UNIT III
Naïve Bays and Text Classification

By
Dr. S. S. Gharde
Dept. of Information Technology/ AIML
Government Polytechnic Nagpur

Contents…

Introduction

- **Classification** lies at the heart of both human and machine intelligence.
- **Examples:**
  - Deciding what letter, word, or image has been presented to our senses
  - Recognizing faces or voices
  - Sorting mail
  - Assigning grades to homeworks

- **Classification for**
  - Text categorization
  - Sentiment analysis

Sentiment Analysis
Positive or negative movie review?

\+ *...zany characters and **richly** applied satire, and some **great** plot twists*

*It was **pathetic**. The **worst** part about it was the boxing scenes...*

\− *...**awesome** caramel sauce and sweet toasty almonds. I **love** this place!*

*...**awful** pizza and **ridiculously** overpriced...*

\+

\−

Why sentiment analysis?

- *Movie*: is this review positive or negative?
- *Products*: what do people think about the new iPhone?
- *Public sentiment*: how is consumer confidence?
- *Politics*: what do people think about this candidate or issue?
- *Prediction*: predict election outcomes or market trends from sentiment

Scherer Typology of Affective States

- **Emotion**: brief organically synchronized … evaluation of a major event
  - *angry, sad, joyful, fearful, ashamed, proud, elated*
- **Mood**: diffuse non-caused low-intensity long-duration change in subjective feeling
  - *cheerful, gloomy, irritable, listless, depressed, buoyant*
- **Interpersonal stances**: affective stance toward another person in a specific interaction
  - *friendly, flirtatious, distant, cold, warm, supportive, contemptuous*
- **Attitudes**: enduring, affectively colored beliefs, dispositions towards objects or persons
  - *liking, loving, hating, valuing, desiring*
- **Personality traits**: stable personality dispositions and typical behavior tendencies
  - *nervous, anxious, reckless, morose, hostile, jealous*

Basic Sentiment Classification

- Sentiment analysis is the detection of **attitudes**
- Simple task we focus on in this chapter
  - Is the attitude of this text positive or negative?
- We return to affect classification in later chapters

Summary: Text Classification

- Sentiment analysis
- Spam detection
- Authorship identification
- Language Identification
- Assigning subject categories, topics, or genres
- …

Text Classification: definition

- *Input*:
  - a document d
  - a fixed set of classes $C = \{c_1, c_2, \ldots, c_J\}$
- *Output*: a predicted class $c \in C$

Classification Methods: Supervised Machine Learning

- *Input:*
  - a document *d*
  - a fixed set of classes $C = \{c_1, c_2, \ldots, c_J\}$
  - A training set of *m* hand-labeled documents $(d_1,c_1),\ldots,(d_m,c_m)$
- *Output:*
  - a learned classifier $\gamma:d \rightarrow c$

Classification Methods: Supervised Machine Learning

- Any kind of classifier
  - Naïve Bayes
  - Logistic regression
  - Neural networks
  - k-Nearest Neighbors
  - …

