Big Data Analytics

Session

Data Stream Mining

Outline

- Introduction
- Data Stream Classification
- Data Stream Clustering
- Novel Class Detection

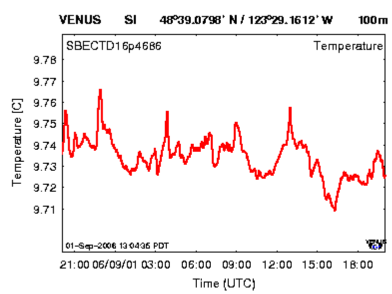
Outline

- Introduction
- Data Stream Classification
- Data Stream Clustering
- Novel Class Detection

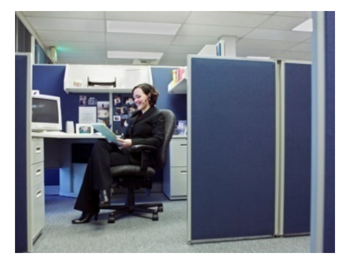
Introduction

- Characteristics of Data streams are:
  - Continuous flow of data
  - Examples

Network traffic



Sensor data



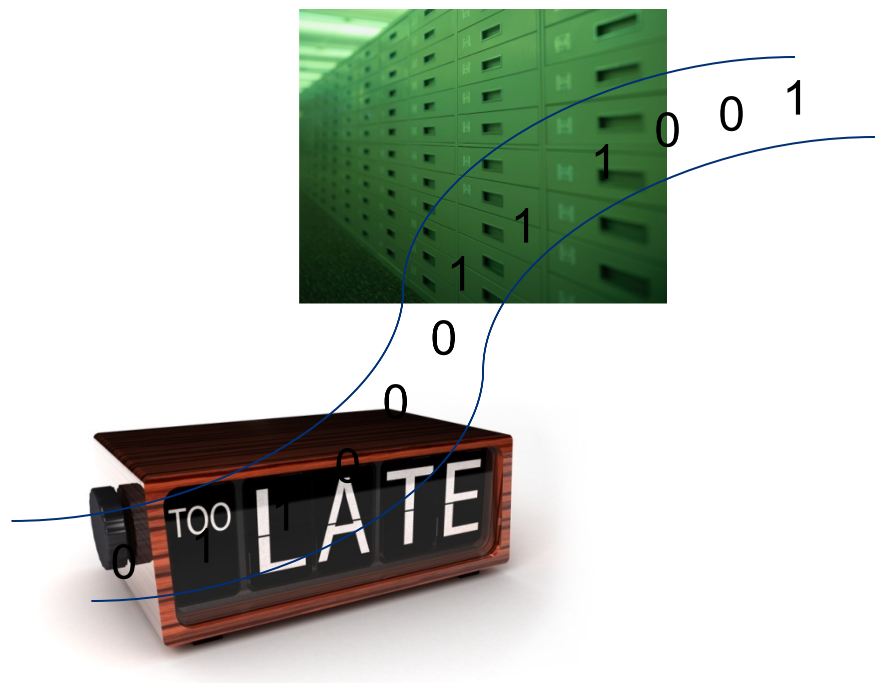
