

# CLASSIFICATION

**JELENA JOVANOVIĆ**

Email: [jeljov@gmail.com](mailto:jeljov@gmail.com)

Web: <http://jelenajovanovic.net>

# OUTLINE

- What is classification?
- Binary and multiclass classification
- Classification algorithms
- Naïve Bayes (NB) algorithm
  - Bayes rule
  - Text classification with the NB algorithm
  - An example in WEKA
- Performance measures for classification algorithms

# WHAT IS CLASSIFICATION?

- A supervised learning task of determining the class of an instance; it is assumed that:
  - feature values for the given instance are known
  - the set of possible classes is known and given
- Classes are given as nominal values; for instance:
  - classification of email messages: spam, not-spam
  - classification of news articles: politics, sport, culture i sl.

# BINARY AND MULTICLASS CLASSIFICATION

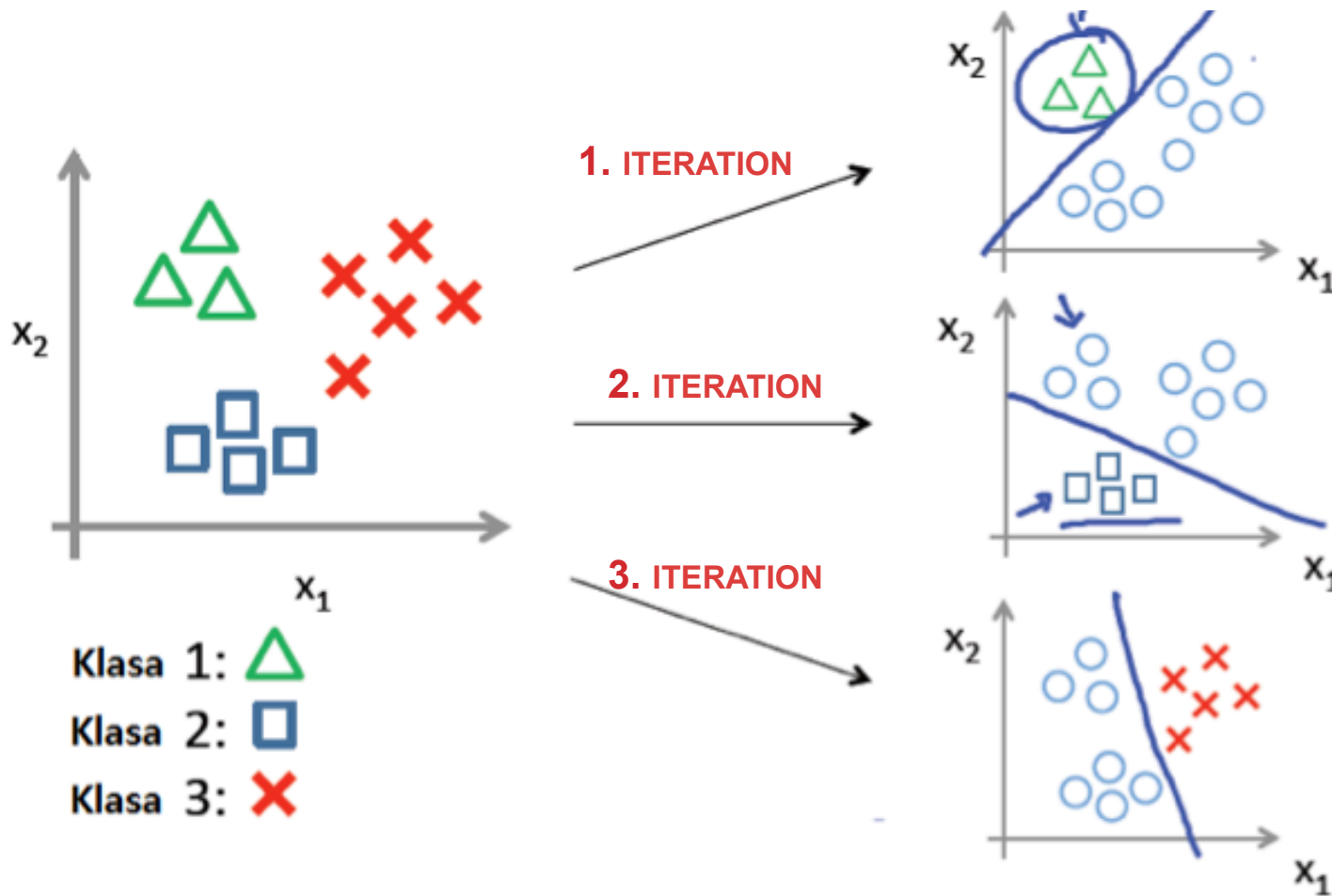
Based on the number of classes, classification can be:

