

# DATA MINING

## ASSIGNMENT 1

Ali Gholami

Department of Computer Engineering & Information Technology  
Amirkabir University of Technology

<http://ceit.aut.ac.ir/~aligholamee>  
[aligholamee@aut.ac.ir](mailto:aligholamee@aut.ac.ir)

### Abstract

In this assignment, several paramount concepts of *Data Analysis* will be explained. we'll discuss the importance of metrics in the first theoretical problem. A quick review on the *Apriori* algorithm for the *Association Rule Mining* will be explained also. We'll also show how *Weka* can be used for *Association Rule Mining*. Furthermore, The effectiveness of *Normalization* concept is proposed. Finally, an *Statistical* point of view will help us to demonstrate and rationalize the relationship between the *Performance* of the *Learning Algorithm* and the amount of *Data* available. A chief section of this assignment is dedicated to solve the *Titanic* problem, which is a great practice of data mining concepts in production. We'll use *Python* programming language and three main libraries; *Scikit-Learn*, *Pandas* and *Numpy* to tackle this problem. The Python implementation of the Titanic problem is provided on a *Jupyter Notebook* attached with this report.

**Keywords.** *Apriori, Association Rule Mining, Normalization, Generalization, Preprocessing, Feature Engineering, Scikit-Learn, Pandas, Numpy, Python 3.5.*

## 1 Performance Metrics Analysis

Given the following *Confusion Matrix* for a prediction about cancer.

		Predicted Class		Total
		Cancer = Yes	Cancer = No	
Actual Class	Cancer = Yes	60	290	350
	Cancer = No	150	9500	9650
Total		210	9790	10000

Table 1.1: Confusion matrix of cancer prediction.

Compute each of these performance metrics.

(a) Accuracy

- (b) Sensitivity
- (c) Precision
- (d) Specificity
- (e) F-measure

## Solution

Before getting into the computations, we'll review the *nicknames* and *formulas* to calculate each of these metrics. We have computed each of these metrics in front of them.

$$Accuracy = \frac{TP + TN}{TP + TN + FN + FP} = \frac{60 + 9500}{60 + 9500 + 290 + 150} = 0.956 \quad (1.1)$$

$$TPR = Recall = Sensitivity = \frac{TP}{P} = \frac{TP}{TP + FN} = \frac{60}{60 + 290} = 0.171 \quad (1.2)$$

$$PPV = Precision = \frac{TP}{TP + FP} = \frac{60}{60 + 150} = 0.285 \quad (1.3)$$

$$TNR = Specificity = \frac{TN}{N} = \frac{TN}{TN + FP} = \frac{9500}{9500 + 150} = 0.984 \quad (1.4)$$

$$F - measure = \frac{2 * TP}{2 * TP + FP + FN} = \frac{2 * 60}{2 * 60 + 150 + 290} = 0.214 \quad (1.5)$$

Our mission is done! Nevertheless, we continue the explanation for almost each of these metrics. We'll discuss why *Accuracy* itself would be a bad metric in most of the challenging cases.

## Why Performance Metrics Are Important

Metrics are important because they allow us to judge about models ability in prediction task. Without metrics we won't be able to compare models; Thus we won't be able to improve each.

## What Kind of Performance Metrics Are Useful

Not all of metrics describe this ability correctly in different conditions. Generally speaking, we need an *Objective* metric. A metric could exhibit a great number for a classifier that classifies data as *True* always. This can happen in *Imbalanced Datasets*. The important thing is that, we need to establish a *Baseline* before getting into these numbers. We need to measure the performance for a simple system before start tuning these numbers up. The absolute maximum performance that a machine learning system can achieve, is called *Ceiling*. The performance we get with respect to the numbers (*like numbers calculated above*), is bound between *Baseline* and *Ceiling* values.

$$\text{Baseline} < \text{Performance} < \text{Ceiling} \quad (1.6)$$

## Possibility of Getting Complete Performance

No, its not possible! Even using 2 humans to classify some data, *they might not agree 100% of the times*.

## Accuracy Paradox

Accuracy is the proportion of the correct results that a classifier achieved. Assume a classifier who classifies its inputs as *true* always. The denominator for the *Accuracy* formula is the size of the dataset, which is a constant. The numerator while, contains  $TP + TN$ . This classifier predicts a great number of  $TP$  and a small number of  $TN$ . If the assumption changes to be that classification always turns out to be *false*, we'll get a huge value for  $TN$  and a small value for  $TP$ . The addition is the same by the way. Thus the accuracy of a *dummy* model can be amazing in both criteria. Thus, *Accuracy* is not a reliable metric in machine learning problems. We call this *Accuracy Paradox*.

## Recall

This metric describes that, out of all the positive examples there were, what fraction did the classifier pick up?

## Precision

This metric states that, out of all the examples the classifier labeled as positive, what fraction were correct?

## Combination of Recall & Precision

Combination of these metrics, causes the results to be *balanced*.

