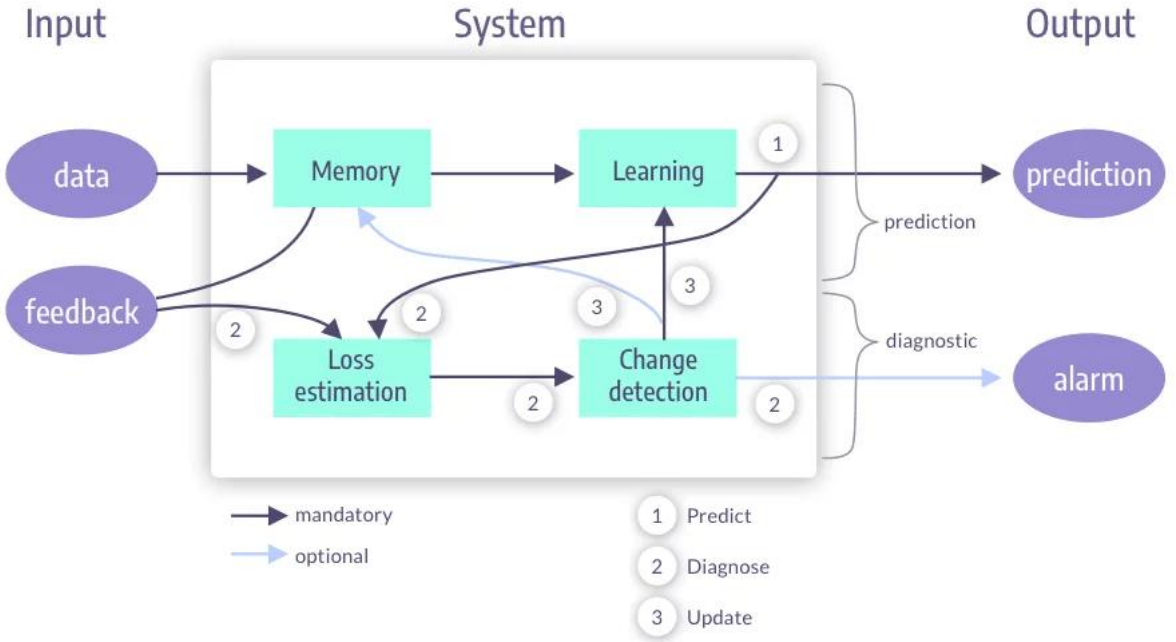
Concept Drift

Drift Aware Systems

Main Component

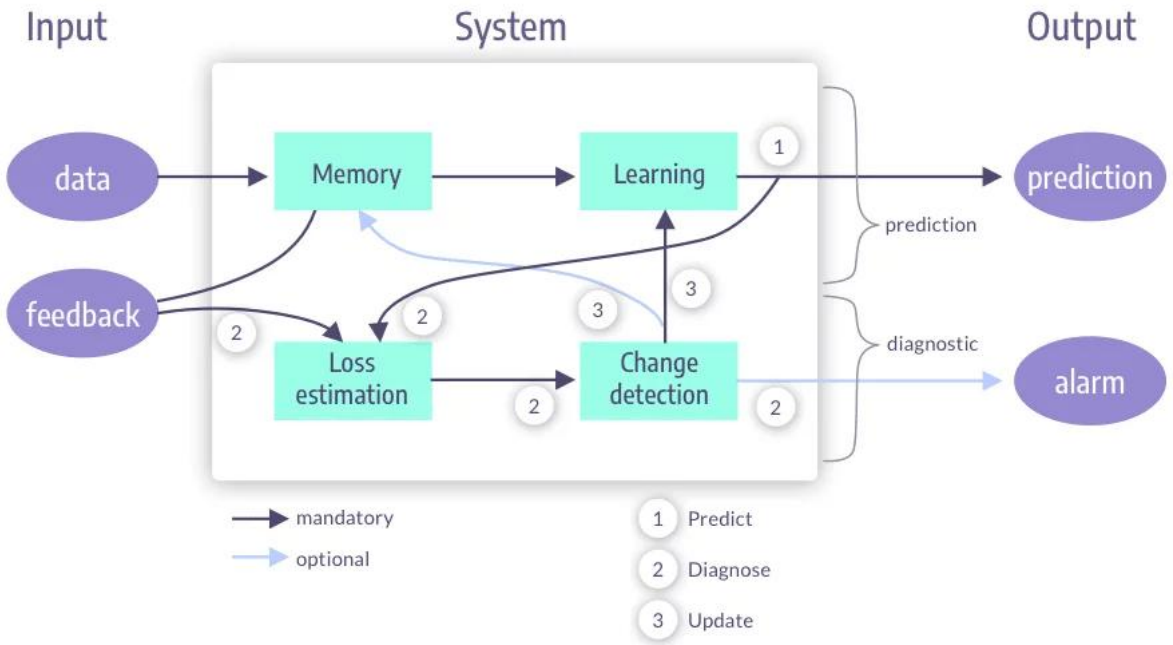
- Change detection
- Memory
- Learning
- Loss estimation: Bad Prediction



Drift Aware Systems

System Components : Memory

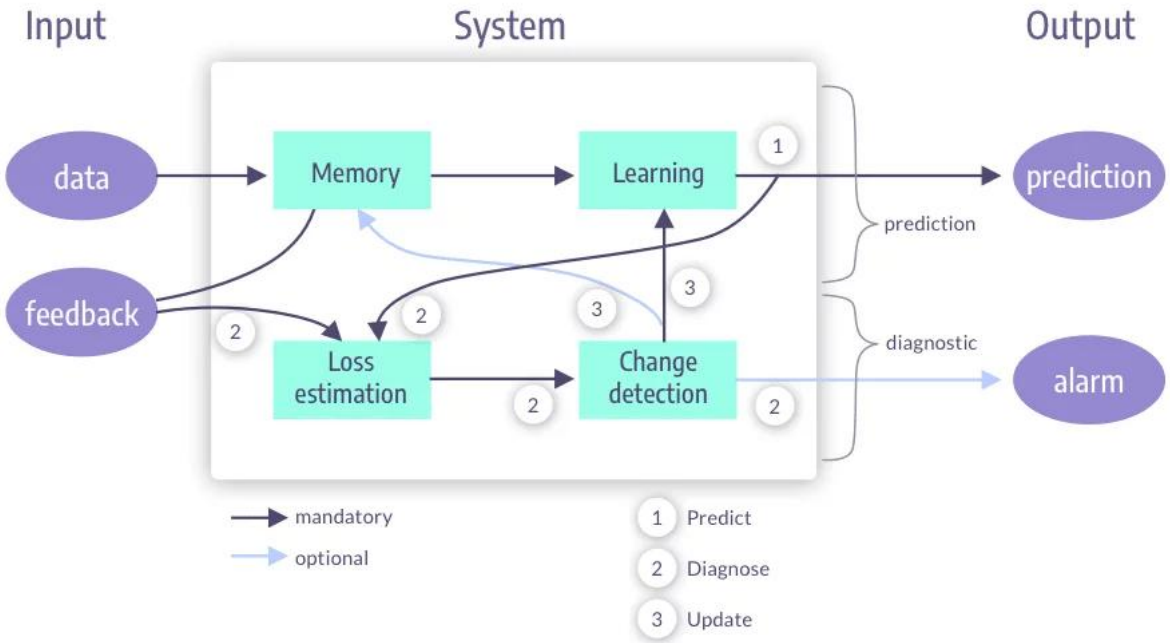
- Stores incoming data and past information.
- Supplies information to the Learning module for training models.
- Can also interact with feedback to improve stored knowledge.



Drift Aware Systems

System Components : Learning

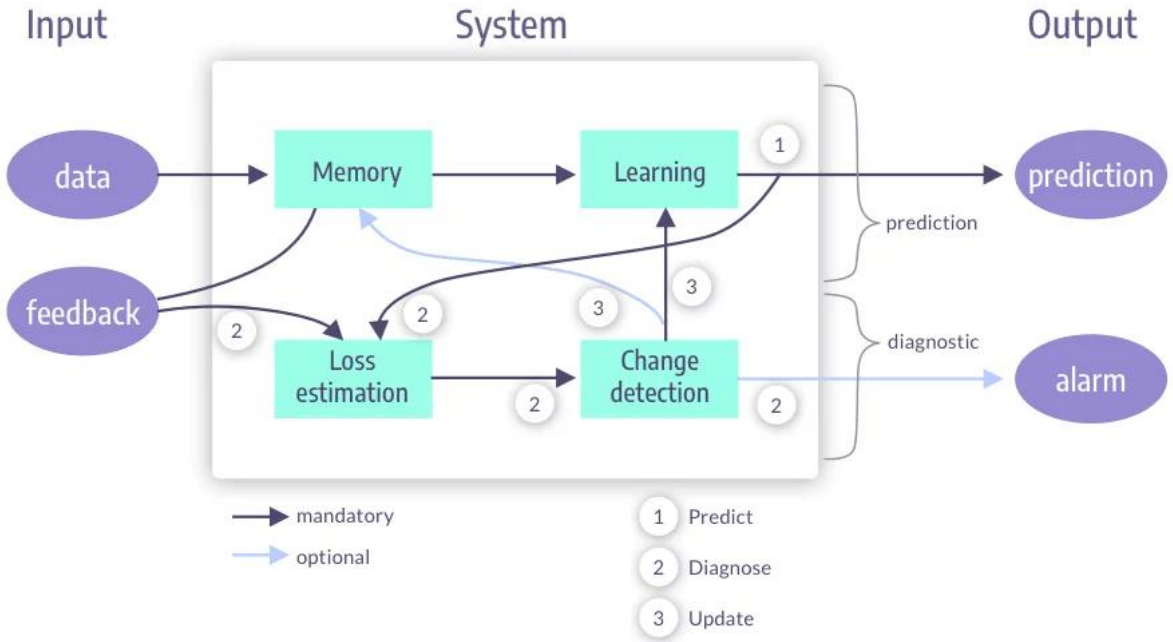
- Core module that builds predictive models using data and memory.
- Responsible for generating the prediction output.
- Continuously updated based on feedback and change detection.



Drift Aware Systems

System Components : Loss Estimation

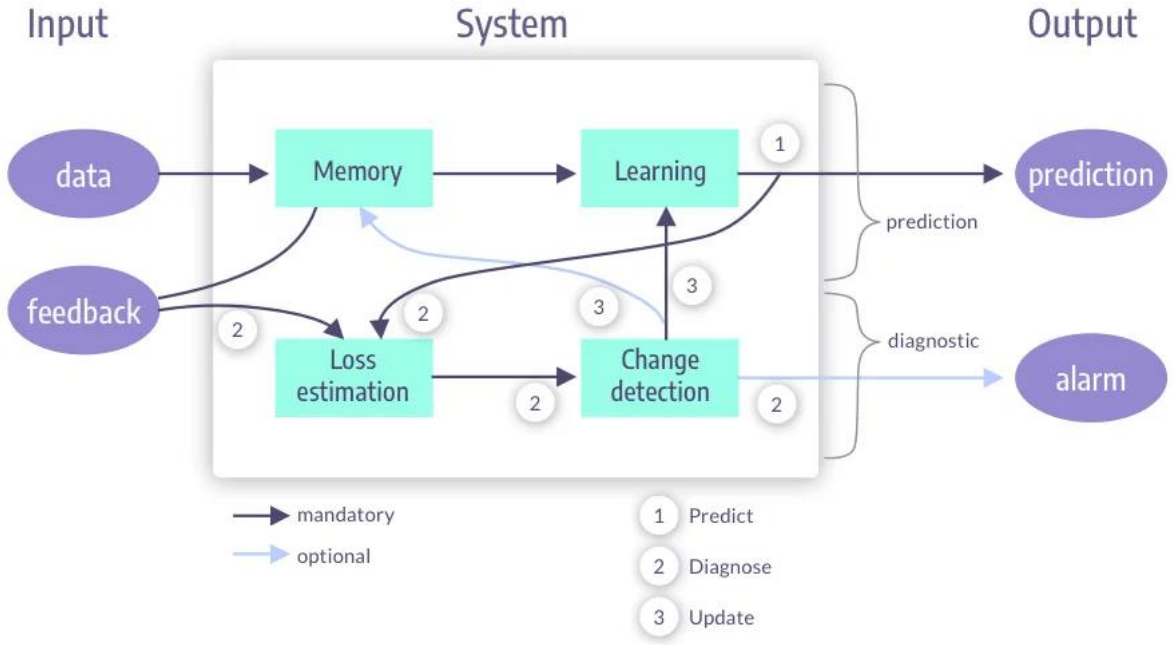
- Calculates errors or mismatches between predicted outcomes and feedback.
- Provides diagnostics about model performance.
- Feeds signals to Change Detection to identify when the system should adapt.



