Basic Sentiment Analysis

PREPARED BY: AHMAD ALAA ALDINE

What is Sentiment Analysis?

The Core Definition

Sentiment analysis is the subfield of NLP that uses computational methods to identify, extract, and quantify subjective states from unstructured text.

It aims to determine the **attitude** of a speaker toward a specific topic or document.

Subjective vs. Objective

Objective: "The smartphone has 8GB of RAM." (Factual, no sentiment).

Subjective: "8GB of RAM is more than enough for daily tasks." (Judgment-based, positive sentiment).

Levels of Granularity



Document Level

Classifies the sentiment of a whole document as positive or negative.



Sentence Level

Analyzes each sentence to distinguish objective facts from subjective opinions.



Aspect Level

Focuses on specific features (e.g., "The battery life is great but the screen is dim").

The Contextual Hurdle

Sentiment analysis is not just about word lookup. It requires understanding:

- Negations: "Not bad" vs "Not good".
- Sarcasm: "Great, another software update to slow my phone down."
- Domain Nuance: "Small" is positive for a laptop, negative for a hotel room.

Polarity Classification

Dimensions of Polarity:



Binary

The simplest task: classifying text as either **Positive** or **Negative**. Used widely in basic spam vs. ham or buy vs. sell signals.



Fine-Grained

Expanding to multi-class (e.g., 5-star scales). Includes levels like *Very Negative, Negative, Neutral, Positive, Very Positive*.



Intensity

Quantifying the strength of the sentiment. Often mapped to a floating point scale from **[-1.0 to +1.0]** or **[0.0 to 1.0]**.

Polarity Classification

The "Neutral" Problem:

60%

Accuracy Gap

Why Neutral is Hard

In NLP benchmarks, the Neutral class is often the primary source of error. It acts as a "junk drawer" for:

- Objective facts without sentiment.
- Ambiguous subjective statements.
- Mixed sentiment (both pos/neg present).

Real-World Applications

Algorithmic Trading

Scanning financial news and earning calls to predict market volatility based on executive sentiment.

Priority Routing

Automatically detecting frustrated or angry customers in support queues to escalate tickets for immediate help.

Brand Health

Tracking social media mentions in real-time to detect public relation crises before they trend globally.

Core Methodological Paradigms



Rule-Based

Systems that rely on manually crafted linguistic rules and sentiment lexicons (dictionaries) to determine polarity.



Supervised ML

Models trained on large annotated datasets to automatically detect patterns and associations between words and sentiments.

Mechanisms of Rule-Based Systems

LEXICONS, HEURISTICS, AND LOGIC

Lexicon-Based Mechanism

Rule-based systems use a predefined set of linguistic rules and Sentiment Lexicons (dictionaries).

- Tokenize text into words/phrases.
- Lookup scores in a dictionary (e.g., "good": +2, "bad": -2).
- Aggregate scores based on heuristics.
- Adjust for negations and intensity.

The Power of Lexicons

- Curation: Lexicons are lists of words labeled as positive, negative, or neutral.
- Valence: Each word is assigned a "valence" score representing its emotional intensity.
- Scanning: Systems tokenize text and sum scores to provide an initial sentiment estimate.

