News Classifier and Trending Event Detection Using Python

SHARANYA KV

SUJITH CP

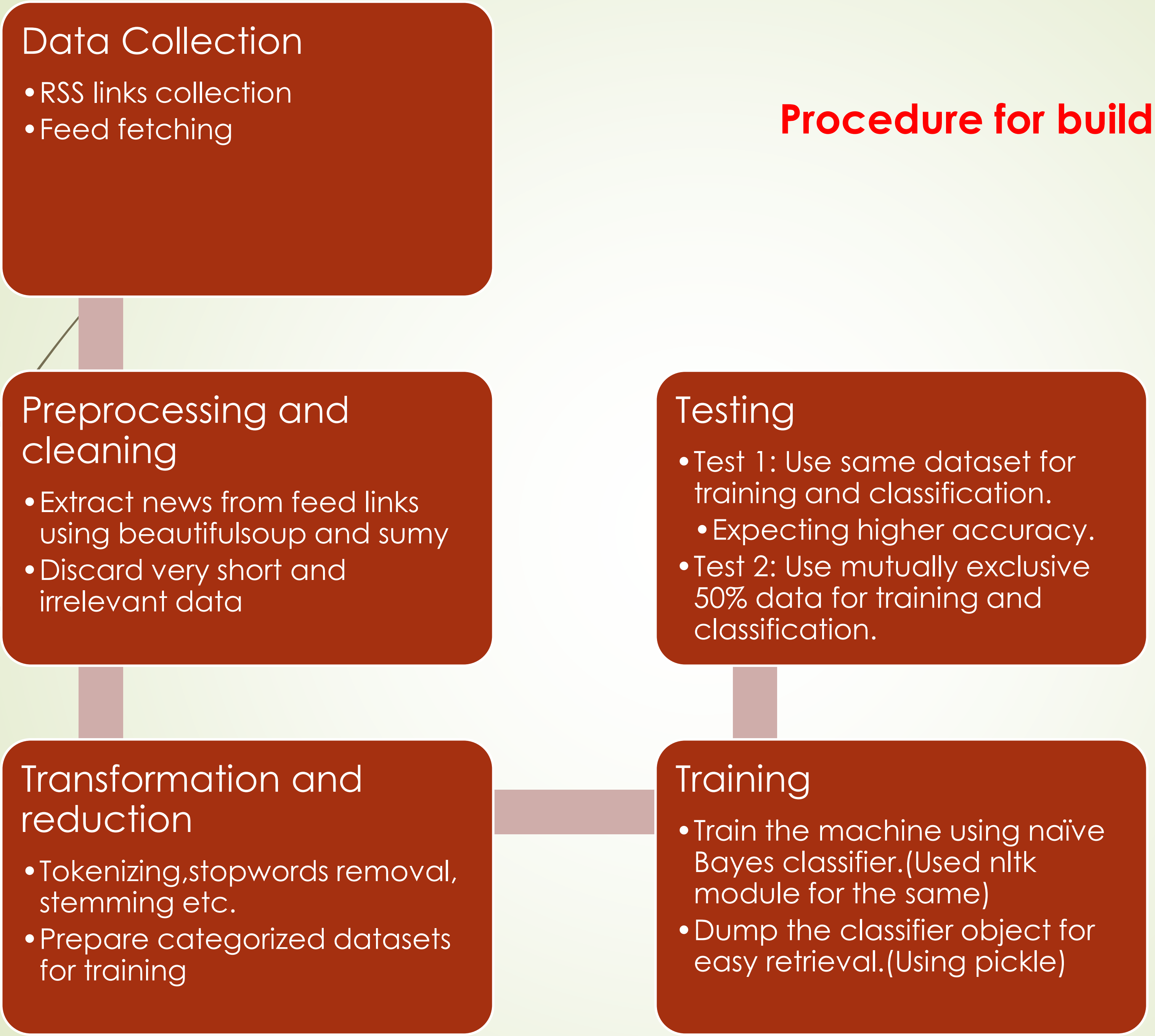
A Brief Introduction

Data classification and information mining are trending and rapidly evolving fields of Computer science. Data mining is the process of identifying patterns and relationships in data that often are not obvious in large, complex data sets.
News event detection and classification are very helpful in socio-economic research and such purposes. As social media and real-time information sharing gains popularity, an automated event detection system can do better in understanding the required events.

