Data Stream Mining- Lecture 4

Classification of Data Streams

Peter Kiss, Teaching Assistant

Eötvös Loránd University, Budapest, Hungary

November 4, 2020

Classification

Source : [Bifet et al., 2018]

Online `https://www.cms.waikato.ac.nz/~abifet/book/chapter_6.html`

Classification

- supervised learning task
- given a list of groups (often called classes), classification seeks to predict which group a new instance may belong to
- output:
  - the given class
  - probability distribution over classes

examples - binary classification:

- spam filtering - is a given email a spam?
- sentiment analysis - is a tweet express positive or negative sentiment

Training classfiers

- (x, y) is a *training example*, where $x = x_1, \ldots, x_d$ is a *feature vector* and $y \in C$ is the *label*, that is the indication of the class, and C is the set of classes
- the goal is to build a classifier : $y = f(x)$
- **training:**
  - load all the training data into the memory,
  - and making multiple passes over the data and *fit* classifier function f.

Classification of Stationary Data vs Streams

- a very important assumption in Data mining /ML an especially in classification is **iid**-ness, that is data is independently and identically distributed
- stationary distribution (doe snot change over time) producing the data in random order,
- In streaming environment not true at all: or certain time periods the labels or classes of instances are correlated.
  - In intrusion detection, there are long periods where all class labels are no-intrusion, mixed with infrequent, short periods of intrusion.

Baseline classifiers

- **Majority class** - new instance will be predicted to be the most frequent class - corresponds to random guess classifier in stationary learning
- **No-Change classifier** - label of the new instance = last label (see intrusion detection example)

Naive Bayes

Bayes theorem:

$$\text{posterior} = \frac{\text{prior} \times \text{likelihood}}{\text{evidence}} \tag{1}$$

$$Pr(c|d) = \frac{Pr(c) \cdot Pr(d|c)}{Pr(d)} \tag{2}$$

where $Pr(c)$ prior the initial probability of event c, $Pr(c|d)$ is the posterior, the probability after accounting d), $Pr(d|c)$ is the *likelihood* of event d given that event c occurs, and $Pr(d)$

Derivation - from conditional probability

$$Pr(c \cap d) = \Pr(c)Pr(d|c) = Pr(d)Pr(c|d) \tag{3}$$

$$Pr(c|d) = \frac{Pr(c) \cdot Pr(d|c)}{Pr(d)} \tag{4}$$

Building NB classifier

incremental -thus well suited for streaming

Let $x_1, \ldots, x_d$ discrete attributes, assuming that x_i may take n_i different values.

Upon receiving an unlabelled instance $I = (x_1 = v_1, \ldots, x_d = v_d)$ the naive Bayes classifier assigns the probability for I belonging to class $c \in C$

$$Pr(C = c|I) = Pr(C = c) \cdot \prod_{i=1}^{d} Pr(x_i = v_i|C = c) \quad (5)$$

$$Pr(C = c|I) = Pr(C = c) \cdot \prod_{i=1}^{d} \frac{Pr(x_i = v_i \wedge C = c)}{Pr(C = c)} \quad (6)$$

$$(7)$$

where $Pr(C = c)$ and $Pr(x_i = v_i \wedge C = c)$ comes from simply counting the occurrences in the training data.

Naive Bayes example for sentiment analysis - Training

Training: First let us assume that we have 4 labelled example

ID	Text	Sentiment
T1	glad happy glad	+
T2	glad joyful glad	+
T3	glad pleasant	+
T4	miserable sad glad	-

Then count how many times a given word appears in the example - transform into vector of attributes, where attributes are element of the dictionary, and values 0 or 1/`true` or false:

Id	glad	happy	joyful	pleasant	miserable	sad	Sentiment
T1	1	1	0	0	0	0	+
T2	1	0	1	0	0	0	+
T3	1	0	0	1	0	0	+
T4	1	0	0	0	1	1	−