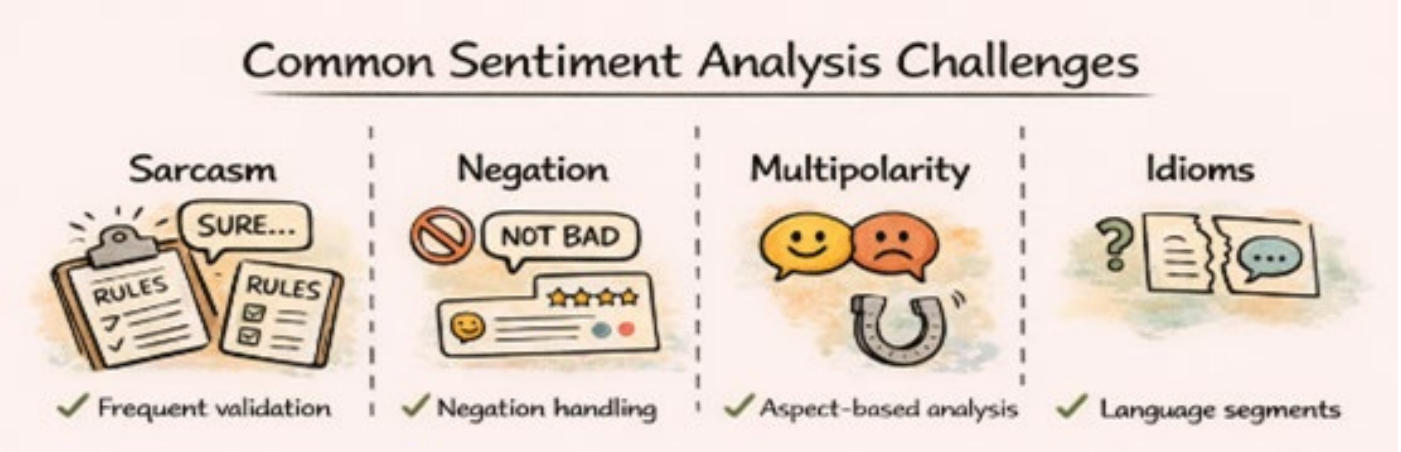
Sentiment Lexicons – Example

Word / Phrase	Sentiment Category	Standard Score
Exceptional / Excellent	Very Positive	+3.0
Good / Pleasant	Positive	+1.5
Neutral / Fine	Neutral	0.0
Bad / Disappointing	Negative	-2.0
Horrific / Abysmal	Very Negative	-4.0

Flipping the Script: Negation

Polarity Reversal

- A simple word count fails when "not" or "never" precedes a sentiment word. Rule-based systems use Valence Shifters to flip scores.
- "Not good" (-1.0)
- "Never fails" (+1.0)



Intensifiers & Dampeners

Degree Modifiers

- Rules apply mathematical weight to adverbs. Intensifiers (boosters) amplify the sentiment score, while dampeners (diminishers) reduce it.

x1.5

"Extremely Happy"

x0.5

"Slightly Happy"

VADER

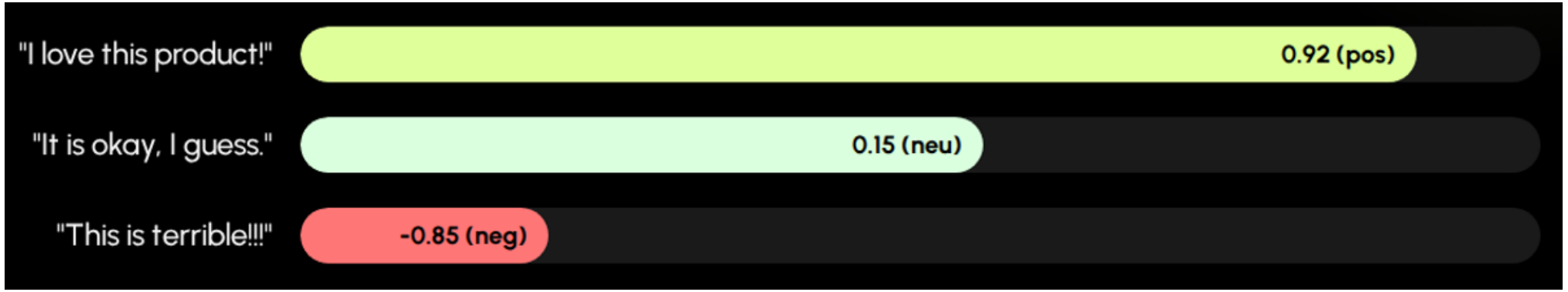
Valence Aware Dictionary and sEntiment Reasoner is a gold standard for short-form text and social media analysis.

Advanced Rules:

- Negation Handling: Flips polarity when words like "not", "never", or "hardly" precede a sentiment word (e.g., "not good").
- Punctuation & Capitalization: Detects emphasis. "GREAT!!!" results in a higher score than "great."
- Degree Modifiers: Handles intensifiers like "extremely" (+ score) or "slightly" (- score).
- Slang & Emojis: Includes scores for common internet tokens like "lol", "meh", and Unicode emojis.

VADER Compound Scoring

The Compound Score is a normalized, weighted composite score summed from the valence of all words in a sentence.



