \dots)$$

Application du modèle (prédiction) sur l'ensemble de test

(Y, \hat{Y})

Identifier les individus mal classés

Les individus bien classés à l'étape courante deviennent l'ensemble de test

Arrêt du processus, lorsque

- Tous les individus de l'ensemble de test sont bien classés
- L'erreur ne décroît plus sur un ensemble d'évaluation à part



Solution 2 – Windowing - Exemple

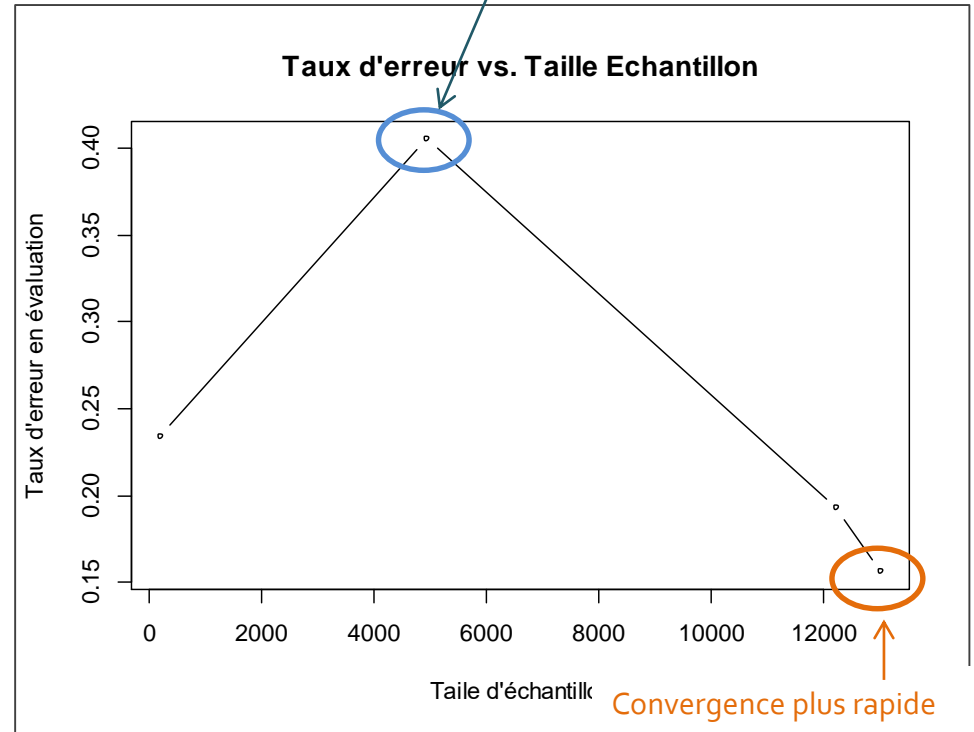
Dégradation à la première itération puisqu'on a introduit toutes les observations à problème (au détriment des données « utiles »).

Base de travail = 20000

Taille initiale échantillon = 200

Taille base évaluation = 80000

Non requis habituellement pour le windowing, mais nous l'utilisons pour suivre de taux d'erreur en généralisation.



Convergence plus rapide (moins d'itérations). Mais taille d'échantillon plus grande.

Constaté
habituellement

- Convergence plus rapide (ici arrêt parce que tous les individus sont bien classés sur le reste de la base de travail). Taille de la base finale $n^* = 12570$
- Mais le modèle final est moins bon parfois (à taille égale par rapport à la solution 1) lorsque la base est bruitée (la méthode peut mettre excessivement l'accent sur le bruit et/ou sur les outliers)
- Bon comportement en revanche sur les bases « propres » (cf. [Hoeksma](#))



Conclusion



Algorithmes d'échantillonnage rapides

- Des variantes plus rapides existent pourvu que l'on puisse « sauter » plusieurs enregistrements vers l'avant lors de la sélection des lignes (soit parce que la source est en accès indexé, soit parce que le saut est moins coûteux que la lecture ligne par ligne)
- Des variantes existent pour l'appréhension des pondérations des individus (tirage avec des probabilités inégales)
- Des algorithmes distribués existent pour tirer parti des clusters de machines

Stratégies d'échantillonnage pour la modélisation prédictive

- L'expérimentation menée ici n'a pas valeur de preuve (une base, un algo de machine learning)
- Quoiqu'il en soit, travailler sur des échantillon peut être bénéfique
- La taille optimale n^* de l'échantillon dépend de la base traitée et de la méthode de machine learning utilisée : on doit passer par l'expérimentation.



Références



- Deville J.C., Grosbras J.M., « Chapitre 10 - Algorithmes de tirage », in « Les sondages », Dreesbeke J.J., Fichet B., Tassi P., éditeurs, Economica, 1988.
- Vitter J.S., « [Random Sampling with a reservoir](#) », in ACM Transactions on Mathematical Software, 11(1), pp. 37-57, 1985.
- Wikipedia, « [Reservoir Sampling](#) ».
- Hoeksma S., « [Machine Learning and Data: Exploring the Potential of Windowing](#) ».