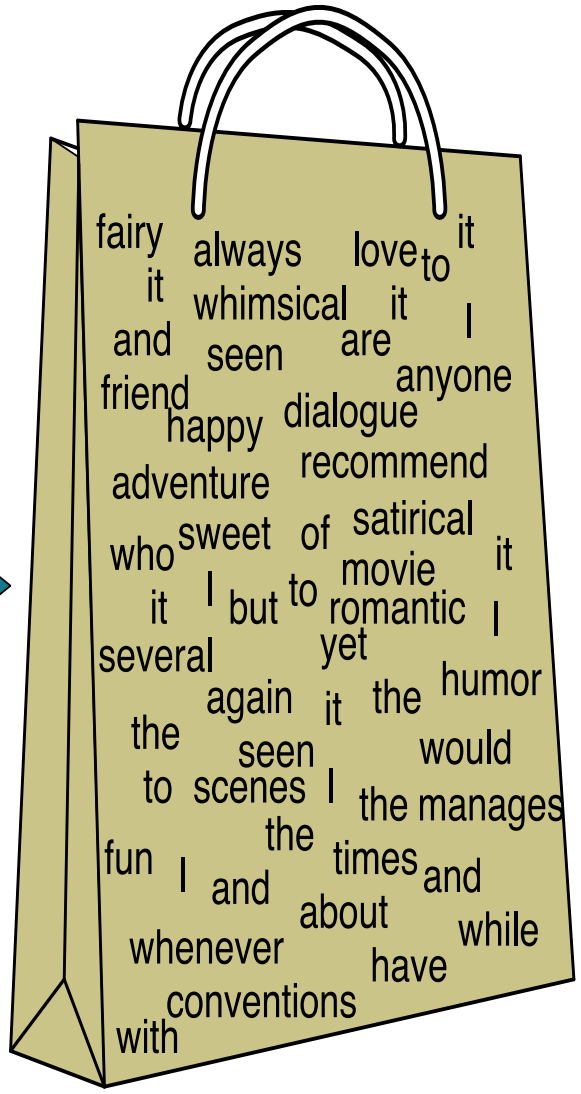
Naive Bayes Classifiers

- Simple ("naive") classification method based on Bayes rule
- Relies on very simple representation of document
  - **Bag of words**

Naive Bayes Classifiers

The Bag of Words Representation

I love this movie! It's sweet, but with satirical humor. The dialogue is great and the adventure scenes are fun... It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it yet!



it	6
I	5
the	4
to	3
and	3
seen	2
yet	1
would	1
whimsical	1
times	1
sweet	1
satirical	1
adventure	1
genre	1
fairy	1
humor	1
have	1
great	1
…	…

Naive Bayes Classifiers

The bag of words representation

$\gamma($

seen	2
sweet	1
whimsical	1
recommend	1
happy	1
…	…

$) = c$

Naive Bayes Classifiers

Bayes' Rule Applied to Documents and Classes

- For a document *d* and a class *c*

$$P(c \mid d) = \frac{P(d \mid c)P(c)}{P(d)}$$

Naive Bayes Classifier (I)

$$c_{MAP} = \underset{c \in C}{\operatorname{argmax}} P(c \mid d)$$

MAP is "maximum a posteriori" = most likely class

$$= \underset{c \in C}{\operatorname{argmax}} \frac{P(d \mid c)P(c)}{P(d)}$$

Bayes Rule

$$= \underset{c \in C}{\operatorname{argmax}} P(d \mid c)P(c)$$

Dropping the denominator

Naive Bayes Classifier (II)

"Likelihood" "Prior"

$$c_{MAP} = \underset{c \in C}{\operatorname{argmax}} P(d \mid c) P(c)$$

$$= \underset{c \in C}{\operatorname{argmax}} P(x_1, x_2, \ldots, x_n \mid c) P(c)$$

Document d represented as features x1..xn

Naive Bayes Classifier

- **Naive Bayes assumption**
- This is the conditional independence assumption that the probabilities P(xi |c) are independent given the class c and hence can be ‘naively’ multiplied as follows:

$$P(x_1,\ldots,x_n \mid c) = P(x_1 \mid c) \bullet P(x_2 \mid c) \bullet P(x_3 \mid c) \bullet \ldots \bullet P(x_n \mid c)$$

Multinomial Naive Bayes Classifier

$$c_{MAP} = \underset{c \in C}{\operatorname{argmax}} P(x_1, x_2, \square, x_n \mid c) P(c)$$

$$c_{NB} = \underset{c \in C}{\operatorname{argmax}} P(c_j) \tilde{\mathrm{O}}_{x \in X} P(x \mid c)$$

Applying Multinomial Naive Bayes Classifiers to Text Classification

positions ← all word positions in test document

$$c_{NB} = \underset{c_j \in C}{\operatorname{argmax}} P(c_j) \prod_{i \in positions} P(x_i \mid c_j)$$

Example

Day	*Outlook*	*Temperature*	*Humidity*	*Wind*	*PlayTennis*
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

$\langle Outlook = sunny, Temperature = cool, Humidity = high, Wind = strong \rangle$

$$P(PlayTennis = yes) = 9/14 = .64$$

$$P(PlayTennis = no) = 5/14 = .36$$

Outlook	Y	N
sunny	2/9	3/5
overcast	4/9	0
rain	3/9	2/5
Tempreature		
hot	2/9	2/5
mild	4/9	2/5
cool	3/9	1/5