Call center records

Challenges

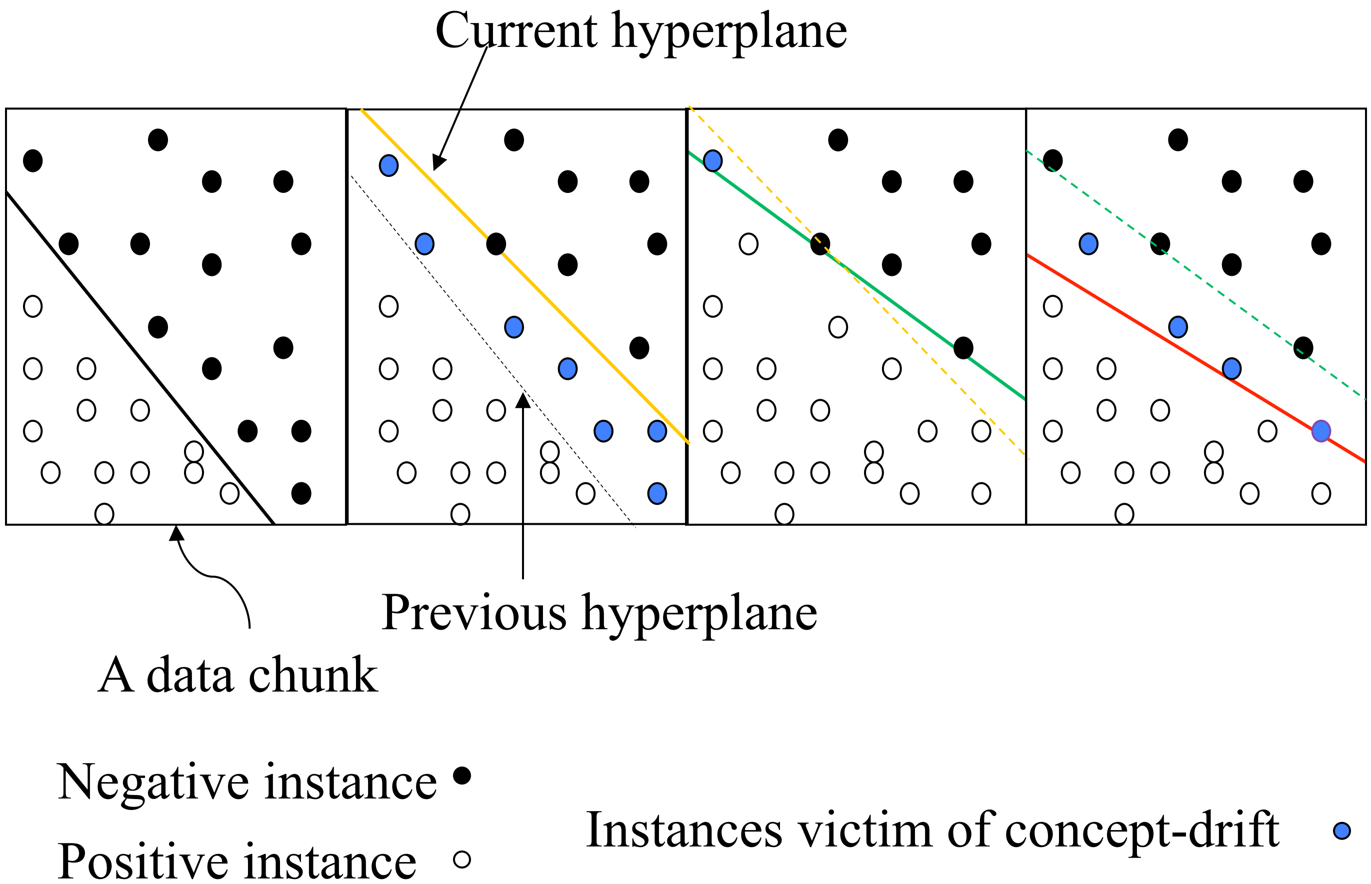
- Infinite length
- Concept-drift
- Concept-evolution
- Feature evolution

Infinite length

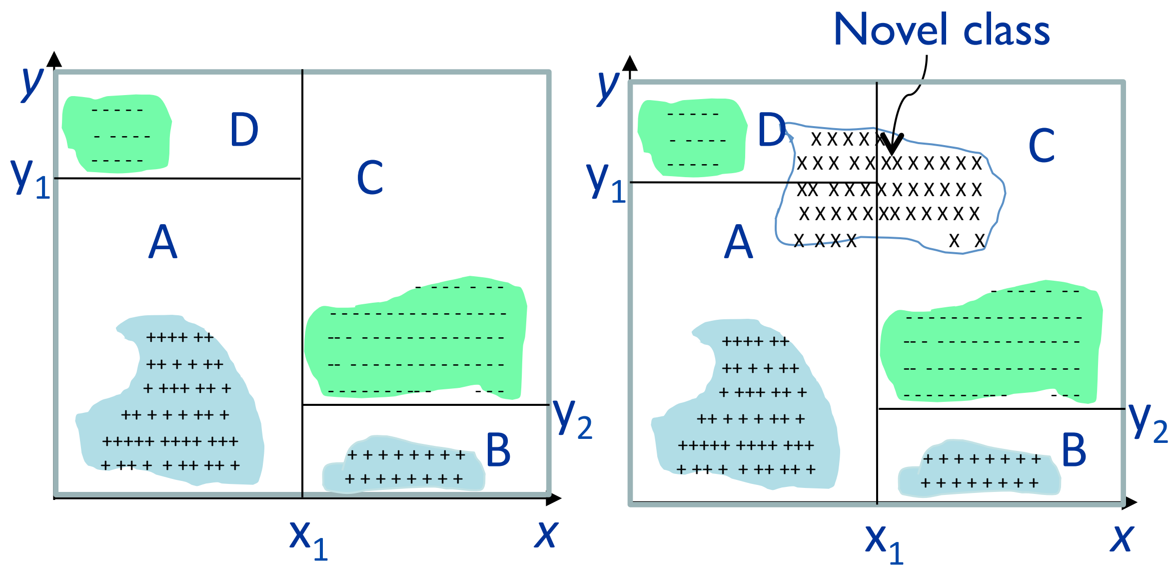
- Impractical to store and use all historical data
  - Requires infinite storage
  - And running time



Concept-Drift



Concept-Evolution



- Classification rules:
  - R1. if (x > x_1 and y < y_2) or ($x < x_1$ and $y < y_1$) then class = +
  - R2. if (x > x_1 and y > y2) or ($x < x_1$ and $y > y_1$) then class = -