Drift Aware Systems

System Components : Change Detection

- Monitors system behavior and loss trends.
- Detects concept drift (changes in data distribution or model validity).
- Sends alarms if performance degrades or data characteristics change.
- Can trigger the Learning module to update the model.

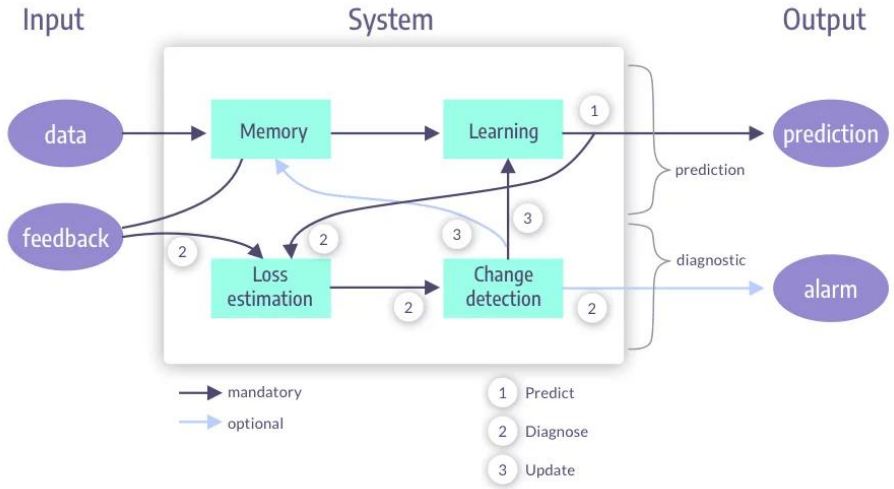


Drift Aware Systems

Connections (Mandatory vs. Optional)

- Mandatory paths (dark arrows): Core system flow,
  - e.g.,Data → Memory → Learning → Prediction
  - Feedback → Loss Estimation → Change Detection
- Optional paths (light arrows): Extensions for adaptive improvement,
  - e.g.,Change Detection → Alarm (diagnostic output)
  - Feedback updating Memory directly
  - Loss Estimation influencing Learning indirectly

This diagram represents a closed-loop adaptive learning system. Data flows into the system to generate predictions, feedback helps in diagnosing performance, and when changes or drifts are detected, the system updates itself. If something critical happens, it raises an alarm.



Criteria for evaluation of the ability of the algorithm to handle concept drift:

Criteria for evaluation of the ability of the algorithm to handle concept drift:

Predictive Performance

Adaptation to Drift

Resource Efficiency

Stability vs. Plasticity

Diagnostic Capability

Drift Type Robustness

Criteria for evaluation of the ability of the algorithm to handle concept drift:

- **Predictive Performance**
  - Accuracy / Error Rate
  - The most basic measure of predictive success.
  - However, in streaming data with drift, accuracy must be monitored over time, since a sudden drop often indicates drift.

Criteria for evaluation of the ability of the algorithm to handle concept drift:

- **Predictive Performance**
  - Especially important in imbalanced datasets (e.g., fraud detection, medical diagnosis) where overall accuracy is misleading.
  - Precision → proportion of positive predictions that are correct.

$$\text{Precision} = \frac{TP}{TP + FP}$$

  - Recall → ability to capture all true positives.

$$\text{Recall} = \frac{TP}{TP + FN}$$

  - F1 Score → harmonic mean of precision and recall. $F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$
  - AUC → **(Area under ROC Curve)**: Measures trade-off between True Positive Rate (TPR) and False Positive Rate (FPR).

Criteria for evaluation of the ability of the algorithm to handle concept drift:

- **Predictive Performance**
  - **Prequential Accuracy**
  - A streaming evaluation approach: test each incoming sample with the current model, then use it for training.
  - This reflects real-time adaptability better than static train/test splits.

$$\text{PreqAcc}(t) = \frac{1}{t}\sum_{i=1}^{t} I(y_i = \hat{y}_i)$$

where each instance is first tested ($\hat{y}_i$), then used for training.

Criteria for evaluation of the ability of the algorithm to handle concept drift:

- **Adaptation to Drift**
  - **Detection Delay:** Time (or number of samples) between drift occurrence and detection. Shorter delay means the model adapts quickly to new realities.

$$D = t_{detect} - t_{drift}$$

  - **Recovery Time:** Number of samples needed to regain stable performance after drift is detected. Important for minimizing business/operational risk.

$$R = t_{stable} - t_{detect}$$

  - **Adaptivity :** Ability of the algorithm to self-adjust without requiring full retraining. Strong adaptivity means handling changes incrementally and autonomously.

Criteria for evaluation of the ability of the algorithm to handle concept drift:

Stability vs. Plasticity

Stability:

- Maintaining knowledge of past stable concepts (avoiding catastrophic forgetting). Important in recurring drifts where old patterns reappear.

$$S = \frac{\text{Performance on old concept}}{\text{Initial performance}}$$

Plasticity

- Flexibility to incorporate new patterns rapidly when drift occurs.
- High plasticity ensures relevance to current data distribution.

Balance:

- The key challenge:
- too much stability → poor adaptation,
- too much plasticity → forgetting.
- Effective algorithms strike a balance (e.g., ensembles, memory-based learning).

$$P = \frac{\text{Performance after drift}}{\text{Final performance}}$$

Catastrophic forgetting

- When a model trained on task A is then trained on task B (without access to A's data), its accuracy on A collapses.
- In streaming terms: after drift or domain shift, updates for the new regime erase the old regime.

Criteria for evaluation of the ability of the algorithm to handle concept drift:

Resource Efficiency

- **Memory Usage:**
  - Algorithms must handle continuous data without storing everything.
  - Efficient memory usage is crucial in high-velocity streams.
- **Computational Complexity**
  - Drift adaptation should not require heavy retraining.
  - Efficient incremental updates allow real-time performance.
- **Scalability**
  - Must scale with high-speed, high-volume streams.
  - In practical deployments (finance, IoT, cybersecurity), scalability is a make-or-break factor.

Criteria for evaluation of the ability of the algorithm to handle concept drift:

Drift Type Robustness

- **Sudden Drift:** Abrupt and radical changes in the data distribution (e.g., regulatory changes in finance). The algorithm should react instantly.
- **Gradual Drift:** Transition happens slowly (e.g., changing user preferences). The model should smoothly adapt without overreacting.
- **Incremental Drift:** Small continuous shifts in data distribution (e.g., sensor degradation). Requires fine-tuned incremental updating.
- **Recurring Drift:** Past concepts reappear cyclically (e.g., seasonal shopping patterns). Algorithms with memory-based mechanisms can reuse old models instead of relearning.

Criteria for evaluation of the ability of the algorithm to handle concept drift:

Diagnostic Capability

- **Alarm Accuracy:**
  - Ability to correctly issue alarms when drift occurs (low false positives and false negatives). False alarms waste resources; missed alarms lead to degraded predictions.
- **Interpretability:**
  - Explaining why a drift occurred and how the model responded (e.g., feature-level analysis). Interpretability builds trust in critical applications.
- **Loss Monitoring:**
  - Continuous monitoring of prediction error/loss provides early-warning signals before major performance collapse. This allows proactive retraining or model switching.

Criteria for evaluation of the ability of the algorithm to handle concept drift:

A strong drift-handling algorithm must:

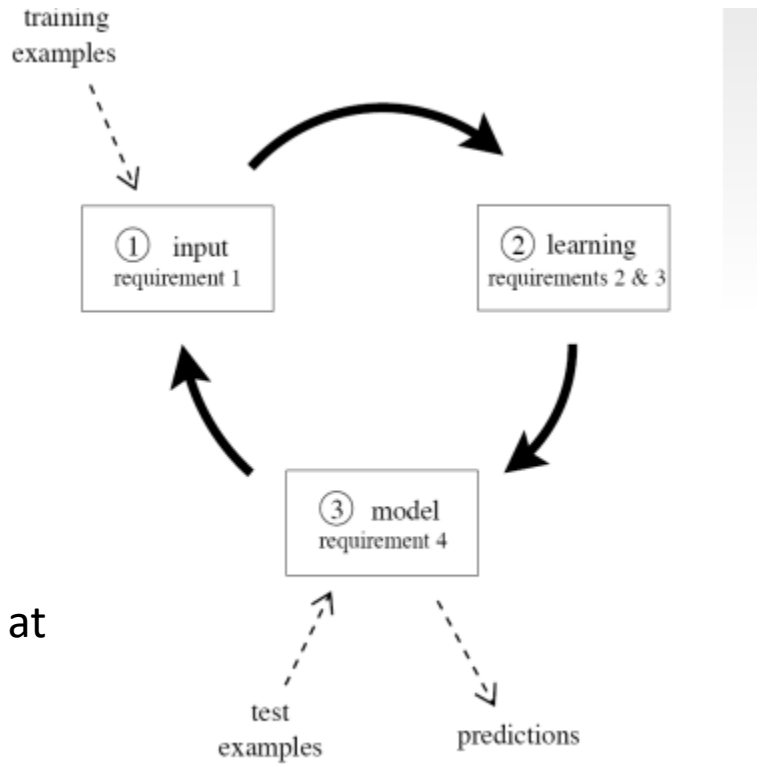
- Maintain high predictive performance,
- Adapt quickly with minimal delay,
- Balance stability and plasticity,
- Be resource-efficient in real-time settings,
- Handle multiple drift types, and
- Provide diagnostic insights for transparency and trust.