TextBlob: Simplicity First

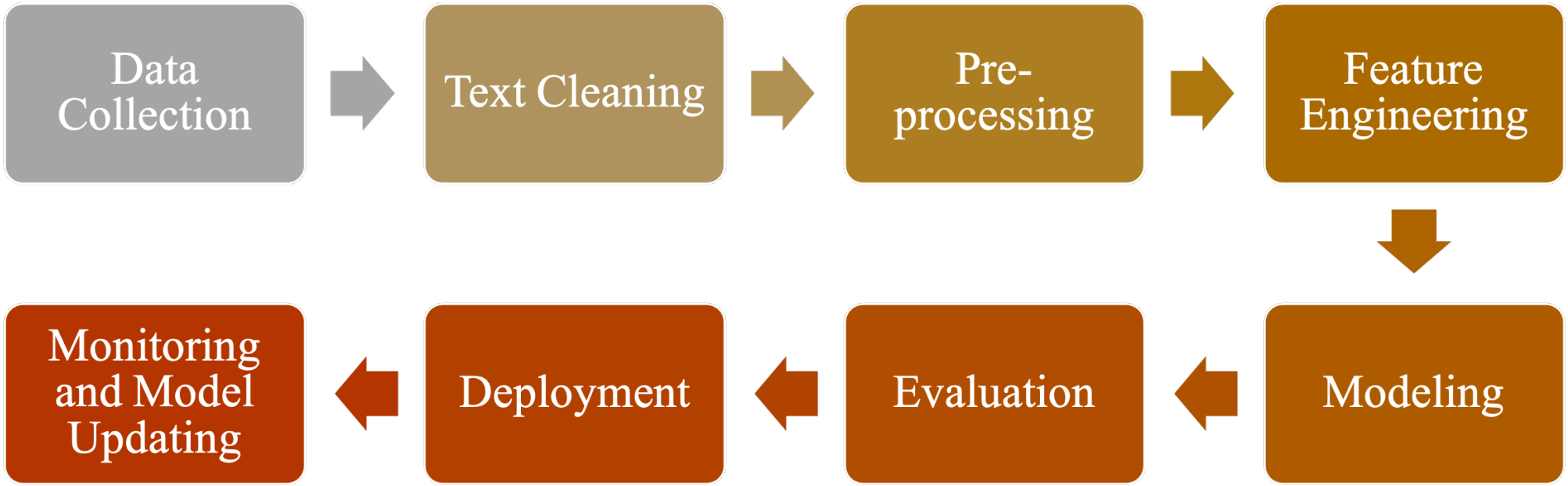
TextBlob provides two distinct scores for every piece of text:

- Polarity [-1, 1]: Measures how positive or negative the text is.
- Subjectivity [0, 1]: Measures how much personal opinion vs. factual info is present.

Feature	VADER	TextBlob
Optimized For	Social Media / Short Text	General NLP
Emoji Support	Full (Integrated)	Requires Pre-processing
Subjectivity	No	Yes
Heuristics	Handles Punctuation/Caps	Uses Averaging Logic