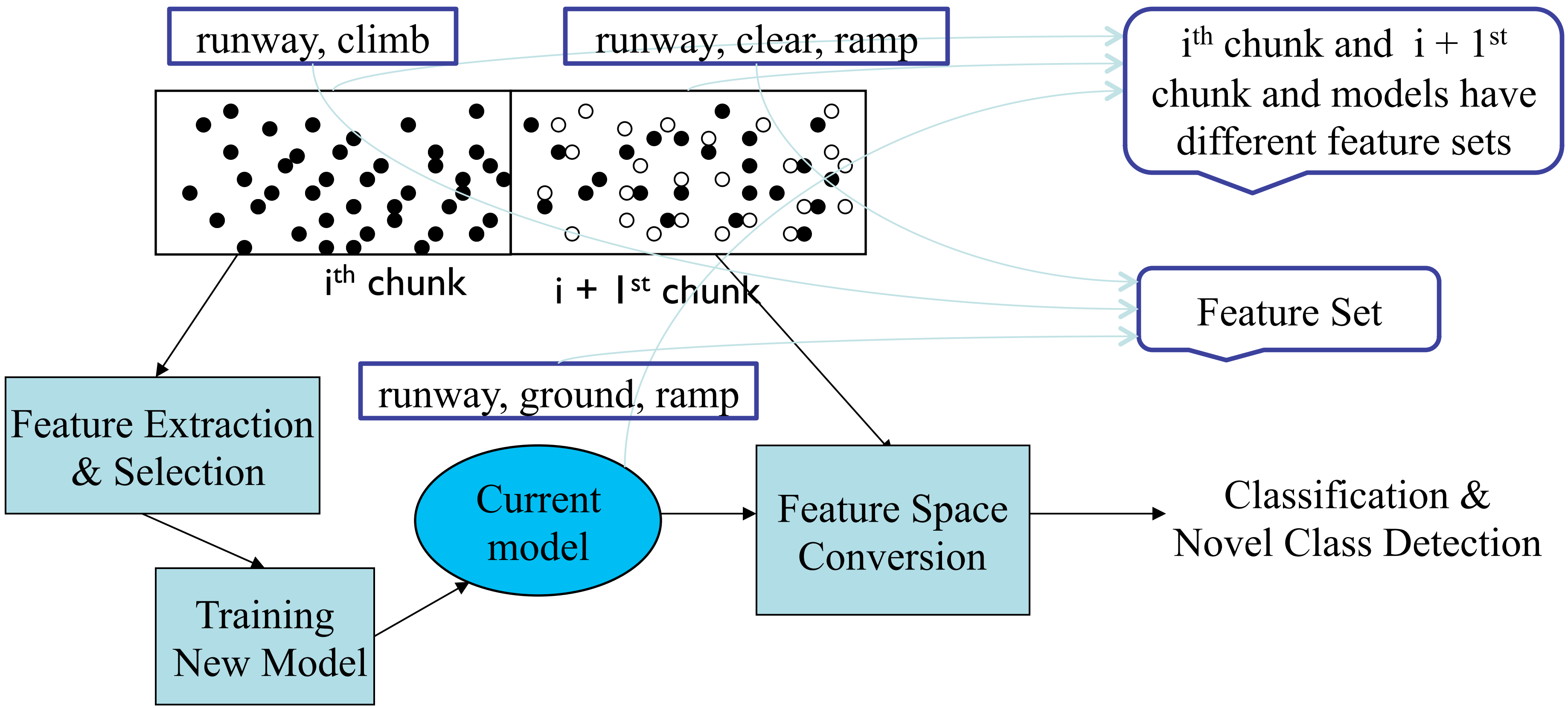
- Existing classification models misclassify novel class instances

Dynamic Features

- Why new features evolving
  - Infinite data stream
    - Normally, global feature set is unknown
    - New features may appear
  - Concept drift
    - As concept drifting, new features may appear
  - Concept evolution
    - New type of class normally holds new set of features

