Regression Models In Data Streams

By Davis Cho and Qiwen Lyu

RoadMap

- Background
- Market Analysis Model
- FTP-DS
- Temporal Patterns
- Missing Values

Background

The problem of data stream:

- One pass and Volume

Solution

- Approximation and adaptivity

Model

- Stream processor and the synopsis maintenance in memory
- Buffer is like a recent part of the data from data streams
- Majority methods were designed for traditional database

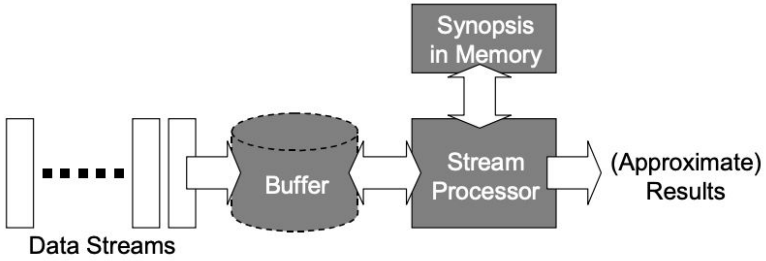
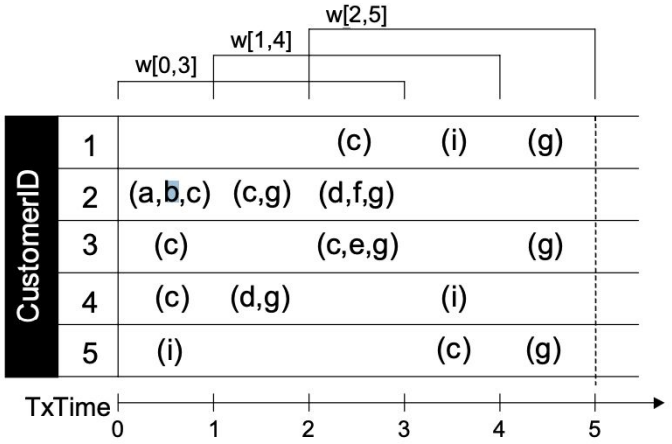


Figure 1: Computation model for data streams

Market Analysis Model



- Market Basket Analysis: Key techniques

used by large retailers.

- Example: Third customer bought item c [0,1], ceg[2,3],g[4,5]
- Problem : difficult to conduct the frequent pattern identification due to the limited time and space constraints

FTP-DS: Frequent Temporal Patterns of Data Streams

Two features

- One data scan for online statistics collection
  - To attain the feature, the data segmentation and the pattern growth scenarios are used
  - Scans online transaction flows and generates candidate frequent patterns in real time
- regression based compact pattern representation
  - Create for pattern representation a compact ATF(Accumulated Time and Frequency)

Temporal Patterns

A temporal pattern is defined as a segment of signals that recurs frequently in the whole temporal signal sequence.

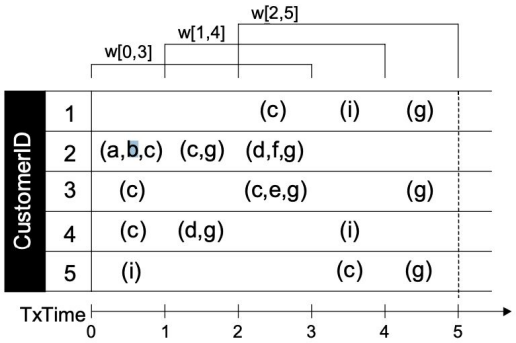
- In many applications:a time constraint is imposed during the mining process to meet the respective constraint.

Support Framework for Temporal Patterns:

- In the previous example, we have <TxTime, CustomerID, itemset>, all the transactions of a customer can be viewed as sequence together.
- Problem arise due to the different support definitions and can not store all the historical data.
- Solution: Formulate a temporal pattern

Definition of the support of a temporal pattern

Definition 1 The support or the occurrence frequency of a temporal pattern X at a specific time t is denoted by the ratio of the number of customers having pattern X in the current time window to the total number of customers.