Concept Drift Handling Methods

- Change detection/adaptation methods:
  - On the Basis of
    - Memory requirement,
    - forgetting mechanism
    - information they use for detecting/adapting to drift,
    - model management (number of base learners)

Data stream classification cycle

1. Process an example at a time, and inspect it only once (at most)
2. Use a limited amount of memory
3. Work in a limited amount of time
4. Be ready to predict at any point



Dimensions of Learning

- **Space -** the available memory is fixed
- **Learning Time -** process incoming examples at the rate they arrive
- **Generalization Power -** how effective the model is at capturing the true underlying concept

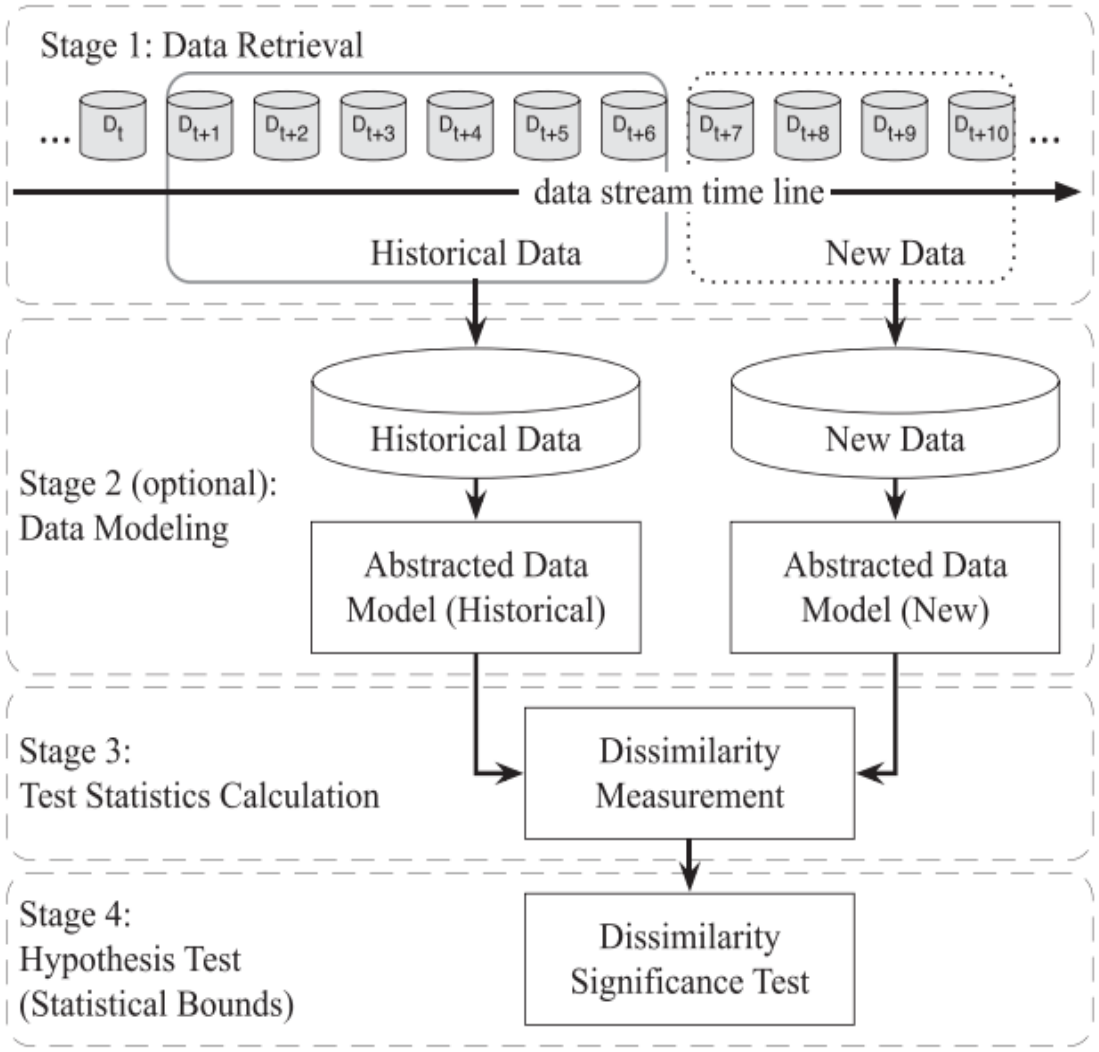
How to detect the Concept Drift

Concept Drift Detection

Drift detection refers to the techniques and mechanisms that characterize and quantify concept drift via identifying change points or change time intervals.

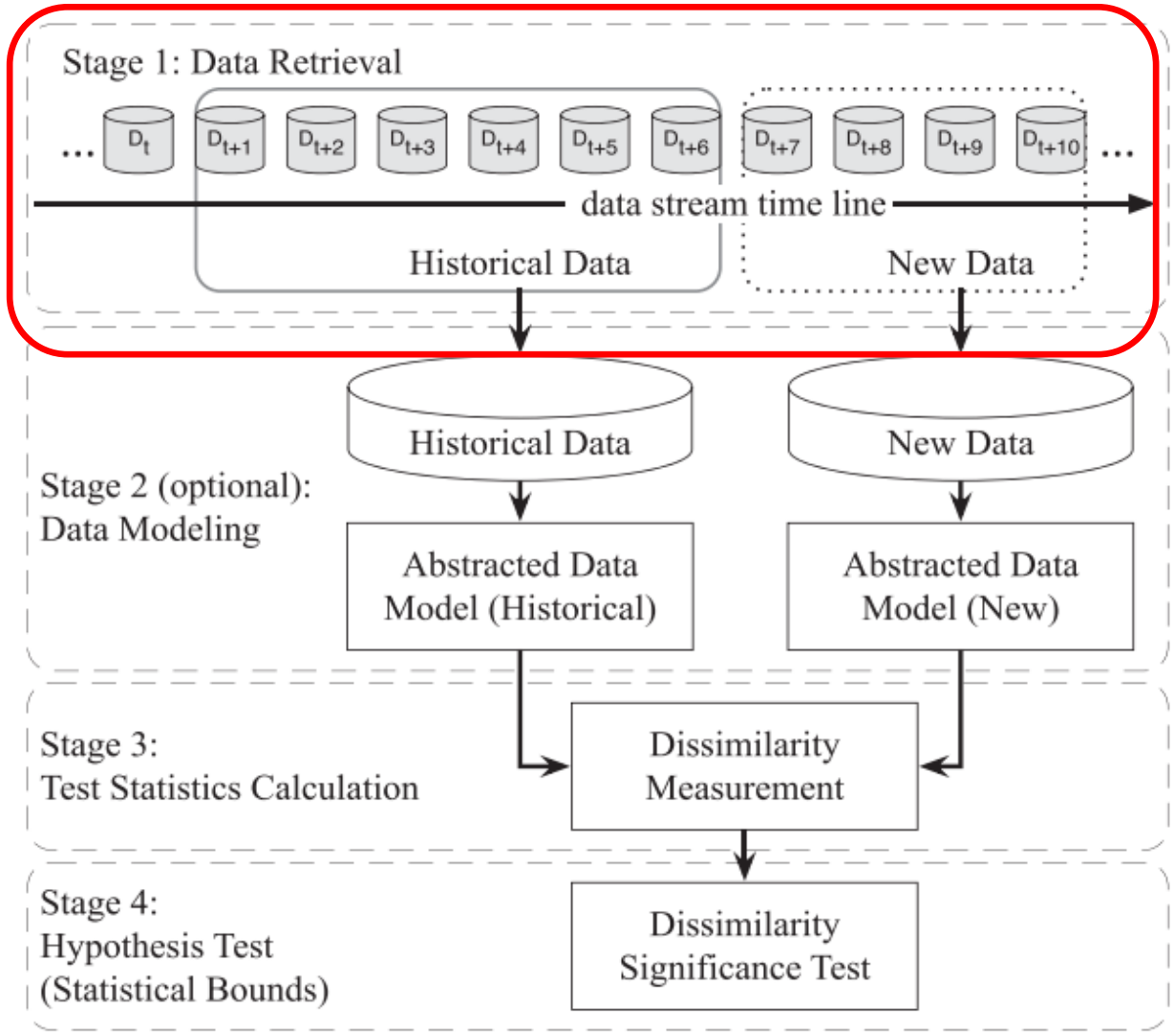
A general framework for

drift detection contains four stages, as shown in Figure



Concept Drift Detection

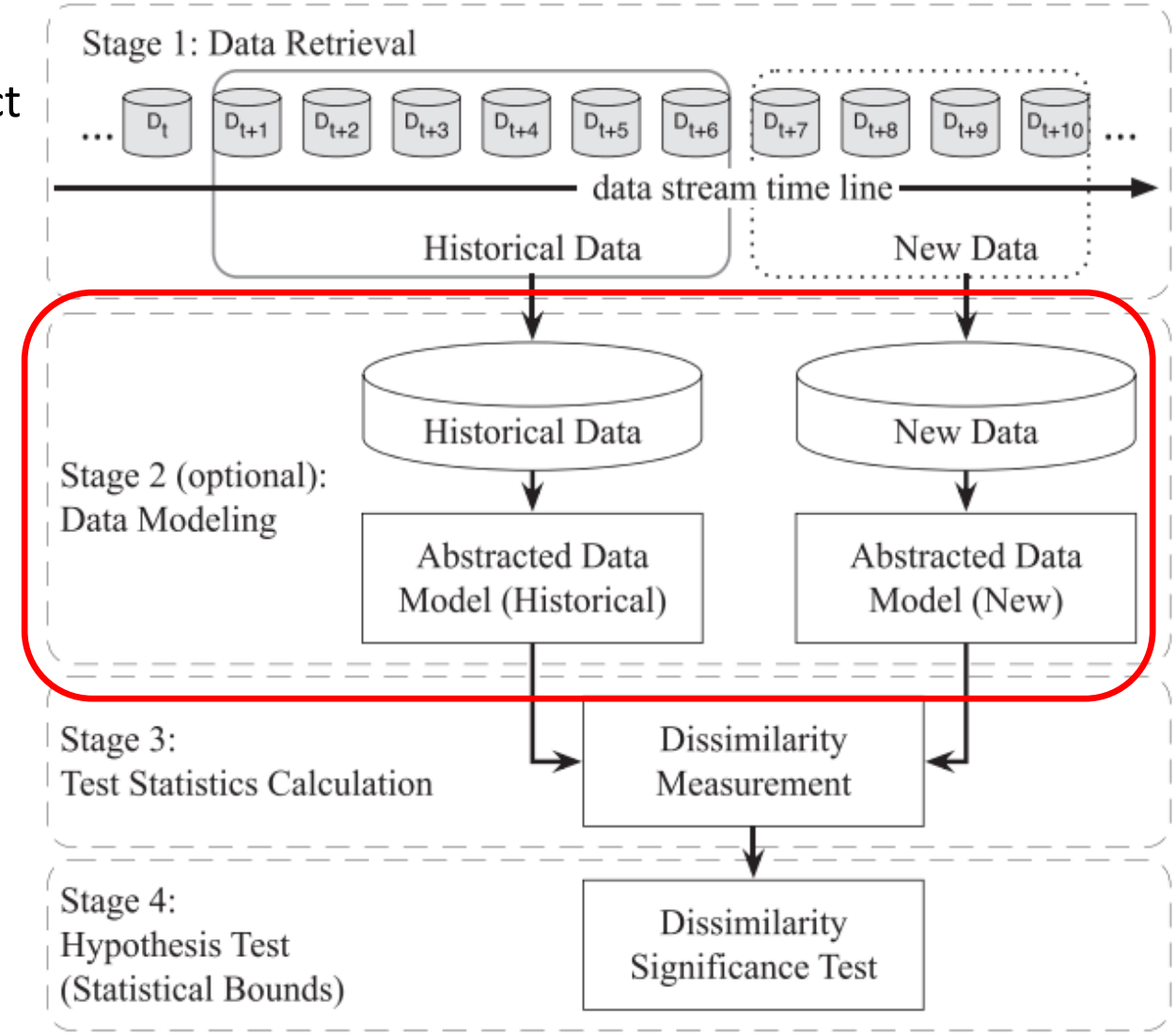
- Stage 1 (Data Retrieval) aims to retrieve data chunks from data streams.
- Since a single data instance cannot carry enough information to infer the overall distribution, knowing how to organize data chunks to form a meaningful pattern or knowledge is important in data stream analysis tasks.