- Different chunks may have different feature sets

Dynamic Features



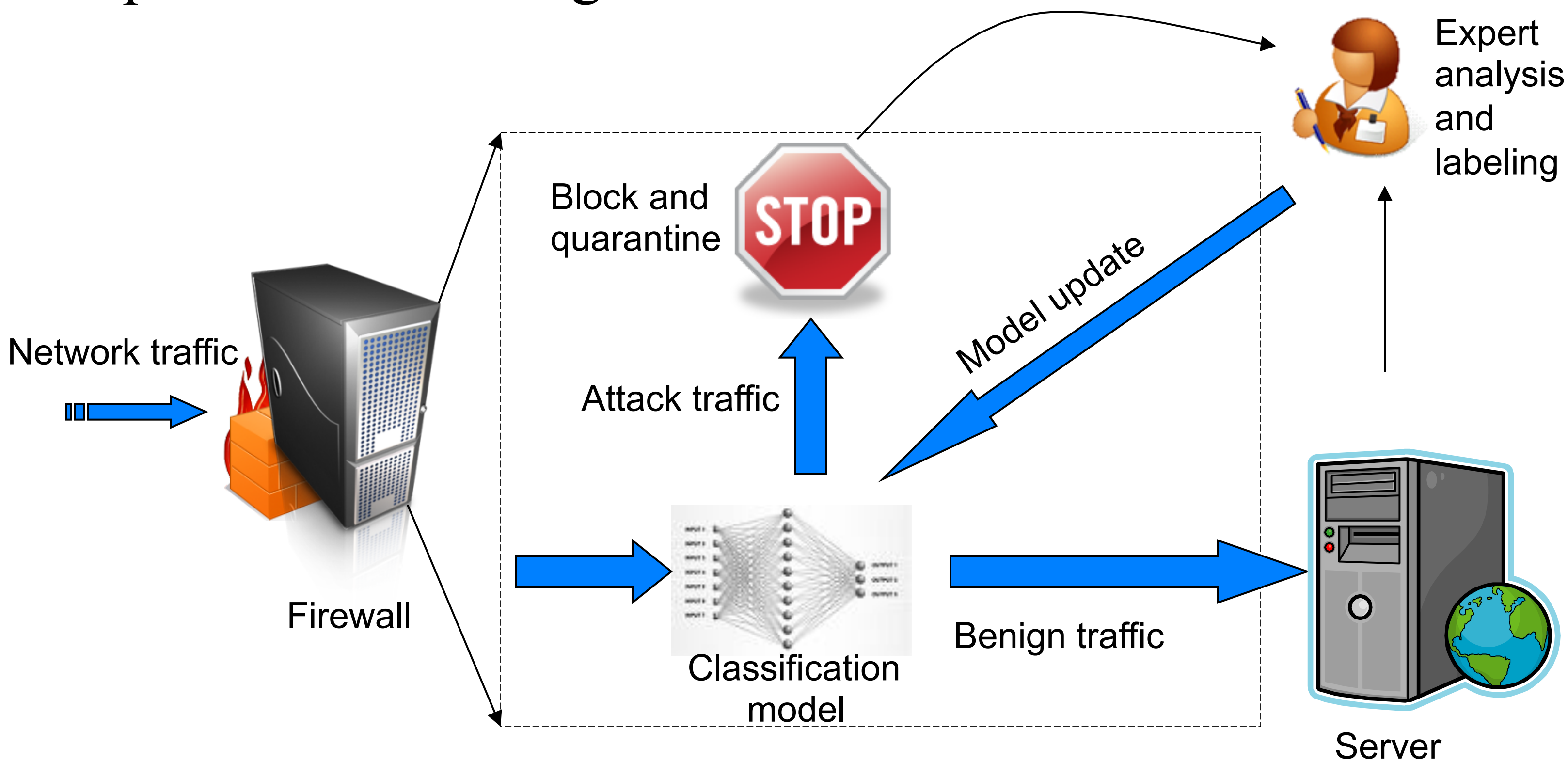
- Existing classification models need complete fixed features and apply to all the chunks.
- Global features are difficult to predict.
- ➢ One solution is using all English words and generate vector.
- Dimension of the vector will be too high.

Outline

- Introduction
- Data Stream Classification
- Data Stream Clustering
- Novel Class Detection

Data Stream Classification

- Uses past ***labeled data*** to build classification model
- Predicts the labels of future instances using the model
- Helps decision making



Data Stream Classification – Applications

- Security monitoring
- Network monitoring and traffic engineering
- Business: credit card transaction flows
- Telecommunication calling records
- Web logs and web page click streams
- Financial market: stock exchange

Data Stream Classification – Approaches

- Single model incremental classification
- Ensemble – Model based classification
  - Supervised
  - Semi-supervised
  - Active learning

Data Stream Classification – Approaches

- Single model incremental classification

- Ensemble – Model based classification
  - Supervised
  - Semi-supervised
  - Active learning

Single Model Incremental Classification

- Introduction
- Problems in mining Data Stream
- Decision Tree
- Very Fast Decision Tree (VFDT)
- Concept-Adapting VFDT (CVFDT)

Problems in Mining Data Streams

- Traditional data mining techniques usually require entire data set to be present.
- Random access (or multiple access) to the data.
- Impractical to store the whole data.
- Simple calculation per data due to time and space constraints.