TxTime		Occurrence(s) of {c, g}	Support
t=1	w[0,1]	none	0
t=2	w[0,2]	CustomerID={2, 4}	2/5=0.4
t=3	w[0,3]	CustomerID={2, 3, 4}	3/5=0.6
t=4	w[1,4]	CustomerID={2, 3}	2/5=0.4
t=5	w[2,5]	CustomerID={1, 3, 5}	3/5=0.6

Table 1: The support values of the inter-transaction itemset $\{c, g\}$

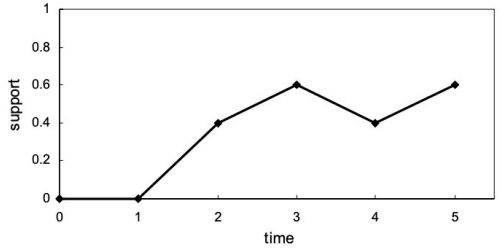


Figure 3: Support variations of the inter-transaction itemset (c, g)

Major Features of Algorithm FTP-DS

One Scan for Statistics Collection:

Both method are designed to obtain approximate answers

- Probabilities error bound
  - Based on the sampling technique
- deterministic error bound
  - Based on data segmentation technique
- Those approaches might not work that well for singleton. -> prohibitive storage and computing overheads.
- Solution: Sliding window model employed, only the occurrences of singleton items are being counted in the first time window.
- After the counting iteration, frequent items which have supports no less than the specified threshold are identified.

Example

MinSup = 0.4

Window size N = 3

The averaged support value is represented by

(accumulated supports over windows)/(number of recorded windows)

- Since a pattern is not taken as a candidate to accumulate its occurrence counts before all its subsets are found frequent

t=1	
{c}	0.6/1

(a)

t=2	
{c}	1.2/2
{g}	0.4/1

(b)

t=3	
{c}	2/3
{d}	0.4/1
{g}	1/2
{c,g}	0.6/1

(c)

t=4	
{c}	2.8/4
{d}	0.8/2
{g}	1.6/3
{i}	0.4/1
{c,g}	1/2
{d,g}	0.4/1

(d)

t=5	
{c}	3.4/5
{g}	2.4/4
{i}	0.8/2
{c,g}	1.6/3

(e)

Table 2: Generation of frequent temporal itemsets (MinSup=0.4)

Major Features of Algorithm FTP-DS

Regression-Based Analysis on Frequent Patterns

A straight-line fit for a
time series s(t), which corresponds to the frequency variation of a temporal pattern, is a linear estimation function bf = αb + βbt that conforms to the principle of least squares. Specifically, the regression parameters αb and βb are chosen to make the residual sum of squares D = summation(fi − α − βt)^2 minimal, where fi is the actual frequency in the i-th recorded point.

To perform the calculations for getting best estimates of $\widehat{\alpha}$ and $\widehat{\beta}$, the following quantities are maintained,

$$\begin{aligned}
\overline{t} &= \frac{1}{n}\sum t, \overline{f} = \frac{1}{n}\sum f, \\
S_{tt} &= \sum (t-\overline{t})^2 = \sum t^2 - \frac{(\sum t)^2}{n}, \\
S_{ff} &= \sum (f-\overline{f})^2 = \sum f^2 - \frac{(\sum f)^2}{n}, \text{ and} \\
S_{tf} &= \sum (t-\overline{t})(f-\overline{f}) = \sum tf - \frac{(\sum t)(\sum f)}{n}.
\end{aligned}$$

Then, the least square estimates of $\widehat{\alpha}$ and $\widehat{\beta}$ are computed by

$$\widehat{\alpha} = \overline{f} - \widehat{\beta}\overline{t}, \text{ and } \widehat{\beta} = \frac{S_{tf}}{S_{tt}}.$$

In addition, the strength of a linear relation is measured by

$$r^2 = \frac{(S_{tf})^2}{S_{tt}S_{ff}}$$

Accumulated Time and Frequency

Definition 2 *The ATF form of the time series corresponding to the frequency variation of a temporal pattern is* $(t_s, \sum tf, \sum f, \sum f^2)$, *where* t_s *is the starting time,* $\sum tf$ *is the accumulated product of time and support, and* $\sum f$ *and* $\sum f^2$ *are, respectively, the sum and the squared sum of pattern frequencies since the pattern is recorded.*