Humidity	Y	N
high	3/9	4/5
normal	6/9	1/5
Windy		
Strong	3/9	3/5
Weak	6/9	2/5

Example

No	Color	Type	Origin	Stolen
1	Red	Sports	Domestic	Yes
2	Red	Sports	Domestic	No
3	Red	Sports	Domestic	Yes
4	Yellow	Sports	Domestic	No
5	Yellow	Sports	Imported	Yes
6	Yellow	SUV	Imported	No
7	Yellow	SUV	Imported	Yes
8	Yellow	SUV	Domestic	No
9	Red	SUV	Imported	No
10	Red	Sports	Imported	Yes

$X = \{ \text{Red}, \text{SUV}, \text{Domestic} \}$

$$P(X|Y) = \frac{P(Y|X) \cdot P(X)}{P(Y)}$$

Training Naïve Bayes Classifier

- For the class prior P(c)
- Let Nc be the number of documents in our training data with class c and N_{doc} be the total number of documents. Then:

$$\hat{P}(c\ \ ) = \frac{N_c}{N_{doc}}$$

$$\hat{P}(w_i \mid c_j) = \frac{count(w_i, c_j)}{\sum_{w \in V} count(w, c_j)}$$

- Since naive Bayes naively multiplies all the feature likelihoods together, zero probabilities in the likelihood term for any class will cause the probability of the class to be zero.
- The simplest solution is the add-one (Laplace) smoothing.

Training Naïve Bayes Classifier

- While Laplace smoothing is usually replaced by more sophisticated smoothing algorithms in language modeling, it is commonly used in naive Bayes text categorization:

$$\hat{P}(w_i \mid c) = \frac{count(w_i, c)}{\sum_{w \in V} \left(count(w, c)\right)}$$

$$= \frac{count(w_i, c) + 1}{\left(\sum_{w \in V} count(w, c)\right) + |V|}$$

- The vocabulary V consists of the union of all the word types in all classes

Training Naïve Bayes Classifier

- Ignore unknown words in test data (not in train data)
- Ignore stop words, very frequent words like ‘the’ and ‘a’.
- Compute Prior Probability P(c)
- Compute Conditional prob. i.e. likelihood Estimation (Since naive Bayes naively multiplies all the feature likelihoods together, zero probabilities in the likelihood term for any class will cause the probability of the class to be zero)
- Apply add-one (Laplace) smoothing

Training Naïve Bayes Classifier

function TRAIN NAIVE BAYES(D, C) **returns** log $P(c)$ and log $P(w|c)$

for each class $c \in C$ # Calculate $P(c)$ terms
  N_{doc} = number of documents in D
  N_c = number of documents from D in class c
  $logprior[c] \leftarrow \log \frac{N_c}{N_{doc}}$
  $V \leftarrow$ vocabulary of D
  $bigdoc[c] \leftarrow$ **append**(d) **for** d $\in$ D **with** class c
  for each word w in V # Calculate $P(w|c)$ terms
    $count(w,c) \leftarrow$ # of occurrences of w in $bigdoc[c]$
    $loglikelihood[w,c] \leftarrow \log \frac{count(w,c) + 1}{\sum_{w' \text{ in } V} (count(w',c) + 1)}$
return *logprior*, *loglikelihood*, V

function TEST NAIVE BAYES(*testdoc*, *logprior*, *loglikelihood*, C, V) **returns** best c

for each class $c \in C$
  $sum[c] \leftarrow logprior[c]$
  for each position i in *testdoc*
    $word \leftarrow testdoc[i]$
    if $word \in V$
      $sum[c] \leftarrow sum[c] + loglikelihood[word,c]$
return $\text{argmax}_c\ sum[c]$

Figure 4.2 The naive Bayes algorithm, using add-1 smoothing. To use add-α smoothing instead, change the +1 to +α for loglikelihood counts in training.