- *binary* – instances should be classified into 2 classes
- *multiclass* – more than 2 classes are used for classifying instances

In both cases, a classification algorithm works in a rather similar manner:

In multiclass classification, an algorithm learns iteratively, so that in each iteration, it learns to differentiate instances of one class from all the other instances

# MULTICLASS CLASSIFICATION



# CLASSIFICATION ALGORITHMS

There are numerous classification models/algorithms:

- Logistic regression
- Naïve Bayes
- Algorithms from the Decision trees family
- Algorithms from the Neural networks family
- k-Nearest Neighbor (kNN)
- Support Vector Machines (SVN)
- ...

# NAÏVE BAYES

# WHY NAÏVE BAYES?

Naïve Bayes (NB) is often cited as an algorithm that is among the first to be considered for any classification task

Rationale:

- Simplicity
- Good performance
- High scalability
- Adaptable to almost any kind of classification task



# TO RECALL: BAYES RULE

$$P(H|E) = P(E|H) * P(H) / P(E)$$

- H – hypothesis
- E – evidence related to the hypothesis H, i.e., the data to be used for validating (accepting/rejecting) the hypothesis H
- P (H) – probability of the hypothesis (*prior probability*)
- P (E) – probability of the evidence i.e., the state of the world described by the gathered data
- P (E | H) – (conditional) probability of evidence E given that the hypothesis H holds
- P (H | E) – (conditional) probability of the hypothesis H given the evidence E

# BAYES RULE – AN EXAMPLE

Let us suppose the following:

- one morning, you wake up with a high temperature
- the previous day, you heard that some virus infection has started spreading through the city, though the infection rate is still rather low, namely 2.5%
- you've also heard that in 50% of cases, the virus goes with a high temperature
- you typically have a high temperature only a couple of times over a year, so the probability that you have a high temp. is 0.065

Question: what is the probability that, since you have a high temperature, you've caught the virus?

# BAYES RULE

Theory	Example
Hypothesis (H)	One has caught a virus infection
P(H)	0.025
Evidence (E)	One has a high temperature
P(E)	0.065
(conditional) probability of E given H P(E H)	Probability that the virus infection causes high temperature 0.50
(conditional) probability of H given E: P(H E)	Probability that given one has a high temperature, he/she also has the virus ?

$$P(H|E) = P(E|H) * P(H) / P(E)$$

$$P(H|E) = 0.50 * 0.025 / 0.065 = 0.19$$

# NAÏVE BAYES IN TEXT CLASSIFICATION

NB is one of the most frequently used algorithms for text classification

Text classification task: for the given text, determine its class, based on the given (annotated) training set

Examples:

- topical classification of news articles, or
- classification of tweets based on the expressed opinion