Machine Learning Technologies for Data Mining

- Inductive Logic Programming
- Genetic Algorithms
- Neural Networks
- **Statistical Methods** **(We use Bayesian method for classification)**
- Decision Trees
- Hidden Markov Models

Procedure for building news classifier



Data Collection

- RSS links collection
- Feed fetching

Preprocessing and cleaning

- Extract news from feed links using beautifulsoup and sumy
- Discard very short and irrelevant data

Transformation and reduction

- Tokenizing,stopwords removal, stemming etc.
- Prepare categorized datasets for training

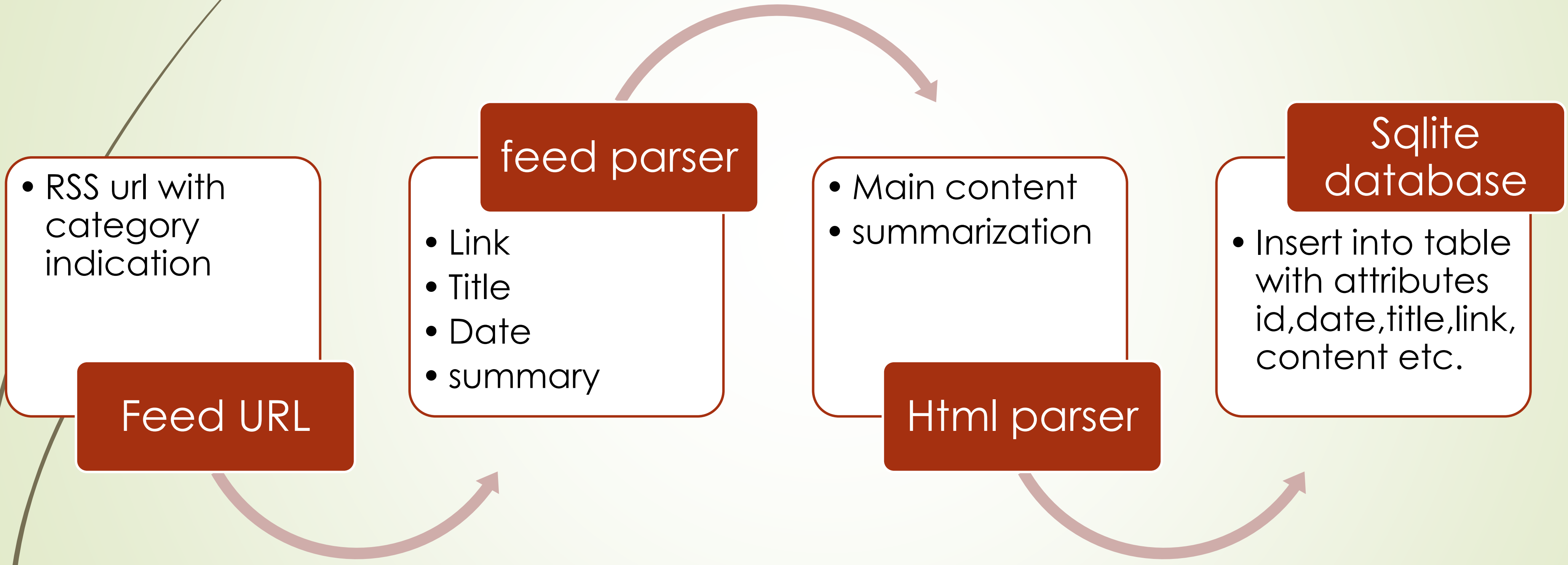
Training

- Train the machine using naïve Bayes classifier.(Used nltk module for the same)
- Dump the classifier object for easy retrieval.(Using pickle)

Testing

- Test 1: Use same dataset for training and classification.
  - Expecting higher accuracy.
- Test 2: Use mutually exclusive 50% data for training and classification.