Worked example

	Cat	Documents
Training	-	just plain boring
	-	entirely predictable and lacks energy
	-	no surprises and very few laughs
	+	very powerful
	+	the most fun film of the summer
Test	?	predictable with no fun

Worked example

	Cat	Documents
Training	-	just plain boring
	-	entirely predictable and lacks energy
	-	no surprises and very few laughs
	+	very powerful
	+	the most fun film of the summer
Test	?	predictable ~~with~~ no fun

1. Prior from training:

$$\hat{P}(c_j) = \frac{N_{c_j}}{N_{total}}$$

P(-) = 3/5
P(+) = 2/5

2. Drop "with"

3. Likelihoods from training:

$$p(w_i|c) = \frac{count(w_i, c) + 1}{(\sum_{w \in V} count(w, c)) + |V|}$$

$$P(\text{“predictable”}|-) = \frac{1+1}{14+20} \qquad P(\text{“predictable”}|+) = \frac{0+1}{9+20}$$

$$P(\text{“no”}|-) = \frac{1+1}{14+20} \qquad P(\text{“no”}|+) = \frac{0+1}{9+20}$$

$$P(\text{“fun”}|-) = \frac{0+1}{14+20} \qquad P(\text{“fun”}|+) = \frac{1+1}{9+20}$$

4. Scoring the test set:

$$P(-)P(S|-) = \frac{3}{5} \times \frac{2 \times 2 \times 1}{34^3} = 6.1 \times 10^{-5}$$

$$P(+)P(S|+) = \frac{2}{5} \times \frac{1 \times 1 \times 2}{29^3} = 3.2 \times 10^{-5}$$

Naive Bayes for other text classification tasks

- **Spam detection**
- Deciding if a particular piece of email is an example of spam (unsolicited bulk email)—one of the first applications of naive Bayes to text classification
- A common solution here, rather than using all the words as individual features, is to predefine likely sets of words or phrases as features.
- For example the open-source SpamAssassin tool predefines features like the phrase "one hundred percent guaranteed", or the feature mentions millions of dollars.

Naive Bayes for other text classification tasks

- **Spam detection**
  - More sample SpamAssassin features:
  - Email subject line is all capital letters
  - Contains phrases of urgency like “urgent reply”
  - Email subject line contains “online pharmaceutical”
  - HTML has unbalanced “head” tags
  - Claims you can be removed from the list

- **Language ID system**—determining what language a given piece of text is written in.
  - The most effective naive Bayes features are character n-grams or r byte n-grams.
  - A widely used naive Bayes system is **langid.py** which begins with all possible n-grams of lengths 1-4.
  - Language ID systems are trained on multilingual text, such as Wikipedia.

Naive Bayes as a Language Model

- A naive Bayes model can be viewed as a set of class-specific unigram language models, in which the model for each class instantiates a unigram language model.
- The model also assigns a probability to each sentence.

Naive Bayes as a Language Model

Each class = a unigram language model

- Assigning each word: P(word | c)
- Assigning each sentence: P(s|c)=∠P(word|c)

Class +

0.1	I
0.1	love
0.01	this
0.05	fun
0.1	film
…	

I	love	this	fun	film
0.1	0.1	.05	0.01	0.1

P(s | +) = 0.0000005

Naïve Bayes as a Language Model

- Which class assigns the higher probability to s?

P("I love this fun film"|+) = 0.1×0.1×0.01×0.05×0.1 = 0.0000005
P("I love this fun film"|−) = 0.2×0.001×0.01×0.005×0.1 = .0000000010