Features (attributes) the algorithm relies upon are words from the given text

- the text to be classified is represented as a simple bag-of-words

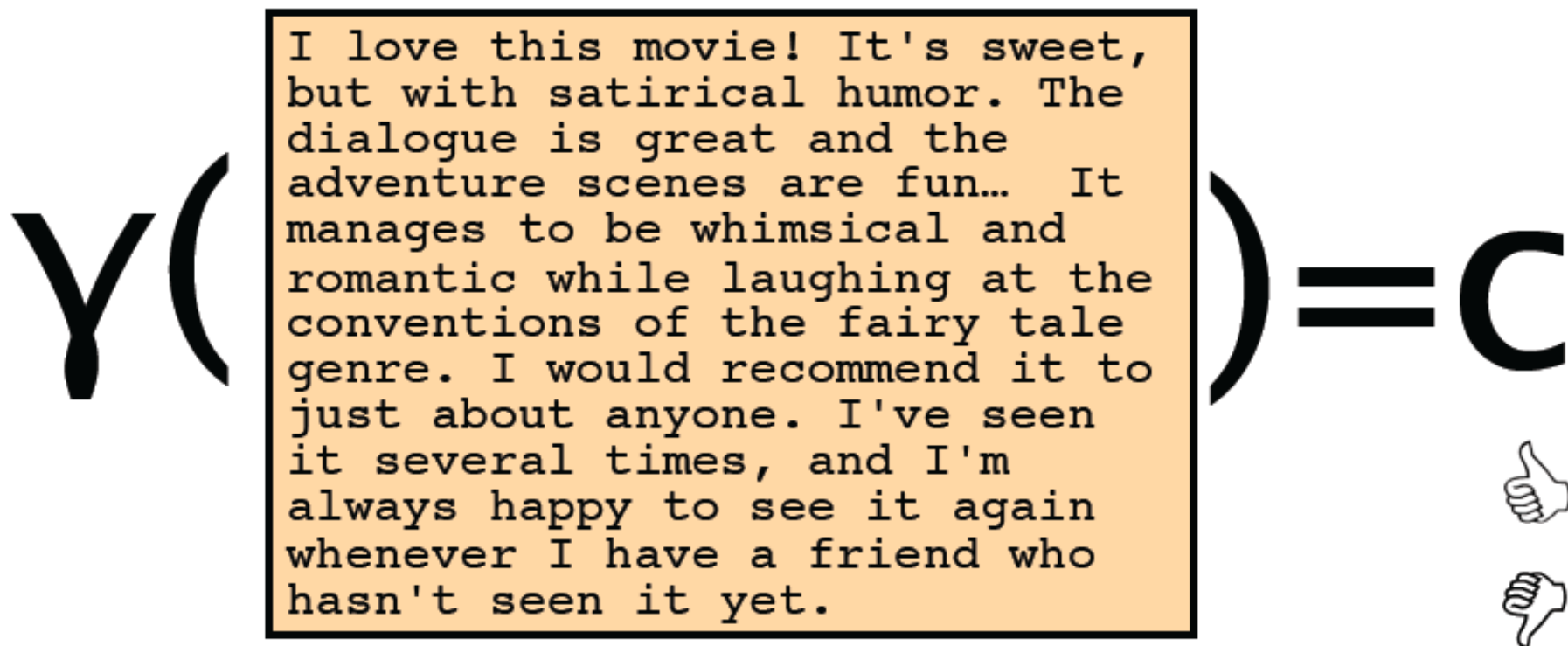
# FEATURE VECTOR FORMATION

There are different approaches for defining a feature vector for the given piece of text

The approach that is often cited as the best one:

- Create a *Dictionary* by extracting words from documents that form the training set (*Corpus*);
- For each document  $\mathbf{d}$  from the *Corpus*, define a feature vector by using words from  $\mathbf{d}$ :
  - For each word  $w_i$  from document  $\mathbf{d}$ , introduce a feature  $x_i$  with the value equal to the index of the word  $w_i$  in the *Dictionary*;
  - Features can be created for all the words from the document  $\mathbf{d}$  or only for those words that are considered relevant for the given classification task

# FEATURE VECTOR FORMATION – AN EXAMPLE



# FEATURE VECTOR FORMATION – AN EXAMPLE

Y (

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.