Naive Bayes example for sentiment analysis- Training

then check in how many examples of each class the words occur

Class	Value	glad	happy	joyful	pleasant	miserable	sad
+	**1**	3	1	1	1	0	0
+	**0**	0	2	2	2	3	3
−	**1**	1	0	0	0	1	1
−	**0**	0	1	1	1	0	0

Naive Bayes example for sentiment analysis- Prediction

ID	Text	Sentiment
T5	glad sad miserable pleasant glad	?

translating into a feature vector(feature = vocabulary)

ID	glad	happy	joyful	pleasant	miserable	sad	Sentiment
T5	1	0	0	1	1	1	?

Naive Bayes example for sentiment analysis- Prediction

ID	glad	happy	joyful	pleasant	miserable	sad	Sentiment
T5	1	0	0	1	1	1	?

Probabilities :

$$Pr(+|T5) = Pr(+) \cdot Pr(glad = 1|+) \cdot Pr(joyful = 0|+) \cdot Pr(happy = 0|+) \quad (8)$$

$$\cdot Pr(pleasant = 1|+) \cdot Pr(miserable = 1|+) \cdot Pr(sad = 1|+) \quad (9)$$

$$= \frac{3}{4} \cdot \frac{3}{3} \cdot \frac{2}{3} \cdot \frac{2}{3} \cdot \frac{1}{3} \cdot \frac{0}{3} \cdot \frac{0}{3} = 0 \quad (10)$$

where $Pr(glad = 1|+)$ means in what fraction of the positive cases occurs the attribute value $glad = 1$ That is $Pr(glad = 1 \wedge C = +)/Pr(glad = 1 \wedge C = +) + Pr(glad = 0 \wedge C = +)$.

Class	Value	glad	happy	joyful	pleasant	miserable	sad
+	**1**	3	1	1	1	0	0
+	**0**	0	2	2	2	3	3
−	**1**	1	0	0	0	1	1
−	**0**	0	1	1	1	0	0

Naive Bayes example for sentiment analysis- Prediction

ID	glad	happy	joyful	pleasant	miserable	sad	Sentiment
T5	1	0	0	1	1	1	?

$$Pr(-|T5) = Pr(-) \cdot Pr(glad = 1|-) \cdot Pr(joyful = 0|-) \cdot Pr(happy = 0|-) \quad (11)$$

$$\cdot Pr(pleasant = 1|-) \cdot Pr(miserable = 1|-) \cdot Pr(sad = 1|-) \quad (12)$$

$$= \frac{1}{4} \cdot \frac{1}{1} \cdot \frac{1}{1} \cdot \frac{1}{1} \cdot \frac{0}{1} \cdot \frac{1}{1} \cdot \frac{1}{1} = 0 \quad (13)$$

Class	Value	glad	happy	joyful	pleasant	miserable	sad
+	**1**	3	1	1	1	0	0
+	**0**	0	2	2	2	3	3
−	**1**	1	0	0	0	1	1
−	**0**	0	1	1	1	0	0

Naive Bayes example for sentiment analysis- Prediction

There is a minor problem however:

$$Pr(-|T5) = \frac{1}{4} \cdot \frac{1}{1} \cdot \frac{1}{1} \cdot \frac{1}{1} \cdot \frac{0}{1} \cdot \frac{1}{1} \cdot \frac{1}{1} = 0 \tag{14}$$

$$Pr(+|T5) = \frac{3}{4} \cdot \frac{3}{3} \cdot \frac{2}{3} \cdot \frac{2}{3} \cdot \frac{1}{3} \cdot \frac{0}{3} \cdot \frac{0}{3} = 0 \tag{15}$$