Data collection and cleaning



Feed URL

- RSS url with category indication

feed parser

- Link
- Title
- Date
- summary

Html parser

- Main content
- summarization

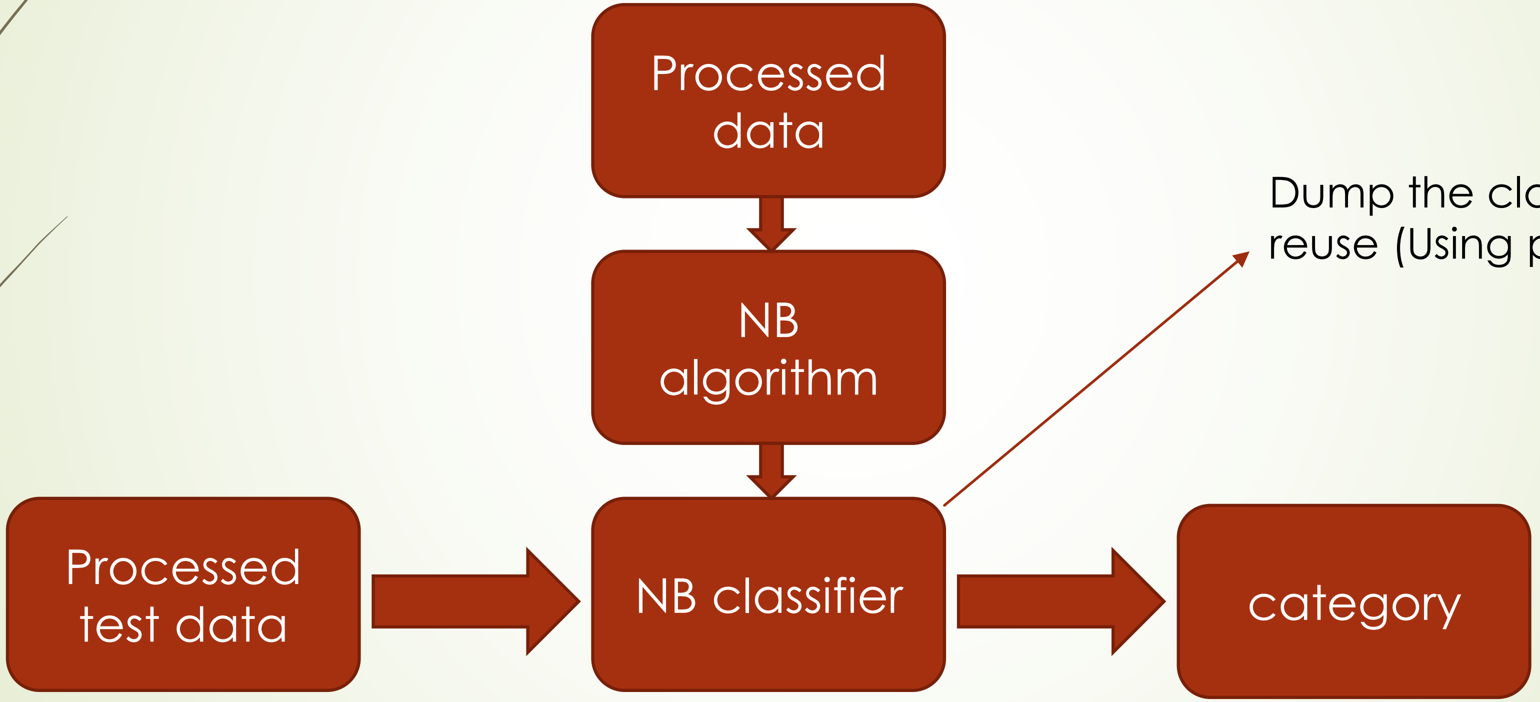
Sqlite database

- Insert into table with attributes id,date,title,link, content etc.