Concept Drift Detection

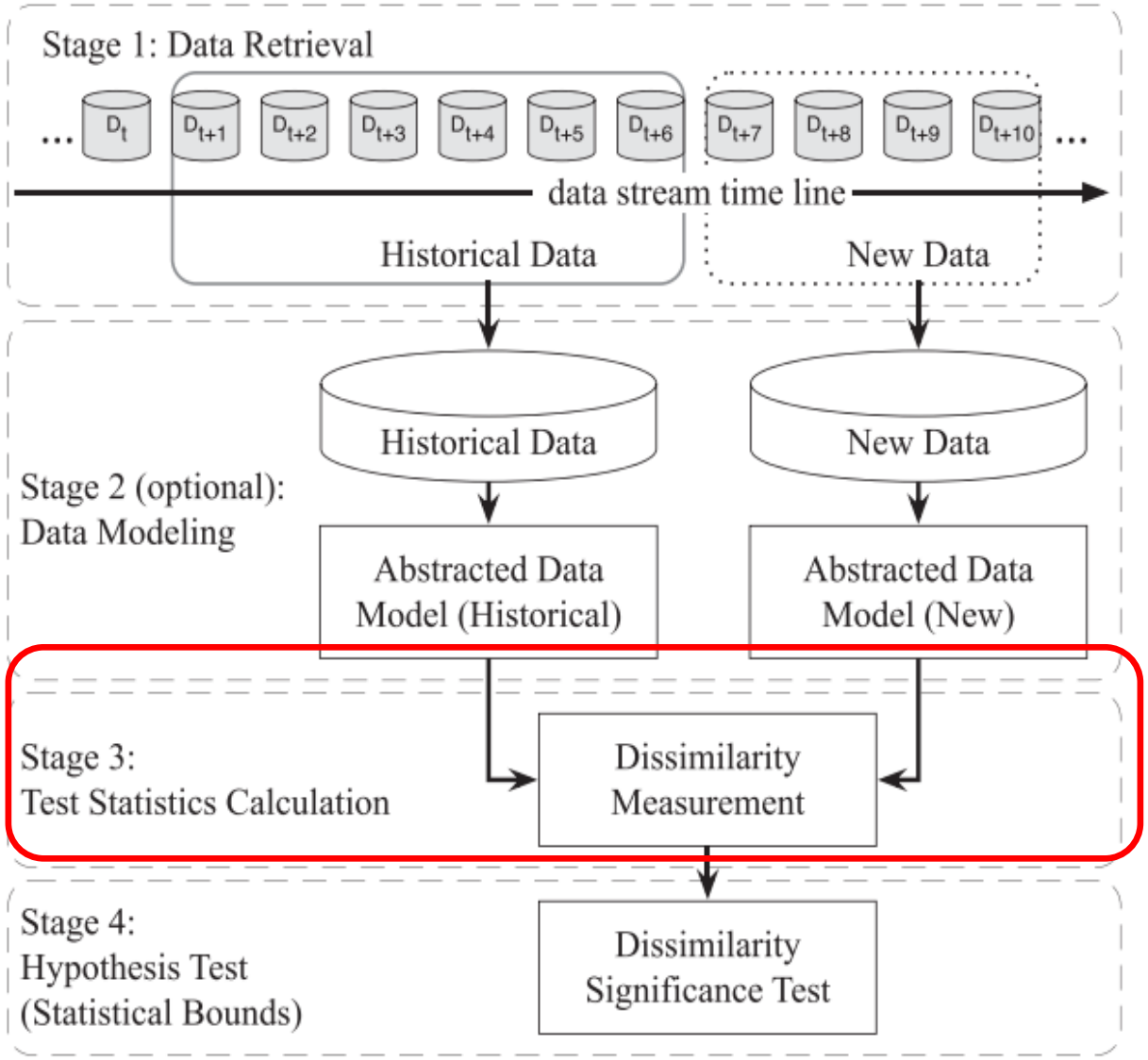
- Stage 2 (Data Modeling) aims to abstract the retrieved data and extract the key features containing sensitive information, that is, the features of the data that most impact a system if they drift.

- This stage is optional, because it mainly concerns dimensionality reduction, or sample size reduction, to meet storage and online speed requirements.



Concept Drift Detection

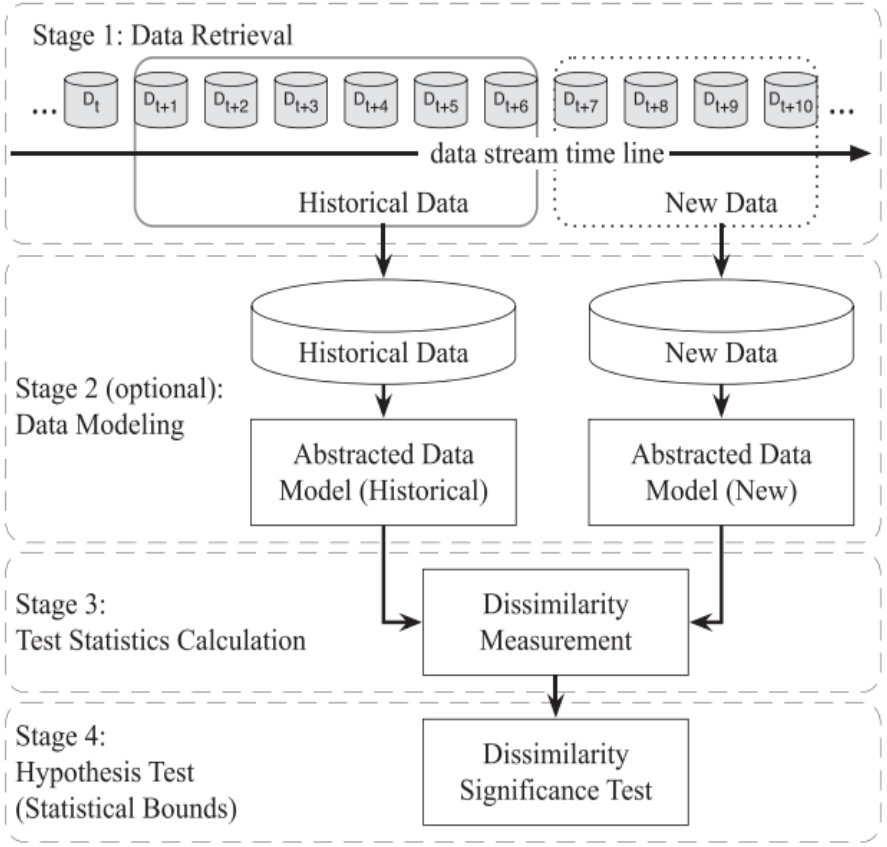
- Stage 3 (Test Statistics Calculation) is measurement of dissimilarity, or distance estimation. It quantifies the severity of the drift and forms test statistics for the hypothesis test.
- It is considered to be the most challenging aspect of concept drift detection.
- The problem of how to define an accurate and robust dissimilarity measurement is still an open question.



Hypothesis to identify Drifts

- **Null hypothesis: This hypothesis proposes that the means of two distinct Data Streams are equal**. When performing statistical tests, the goal becomes to either reject the null hypothesis or prove it correct.

- **Alternative hypothesis: Alternative hypotheses propose that there is a significant difference between Data Streams** and the variations between the samples result in unequal means.
- If you arrive at an alternative hypothesis during statistical analysis, it can indicate a rejection of the null hypothesis.



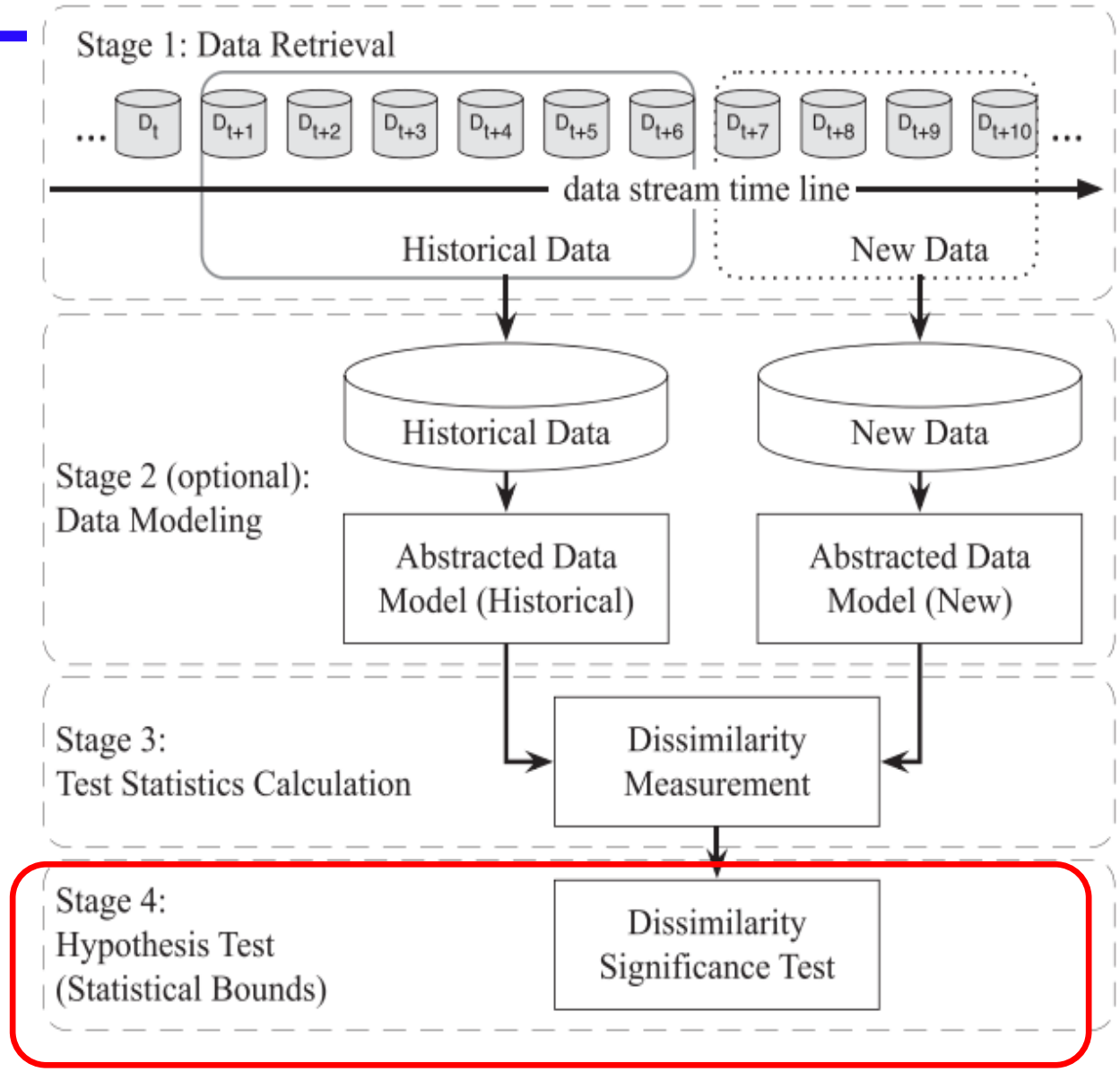
Test statistics for the hypothesis test

Test Statistic	Formula	Finding
T-value for 1-sample t-test	$t = \frac{\bar{x} - \mu_0}{s/\sqrt{n}}$	Take the sample mean, subtract the hypothesized mean, and divide by the standard error of the mean.
T-value for 2-sample t-test	$t = \frac{\bar{x}_1 - \bar{x}_2}{s_p\sqrt{\frac{1}{n_1} + \frac{1}{n_2}}}$	Take one sample mean, subtract the other, and divide by the pooled standard deviation.
F-value for F-tests and ANOVA	$F = \frac{s_1^2}{s_2^2}$	Calculate the ratio of two variances.
Chi-squared value (χ^2) for a Chi-squared test	$\chi^2 = \sum \frac{(Observed - Expected)^2}{Expected}$	Sum the squared differences between observed and expected values divided by the expected values.