Proof: In a data stream environment, the current time t_{now} is always known and up-to-date as time advances. Using the starting time t_s, the accumulated values of $\sum t$ and $\sum t^2$ in the ATF form of a pattern can be obtained by $\sum t = \sum_{t_i=t_s}^{t_{now}} t_i$ and $\sum t^2 = \sum_{t_i=t_s}^{t_{now}} t_i^2$, since the corresponding time series for a pattern is composed of the averaged support values at every time unit during $t=[t_s, t_{now}]$. In addition, the number of recorded points is $n=t_{now}-t_s+1$.

Together with the other three measures, i.e., $\sum tf$, $\sum f$, $\sum f^2$, the values of S_{tt}, S_{ff} and S_{tf} can be obtained through the equations:

$$\begin{aligned} S_{tt} &= \sum t^2 - \frac{(\sum t)^2}{n}, \\ S_{ff} &= \sum f^2 - \frac{(\sum f)^2}{n}, \text{ and} \\ S_{tf} &= \sum tf - \frac{(\sum t)(\sum f)}{n}. \end{aligned}$$

Consequently, the least square estimates of $\widehat{\alpha}$ and $\widehat{\beta}$ are computed by

$$\begin{aligned} \widehat{\beta} &= \frac{S_{tf}}{S_{tt}}, \text{ and} \\ \widehat{\alpha} &= \overline{f} - \widehat{\beta}\overline{t} = \frac{\sum f}{n} - \widehat{\beta} \times \frac{\sum t}{n}. \end{aligned}$$

Therefore the fit line $\widehat{f} = \widehat{\alpha} + \widehat{\beta}t$ for this pattern can be precisely extracted from the ATF compact form. **Q.E.D.**

Algorithm of FTP-DS

Since a frequent pattern does
not always have a very steady frequency, the frequency threshold can be slightly lifted to be higher than the real threshold MinSup. This option can help reducing the overhead for recording unnecessary patterns with the trade-off of a delayed recognition of some frequent patterns.

Algorithm FTP-DS: Frequent Temporal Patterns of Data Streams
Input: The window size N, and the support threshold MinSup
Output: The set of frequent temporal patterns F with their ATF forms

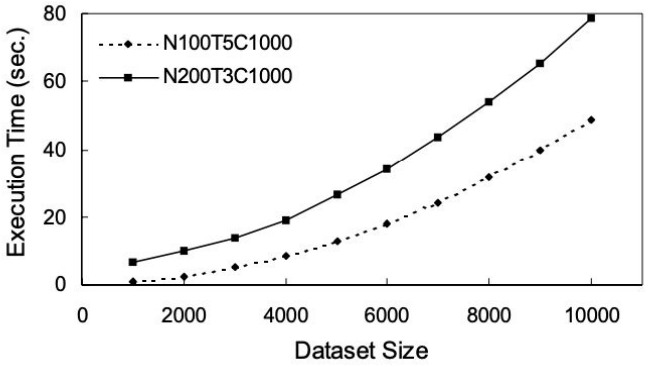
```
1. t=0;
2. buf=NULL;  //buffer for storing recent transactions
3. F={all singleton items};  //initial candidate patterns
4. while(1){
5.    wait until t=t+1;
6.    data_t=transactions in time slot t;
7.    buf=(buf∪data_t)−data_{t−N};  //data_{t−N} is expired
8.    foreach(p∈F){
9.       p.count=0;
10.      p.nWindow+=1;
11.   }
12.   count occurrences of each p∈F from buf;
13.   foreach(p∈F){
14.      p.sup=(p.sup+p.count/nCustomer)/p.nWindow;
15.      if(p.sup<MinSup)  remove p from F;
16.      else              update the ATF forms of p;
17.   }
18.   F=F∪{candidates generated from F};
19.}
```

Advantages of FTP

```
SELECT *
FROM DataStream
WHERE (Query Clause)
BEGIN (BeginTime)
END (EndTime).
```