Training

- Supervised learning using naïve Bayes classifier

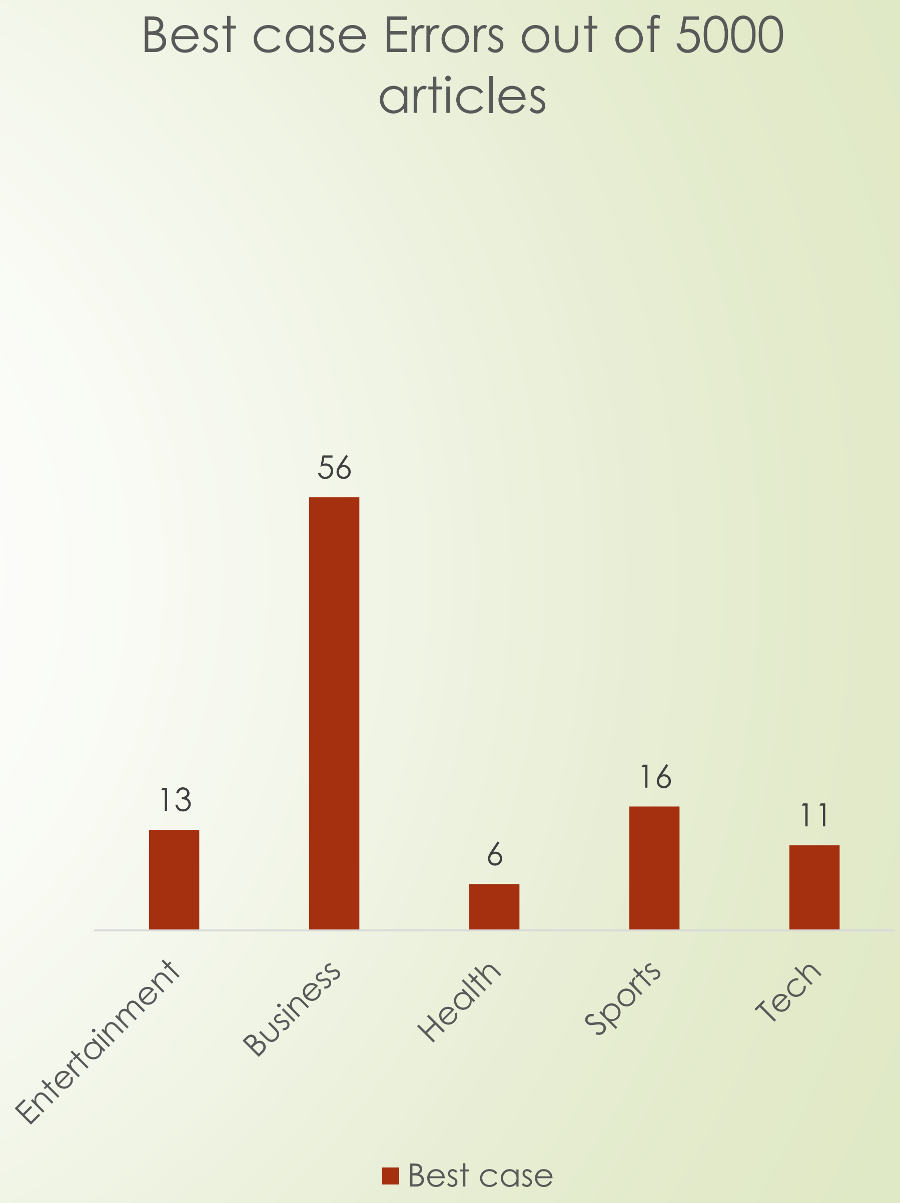
NB classifier works on the Bayes theorem in probability mathematics



Dump the classifier object for easy reuse (Using pickle library)

Analysis

- Random testing
  - Data set size : 5000+
  - Accuracy : 94.6
- Best case test
  - No. of fault classification :
  - Data set size :
  - Accuracy :