Naive Bayes example for sentiment analysis- Prediction

There is a minor problem however:

$$Pr(-|T5) = \frac{1}{4} \cdot \frac{1}{1} \cdot \frac{1}{1} \cdot \frac{1}{1} \cdot \frac{0}{1} \cdot \frac{1}{1} \cdot \frac{1}{1} = 0 \quad (14)$$

$$Pr(+|T5) = \frac{3}{4} \cdot \frac{3}{3} \cdot \frac{2}{3} \cdot \frac{2}{3} \cdot \frac{1}{3} \cdot \frac{0}{3} \cdot \frac{0}{3} = 0 \quad (15)$$

Laplace correction : $Pr(d|c) = \frac{n_{dc}+1}{n_d+n_c}$, n_{dc} - how many times the given value occurs in cases where the class is c, n_c is number of classes, n_d number of occurrence of the class
in practice we initialize the counting table with 1s:

Class	Value	glad	happy	joyful	pleasant	miserable	sad
+	**1**	4	2	2	2	1	1
+	**0**	1	3	3	3	4	4
−	**1**	2	1	1	1	2	2
−	**0**	1	2	2	2	1	1

Naive Bayes example for sentiment analysis- Prediction

$$Pr(+|T5) = Pr(+) \cdot Pr(glad = 1|+) \cdot Pr(joyful = 0|+) \cdot Pr(happy = 0|+) \tag{16}$$

$$\cdot Pr(pleasant = 1|+) \cdot Pr(miserable = 1|+) \cdot Pr(sad = 1|+) \tag{17}$$

$$= \frac{3}{4} \cdot \frac{4}{5} \cdot \frac{3}{5} \cdot \frac{3}{5} \cdot \frac{1}{5} \cdot \frac{1}{5} = 0.0128 \tag{18}$$

$$Pr(-|T5) = Pr(+) \cdot Pr(glad = 1|-) \cdot Pr(joyful = 0|-) \cdot Pr(happy = 0|-) \tag{19}$$

$$\cdot Pr(pleasant = 1|-) \cdot Pr(miserable = 1|-) \cdot Pr(sad = 1|-) \tag{20}$$

$$= \frac{1}{4} \cdot \frac{2}{3} \cdot \frac{2}{3} \cdot \frac{2}{3} \cdot \frac{1}{3} \cdot \frac{2}{3} \cdot \frac{2}{3} = 0.0987 \tag{21}$$

Prediction: $0.0128 = Pr(+|T5) < Pr(-|T5) = 0.0987$ -¿ therefore our prediction is **negative**.

Multinomial Naive Bayes (MNB) [McCallum et al., 1998]

- Extension for document classification
- a document is considered as *bag of words*

$$Pr(c|d) = \frac{Pr(c)\prod_{w\in d} Pr(w|c)^{n_{wd}}}{Pr(d)} \tag{22}$$

  for $Pr(d)$ normalization factor (thus we omit it usually), n_{wd} is the number of times that word w occurs in document d.

- Key difference is the interpretation of the likelihoods $Pr(w|c)$ - here stands for number of occurrences of w in documents of class c over the total number of words in documents in c. That is the **probability of observing w at any position of a document belonging to c.**

MNB

$$Pr(c|d) = \frac{Pr(c)\prod_{w\in d} Pr(w|c)^{n_{wd}}}{Pr(d)} \tag{23}$$

- Thus absence of a word in a document does not make any class more likely than any other, since it means that the exponent of the probability in 0
- $n_{wd}, PR(w|c, Pr(c))$ are trivial to estimate on a data stream, by keeping the appropriate counts.
- (Laplace correction can be also used, as we do here too)

MNB

ID	Text	Sentiment
T1	glad happy glad	+
T2	glad joyful glad	+
T3	glad pleasant	+
T4	miserable sad glad	-

compute the occurrences of w per class c (all the documents, all the occurrences)

Class	glad	happy	joyful	pleasant	miserable	sad	Total
+	6	2	2	2	1	1	8+6=14
−	2	1	1	1	2	2	3+6=9