## $F_\beta$ Score

$F_\beta$  combines *Precision* and *Recall*. We'll talk about its advantages later in this assignment.

## 2 The Concept of Harmonic Mean and $F_\beta$ Score

What is  $F_\beta$  score? Describe how the term  $\beta$  affects the final score.

### Solution

We saw how the  $F_1$  score is computed. Before getting into the details, let's recall something interesting; **Average**. The simplest form of average is what we describe as *sum of all values divided by their count*. There is another average (*mean*), which is called *Harmonic mean*.

Harmonic mean is represented as:

$$\frac{1}{H} = \frac{1}{n} \sum_{i=1}^n \frac{1}{x_i} \quad (2.1)$$

Lets find the harmonic mean for two scores; Precision and Recall. The equation (2.1) turns out to be:

$$\frac{1}{H} = \frac{1}{2} \sum_{i=1}^2 \frac{1}{x_i}$$

Replacing the terms, Recall and Precision:

$$H = \frac{1}{\frac{1}{2}(\frac{1}{Recall} + \frac{1}{Precision})} = \frac{2 * Recall * Precision}{Recall + Precision} \quad (2.2)$$

Generalizing this idea, we can multiply  $H$  by both sides of (2.1):

$$\frac{1}{n} \sum_{i=1}^n \frac{H}{x_i} = 1 \quad (2.3)$$

In other words, the average of ratios between the harmonic mean and the data points is unity. We call  $H$  the  $F_1$  score. The important thing is that the effect of each score on the  $H$  is considered the same here ( $\frac{1}{2}$ ). We call this effectiveness as the *weight* of each score in forming the  $F_1$  score. The summation of weights has to remain 1. Considering the weight of  $\frac{1}{\beta+1}$  as the weight for the *Precision*, we'll have  $\frac{\beta}{\beta+1}$  as the weight for the *Recall*. Because we can get

$$\frac{\beta}{\beta+1} + \frac{1}{\beta+1} = \frac{\beta+1}{\beta+1} = 1$$

Rewriting the equation (2.2) by changing the weights of its parameters:

$$F_\beta = \frac{1}{\frac{\beta}{\beta+1} \frac{1}{Recall} + \frac{1}{\beta+1} \frac{1}{Precision}} = \frac{(\beta+1) * Recall * Precision}{Recall + \beta * Precision} \quad (2.4)$$

### The Effect of $\beta$

In this subsection, we'll focus on the measurements of effectiveness; *Precision* and *Recall*. We'll discover the *effectiveness* of *Precision* and *Recall* on the  $F_\beta$  score. There are following possible conditions for the *Precision* and *Recall* weights.

$$W_{Precision} > W_{Recall}$$

$$W_{Precision} < W_{Recall}$$

$$W_{Precision} = W_{Recall}$$

According to the equation (2.3), the summation of these weights is equal to 1. Thus, in order to get  $W_{Precision} = W_{Recall}$ , weights are needed to be  $\frac{1}{2}$  each. If  $\beta > \frac{1}{2}$  the weight of *Precision* is higher than the *Recall* and vice versa. In the measurement process, the users can attach different relative importance to *Precision* and *Recall*. What we want is therefore a parameter( $\beta$ ) to characterize the measurement function in a way that *It measures the effectiveness of prediction with respect to a user who attaches  $\beta$  times as much importance to Recall as Precision.*

## 3 The Concept of Normalization

Describe the term *Normalization* in data engineering. What do we mean by *Normalizing* the data?

### Solution

As the word implies, *Normalization* is the process of adjusting values into alignment. The *transformation* of values in order to represent them in a uniform range, is also called *Normalization*. This topic can have different meaning in different applications.

### Normalization of Inputs in Neural Networks

The inputs must have same range of values, otherwise we'll be left with an *ill-conditioned* model after the training.

### Convergence of Weights & Biases in Distance Based Classifiers

While using *Distance Based* classifiers, it is important not to get conditioned by features with wider range of possible values. *Normalization* is used to guarantee the *Convergence* of *Weights and Biases* in such conditions. We often call this, the *Convergence of Gradient Descent*, since the gradient descent is used most of the times to optimize the loss.

## 4 The Effect of Data

*“More better data lets the model grasp a better intuition of an specific random event.”* This is simple expression of the *Law of Large Numbers* in probability theory. The parameters of our model is tuned to predict the real-world data. It could be the data that the model has never seen while being trained. Thus, we can infer that the input data distribution will comply with the *LLN* most of the times. We’ll be reviewing two other effects of data in the upcoming subsections; *High Variance* and *High Bias* conditions.

### High Variance Condition

Occurs when a model is too complicated and relatively, the amount of data is not enough. We often call this as *Overfitting* in machine learning. The result of the criterion will be

$$Error_{Training} \ll Error_{Testing}$$

This problem can be solved by reducing the number of features. As an example, assume a language model in which roughly every word in the vocabulary can be considered as a feature. *In this case, adding more data would help.*

### High Bias Condition

Occurs when a model is too simple to explain our data. *Adding more data won’t help in this case.*