) = C





# FEATURE VECTOR FORMATION – AN EXAMPLE

**Y** (

words ( $w_i$ )	features ( $x_i$ )
$w_1 = \text{love}$	$x_1 = 04567$
$w_2 = \text{sweet}$	$x_2 = 14321$
$w_3 = \text{satirical}$	$x_3 = 14007$
...	
$w_{11} = \text{happy}$	$x_{11} = 02364$
$w_{12} = \text{again}$	$x_{12} = 00012$

) = **C**

Index of the word  $w_i$  in the *Dictionary*



# NB IN TEXT CLASSIFICATION

If there is a class  $\mathbf{c}$  and a document  $\mathbf{d}$ , the probability that  $\mathbf{c}$  is the class of the document  $\mathbf{d}$  is given as:

$$P(\mathbf{c}|\mathbf{d}) = P(\mathbf{d}|\mathbf{c}) * P(\mathbf{c}) / P(\mathbf{d}) \quad (1)$$

For the given set of classes  $\mathbf{C}$  and the document  $\mathbf{d}$ , we want to find the class  $\mathbf{c}$ , from the set  $\mathbf{C}$ , with the highest conditional probability for the document  $\mathbf{d}$ ; this leads to the function:

$$f = \operatorname{argmax}_{\mathbf{c} \in \mathbf{C}} P(\mathbf{c}|\mathbf{d}) \quad (2)$$

By applying the Bayes rule, we get:

$$f = \operatorname{argmax}_{\mathbf{c} \in \mathbf{C}} P(\mathbf{d}|\mathbf{c}) * P(\mathbf{c}) \quad (3)$$

# NB IN TEXT CLASSIFICATION

$$f = \operatorname{argmax}_{c \in \mathcal{C}} P(d|c) * P(c) \quad (3)$$

Now, we need to *estimate* the probabilities  $P(c)$  and  $P(d|c)$

$P(c)$  can be computed rather easily: by counting the number of occurrences of the class  $c$  in the training set (*Corpus*)

$P(d|c)$  – probability that in the class  $c$  one “finds” the document  $d$  – not that easy to determine, so we introduce assumptions for which this algorithm got the epithet “naïve”

# NB IN TEXT CLASSIFICATION

How do we determine  $P(d|c)$ ?

- we represent document  $d$  as a feature vector  $(x_1, x_2, \dots, x_n)$
- so, instead of  $P(d|c)$ , we'll have  $P(x_1, x_2, x_3, \dots, x_n|c)$
- to compute  $P(x_1, x_2, x_3, \dots, x_n|c)$ , we introduce 2 naïve assumptions:
  - document  $d$  is treated as a simple bag-of-words; i.e., the position and order of words in the text are considered unimportant
  - the presence of a certain word in the given class  $c$  is independent of the presence of any other word in the same class

# NB IN TEXT CLASSIFICATION

The introduced assumptions

- lead to a significant loss of information that could have been derived from the data, *but*,
- simplify the computation of  $P(x_1, x_2, \dots, x_n | c)$ , and thus simplify the overall classification task

# NB IN TEXT CLASSIFICATION

Based on the introduced assumptions,  $P(x_1, x_2, \dots, x_n | c)$  can be represented as a product of individual conditional probabilities

$$P(x_1, x_2, \dots, x_n | c) = P(x_1 | c) * P(x_2 | c) * \dots * P(x_n | c)$$

Thus, we arrive to the general equation of the NB algorithm:

$$f = \operatorname{argmax}_{c \in \mathcal{C}} P(c) * \prod_{i=1, n} P(x_i | c)$$

# NB IN TEXT CLASSIFICATION

The probabilities are estimated on the training set, based on the following equations:

$P(c) = \# \text{ docs of the class } c / \text{ total } \# \text{ docs in the training set}$

$P(x_i | c) = \# \text{ occurrences of the word } w_i \text{ in docs of the class } c / \text{ total } \# \text{ words from the Dictionary that are of the class } c$

# means “number of”

APPLYING THE NB ALGORITHM FOR  
TEXT CLASSIFICATION  
USING THE WEKA FRAMEWORK:  
AN EXAMPLE

# PRIMER PRIMENE NB ALGORITMA KORIŠĆENJEM WEKA FRAMEWORK-A

Primer je preuzet iz GitHub projekta TMWeka:

<https://github.com/jmgomezh/tmweka>

i raspoloživ je na sledećoj adresi:

<https://github.com/jmgomezh/tmweka/tree/master/FilteredClassifier>

U okviru TMWeka projekta, ima jos nekoliko interesantnih primera klasifikacije teksta primenom ML algoritama



# CHARACTERISTICS OF THE NB ALGORITHM

- Very fast and efficient
- Often produces good results
  - often turns out to be better or at least equally good as other, more sophisticated algorithms
- Does not require much memory
- Has low affinity for over-fitting
- Suitable when we do not have much training data

# CHARACTERISTICS OF THE NB ALGORITHM

- “Resistant” to the low-importance attributes
  - attributes that are equally distributed through the overall training set, and thus do not have significant impact on the class label
- Primarily suitable for use with nominal attributes; in the case of numerical attributes
  - Discretize the attribute values, or
  - Use probability distribution of the attributes (typically, Normal dist.) to estimate the probability of each attribute value

# PERFORMANCE MEASURES

The most frequently used metrics:

- Confusion Matrix
- Accuracy
- Precision and Recall
- F measure
- Area Under the ROC Curve

# CONFUSION MATRIX

Serves as the basis for calculating other performance measures

		Predicted Class	
		Yes	No
Actual Class	Yes	TP	FN
	No	FP	TN

TP = True Positive

FP = False Positive

TN = True Negative

FN = False Negative

# ACCURACY

Accuracy is the percentage of correctly classified instances

$$\text{Accuracy} = (TP + TN) / N$$

where:

- TP – True Positive; TN – True Negative
- N – the total number of instances in the dataset

		Predicted Class	
		Yes	No
Actual Class	Yes	TP	FN
	No	FP	TN

# ACCURACY

In the case of highly unequal distribution of instances across classes (so called skewed classes), this measure is unreliable

An example:

- in the case of message classification as spam vs. not-spam, the training set might contain 0.5% of spam messages
- if we apply a biased classifier that classifies each message as not-spam, we get very high accuracy – 99.5%
- obviously, this metric is unreliable and in the case of skewed classes, other metrics are needed

# PRECISION AND RECALL

**Precision** =  $TP / \# \text{ predicted positive} = TP / (TP + FP)$

Example: out of all the messages *marked as spam*, the percentage of those that are *really spam* messages

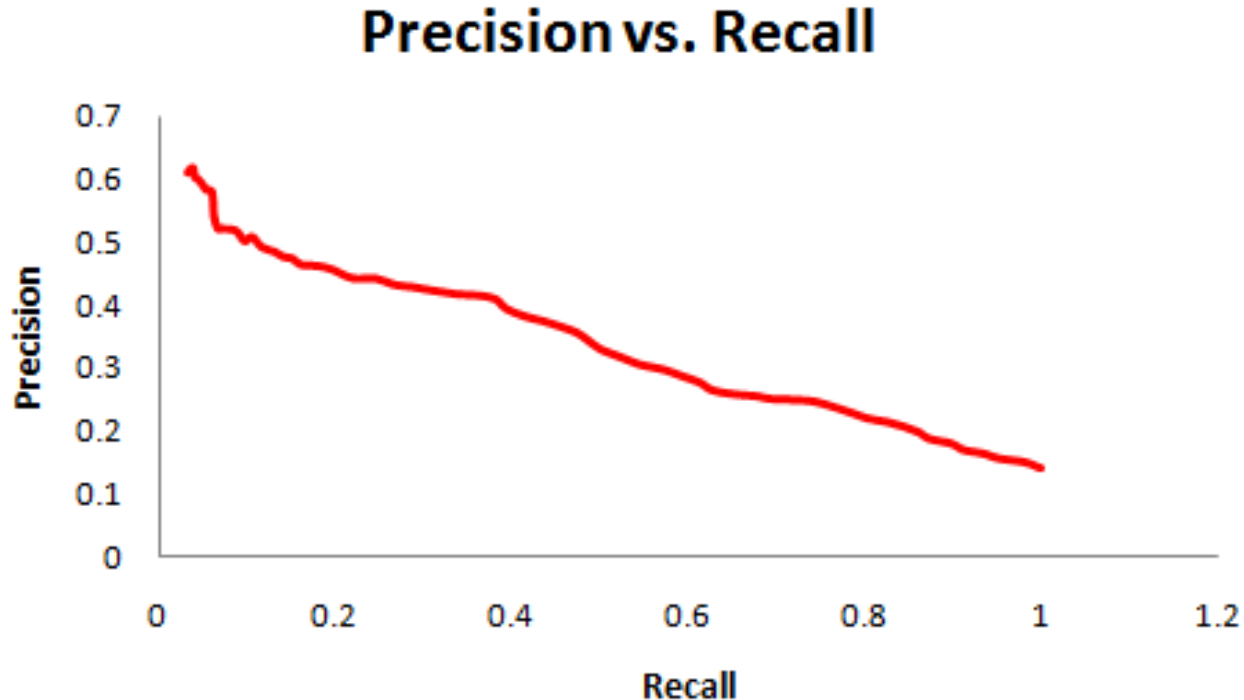
**Recall** =  $TP / \# \text{ actual positive} = TP / (TP + FN)$