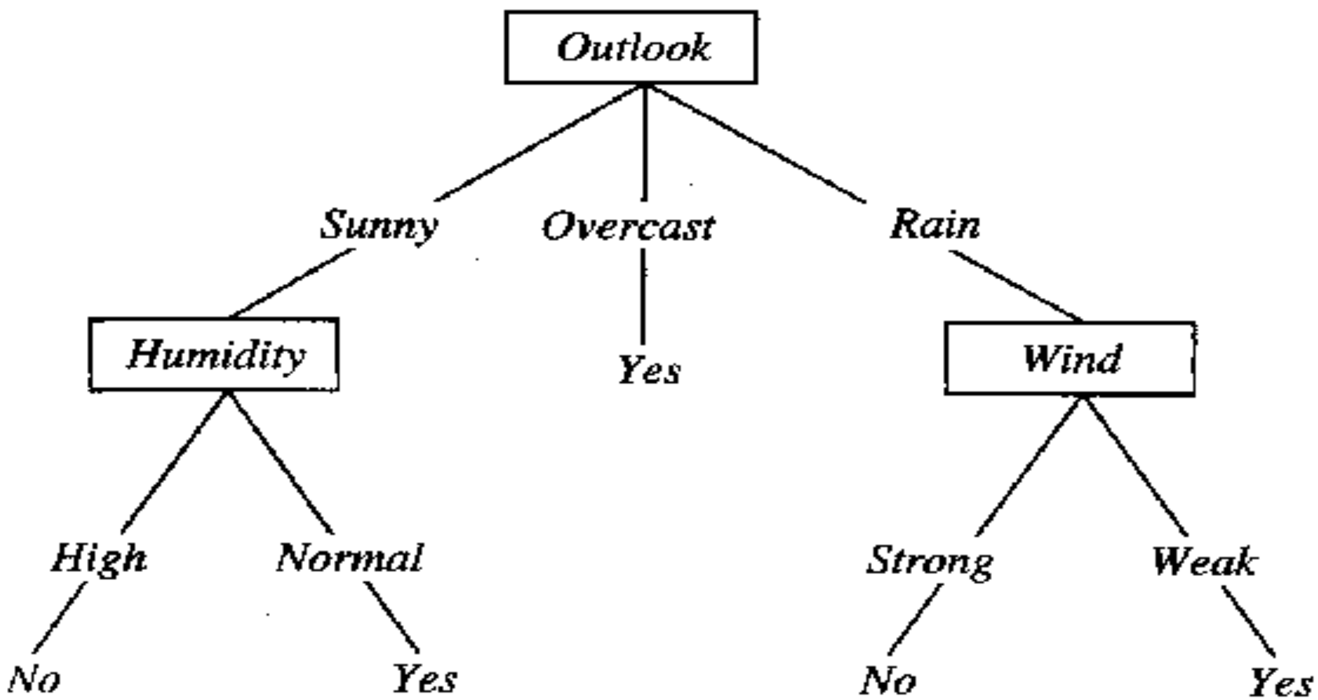
Revision: Tree-Based Classification

- Decision tree is a classification model. Its basic structure is a general tree structure
  - Internal node: test on example's attribute value
  - Leaf node: class labels

- Key idea:
  - 1) pick an attribute to test at root
  - 2) divide the training data into subsets D_i for each value the attribute can take on
  - 3) build the tree for each D_i and splice it in under the appropriate branch at the root

Decision Tree Example

- Shall we go outside today?



Decision Tree – Algorithm

- How to build a decision tree?

Main loop:

1. $A \leftarrow$ the "best" decision attribute for next *node*
2. Assign A as decision attribute for *node*
3. For each value of A, create new descendant of *node*
4. Sort training examples to leaf nodes
5. If training examples perfectly classified, Then STOP, Else iterate over new leaf nodes

Decision Trees – Limitations and Goal

- Limitations
  - Classic decision tree learners assume all training data can be simultaneously stored in main memory.
  - Disk-based decision tree learners repeatedly read training data from disk sequentially
- Goal
  - Design decision tree learners that read each example at most once, and use a small constant time to process it.

Very Fast Decision Trees (VFDT)

- In order to find the best attribute at a node, it may be sufficient to consider only a small subset of the training examples that pass through that node.
  - Given a stream of examples, use the first ones to choose the root attribute.
  - Once the root attribute is chosen, the successive examples are passed down to the corresponding leaves, and used to choose the attribute there, and so on recursively.
- Use Hoeffding bound to decide how many examples are enough at each node

How to choose an attribute?

- In Session 5, we choose the attributes that can minimise the error rate.
- Recall: $$\text{Error rate} = \frac{\#\text{ wrong predictions}}{\text{total }\#\text{ of predictions}} = \frac{f_{10} + f_{01}}{f_{11} + f_{10} + f_{01} + f_{00}}$$
- Here we introduce a more algorithmic criterion: information gain
- Gain(A) = Info(D) - Info_A(D)
- Info(D): the info needed to classify a tuple in D
- Info_A(D): the info needed to classify D after using A to split D
- Gain(A): information gain by branching on attribute A
- Calculate the Info Gain for each attribute and pick the one with the largest gain

VFDT (contd.)