Best case Errors out of 5000 articles

Analysis

- Graphs

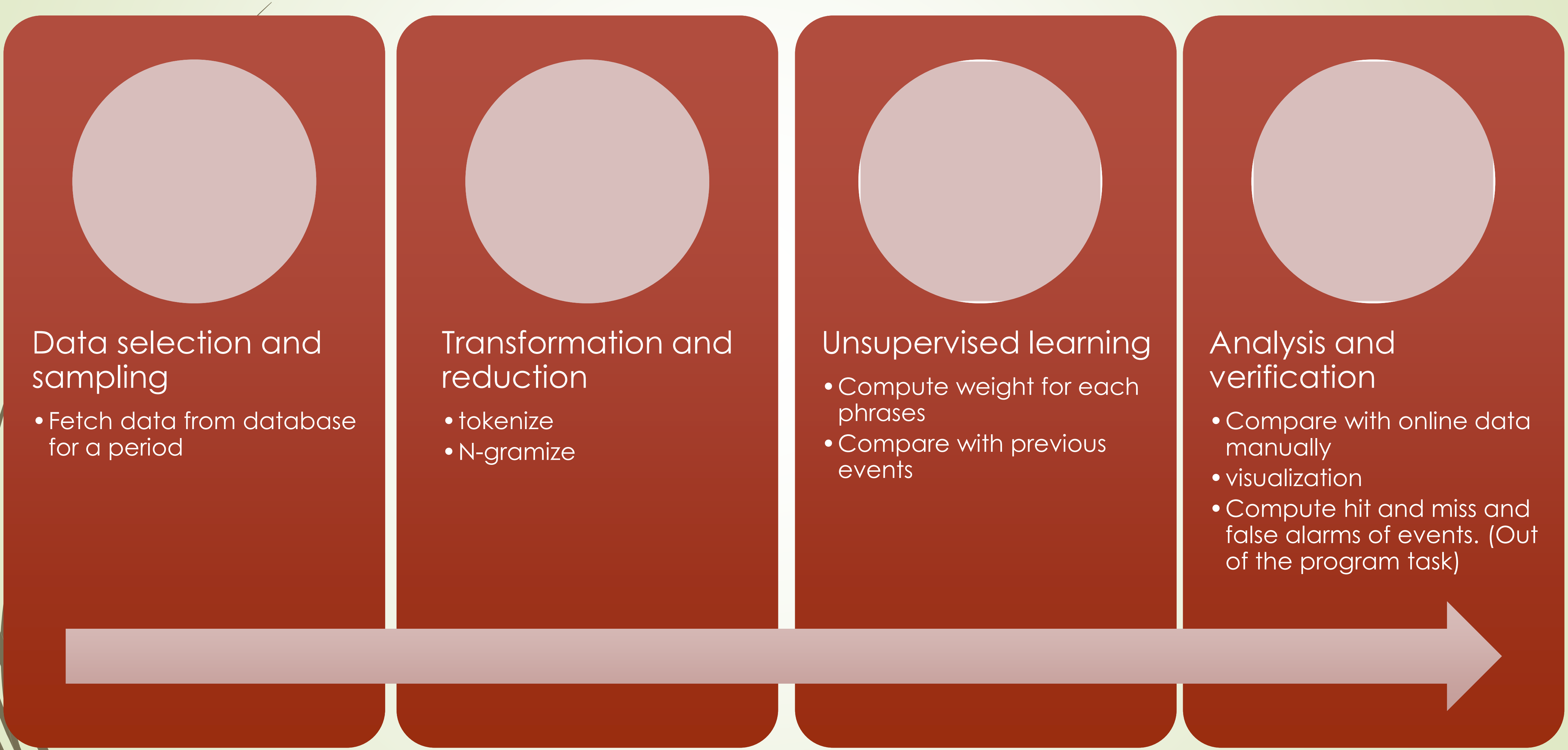
Proposed Enhancements

- Multi-category tagging for documents
- Dynamic categorization during fetching

Event detection

- Event detection is an unsupervised learning task.
- We used ***retrospective event detection*** which identifies previously unidentified events in chronological order.
- Events are those phrases which occur in a peaking frequency for a short period of time.

Event detection Procedure



Data selection and sampling

- Fetch data from database for a period

Transformation and reduction

- tokenize
- N-gramize

Unsupervised learning

- Compute weight for each phrases
- Compare with previous events

Analysis and verification

- Compare with online data manually
- visualization
- Compute hit and miss and false alarms of events. (Out of the program task)

Data transformation

- The data used for learning purpose must be cleansed before the task
- Main processing steps are
  - Tokenize the news data and remove stopwords.

    “Brain cancer detected in the Assam state

    ↓

    { brain, cancer, detected, assam, state }
  - Create unigrams, bigrams, and trigrams and add it to the phrases list (learning vocabulary)

```
{     {brain, cancer, . . . . }
      {brain cancer, cancer detected,  . . . . . }
      {brain cancer detected, cancer detected assam, . . . . . }
}
```

Event detection working

Weight of a term is calculated as,

$$\boldsymbol{w(t,d)} = \frac{\boldsymbol{\left(1+tf(t,d)\right) \times log(N/n_t)}}{\boldsymbol{||\vec{d}||}}$$

- $w(t,d)$ *is the weight of term t in document d*
- $tf(t,d)$ *is the term frequency (TF)* $tf(t,d) = a + (1-a) . \frac{f_{t,d}}{max_{\{t' \in d\}} f_{t',d}}$
- $\log(N/n_t)$ *is the Inverted Document Frequency*
- *N is the size of the training corpus*
- n_t *is the no. of documents containing term t*
- $||\vec{d}||$ = $\sqrt{\sum_t w(t,d)^2}$ is the 2-norm vector

Graphs

Proposed Enhancements

- Clustering and event merging.
- Real-time online reference, comparison, and verification.