Example: out of all the messages that are *really spam*, the percentage of those that have been *detected/classified as spam*

		Predicted Class	
		Yes	No
Actual Class	Yes	TP	FN
	No	FP	TN

# PRECISION VS. RECALL

In practice, one always needs to make a compromise between these two metrics: by increasing Recall, we decrease (though unwillingly) Precision, and vice versa





# F MEASURE

F measure combines Precision and Recall and allows for easier comparison of two or more algorithms

$$F = (1 + \beta^2) * \text{Precision} * \text{Recall} / (\beta^2 * \text{Precision} + \text{Recall})$$

Parameter  $\beta$  controls the extent to which we want to favor Recall over Precision

In practice, F1 measure is typically used; it is called “balanced” F measure as it equally weights Precision and Recall:

$$F1 = 2 * \text{Precision} * \text{Recall} / (\text{Precision} + \text{Recall})$$

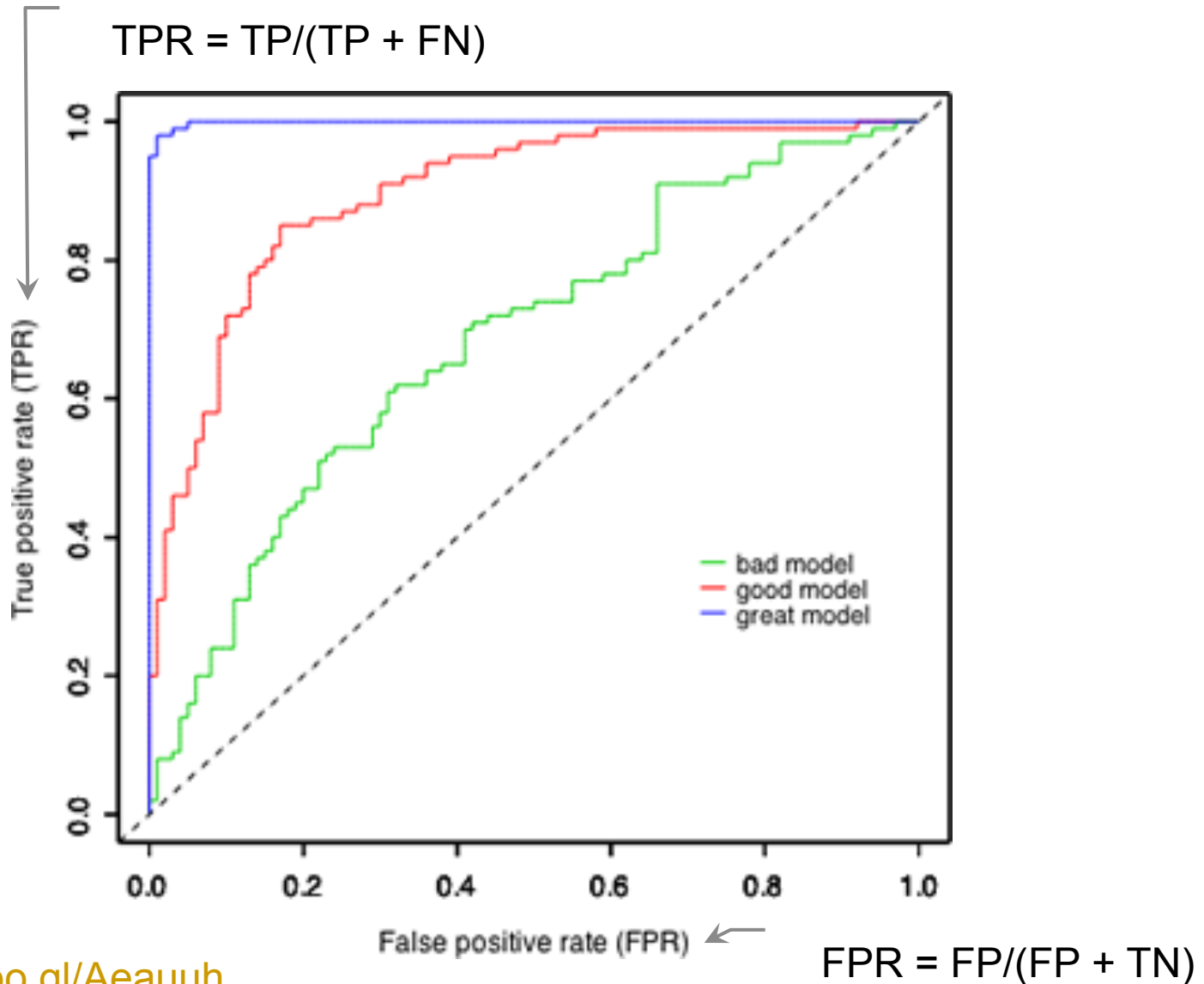
# AREA UNDER THE ROC\* CURVE (AUC)

- It measures discriminatory power of a classifier, i.e., its ability to correctly differentiate instances of different classes
- It is used for measuring performance of binary classifiers
- It takes values from the 0-1 interval
- In the case of random classification,  $AUC = 0.5$ ; so, as the AUC value is greater than 0.5, the classifier is better