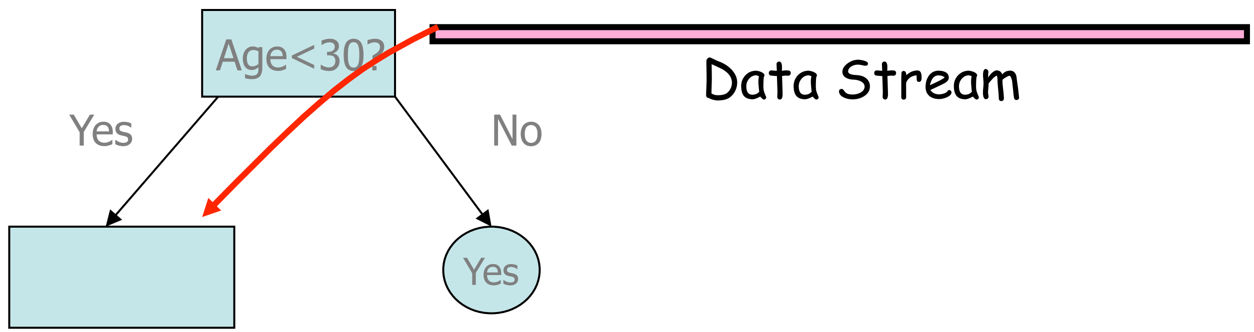
- Calculate the information gain for the attributes and determines the best two attributes A and B
  - Pre-pruning: consider a "null" attribute that consists of not splitting the node

- At each node, check for the condition

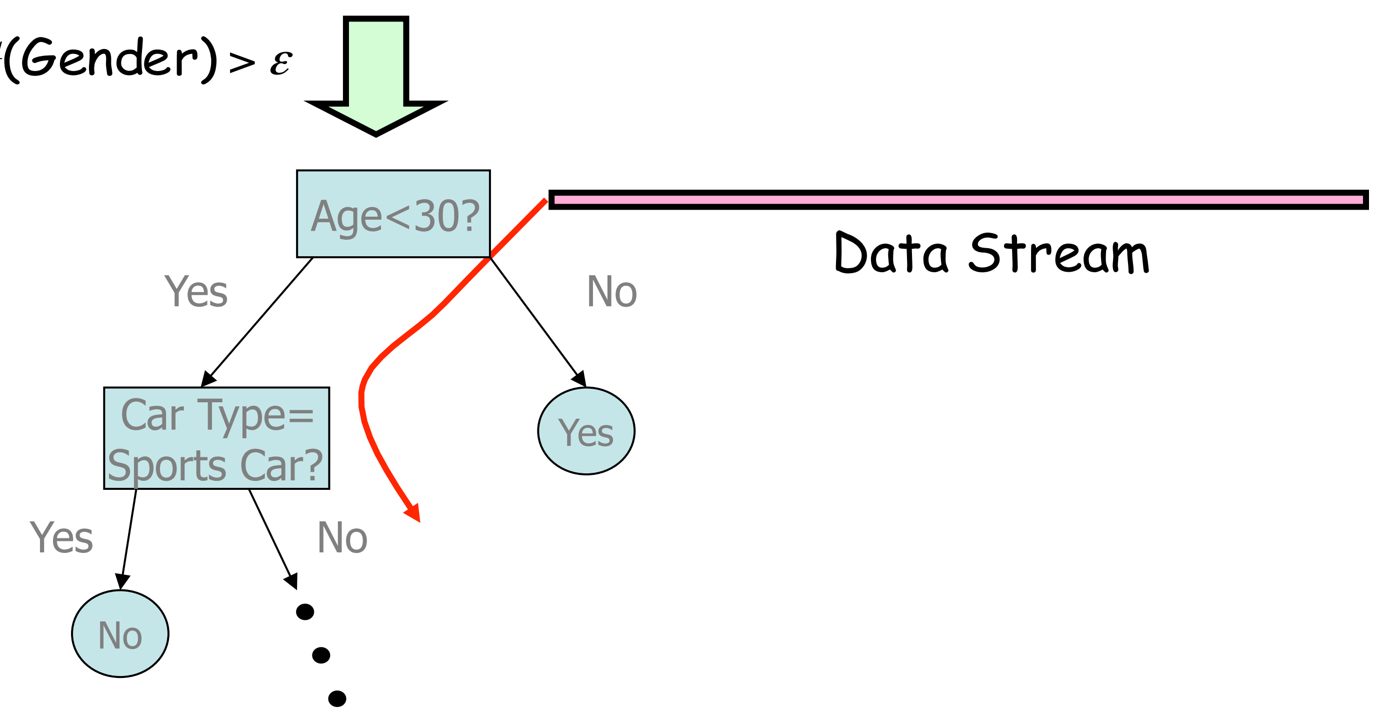
$$\Delta \text{Gain} = \text{Gain}(A) - \text{Gain}(B) > \varepsilon$$