Supervised Sentiment Classification

NLP Pipeline: Supervised Classification



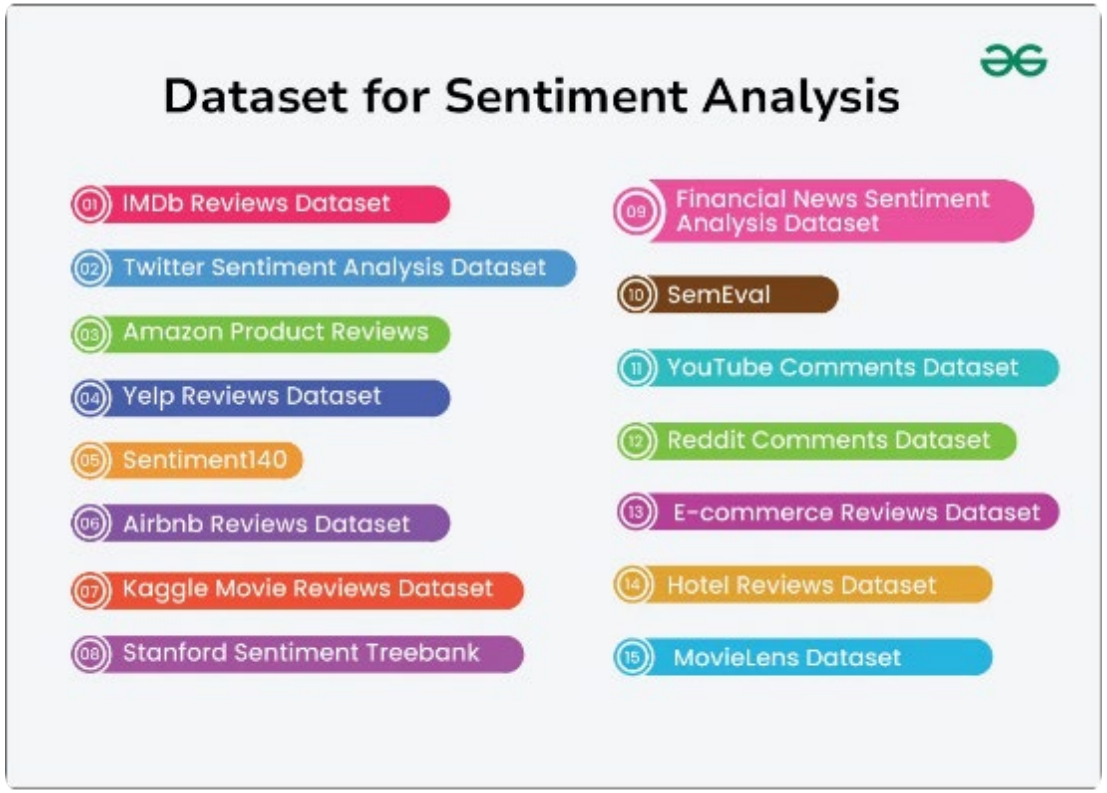
The Foundation: Dataset & Labels

Supervised learning requires a **ground truth** corpus where text is paired with human-annotated sentiment scores.

Input (X): Raw text (reviews, tweets).

Labels (Y): Discrete categories.

Common labels: Binary (Positive/Negative) or Multi-class (Neutral included).



Text Cleaning & Preprocessing

1. Tokenization
2. Lowercasing Removal
3. Punctuation Removal

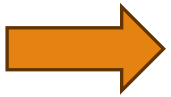
Sample Text	Cleaned Tokens	Sentiment Label
"The movie was absolutely fantastic!"	[movie, absolutely, fantastic]	Positive (1)
"I wasted my money on this product."	[wasted, money, product]	Negative (0)
"It was an average experience."	[average, experience]	Neutral (2)

Feature Engineering: Vectorization

The simplest way to turn text into numbers is to treat each document as a "bag" of its constituent words, ignoring order.

- Bag-of-Words
- Bag-of-Ngrams
- TF-IDF

Text Data	Label
Nice guy	Positive (1)
Bad boy	Negative (0)