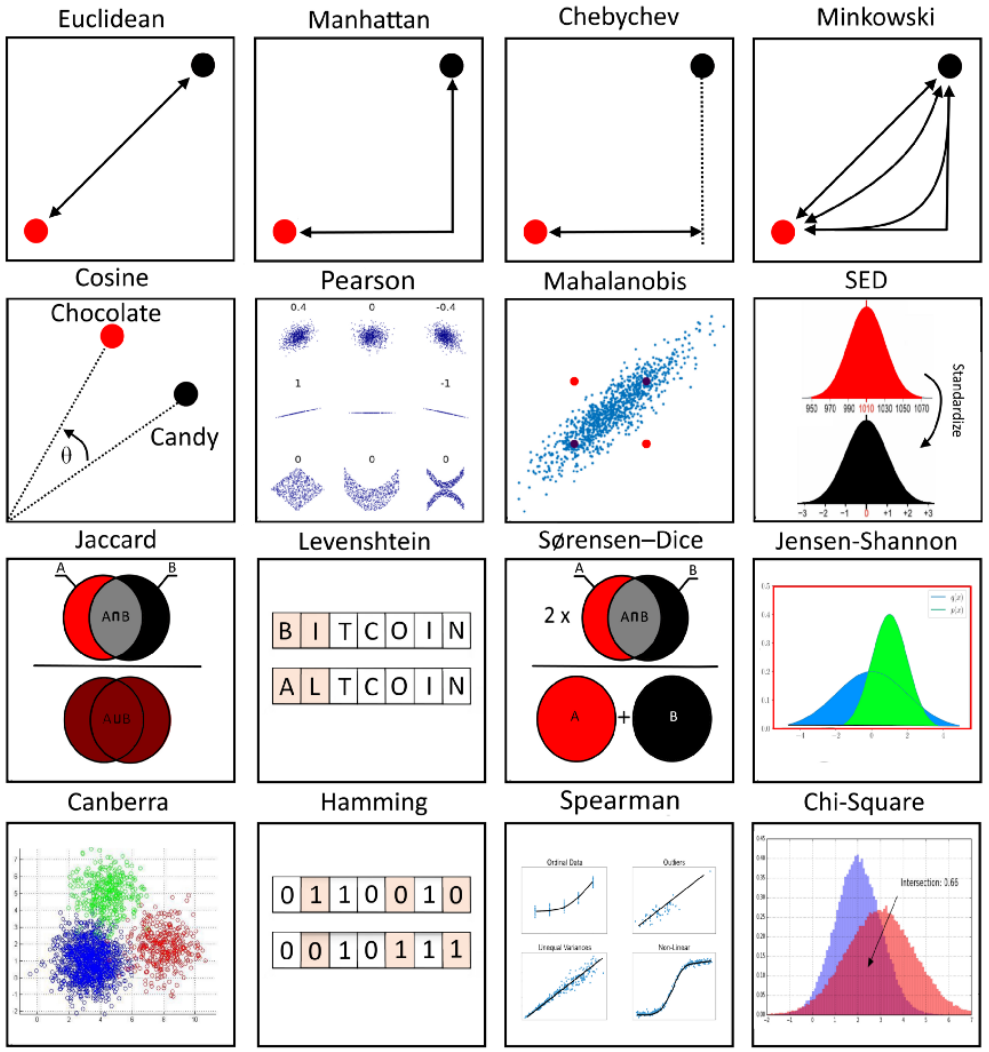
Concept Drift Detection

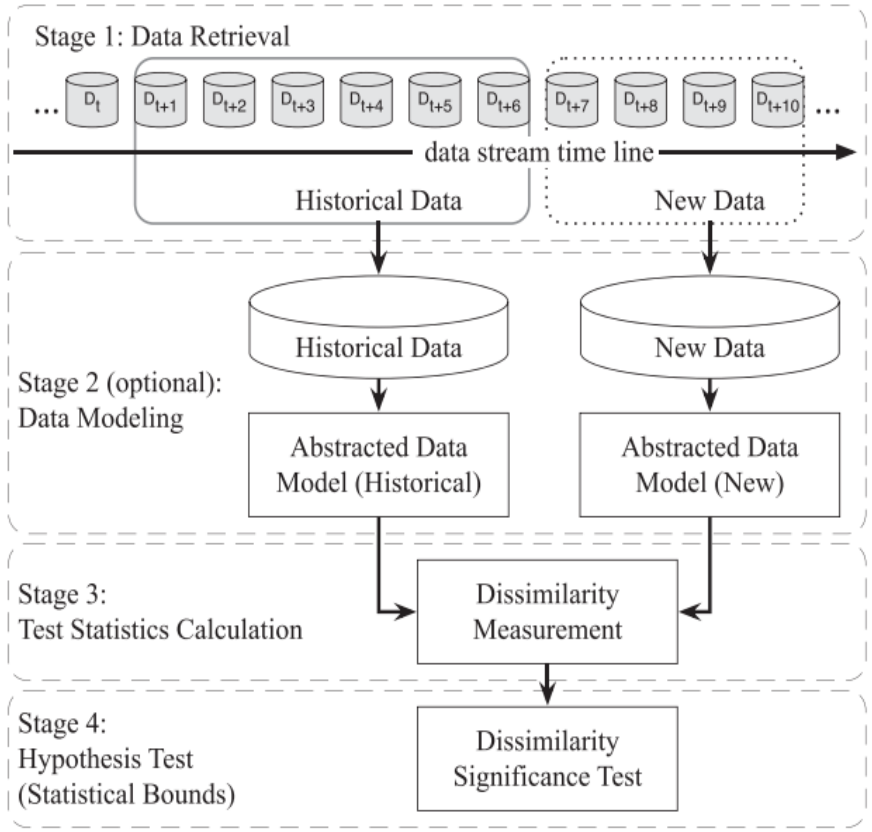
- Stage 4 (Hypothesis Test) uses a specific hypothesis test to evaluate the statistical significance of the change observed in Stage 3.

- They are used to determine drift detection accuracy by proving the statistical bounds of the test statistics proposed in Stage 3.



Types of dissimilarity measures





Drift Detection approaches

	Explicit drift detection (Supervised)	Implicit drift detection (Unsupervised)
1	Sequential analysis	Novelty detection/ clustering methods
2	Statistical Process Control	Multivariate distribution monitoring
3	Window based distribution monitoring	Model dependent monitoring

Explicit concept drift detection methodologies

- **Sequential analysis methodologies-**
  - Continuously monitor the sequence of performance metrics , such as
    - Accuracy
    - F-measure
    - precision and recall;
  - to signal a change, in the event of a significant drop in these values.
- Methodologies comes under the Sequential analysis-
  - CUSUM (Cumulative Sum approach)
  - PHT (PageHinckley Test)

Sequential analysis methodologies

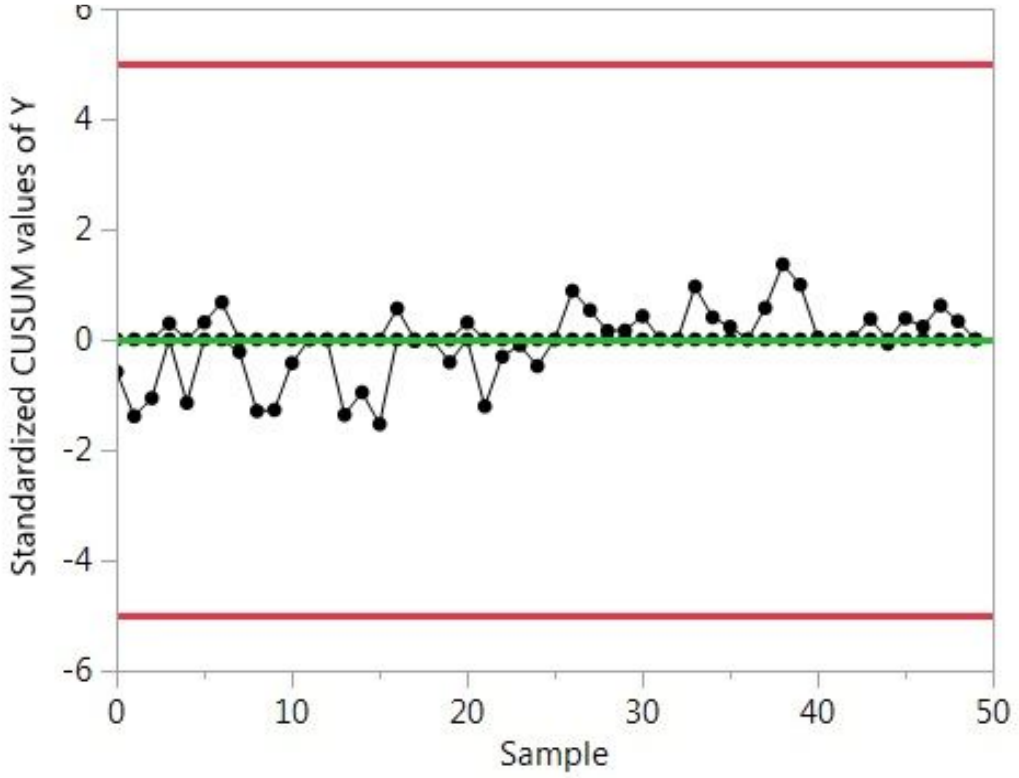
- CUSUM(Cumulative Sum approach)- This approach signals an alarm when the mean of the sequence significantly deviates from 0.
- The CUSUM test monitors a metric M, at time t, on an incoming sample's performance εt , using parameters v for acceptable deviation and θ for the change threshold as given in the equation.

$$M_0 = 0; \quad M_t = max(0, M_{t-1} + \epsilon_t - v)$$
$$if \quad M_t > \theta \quad then \quad 'alarm' \quad and \quad M_t = 0$$

- Max function is used to test changes in positive direction. For reverse effect a min function can be used.
- Memory-less and can be used incrementally.

CUSUM(Cumulative Sum approach)-

- X-axis (Sample): the observation index (1…50).
- Y-axis: standardized CUSUM of Y.
- Each point is the running sum of standardized deviations from the target mean (so units are in σ's).
- Green dotted line at 0: the target (in-control) level.
- Red horizontal lines (~±5): the decision limits $\pm h$.
- Crossing one of these is a drift/out-of-control signal.



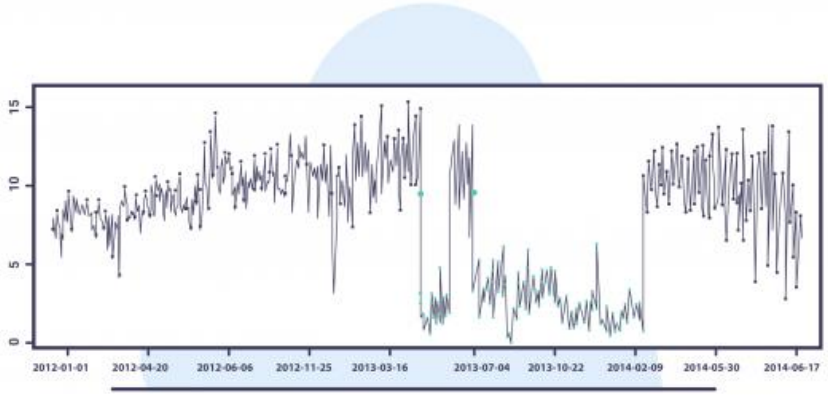
- When the curve slants upward, recent observations have, on average, been above the target → possible positive shift.
- When it slants downward, they've been below the target → possible negative shift.