Model +	
0.1	I
0.1	love
0.01	this
0.05	fun
0.1	film

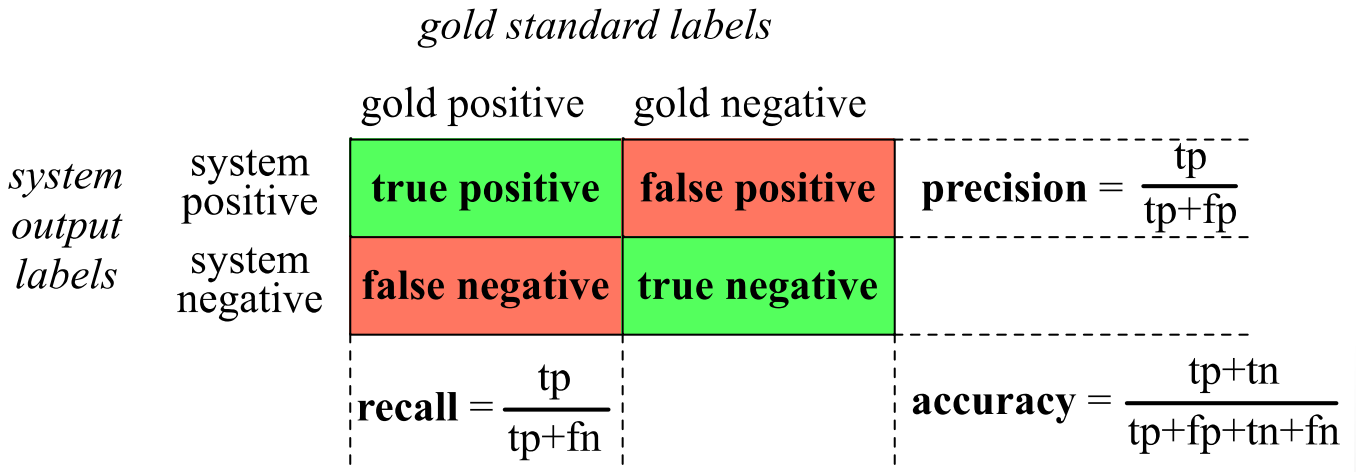
Model -	
0.2	I
0.001	love
0.01	this
0.005	fun
0.1	film

I	love	this	fun	film
0.1	0.1	0.01	0.05	0.1
0.2	0.001	0.01	0.005	0.1

P(s|+) > P(s|-)

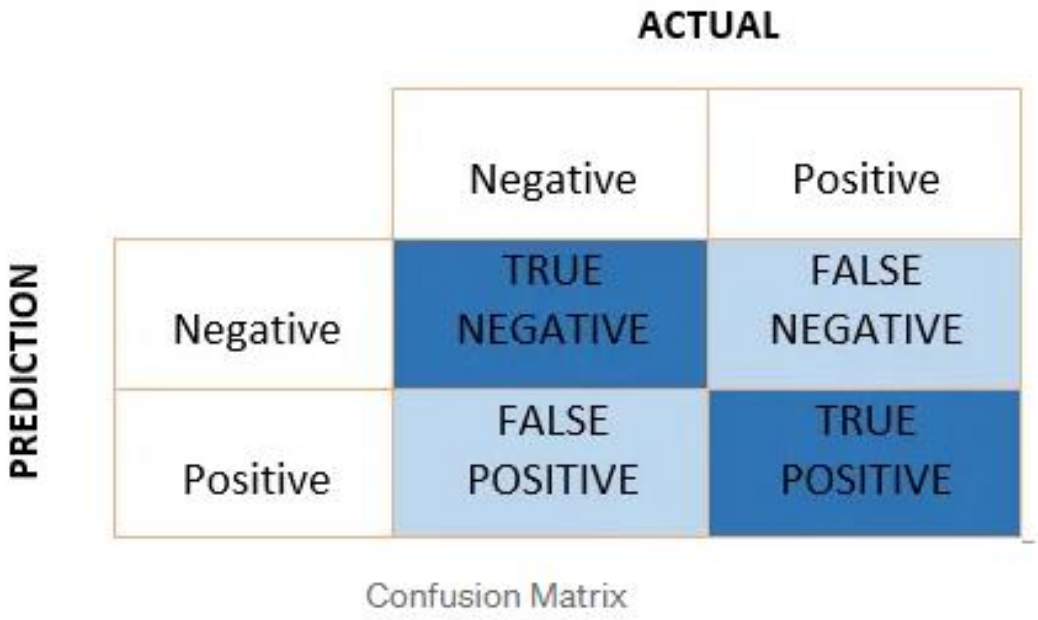
Evaluation: Confusion Matrix

- The methods for evaluating text classification.
- The human-defined labels for each document that we are trying to match are refer as **the gold labels**.
- A **confusion matrix** is a table for visualizing how an algorithm performs with respect to the human gold labels, using two dimensions (system output and gold labels), and each cell labeling a set of possible outcomes.



Evaluation: Confusion Matrix

- **Confusion matrix**
- It is a matrix of size 2×2 for binary classification with actual values on one axis and predicted on another.