  - 0.7–0.8 is considered fair; 0.8–0.9 good;  $> 0.9$  excellent

\*ROC = Receiver Operating Characteristic;

[http://en.wikipedia.org/wiki/Receiver\\_operating\\_characteristic](http://en.wikipedia.org/wiki/Receiver_operating_characteristic)

# AREA UNDER THE ROC\* CURVE



# ACKNOWLEDGEMENTS AND RECOMMENDATIONS

# ACKNOWLEDGEMENTS AND RECOMMENDATIONS

## MACHINE LEARNING @ STANFORD

- Coursera: <https://www.coursera.org/course/ml>
- Stanford YouTube channel:  
[http://www.youtube.com/view\\_play\\_list?p=A89DCFA6ADACE599](http://www.youtube.com/view_play_list?p=A89DCFA6ADACE599)

## MACHINE LEARNING @ Carnegie Mellon University

- Lectures by Andrew W. Moore, especially, the lecture on Bayes Rule and Bayes Classifiers:  
[http://www.autonlab.org/tutorials/prob\\_and\\_naive\\_bayes.pdf](http://www.autonlab.org/tutorials/prob_and_naive_bayes.pdf)

# RECOMMENDATIONS

## DATA STORIES PODCASTS

- <http://datastori.es/>
- especially podcast #27 on Big Data Skepticism
- the Cambridge study mentioned in podcast #27 – using FB Likes to determine people’s demographic traits:  
<http://www.pnas.org/content/early/2013/03/06/1218772110.full.pdf>

(Anonymous) questionnaire for your  
critiques, comments, suggestions:

<http://goo.gl/cqdp3l>