Sequential analysis methodologies

- Page Hinckley Test (PHT) is a variant of CUSUM approach.
- **PHT monitor the metric as an accumulated difference between its mean and current values, as shown below.**

$$M_0 = 0; \quad M_t = M_{t-1} + (\epsilon_t - v); \quad M_{Ref} = \min(V)$$
$$if \quad M_t - M_{Ref} > \theta \quad then \quad 'alarm' \quad and \quad M_t = 0$$

  - Where, M_0 is the initial metric at time t = 0.
  - M_t is the current metric computed far (M_t−1) and the sample's performance at time t = εt and v denotes acceptable deviation from mean and θ is the change detection threshold.



Page-Hinkley

Statistical Process Control based methodologies-

- Monitor the online trace of error rates, and detects deviations based on ideas taken from control charts.
- A significantly increased error rate violates the model and as such is assumed to be a result of concept drift.
- Methodologies under this category are-

  - DDM (Drift Detection Method)

  - EDDM (Early Drift Detection Method)

  - STEPD (Statistical Test of Equal Proportion Distribution)

  - EWMA (Exponentially Weighted Moving Average)

DDM (Drift Detection Method)

Suppose the model produces this error indicator over time (1 = wrong):

$$\underbrace{0\,0\,0\,1}_{i_1=4}\,0\,0\,0\,0\,\underbrace{1}_{i_2=9}\,0\,0\,0\,\underbrace{1}_{i_3=13}\,0\,0\,\underbrace{1}_{i_4=16}\,\ldots$$

- We look at the **error stream** $e_1, e_2, \ldots$ where

$$e_t = \begin{cases} 1 & \text{if the model's prediction at time } t \text{ is wrong} \\ 0 & \text{if it is correct.} \end{cases}$$

- Because each e_t is either 0 or 1, we can model it as a **Bernoulli** variable.

- The **running error rate** (sample mean) up to time t is

$$p_t = \frac{1}{t}\sum_{i=1}^{t} e_i.$$

- Under a Bernoulli model, the **standard deviation of the sample mean** is

$$s_t = \sqrt{\frac{p_t(1-p_t)}{t}}.$$

Drift Detection Method (DDM)

- As data arrives, track the best (lowest) pair seen so far:
  - $p_{\min}$: lowest error rate observed historically,
  - $s_{\min}$: the corresponding standard deviation at that time.

"Warning if $p_t + s_t \geq p_{\min} + 2\, s_{\min}$"

Drift if $p_t + s_t \geq p_{\min} + 3\, s_{\min}$"

Drift Detection Method (DDM)

Pros:

- DDM shows good performance when detecting gradual changes (if they are not very slow) and abrupt changes (incremental and sudden drifts).
- **Simple & fast**: you only update two running numbers ptp_tpt and sts_tst; no heavy statistics.

Cons:

- DDM has difficulties detecting drift when the change is slowly gradual.
- It is possible that many samples are stored for a long time, before the drift level is activated and there is the risk of overflowing the sample storage.

Statistical Process Control based methodologies

- EDDM(Early Drift Detection Methodology)
  - An extension of DDM, and was made suitable for slow moving gradual drifts, where DDM previously failed.
  - EDDM monitors the number of samples between two classification errors, as a metric to be tracked online for drift detection.
  - Based on the model, it was assumed that, in stationary environments, the distance (in number of samples) between two subsequent errors would increase.
  - A violation of these condition was seen to be indicative of drift.

EDDM(Early Drift Detection Methodology)

- If the model is doing well, ERRORS are far apart (long distances).
- As the concept drifts, ERRORS get closer together (short distances).
- EDDM tracks the mean and spread of those inter-error distances and compares them to the best (largest) regime seen so far.

EDDM(Early Drift Detection Methodology)

Let $e_t \in \{0, 1\}$ be the error indicator at time t (1 if wrong, 0 if correct).

Let $i_1 < i_2 < \cdots < i_j$ be the **time indices where errors occurred** (i.e., $e_{i_k} = 1$).

- **Inter-error distance (at the j-th error):**

$$d_j = i_j - i_{j-1} \quad (j \geq 2)$$

- **Running mean and std** of the distances after observing j errors:

$$\mu_d(j) = \frac{1}{j-1} \sum_{k=2}^{j} d_k, \qquad \sigma_d(j) = \sqrt{\frac{1}{j-2} \sum_{k=2}^{j} \big(d_k - \mu_d(j)\big)^2}$$

Suppose the model produces this error indicator over time (1 = wrong):

$$\underbrace{0\,0\,0\,1}_{i_1=4}\,0\,0\,0\,0\,\underbrace{1}_{i_2=9}\,0\,0\,0\,\underbrace{1}_{i_3=13}\,0\,0\,\underbrace{1}_{i_4=16}\,\ldots$$

Distances:

- $d_2 = i_2 - i_1 = 9 - 4 = 5$
- $d_3 = 13 - 9 = 4$
- $d_4 = 16 - 13 = 3$ (errors getting closer)

EDDM(Early Drift Detection Methodology)

- **Score used for detection (with a "two-sigma" cushion):**

$$p_j = \mu_d(j) + 2\,\sigma_d(j)$$

- Track the **historical maximum** of this score:

$$p^* = \max_{k \le j} p_k$$

- **Normalized ratio (current vs best):**

$$r_j = \frac{p_j}{p^*}$$

If $r_j < \alpha_d$: **Drift** → trigger adaptation (reset/retune/replace model), reset counters.

Else if $r_j < \alpha_w$: **Warning** → start buffering recent data, monitor closely.

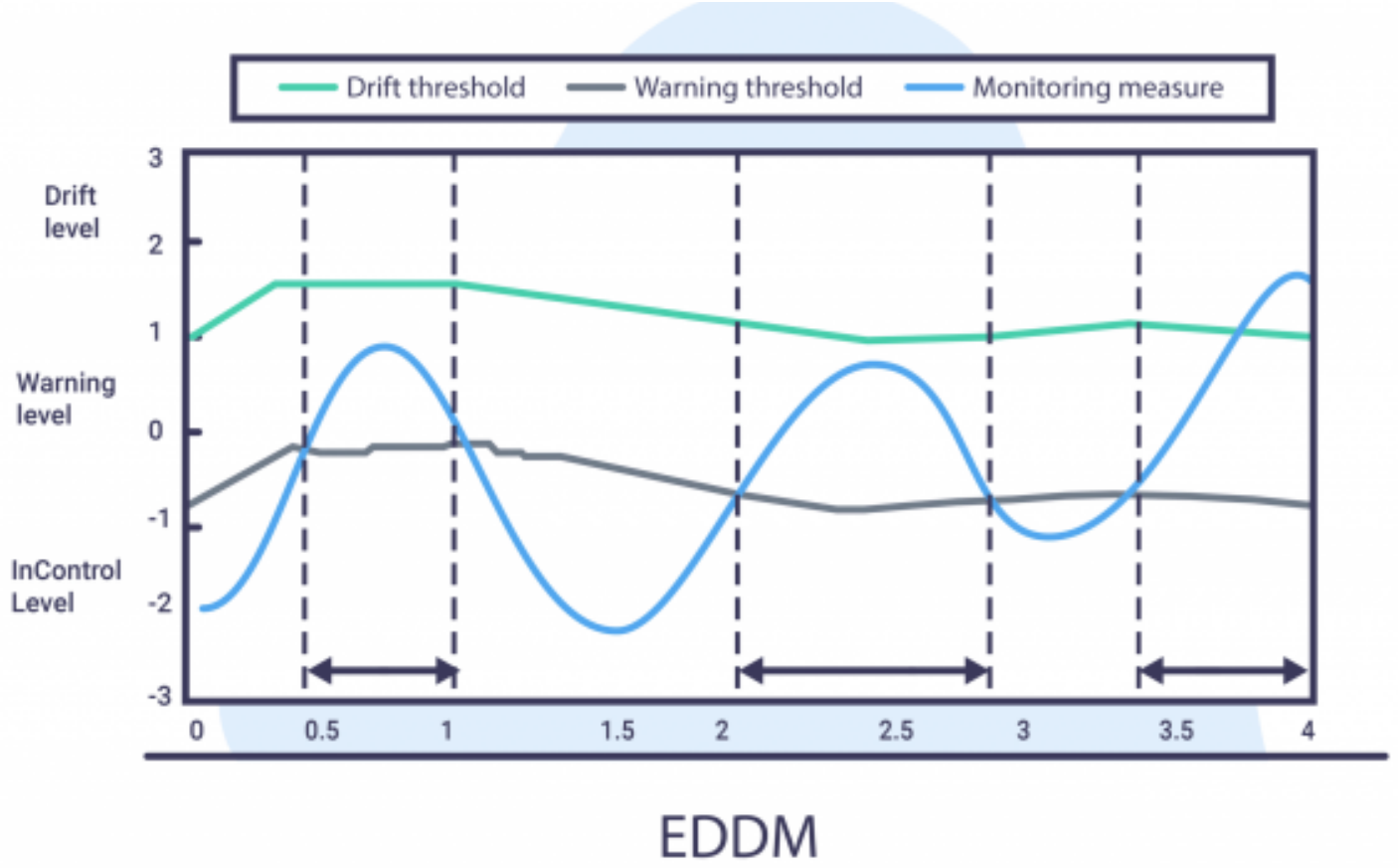
EDDM(Early Drift Detection Methodology)

- Pros
  - Better than DDM at gradual drift (focuses on clustering of errors).
  - More noise-robust than monitoring instantaneous error rate.

  Cons
  - Needs enough errors to estimate distances (can be slow when the model is very accurate or the positive class is rare).
  - Like DDM, it’s supervised (relies on labels, possibly delayed).When to prefer EDDM

EDDM

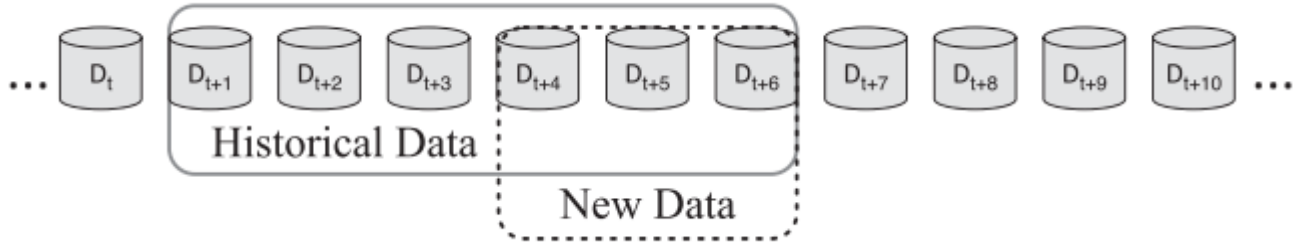
- A warning when $\frac{p_t+2\sigma_t}{p_{max}+2\sigma_{max}} < \alpha$
- An alarm when $\frac{p_t+2\sigma_t}{p_{max}+2\sigma_{max}} < \beta$, where β is usually 0.9



EDDM

Statistical Process Control based methodologies

- STEPD(Statistical Test of Equal Proportions)
  - Computes the accuracy of a chunk C of recent samples and compares it with the overall accuracy from the beginning of the stream, using a chi-squares test to check for deviation.