- If the condition is satisfied, create child nodes based on the test at the node
- If not, stream in more examples and perform calculations till condition satisfied

VFDT (contd.)



$$\bar{G}(\text{Car Type}) - \bar{G}(\text{Gender}) > \varepsilon$$



Limitations of VFDT

- VFDT assumes

  training data is a sample drawn from *stationary distribution.*

- Most large databases or data streams violate this assumption
  - Concept Drift: data is generated by a *time-changing* concept function, e.g.
    - Seasonal effects
    - Economic cycles

- Goal:
  - Mining continuously changing data streams
  - Scale well

Improvements for VFDT

- Common Approach: Sliding window
  - when a new example arrives, reapply a traditional learner to a sliding window of w most recent examples

- Drawbacks of sliding window
  - Sensitive to window size
    - If w is small relative to the concept shift rate, assure the availability of a model reflecting the current concept
    - Too small w may lead to insufficient examples to learn the concept
  - If examples arrive at a rapid rate or the concept changes quickly, the computational cost of reapplying a learner may be prohibitively high.

CVFDT – Why?

- Concept-adapting very fast decision trees (CVFDT)
  - Extend VFDT
  - Maintain VFDT's speed and accuracy
  - Detect and respond to changes in the example-generating process

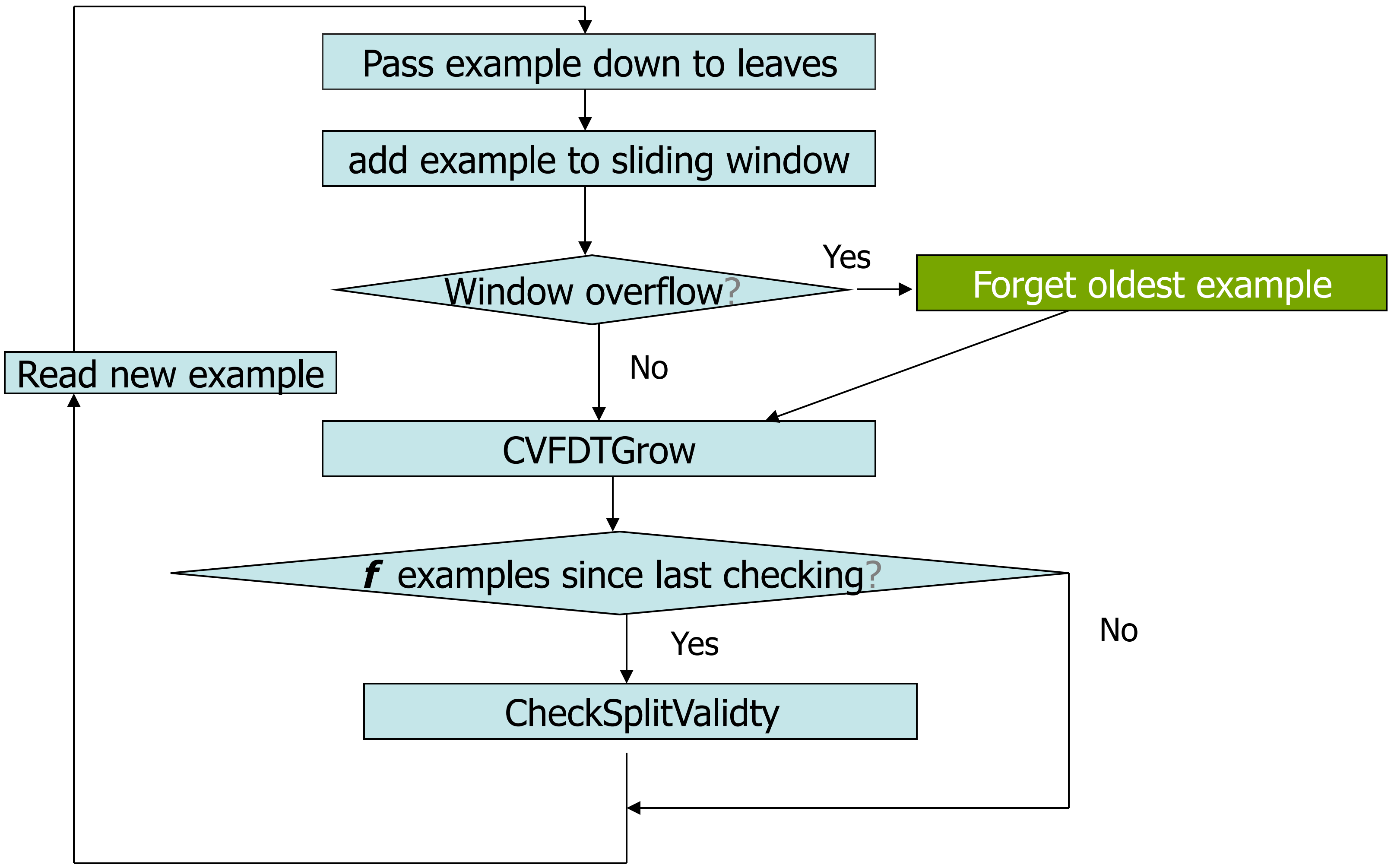
CVFDT – How?