Frame Title

Now the prediction:

$$Pr(+|T5) = Pr(+) \cdot Pr(glad = 1|+) \cdot Pr(joyful = 0|+) \cdot Pr(happy = 0|+) \tag{24}$$

$$\cdot Pr(pleasant = 1|+) \cdot Pr(miserable = 1|+) \cdot Pr(sad = 1|+) \tag{25}$$

$$= \frac{3}{4} \cdot (\frac{6}{14})^2 \cdot (\frac{2}{14})^1 \cdot (\frac{1}{14})^1 \cdot (\frac{1}{14})^1 = 10.04 \cdot 10^{-5} \tag{26}$$

$$Pr(-|T5) = Pr(+) \cdot Pr(glad = 1|-) \cdot Pr(joyful = 0|-) \cdot Pr(happy = 0|-) \tag{27}$$

$$\cdot Pr(pleasant = 1|-) \cdot Pr(miserable = 1|-) \cdot Pr(sad = 1|-) \tag{28}$$

$$= \frac{1}{4} \cdot (\frac{2}{9})^2 \cdot (\frac{1}{9})^1 \cdot (\frac{2}{9})^1 \cdot (\frac{2}{9})^1 = 6.77 \cdot 10^{-5} \tag{29}$$

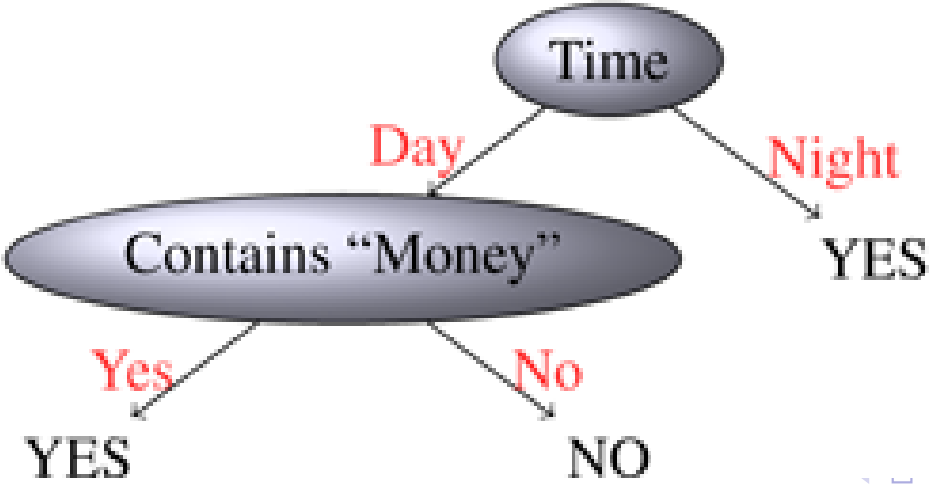
thus now $Pr(+|T5) > Pr(-|T5)$, and our prediction is **positive**.

Decision tree

- very popular, easy to interpret and visualize
- each internal node corresponds to an attribute that splits into a branch for each attribute value and the leaves correspond to classification predictor usually majority class classifiers.
- Building decision tree
  1. if instances are classified perfectly stop
  2. let attribute A be the best decision criterion of the actual leaf node
  3. for each value of A create a new leaf

Decision Tree - Spam classification example[Bifet et al., 2018]

Contains "Money"	Domain type	Has attach.	Time received	spam
yes	com	yes	night	yes
yes	edu	no	night	yes
no	com	yes	night	yes
no	edu	no	day	no
no	com	no	day	no
yes	cat	no	day	yes