Two time windows for concept drift detection. The new data window has to be defined by the user.

Window based distribution monitoring methodologies

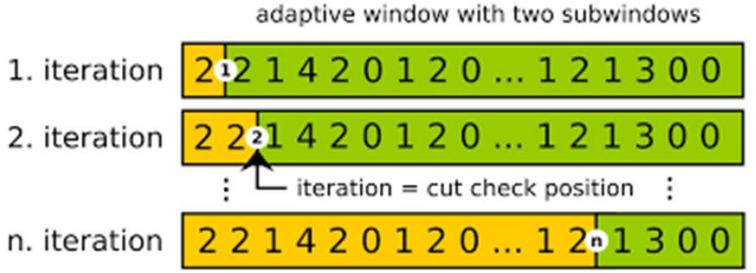
- Window based approaches use a chunk based or sliding window approach over the recent samples, to detect changes.
- Deviations are computed by comparing the current chunk's distribution to a reference distribution, obtained at the start of the stream, from the training dataset.
- These approaches provide precise localization of change point, and are robust to noise and transient changes.
- Extra memory is required to store these two distributions over time.

Window based distribution monitoring methodologies

- ADWIN(Adaptive Windowing)
- It keeps a **single, variable-length window W** of recent values (e.g., losses, scores, a key feature). Whenever the **average in an older part** W0 and the **average in a newer part** W1 differ **significantly** (by a statistical test), ADWIN **cuts** W at that point and signals **change**. If not, it **grows** the window automatically.

Window based distribution monitoring methodologies

- ADWIN(Adaptive Windowing)
  - This algorithm of uses a variable length sliding window, whose length is computed online according to the observed changes.
  - Whenever two large enough sub windows of the current chunk exhibit distinct averages of the performance metric, a drift is detected.
  - Hoeffding bounds are used to determine optimal change threshold and window parameters.



ADWIN0: ADAPTIVE WINDOWING ALGORITHM

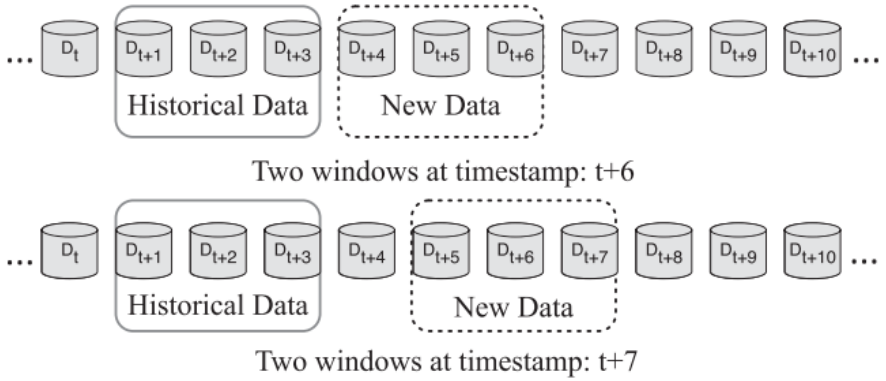
1 Initialize Window W
2 for each $t > 0$
3 do $W \leftarrow W \cup \{x_t\}$ (i.e., add x_t to the head of W)
4 repeat Drop elements from the tail of W
5 until $|\hat{\mu}_{W_0} - \hat{\mu}_{W_1}| < \epsilon_{cut}$ holds
6 for every split of W into $W = W_0 \cdot W_1$
7 output $\hat{\mu}_W$

Window based distribution monitoring methodologies

- **DoD (Degree of Drift)**
  - Detects drifts by computing a distance map of all samples in the current chunk and their nearest neighbors from the previous chunk.
  - If the distance increases more than a parameter θ, a drift is signaled.
  - Drift is managed by replacing the stable model with the reactive one and setting the circular disagreement list to all zeros.

Implicit drift detection methodologies

- **Novelty detection / Clustering based methods**
  - Capable of identifying uncertain suspicious samples, which need further evaluation.
  - An additional 'Unknown' class label to indicate that these suspicious samples do not fit the existing view of the data.
  - Clustering and outlier based approaches are popular for detecting novel patterns, as they summarize current data and can use dissimilarity metrics to identify new samples.

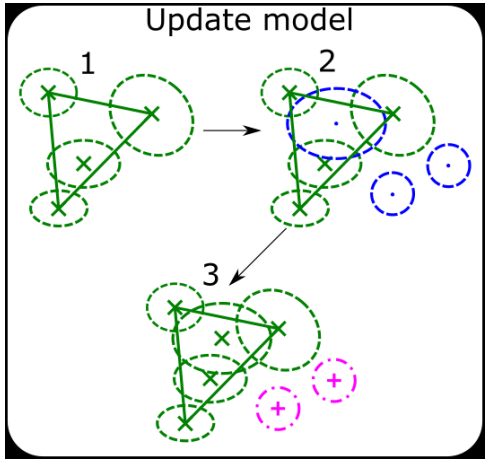


Two sliding time windows, of fixed size. The historical data window will be fixed while the new data window will keep moving.

Novelty detection / Clustering based methods

OLINDDA(OnLIne Novelty and Drift Detection Algorithm)

- Uses K-means data clustering to continuously monitor and adapt to emerging data distribution.
- Unknown samples are stored in a short term memory queue, and are periodically clustered and then either merged with existing similar cluster profiles or added as a novel profile to the pool of clusters.



Novelty detection / Clustering based methods

- All novelty detection techniques rely on clustering to recognize new regions of space, which were previously unseen.
- They suffer from the curse of dimensionality, being distance dependent, and also the problem of dealing with binary data spaces.
- Additionally, they are suitable to detect only specific type of cluster-able drifts.

Multivariate distribution monitoring methods

- These approaches are primarily chunk based and store summarized information of the training data chunk (as histograms of binned values), as the reference distribution, to monitor changes in the current data chunk.

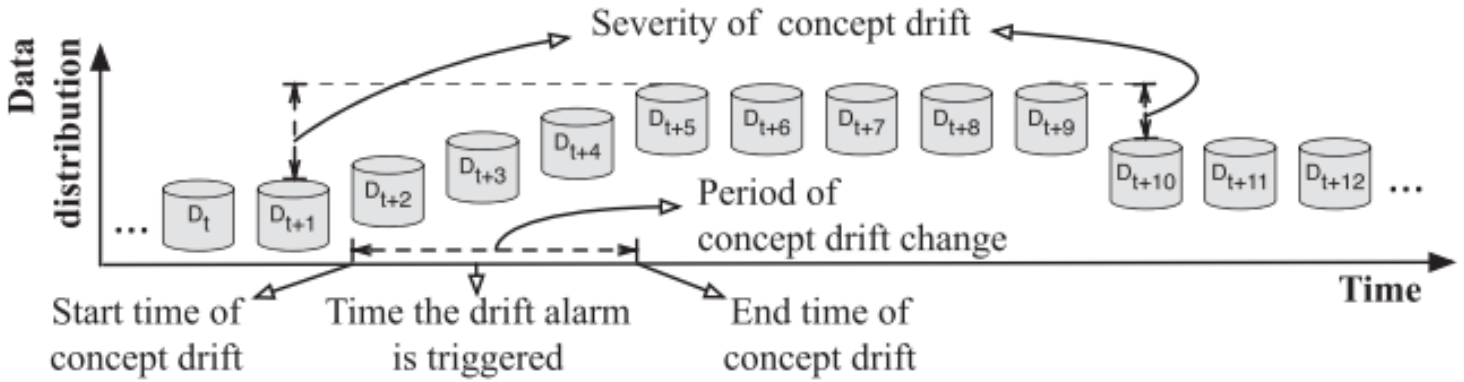
Multivariate distribution monitoring methods

- **Change of Concept(CoC)**

- This technique considers each feature as an independent stream of data and monitors correlation using Pearson correlation between the current chunk and the reference training chunk.
- Change in the average correlation over the features is used as a signal of change.

Concept Drift Understanding

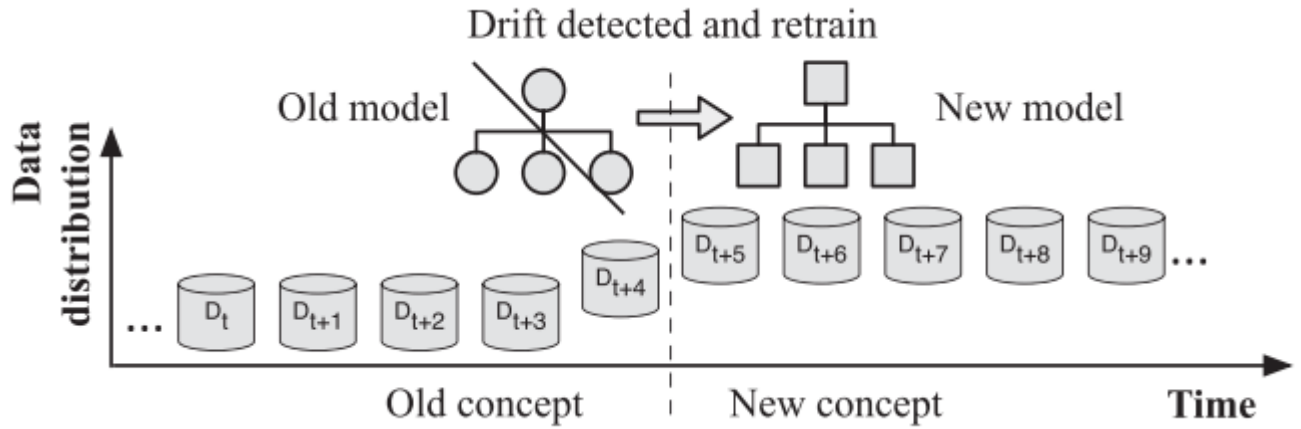
- Drift understanding refers to retrieving concept drift information about
  - “When” (the time at which the concept drift occurs and how long the drift lasts),
  - “How” (the severity /degree of concept drift), and
  - “Where” (the drift regions of concept drift).
- This status information is the output of the drift detection algorithms, and is used as input for drift adaptation.



Drift Adaptation Techniques

- It focuses on strategies for updating existing learning models according to the drift.
- There are three main groups of drift adaptation methods:
  - simple retraining
  - ensemble retraining
  - model adjusting

Training New Models for Global Drift



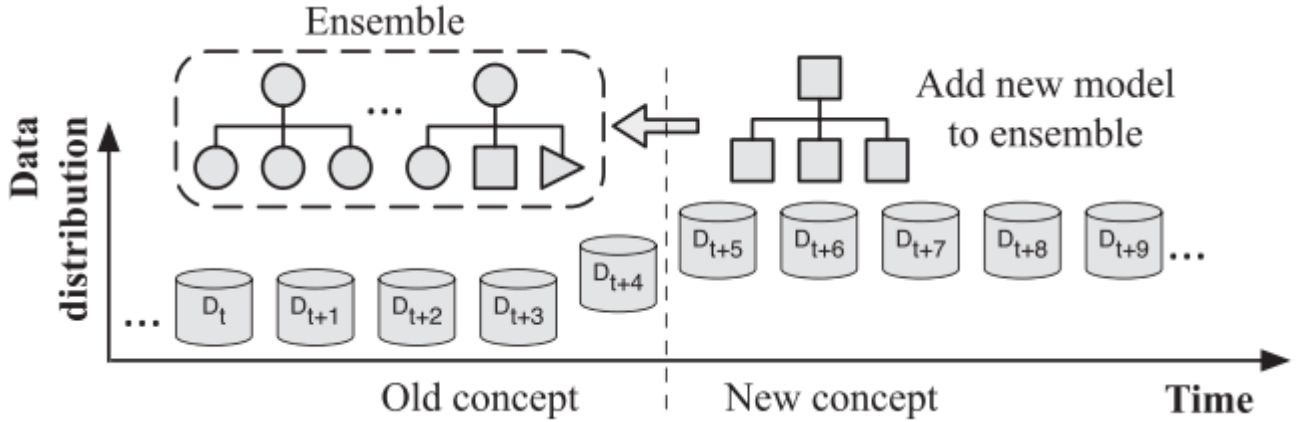
- A new model is trained with latest data to replace the old model when a concept drift is detected.