- With a time-changing concept, the current splitting attribute of some nodes may not be the best any more.

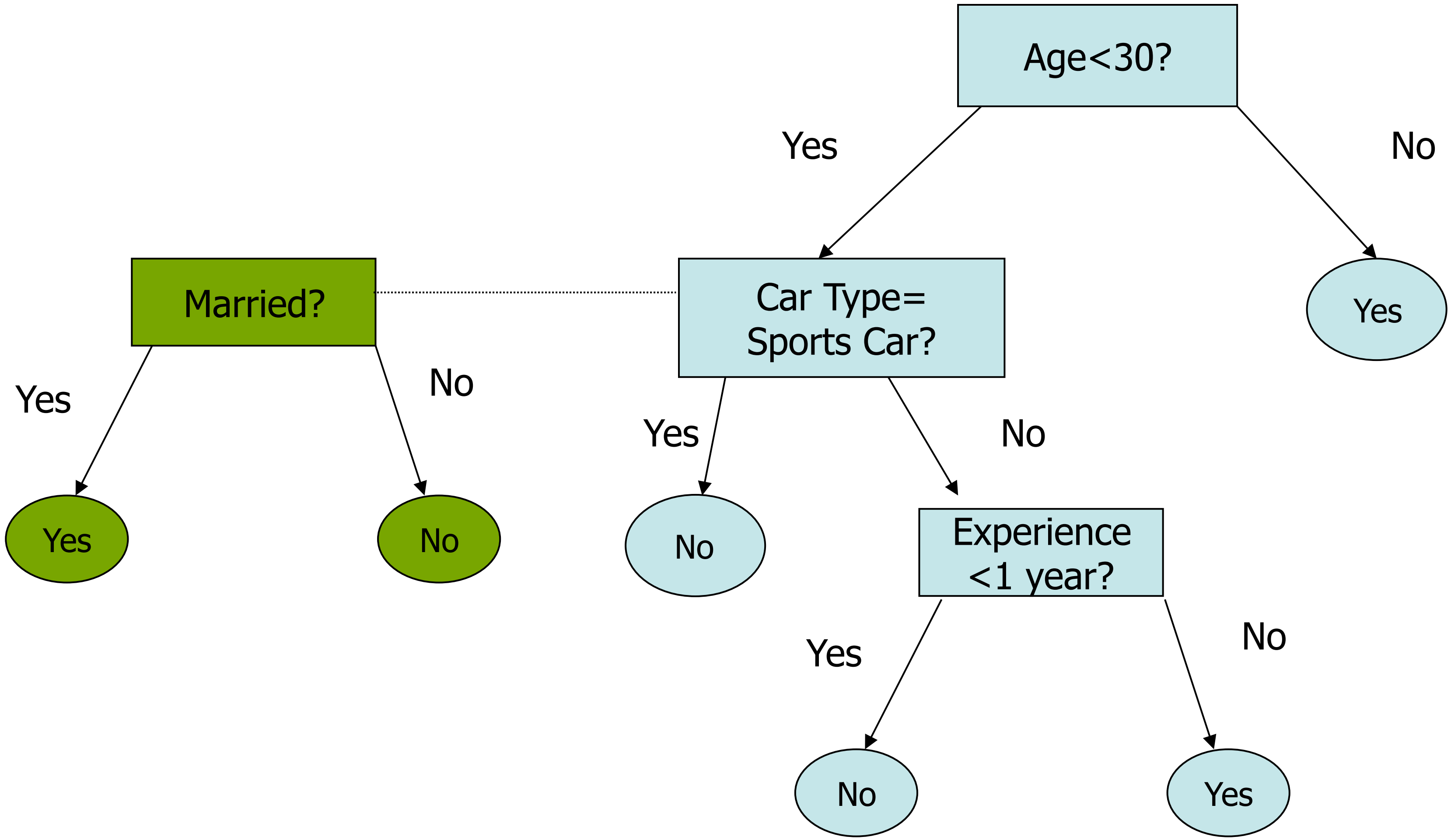
- An outdated sub tree may still be better than the best single leaf, particularly if it is near the root.
  - Grow an alternative sub tree with the new best attribute at its root, when the old attribute seems out-of-date.

- Periodically use a bunch of samples to evaluate qualities of trees.
  - Replace the old sub tree when the alternate one becomes more accurate.

CVFDT – The algorithm



CVFDT – Example



CVFDT – Experimental Result