Vectors				Label
1	1	0	0	1
0	0	1	1	0

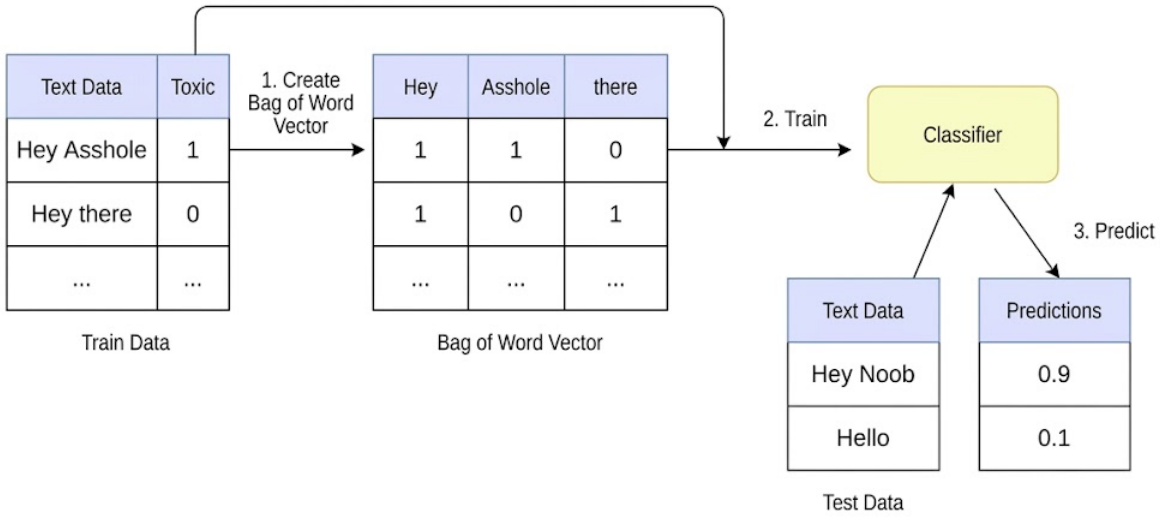
Modeling

Fed Labeled vectors into machine learning algorithm to train a classifier model.

Then the classifier can be used to predict sentiment labels for unseen text data during training.

Example of algorithms:

- Naïve Bayes
- Logistic Regression



Modeling: Naïve Bayes

Probabilistic Classifier

Based on Bayes' Theorem with the "naive" assumption that features are independent given the class.

It calculates the probability of a sentiment given the words in the text. Extremely fast and effective for high-dimensional text data.

$$\text{P(Positive}|\text{Text)} = \frac{P(Text|Positive)*P(Positive)}{P(Text)}$$

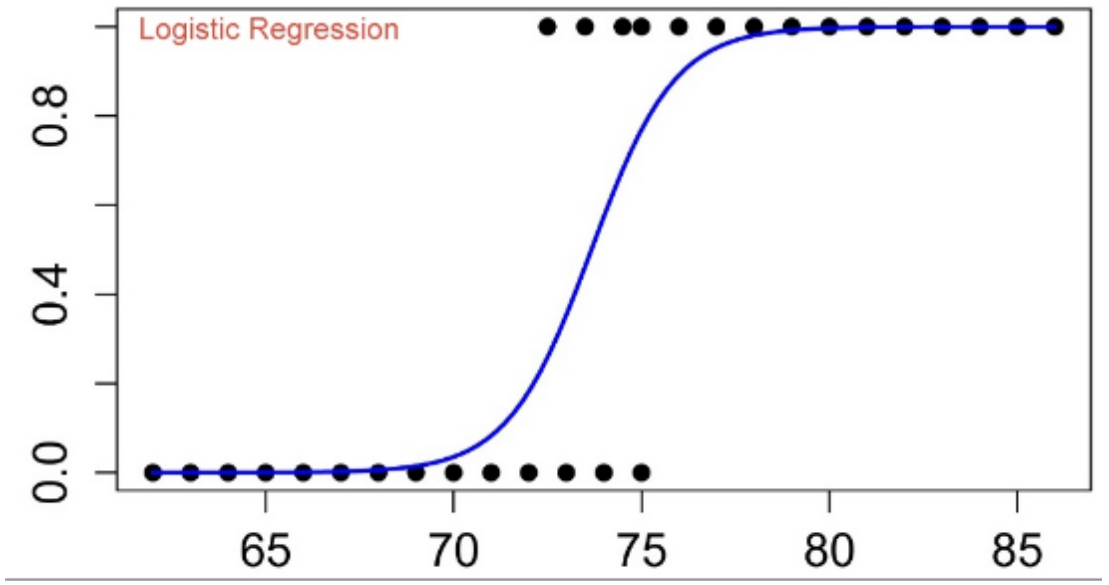
$$\text{P(Negative}|\text{Text)} = \frac{P(Text|Negative)*P(Negative)}{P(Text)}$$

Modeling: Logistic Regression

Modeling Probabilities

Unlike standard regression, Logistic Regression uses the **Sigmoid function** to map real-valued scores to a [0, 1] range.

- Optimizes weights for specific feature words.
- Highly interpretable: positive coefficients indicate "positive" words.
- Often outperforms Naive Bayes on larger datasets.



Evaluation: Model Performance

Evaluating machine learning model performance involves using specific metrics to ensure the model generalizes to new data (testing data) and avoids overfitting.