Splitting nodes

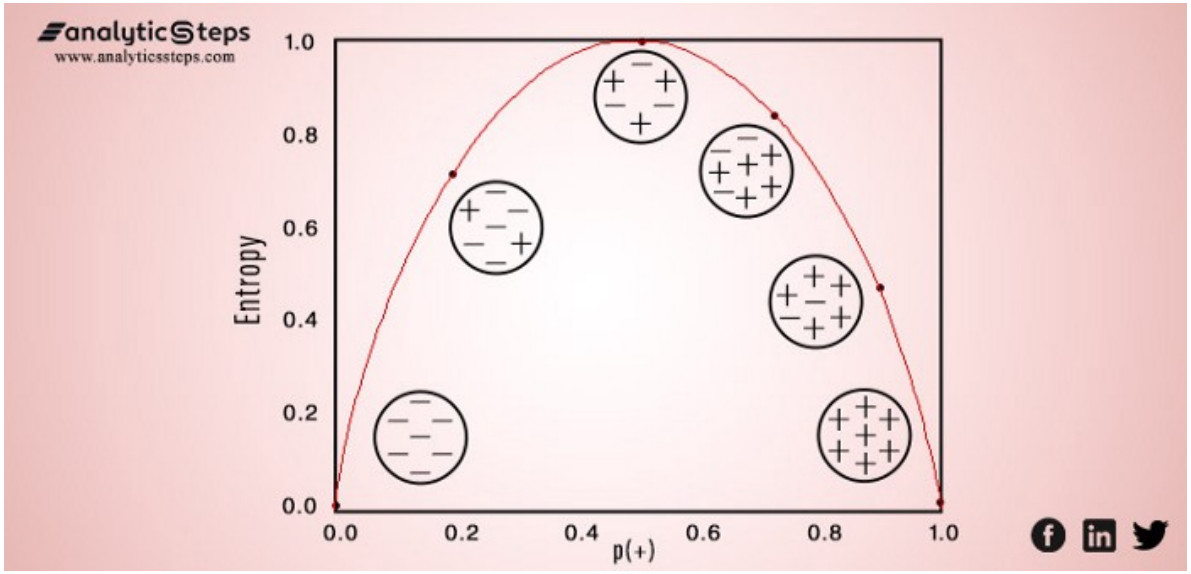
How to pick A?
Our goal is to have leaves in the tree, that sort data points into *pure* groups
MEasures of purity:

- Entropy
- Giniindex

Split the nodes if it reduces *impurity*

Entropy

Entropy $H(S) = -\sum_c p_c \log(p_c)$ - a measurement of the impurity or randomness in the data points



Information gain

- which feature provides maximal information
- difference between entropy of the class before and after splitting
- after splitting the attribute: $H(S, A) = \sum_a H(S_a)|S_a|/|S|$ for S_a the subset of S where $A = a$
- the information gain $IG(S, A) = H(S) - H(S, A)$. for C4.5 DT algorithm

Gini

- Gini index =gini impurity calculates the amount of probability of a specific feature that is classified incorrectly when selected randomly. 0 means; purity 0.5 of the Gini Index shows an equal distribution of elements over some classes, that is the less is the better. It is a nonlinear measure of dispersion of C:
  $G(C) = \sum_c p_c(1 - p_c) = \sum_{c \neq c'} p_c p_{c'}$
- Gini impurity reduction: difference between the Gini index before and after splitting the attribute (CART DT alg)

$$Gini(S, A) = \sum_{a \in A} \sum_{c \in C} p_c(1 - p_c)|S_a|/|S| \tag{30}$$

$$= \sum_{a \in A} (1 - \sum_{c \in C} p_c^2)|S_a|/|S| \tag{31}$$

  (since $\sum_c p_c = 1$), with p_c here $p_c = Pr(C = c \wedge A = a)$
- Gini reduction or what : $\Delta Gini(S, A) = Gini(S) - Gini(S, A)$

Attribute selection

we want an attribute, where the sum of the gini / entropy is minimal if we split across that is

$$\max_A \Delta Gini(S, A) \qquad \text{or} \qquad \max_A IG(S, A) \tag{32}$$

Hoeffding Tree

When should we split: confidence interval for estimating entropy of the node:

$$\epsilon = \sqrt{\frac{R^2 \ln 1/\delta}{2n}} \tag{33}$$

where R is the range of the random variable, δ the desired probability of an estimate not being within ϵ of its expected value, and n is the number of examples collected at the node.
(using Hoeffding bound is formally incorrect but still widely used,)
for Information gain the entropy is in the range $[0, \ldots, \log n_c]$

