ORES Custom Documentation V

Disclaimer: No guarantee for the correctness of information / explanations / sources is given.

Goals

1. Metrics list:
   - Add examples ✓
   - Correct the descriptions of counts and rates ✓
   - Improve descriptions of roc_auc and pr_auc ✓
   - Add the new standalone version to the repo ✓
2. Research what form of revision data is needed (for existing visualizations, but also in general) → also check RecentChanges and new Filters and what API calls look like in that context
3. - Watch “ROC curves and Area Under the Curve explained” and add insights to the **roc_auc** in the list ✓
   - and think about what parameters could be used in which ways to filter the output of the current UI (currently: X inputs → X outputs)
   - Also check out: Precision-Recall AUC vs ROC AUC discussion ✓
4. Read A Review of User Interface Design for Interactive Machine Learning
5. Think about what could be the goal of this thesis

1 Crucial metrics: damaging-model

Examined metrics:

- !f1
- !precision
- !recall
- accuracy
- counts
- f1
- filter_rate
- fpr
- match_rate
- pr_auc
- precision
- rates
- recall
- roc_auc

For each metric (if possible) there will be:

1. The formula based on the **confusion matrix**
2. A definition
3. An intuitive explanation with an example
4. Its meaning based on the **loan threshold** representation by Google (Link)
5. Additional information (if necessary)

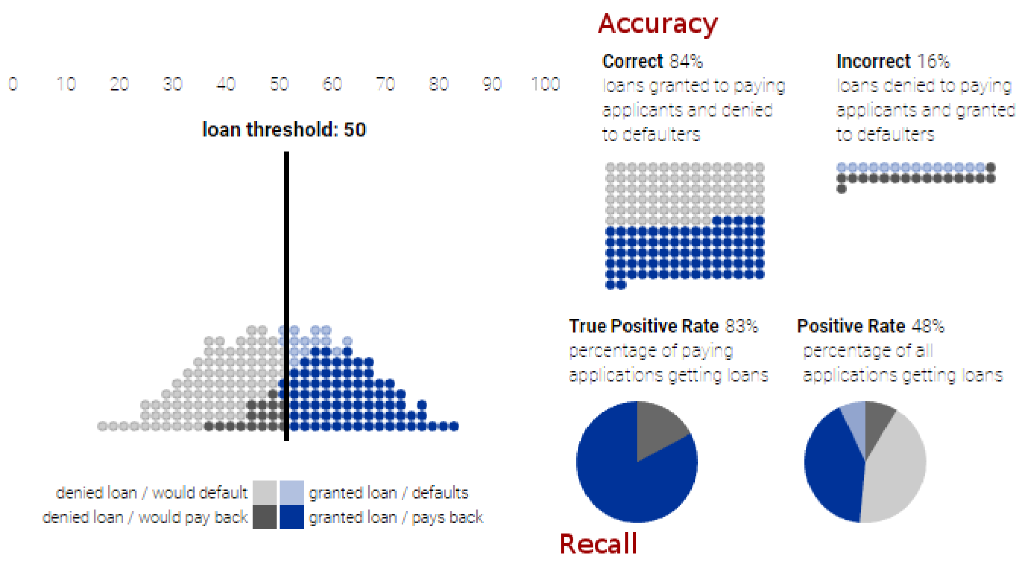
Explanations: References

- Confusion Matrix

		Actual	
		Positive	Negative
Predicted	Positive	**True Positive**	**False Positive**
	Negative	**False Negative**	**True Negative**

Abbreviations: **TP**, **FP**, **FN** and **TN**.