Key techniques include cross-validation and confusion matrices to, respectively, split data, validate stability, and analyze prediction errors.

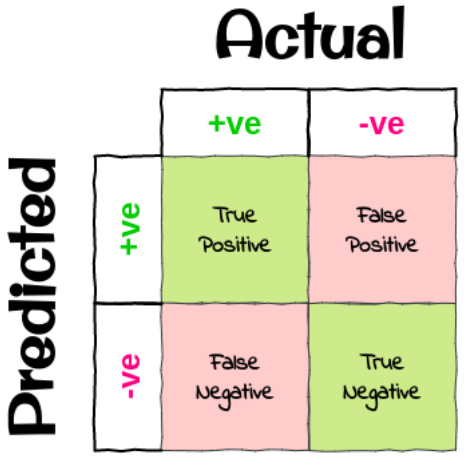


Fig 1: general representation of a confusion matrix

Evaluation Metrics

Accuracy: Overall correct predictions (best for balanced datasets)

Precision: Proportion of positive identifications that were actually correct (use when false positives are costly).

Recall: Proportion of actual positives identified correctly (use when false negatives are costly).

F1-score: Harmonic mean of precision and recall, balancing both.

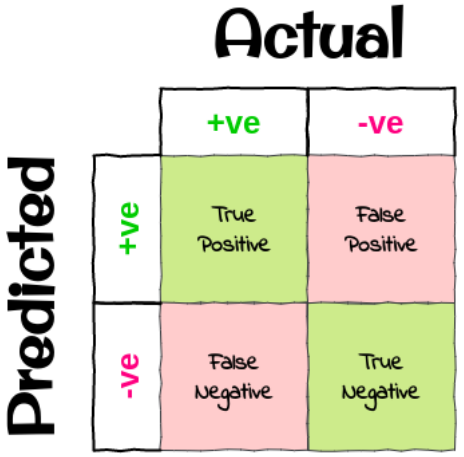


Fig 1: general representation of a confusion matrix

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

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

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

$$F_1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$$

Model Performance Comparison

Illustrative accuracy scores based on typical benchmarks (e.g., IMDB reviews).

NB (BoW)	NB (TF-IDF)	LR (BoW)	LR (TF-IDF)
75%	82%	85%	91%

A Comparative Analysis: Rule-Based vs. Supervised Machine Learning

Strategic Comparison Matrix

Feature	Rule-Based (Lexicon)	Supervised Learning
Data Needs	Minimal (Rules + Lexicons)	High (Labeled Training Data)
Context Aware	Limited (Hard-coded)	High (Learned Context)
Setup Speed	Fast (Plug-and-play)	Slow (Training + Tuning)
Interpretability	High (Transparent Logic)	Low (Black Box Models)
Maintenance	Manual Rule Updates	Re-training with New Data

Rule-Based Analysis Profile

Key Advantages

- Immediate implementation without training.
- Total control over specific business logic.
- Requires zero specialized hardware (GPUs).
- Highly effective for short, clear texts.

Strategic Limitations

- Fails at irony, sarcasm, and nuance.
- Rules become unmanageable at scale.
- Hard to adapt to new linguistic trends.
- Dictionary bias can skew global results.

Supervised ML Analysis Profile

Key Advantages

- Superb handling of complex linguistic context.
- Scales automatically with data volume.
- Handles sarcasm through structural patterns.
- Adapts to new domains via transfer learning.

Strategic Limitations

- "Cold Start" problem: requires data first.
- Expensive data labeling/annotation costs.
- Opaque decision-making process.
- Prone to overfitting on noisy data.

When to Deploy Rule-Based Systems



Quick Prototyping

When you need to ship a proof-of-concept within days and lack a pre-existing annotated dataset.



Edge Computing

For on-device analysis where memory and processing power are extremely constrained (IoT, mobile).



Audit Requirements

In highly regulated industries where every sentiment decision must be manually verifiable and auditable.

When to Deploy Supervised ML



Big Data Scale

Processing millions of social media mentions where manual rule creation is physically impossible.



High Nuance Tasks

Analyzing customer complaints or political discourse where sarcasm and subtle tone are critical.



Dynamic Content

When the language evolves rapidly and models need constant updates.