Hoeffding Tree - Streaming DT [Domingos and Hulten, 2000]

- maintain in each node statistics for splitting
- for discrete attributes it is the same as for NB: for each triple count (x_i, v_j, c) $n_{i,j,c}$ where $x_i = v_j$ and a one dimensional table for the counts $C = c$ that is for a class a given attribute how many times takes the different possible values:

Class	Value	glad	happy	joyful	pleasant	miserable	sad
+	**1**	3	1	1	1	0	0
+	**0**	0	2	2	2	3	3
−	**1**	1	0	0	0	1	1
−	**0**	0	1	1	1	0	0

- memory depends on the number of leaves not examples
- if at the node the examples seen so far are not from the same class compute G for each attribute - (G=IG or Gini reduction(TODO name?))we want to split at the biggest "impurity"

  if $G(bestattribute) - G(secondbestattribut) > \sqrt{\frac{R^2 \ln 1/\delta}{2n}}$ split the node on the best

Very Fast Decision Tree (VFDT)

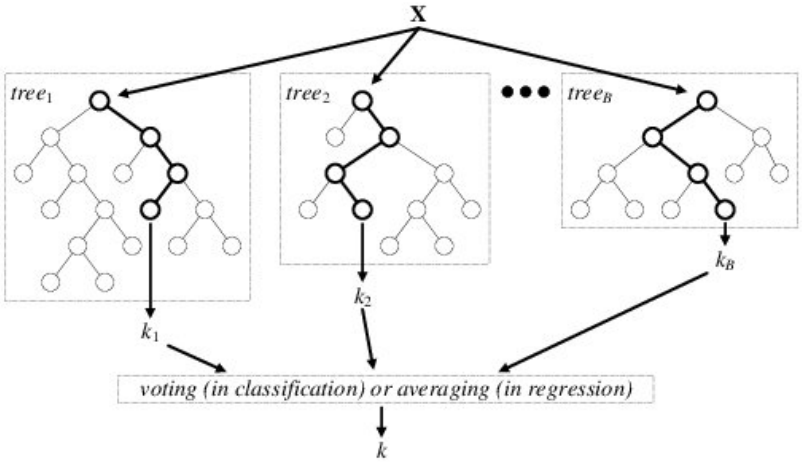
- Instead of computing the best attributes to split at every new instance, wait for n_{min} instances
- to reduce memory usage - deactivate least promising node: that is lowest number for $p_l \times e_l$ for
  - p_l is the probability to reach leaf l
  - e_l error in node l
- method can be strarted from any existing tree $\rightarrow$ possibility to pretrain model on existing data (HT grows slowly and performance might be poor initially)

Numeric attirbutes

- much harder for streaming - **discretization**:
  - *Equal width* equal length bins simplest - but vulnerable to outliers and skewed distributions
  - *equal frequency* -same number of elements - need for sorting elements more processing time
  - *Fayyad and Irani's method* finding best cutting points as in decision trees - first sort then take each pair as a candidate - maximum IG - recursively - stop when intervals become clear

Ensemble

- combinations individual predictions of smaller models to form a final prediction
- voting, averaging, ...
- tend to improve prediction accuracy
- more time and memory
- parallelizable



Source: [Verikas et al., 2016]

Weighted voting

- simplest method
- returns most popular class
- for C_i being class predicted by ith model, for binary classification and a suitable threshold θ:

$$C(x) = \begin{cases} 1 & \text{if } \sum_i w_i C_i(x) > \theta \\ 0 & \text{otherwise} \end{cases} \tag{34}$$