- Flexible Time intervals
  - Example
  - In the proposed framework, since the frequency variations of frequent patterns are recorded, the answers to queries with variable time intervals are directly supported.
- Trend detection
  - Can detect pattern
  - Even if pattern changes, can still detect it in later intervals

Feasibility and Scalability of FTP-DS



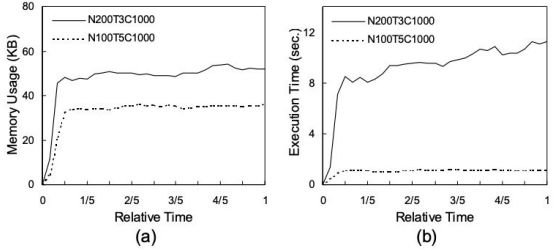
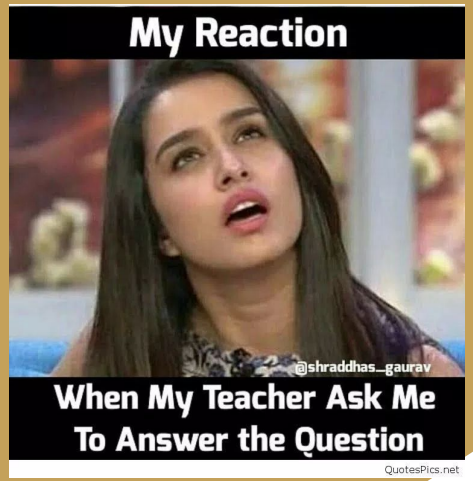


Figure 8: Required resources for synthetic datasets: (a) memory; (b) execution time

FTP-DS

- Two major features
  - One data scan for online statistics collection
  - Regression-based compact pattern representation
- Variable time intervals
- Trent detection effectively

What if There Are Missing Values In the Data Stream?



What Do You Mean Missing Values?

- What are some applications of data streams?
  - Network traffic
  - Call center records
  - Many sensors throughout a scattered area
- What happens if a few of the sensors malfunction and are unable to send any data? Ie. a few sensors get destroyed by weather conditions
- Most algorithms assume that we have standard operation, in that nothing has malfunctioned.

Why Is This a Problem?

- In order to use the linear regression model, we cannot have empty probability values in the data input.
- Can we just ignore the data point with missing values?
- We cannot ignore the data point with the missing values because we need to continuously predict the outcomes of data points as they come in. If we ignore them, we would have no output at certain points.
    - Even worse, if a period of time has missing variables, then we would have no output at all for that period of time!

How To Fix the Problem?

- Since we cannot ignore any data points that have missing values, this means we must use the method of imputation to patch missing values. (Fill in the blanks)
- The two most common and popular ways to input missing values in datasets are:
  1. Impute a single value
  2. Use a model to impute the value

Im·pute /imˈpyo͞ot/ *verb*
represent (something, especially something undesirable) as being done, caused, or possessed by someone; attribute.

Single Value Impute

Constant Value Impute (ie. a sample mean, a constant c, etc.)

+ Computationally very simple
- Ignores relationship between variables in the data, thus weakening variance and covariance estimates

Regressional Value Impute

+ Takes into account relationships between variables using regression models(uses the non-missing features as the input to find out the missing feature)
- Requires 2^r imputation models, where r is the number of input features. (Assumes binary features)
- is impractical and often computationally infeasible for streaming data applications

Model-Based Impute

- All very popular and also powerful, but very costly in computation
- Maximum Likelihood Imputation
  - \+ Chooses as estimate the value that maximizes the probability of observing non-missing values
  - \- Requires iterative calculations
- Multiple Imputation
  - \+ Averages across multiple imputed data sets, using randomized imputation method
  - \- Requires multiple imputation calculations, and then averaging them for each input
- In general, Model-Based inputs are accurate, but mostly for static datasets that are not streams, and want to reconstruct the data as accurately as possible