Training New Models for Global Drift

- Paired Learners follows this strategy and uses two learners:
  - The **stable learner** and the **reactive learner.**
  - If the stable learner frequently misclassifies instances that the reactive learner correctly classifies, a new concept is detected and the stable learner will be replaced with the reactive learner.
- This method is simple to understand and easy to implement, and can be applied at any point in the data stream.

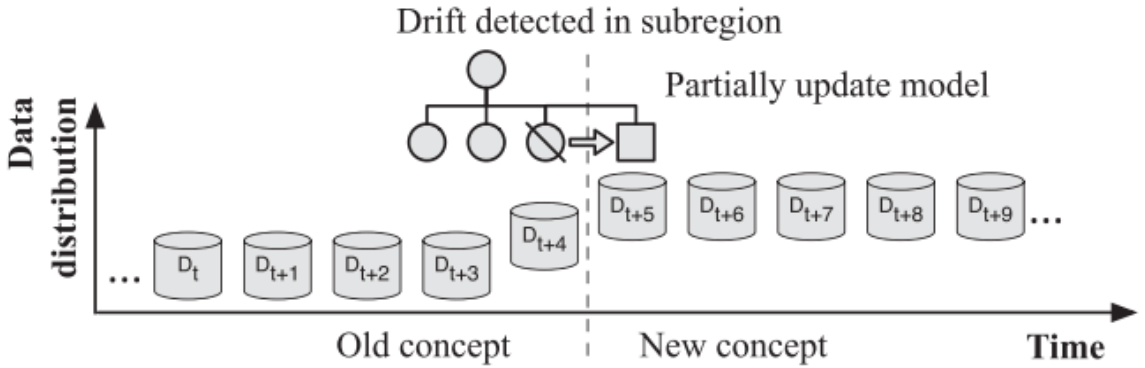
Model Ensemble for Recurring Drift

- In the case of recurring concept drift, preserving and reusing old models can save significant effort to retrain a new model for recurring concepts.
- This is the core idea of using ensemble methods to handle concept drift



Adjusting Existing Models for Regional Drift

- An alternative to retraining an entire model is to develop a model that adaptively learns from the changing data. Such models have the ability to partially update themselves when the underlying data distribution changes, as shown in Figure:



A decision tree node is replaced with a new one as its performance deteriorates when a concept drift occurs in a subregion

- This approach is arguably more efficient than retraining when the drift only occurs in local regions.
- Many methods in this category are based on the decision tree algorithm because trees have the ability to examine and adapt to each sub-region separately.

References

- “NoSQL -- Your Ultimate Guide to the Non - Relational Universe!” http://nosql-database.org/links.html
- “NoSQL (RDBMS)” http://en.wikipedia.org/wiki/NoSQL
- PODC Keynote, July 19, 2000. *Towards Robust*. *Distributed Systems*. Dr. Eric A. *Brewer*. Professor, UC Berkeley. Co-Founder & Chief Scientist, Inktomi . *www.eecs.berkeley.edu/~**brewer**/cs262b-2004/PODC-keynote.pdf*
- “Brewer's CAP Theorem” posted by Julian Browne, January 11, 2009. http://www.julianbrowne.com/article/viewer/brewers-cap-theorem
- “How to write a CV” Geek & Poke Cartoon http://geekandpoke.typepad.com/geekandpoke/2011/01/nosql.html

References

- “Exploring CouchDB: A document-oriented database for Web applications”, Joe Lennon, Software developer, Core International. http://www.ibm.com/developerworks/opensource/library/os-couchdb/index.html
- “Graph Databases, NOSQL and Neo4j” Posted by Peter Neubauer on May 12, 2010 at: http://www.infoq.com/articles/graph-nosql-neo4j
- “Cassandra vs MongoDB vs CouchDB vs Redis vs Riak vs HBase comparison”, Kristóf Kovács. http://kkovacs.eu/cassandra-vs-mongodb-vs-couchdb-vs-redis
- “Distinguishing Two Major Types of Column-Stores” Posted by Daniel Abadi onMarch 29, 2010 http://dbmsmusings.blogspot.com/2010/03/distinguishing-two-major-types-of_29.html

References

- “MapReduce: Simplified Data Processing on Large Clusters”, Jeffrey Dean and Sanjay Ghemawat, December 2004. http://labs.google.com/papers/mapreduce.html
- “Scalable SQL”, ACM Queue, Michael Rys, April 19, 2011 http://queue.acm.org/detail.cfm?id=1971597
- “a practical guide to noSQL”, Posted by Denise Miura on March 17, 2011 at http://blogs.marklogic.com/2011/03/17/a-practical-guide-to-nosql/

Books

- “CouchDB *The Definitive Guide*”, J. Chris Anderson, Jan Lehnardt and Noah Slater. O’Reilly Media Inc., Sebastopool, CA, USA. 2010
- “Hadoop *The Definitive Guide*”, Tom White. O’Reilly Media Inc., Sebastopool, CA, USA. 2011
- “MongoDB *The Definitive Guide*”, Kristina Chodorow and Michael Dirolf. O’Reilly Media Inc., Sebastopool, CA, USA. 2010

Bibliography

Bean, Randy. “Why Becoming a Data-Driven Organization Is So Hard.” *Harvard Business Review*, 24 Feb. 2022. Accessed Oct. 2022.

Brown, Annie. “Utilizing AI And Big Data To Reduce Costs And Increase Profits In Departments Across An Organization.” *Forbes*, 13 April 2021. Accessed Oct. 2022.

Burciaga, Aaron. “Five Core Virtues For Data Science And Artificial Intelligence.” *Forbes*, 27 Feb. 2020. Accessed Aug. 2022.

Cadwalladr, Carole, and Emma Graham-Harrison. “Revealed: 50 million Facebook profiles harvested for Cambridge Analytica in major data breach.”
The Guardian, 17 March 2018. Accessed Aug. 2022.

Carlier, Mathilde. “Connected light-duty vehicles as a share of total vehicles in 2023.” *Statista*, 31 Mar. 2021. Accessed Oct. 2022.

Carter, Rebekah. “The Ultimate List of Big Data Statistics for 2022.” Findstack, 22 May 2021. Accessed Oct. 2022.

- Castelvecchi, Davide. “Underdog technologies gain ground in quantum-computing race.” *Nature*, 6 Nov. 2023. Accessed Feb. 2023.

Clark-Jones, Anthony, et al. “Digital Identity:” *UBS*, 2016. Accessed Aug 2022.

“The Cost of Bad Data – Infographic.” *Pragmatic Works*, 25 May 2017. Accessed Oct. 2022.

Demchenko, Yuri, et al. “Data as Economic Goods: Definitions, Properties, Challenges, Enabling Technologies for Future Data Markets.“ *ITU Journal: ICT Discoveries*, Special Issue, no. 2, vol. 23, Nov. 2018. Accessed Aug 2022.

Feldman, Sarah. ”20 Years of Quantum Computing Growth.” *Statista*, 6 May 2019. Accessed Oct. 2022.

“Genomic Data Science.” *NIH, National Human Genome Research Institute*, 5 April 2022. Accessed Oct. 2022.

Bibliography

Hasbe, Sudhir, and Ryan Lippert. “The democratization of data and insights: making real-time analytics ubiquitous.” *Google Cloud*, 15 Jan. 2021. Accessed Aug. 2022.

Helmenstine, Anne. “Viscosity Definition and Examples.” *Science Notes*, 3 Aug. 2021. Accessed Aug. 2022.

“How data storytelling and augmented analytics are shaping the future of BI together.” *Yellowfin*, 19 Aug. 2021. Accessed Aug. 2022.

“How Netflix Saves $1B Annually using AI?” *Logidots*, 24 Sept. 2021. Accessed Oct. 2022

Hui, Kenneth. “The AWS Love/Hate Relationship with Data Gravity.” *Cloud Architect Musings*, 30 Jan. 2017. Accessed Aug 2022.

ICD. “The Growth in Connected IoT Devices Is Expected to Generate 79.4ZB of Data in 2025, According to a New IDC Forecast.” *Business Wire*, 18 June 2019. Accessed Oct 2022.

Internet of Things (IoT) and non-IoT active device connections worldwide from 2010 to 2025” *Statista*, 27 Nov. 2022. Accessed Nov. 2022.

Koch, Gunter. “The critical role of data management for autonomous driving development.” *DXC Technology*, 2021. Accessed Aug. 2022.

Morris, John. “The Pull of Data Gravity.” *CIO*, 23 Feb. 2022. Accessed Aug. 2022.

Nield, David. “Google's Quantum Computer Is 100 Million Times Faster Than Your Laptop.” *ScienceAlert*, 9 Dec. 2015. Accessed Oct. 2022.

Redman, Thomas C. “Seizing Opportunity in Data Quality.” *MIT Sloan Management Review*, 27 Nov. 2017. Accessed Oct. 2022.

Segovia Domingo, Ana I., and Álvaro Martín Enríquez. “Digital Identity: the current state of affairs.” *BBVA Research*, 2018. Accessed Aug. 2022.

Bibliography

"State of IoT 2022: Number of connected IoT devices growing 18% to 14.4 billion globally." *IOT Analytics*, 18 May 2022. Accessed. 14 Nov. 2022.

Strod, Eran. "Data Observability and Monitoring with DataOps." *DataKitchen*, 10 May 2021. Accessed Aug. 2022.

Sujay Vailshery, Lionel. "Edge computing market value worldwide 2019-2025." *Statista*, 25 Feb. 2022. Accessed Oct 2022.

Sujay Vailshery, Lionel. "IoT and non-IoT connections worldwide 2010-2025." *Statista*, 6 Sept. 2022. Accessed Oct. 2022.

Sumina, Vladimir. "26 Cloud Computing Statistics, Facts & Trends for 2022." *Cloudwards*, 7 June 2022. Accessed Oct. 2022.

Taulli, Tom. "What You Need To Know About Dark Data." *Forbes*, 27 Oct. 2019. Accessed Oct. 2022.

Taylor, Linnet. "What is data justice? The case for connecting digital rights and freedoms globally." *Big Data & Society*, July-Dec 2017. Accessed Aug 2022.

"Twitter: Data Collection With API Research Paper." *IvyPanda*, 28 April 2022. Accessed Aug. 2022.

"Using governance automation to reduce data risk." *Nintex*, 15 Nov. 2021. Accessed Oct. 2022

"Volume of data/information created, captured, copied, and consumed worldwide from 2010 to 2020, with forecasts from 2021 to 2025." *Statista*, 8 Sept. 2022. Accessed Oct 2022.

Wang, R. "Monday's Musings: Beyond The Three V's of Big Data – Viscosity and Virality." *Forbes*, 27 Feb. 2012. Accessed Aug 2022.

"What is a data fabric?" *IBM*, n.d. Accessed Aug 2022.

Yego, Kip. "Augmented data management: Data fabric versus data mesh." *IBM*, 27 April 2022. Accessed Aug 2022.

Thank you