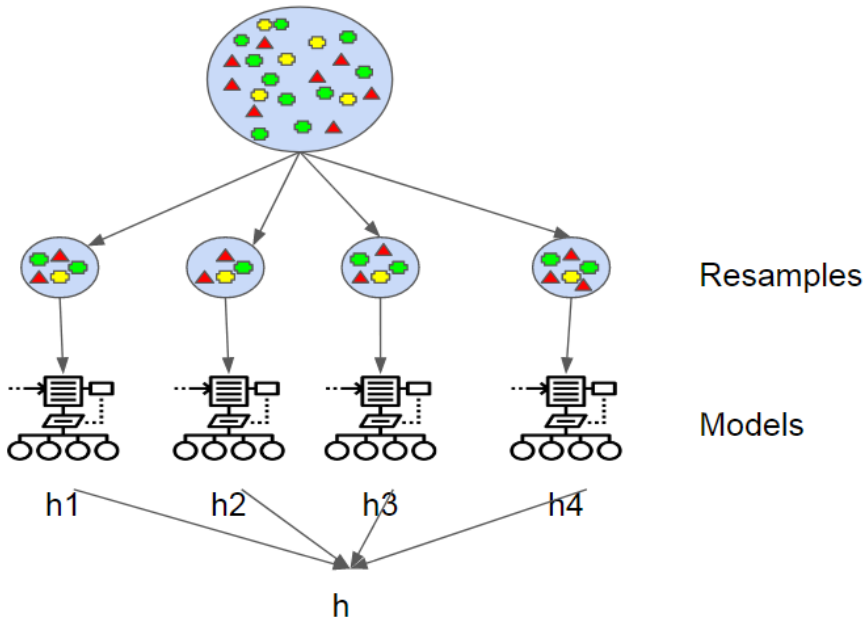
- weights can be fixed, or vary over time

Accuracy-Weighted Ensembles(AWE)[Wang et al., 2003]

- mine evolving datastreams using nonstreaming learners
- stream is processed in chunks
- new classifier for each chunk
- every time new classifier added, oldest removed
- [Wang et al., 2003] proposes to have $w_i \propto err_r - err_i$ for err_r the error of random classifier

Bagging[Breiman, 1996]

- use a base learning algorithm to produce M model by *bootstrapping*:
  - take samples from the data with *replacement*
- prediction by majority vote



Source:
https://commons.wikimedia.org/wiki/File:Bagging.png

Online bagging

- difficult to implement sampling with replacement
- simulation instead:
  - in a bootstrap replica the number of copies K for each of the n samples:

$$P(K = k) = \binom{n}{k} p^k (1-p)^{n-k} = \binom{n}{k} \frac{1}{n}^k (1 - \frac{1}{n})^{n-k} \quad (35)$$

  - for large value of n sample Binomial$\rightarrow$Poisson(1) distribution, for

$$\text{Poisson}(1) = \frac{e^{-1}}{k!} \quad (36)$$

  - OzaBag[Oza and Russell, 2001][Oza, 2005]: for each example the models will be updated with the weight of the example Poisson(1)

OzaBag

ONLINE BAGGING(*Stream*, M)

Input: a stream of pairs (x, y), parameter M = ensemble size
Output: a stream of predictions $\hat{y}$ for each x

1 initialize base models h_m for all $m \in \{1, 2, \ldots, M\}$
2 **for** each example (x, y) in *Stream*
3 **do** predict $\hat{y} \leftarrow \arg\max_{y \in Y} \sum_{t=1}^{T} I(h_t(x) = y)$
4 **for** $m = 1, 2, \ldots, M$
5 **do** $w \leftarrow Poisson(1)$
6 update h_m with example (x, y) and weight w

Bernoulli

Describes a single trial with 2 possible outcome: success $X = 1$ and failure $X = 0$
Pmf: $P(X = x) = p^x(1 - p)^{1-x}$ for $x = 0$ or 1
$\mu = p$
$\sigma^2 = p(1 - p)$

Significance

Common discrete prob distributions are built on the assumption of independent Bernoulli trials:

- **Binomial**: number of successes in n independent Bernoulli trials,
- **Geometric**: distribution of the number of trials to get the first success in independent Bernoulli trials,
- **Negative Binomial**: distribution of the number of trials to get the rth success in independent Bernoulli trials.