Verdict: Single Value Impute

- Because of how much costly computations are required to do model-based imputes, the best choice is to use single value imputes using a sample mean (this minimizes MSE)
    - This minimizes MSE and accounts for the problem of increased variance and covariance mentioned by (Allison 2001), simply by only updating the model when there is no missing values.
- True that It ignores relations between input variables, but we don't necessarily need to replicate the dataset exactly - we just need to be accurate enough by satisfying a specified Mean Square Error (MSE)

$$MSE = \frac{1}{n}\sum_{l=1}^{n}(o^{(l)} - y^{(l)})^2 = \frac{1}{n}(\mathbf{o} - \mathbf{y})^T(\mathbf{o} - \mathbf{y}),$$

Where y is the actual value, o is the predicted value, and n is the number of values in testing set

Predicting the Output

- The prediction vector is calculated using least squares

$$\hat{\beta}_{OLS} = \arg\min_{\beta} \left((\mathbf{y} - \mathbf{X}\beta)^T (\mathbf{y} - \mathbf{X}\beta) \right) = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y},$$

  where B is the prediction vector, y is the training output, and X is the training input.

- In tandem with the MSE equation, we get the following equations based on certain criteria, ie. whether or not each variable in the input vector is independent, or whether or not missing values appear at a fixed probability.

MSE Values Vary

Notation	Proposition	Missing values	Equal probabilities of missing	Standardized inputs and target
MSE_0	Eq. 4	No	–	–
$MSE_{\mathbf{p}}$	Prop. 1	Yes	No	No
$MSE^{\star}_{\mathbf{p}}$	Prop. 2	Yes	No	Yes
$MSE^{\star}_{p}$	Prop. 3	Yes	Yes	Yes

β is regression coefficients, C is the covariance matrix of input data, z is relation vector of input data and the target variable, D = diag(C), H = I−diag(p), Ir×r is the identity matrix, p=probability of missing value

Equation

$$E[MSEp] = \beta^T HD\beta + \beta^T H(C - D)H\beta - 2\beta^T Hz + Var0[y] + (\bar{x}\beta - \bar{y})^2$$

$$E[MSE^{*}\mathbf{p}] = \beta^T H\beta + \beta^T H(C - I)H\beta - 2\beta^T Hz + 1$$

$$E[MSE^{*}p] = (1 - p)E[MSE0] + p - p(1 - p)\beta^T (C - I)\beta.$$

Memory Needed

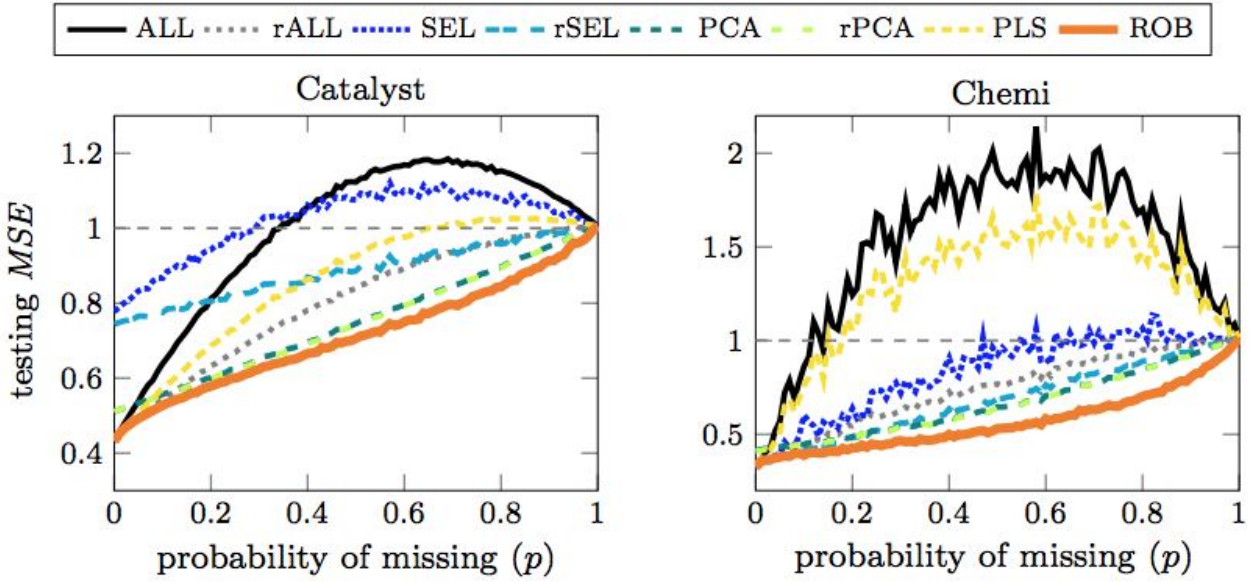
	Estimate	Size	Memory
C	Covariance matrix	$r \times r$	r^2
z	Input-target relation	$r \times 1$	r
p	Probabilities of missing	$r \times 1$	r
	Mean and SD of inputs	$2 \times r \times 1$	$2r$
	Mean and SD of target	$2 \times 1 \times 1$	2
Total memory			$r^2 + 4r + 2$