Confusion Matrix

Evaluation: Confusion Matrix

• **True Negative** • **(Predicted Negative and Actual Negative)**	False Positive **(Predicted Positive and Actual negative)**
False Negative **(Predicted Negative and Actual Positive)**	**True Positive** (Predicted Positive and Actual Positive)

Figure 5.10: Confusion Matrix

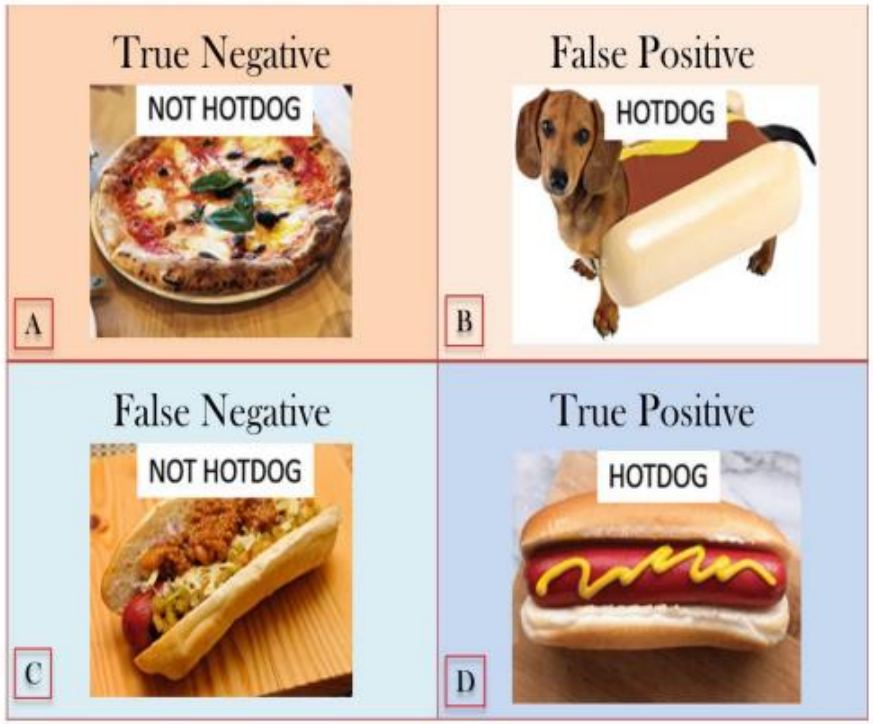


Figure 5.11: Confusion Matrix Example

Evaluation: Confusion Matrix

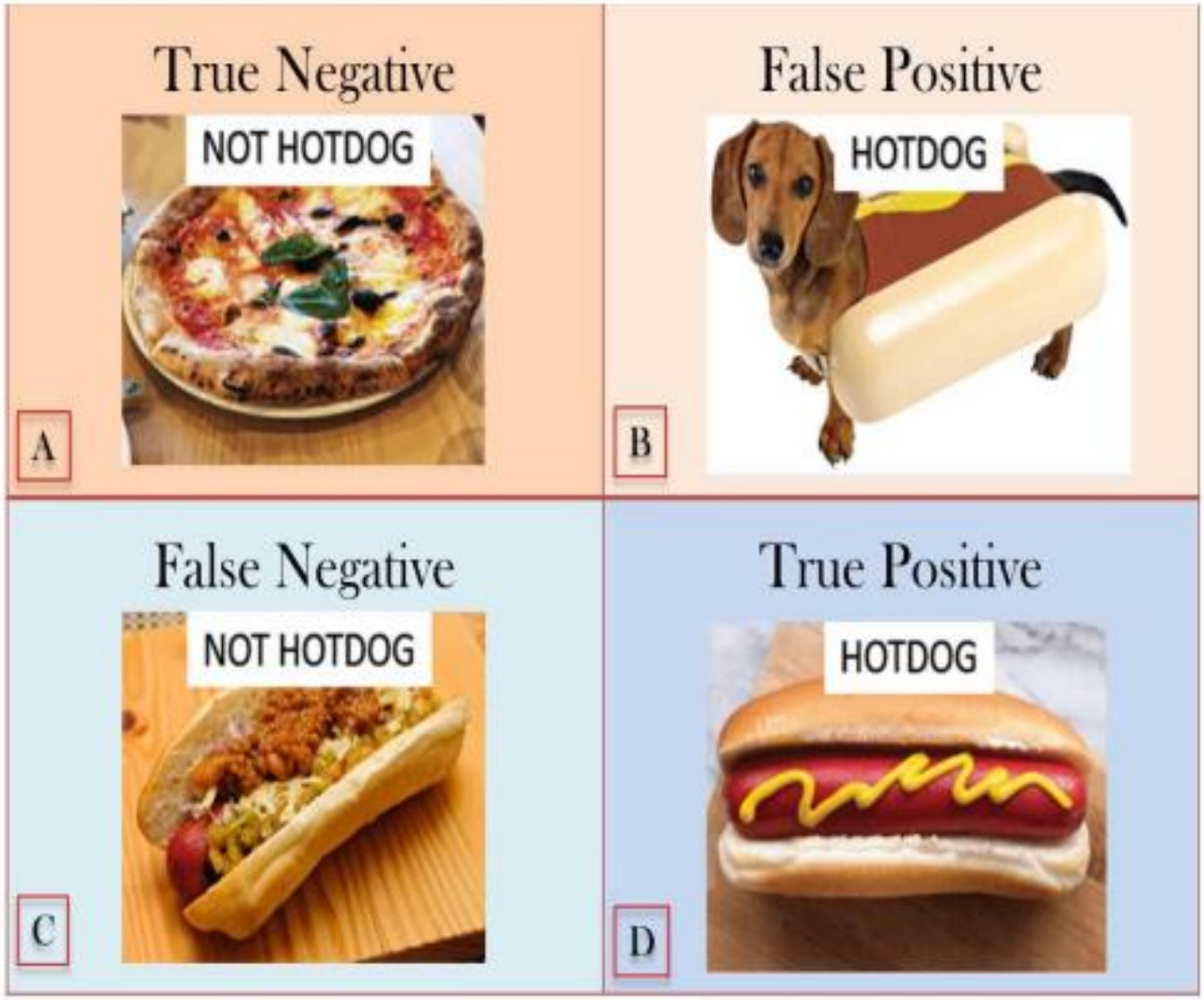


Figure 5.11: Confusion Matrix Example

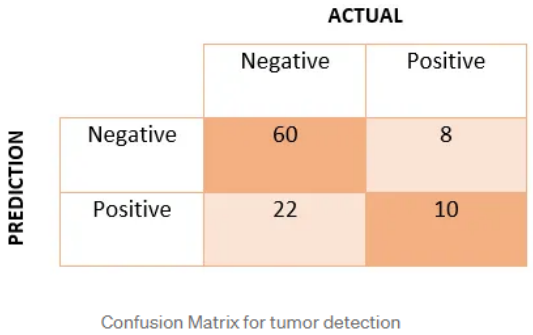
Evaluation: Confusion Matrix

- Let's understand the confusing terms in the confusion matrix: **true positive, true negative, false negative, and false positive** with an example.
- A machine learning model is trained to predict tumor in patients. The test dataset consists of 100 people.

		ACTUAL	
		Negative	Positive
PREDICTION	Negative	60	8
	Positive	22	10

Confusion Matrix for tumor detection

Evaluation: Confusion Matrix



Confusion Matrix for tumor detection

- **True Positive (TP)** — model correctly predicts the positive class (prediction and actual both are positive). *In the above example, **10 people** who have tumors are predicted positively by the model.*

- **True Negative (TN)** — model correctly predicts the negative class (prediction and actual both are negative). *In the above example, **60 people** who don't have tumors are predicted negatively by the model.*

- **False Positive (FP)** — model gives the wrong prediction of the negative class (predicted-positive, actual-negative). *In the above example, **22 people** are predicted as positive of having a tumor, although they don't have a tumor.* FP is also called a **TYPE I** error.

- **False Negative (FN)** — model wrongly predicts the positive class (predicted-negative, actual-positive). *In the above example, **8 people** who have tumors are predicted as negative.* FN is also called a **TYPE II** error.