Binomial Distribution

$$\text{Binomial coefficient: } \binom{n}{x} = \frac{n!}{x!(n-x)!} \tag{37}$$

Assumptions

- n independent trials = an outcome does not give any information on other trials
- each trial result in one of two possible outcomes success or failure

$p = P(success), P(failure) = 1 - p$

Binomial distribution

number of successes in n independent Bernoulli trials has a binomial distribution
X represents the number of successes in n trials
$P(X = x) = \binom{n}{x} p^x (1-p)^{n-x}, for x = 1, \ldots, n$
$\mu = \mathbb{E}(X) = np \; \sigma^2 = np(1-p)$
$p^x (1-p)^{n-x}$ probability of one specific ordering, and there are $\binom{n}{x}$ possible orderings

Poisson Distribution

Counting the number of occurrences of an event in a given unit of time.

Assumption

- events occur independently - given that one event happened gives no information at all when another even will occur

probability that an event occurs in a given length of time does not change over time

Poisson distribution

X the number of events in a fixed unit of time has a Poisson distribution
Pmf: $p(x) = P(X = x) = \frac{\lambda^x e^{-\lambda}}{X!}$ $x = 0, 1, \ldots, \infty$
Mean: $\mu = \lambda$
Variance: $\sigma^2 = \lambda$

Binomial and Poisson

Relationship

- is useful to decide whether a variable has a Poisson distribution or not
- The Poisson distribution with $\lambda = np$ closely approximates the Binomial distribution if n large and p small
- that is Bin$\rightarrow$Poi, as $m \rightarrow \infty$, $p \rightarrow 0$,

YouTube - for the distributions

Bernoulli: `https://www.youtube.com/watch?v=bT1p5tJwn_0&list=PLvxOuBpazmsNIHP5cz37oOPZx0JKyNszN&index=3`

Binomial:`https://www.youtube.com/watch?v=qIzC1-9PwQo&list=PLvxOuBpazmsNIHP5cz37oOPZx0JKyNszN&index=4`

Poisson:

`https://www.youtube.com/watch?v=jmqZG6roVqU&t=165s`

Bibliography I

Bifet, A., Gavaldà, R., Holmes, G., and Pfahringer, B. (2018).
Machine Learning for Data Streams with Practical Examples in MOA.
MIT Press.
https://moa.cms.waikato.ac.nz/book/.

Breiman, L. (1996).
Bagging predictors.
Machine learning, 24(2):123–140.

Domingos, P. and Hulten, G. (2000).
Mining high-speed data streams.
In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 71–80.

McCallum, A., Nigam, K., et al. (1998).
A comparison of event models for naive bayes text classification.
In *AAAI-98 workshop on learning for text categorization*, volume 752, pages 41–48. Citeseer.

Oza, N. C. (2005).
Online bagging and boosting.
In *2005 IEEE international conference on systems, man and cybernetics*, volume 3, pages 2340–2345. Ieee.

Bibliography II

Oza, N. C. and Russell, S. (2001).

Experimental comparisons of online and batch versions of bagging and boosting.
In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 359–364.

Verikas, A., Vaiciukynas, E., Gelzinis, A., Parker, J., and Olsson, M. C. (2016).

Electromyographic patterns during golf swing: Activation sequence profiling and prediction of shot effectiveness.
Sensors, 16:592.

Wang, H., Fan, W., Yu, P. S., and Han, J. (2003).

Mining concept-drifting data streams using ensemble classifiers.
In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 226–235.