The Robust Algorithm

Data: stream of observations $\mathbf{x}$,
stream of true target values y,
forgetting factors $\alpha \in (0, 1)$, $\gamma \in (0, 1)$
Result: stream of predictions o

Initialization: $\mathbf{C} \leftarrow \mathbf{I}$, $\mathbf{z} \leftarrow (1, \ldots, 1)^T$, $\mathbf{p} \leftarrow (0, \ldots, 0)^T$, $\beta \leftarrow (0, \ldots, 0)^T$;
for $t \leftarrow 1$ **to** ∞ **do**
    new observation $\mathbf{x}$ arrives, predict $o = \mathbf{x}\beta$;
    true target value y arrives ;
    if *no missing values in* $\mathbf{x}$ **then**
        update covariance $\mathbf{C} \leftarrow \alpha \mathbf{x}^T \mathbf{x} + (1 - \alpha)\mathbf{C}$;
        update relation $\mathbf{z} \leftarrow \alpha \mathbf{x}^T y + (1 - \alpha)\mathbf{z}$;
        update missing value estimate $\mathbf{p} \leftarrow (1 - \gamma)\mathbf{p}$;
    end
    else
        record missing value indicator $\mathbf{m}$, where $m_i = 1$ if i^{th} value in $\mathbf{x}$ is missing, and 0 otherwise ;
        update missing value estimates $\mathbf{p} \leftarrow \gamma \mathbf{m} + (1 - \gamma)\mathbf{p}$;
    end
    update model $\beta = (\mathbf{C} - diag(\mathbf{p})(\mathbf{C} - \mathbf{I}))^{-1}\, \mathbf{z}$;
end

Testing the Imputation and Prediction



Winner!

Table 5 Empirical $-\beta^T(\mathbf{C}-\mathbf{I})\beta$, the smaller the better

	ALL	rALL	SEL	rSEL	PCA	rPCA	PLS	ROB
Catalyst	1.6	0.8	0.5	0.1	0.0	−0.1	0.9	**−0.4**
Chemi	6.4	3.1	0.4	0.2	−0.2	−0.2	5.9	**−0.8**
Concrete	0.6	0.3	0.0	0.0	**−0.0**	**−0.0**	0.5	0.1
CPU	−0.1	−0.2	−0.2	−0.3	−0.3	−0.3	−0.1	**−0.8**
Gaussian	0.6	−0.1	−0.1	−0.2	−0.3	−0.3	−0.0	**−0.6**
Protein	−0.3	−0.4	−0.2	−0.2	−0.7	−0.7	−0.6	**−2.2**
Wine	0.2	0.1	0.1	0.1	**−0.1**	**−0.1**	0.1	**−0.1**
Year	−0.3	−0.3	−0.3	−0.3	−0.4	−0.4	−0.3	**−1.4**

The best result on each dataset is in bold

NO QUESTIONS ALLOWED

Just kidding.

References

1. *A Regression-Based Temporal Pattern Mining Scheme for Data Streams*, by Wei-Guang Teng, Ming-Syan Chen, Philip S. Yu, in the International Conference on Very Large Data Bases (VLDB) 2003.
2. Allison, P. (2001). Missing data. Thousand Oaks: Sage.
3. Žliobaitė, I. & Hollmén, J. “Optimizing regression models for data streams with missing values.” Mach Learn (2015) 99: 47. https://doi.org/10.1007/s10994-014-5450-3