Evaluation: Accuracy

- **Accuracy** represents the number of correctly classified data instances over the total number of data instances.

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

- Accuracy may not be a good measure if the dataset is not balanced.
- Instead of accuracy, precision and recall are preferred.

Evaluation: Precision

- **Precision** measures the percentage of items the system detected (i.e., items the system labeled as positive) that are in fact positive (according to the human gold labels).
- **Precision** is defined as

$$\textbf{Precision} = \frac{\text{true positives}}{\text{true positives} + \text{false positives}}$$

Evaluation: Recall

- **Recall** measures the percentage of items actually present in the input that were correctly identified by the system.
- **Recall** is defined as

$$\textbf{Recall} = \frac{\text{true positives}}{\text{true positives} + \text{false negatives}}$$

- There are many ways to define a single metric that incorporates aspects of both precision and recall. The simplest of these **combinations is the F-measure.**

Evaluation: F measure

- F measure: a single number that combines P and R:

$$F_\beta = \frac{(\beta^2 + 1)PR}{\beta^2 P + R}$$

- The β parameter differentially weights the importance of recall and precision, based perhaps on the needs of an application.
- Values of β > 1 favor recall, while values of β < 1 favor precision.
- We almost always use balanced F_1 (i.e., $\beta = 1$)

$$F_1 = \frac{2PR}{P + R}$$

Evaluation: F measure

- ***F1-score*** is a metric which takes into account both ***precision*** and ***recall*** and is defined as follows:

$$F_1 = \frac{2PR}{P + R}$$

- ***F1 Score*** becomes 1 only when ***precision*** and ***recall*** are both 1.
- ***F1 score*** becomes high only when both ***precision*** and ***recall*** are high.
- ***F1 score*** is the harmonic mean of ***precision*** and ***recall*** and is a better measure than ***accuracy***.

Test sets and Cross-validation

- We use the training set to train the model, then use the development test set (also called a devset) to perhaps tune some parameters, and decide what the best model is. Run it on the test set to report its performance.
- **Cross-validation:** if we use all our data for training and also use all our data for testing. We can do this by cross-validation.
- In cross-validation, we choose a number k, and partition our data into k disjoint subsets called **folds.**
- Now we choose one of those k folds as a test set, train our classifier on the remaining k − 1 folds, and then compute the error rate on the test set. Then we repeat with another fold as the test set.
- If we choose k = 10, we would train 10 different models (each on 90% of our data), test the model 10 times, and average these 10 values. This is called 10-fold cross-validation.

Statistical Significance Testing

- need to compare the performance of two systems.
- In the paradigm of statistical hypothesis testing, we perform test by formalizing two hypotheses.
- $H0 : \delta(x) \leq 0$ $H1 : \delta(x) > 0$
- The hypothesis H0, called the null hypothesis, supposes that $\delta(x)$ is actually negative or zero, meaning that A is not better than B.
- We would like to know if we can confidently rule out this hypothesis, and instead support H1, that A is better.
- if the null hypothesis H0 was correct, formalize this likelihood as the p-value: the probability, assuming the null hypothesis H0 is true, of seeing the $\delta(x)$ that we saw or one even greater
- $$P(\delta(X) \geq \delta(x) | H0 \text{ is true})$$

Statistical Significance Testing

- A very small p-value means that the difference we observed is very unlikely under the null hypothesis, and we can reject the null hypothesis. A value of .01 means that if the p-value is less than .01, we reject the null hypothesis and assume that A is indeed better than B.
- We say that a result (e.g., "A is better than B") is statistically significant if the δ has a probability that is below the threshold and we therefore reject this null hypothesis.
- To compute p-value in NLP we usually use non-parametric tests based on sampling.
- There are two common non-parametric tests used in NLP: approximate randomization and the bootstrap test

Avoiding Harms in Classification

- It is important to avoid harms that may result from classifiers,
- One class of harms is representational harms.

- Harms caused by a system that demeans a social group, for example by perpetuating negative stereotypes about them.

- In other tasks classifiers may lead to both representational harms and other harms, such as censorship.

- For example the important text classification task of toxicity detection is the task of detecting hate speech, abuse, harassment, or other kinds of toxic language.

End of UNIT III