- Loan Threshold

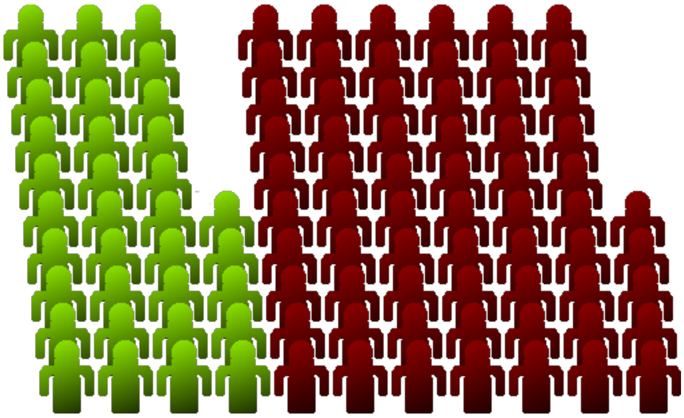


The example scenario

To ease the understanding, let's stick to the following scenario and refer to it for each metric:

- 100 people represent our total population
- 35% of our population is infected with disease X

  That leaves us with the following **labels**: **35** **positives** and **65** **negatives** (being tested for disease X;)



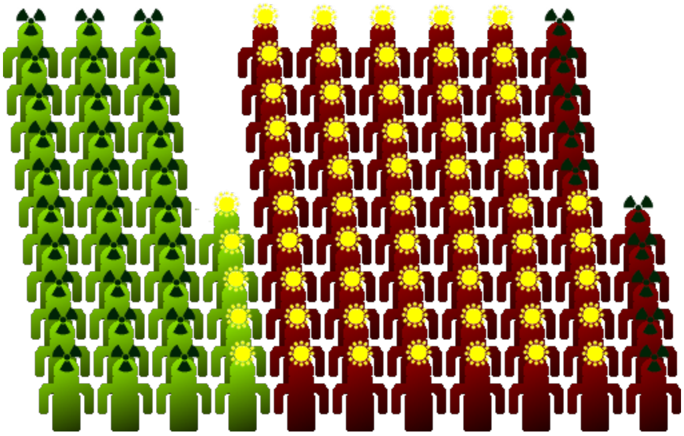
- A classifier is now supposed to classify every person of our population (based on their visible symptoms for example). This is the **prediction**:
  - If our algorithm says a person is infected, that person will be classified as a **positive**, and marked with radioactive symbol:

  - If the prediction results in a **negative**, the person will be marked with a sun symbol:

- The classifier may predict that:
  - out of the 35 infected people, **30** are infected (those 30 are what we call **true positives**) and **5** are not (those 5 are **false negatives**)
  - out of the 65 non infected people, **10** are infected (**false positives**) and **55** are not (**true negatives**)



 **true positive**: infected and correctly predicted (**30**)

 **false negative**: infected and incorrectly predicted (**5**)

 **true negative**: not infected and correctly predicted (**55**)

 **false positive**: not infected and incorrectly predicted (**10**)

Let's get started.

1.1 recall

1. $\frac{\mathtt{TP}}{\mathtt{TP}+\mathtt{FN}}$

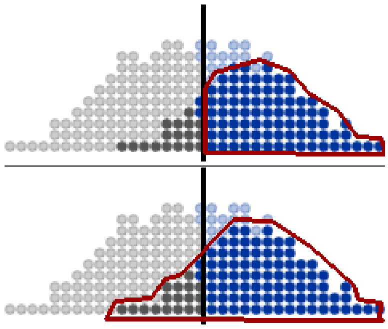
2. Recall ($\equiv$ true positive rate $\equiv$ "sensitivity") is the ability of a model to find **all** relevant cases within the dataset.

3. Now, in our example scenario, the relevant cases are the infected people. We absolutely want to identify those: . The ability of the model to identify those depends on how many will be **correctly** predicted to be infected: .

   In other words, we are looking for the ratio of correctly predicted to be infected people to all infected people.

That leads to $\frac{\sum}{\sum}$, with $=$ $+$, which is equivalent to the formula in **1.**, if you replace the symbols with their confusion matrix counterpart according to the legend in **The example scenario**.

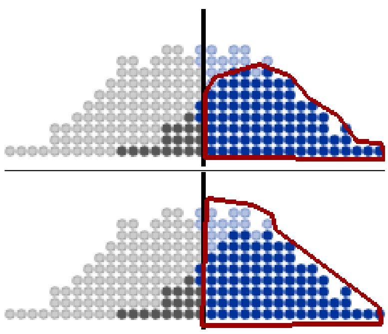
In terms of numbers for our example that would be $\frac{30}{30+5} \approx 0.86$

4. 

1.2 precision

1. $\frac{\mathtt{TP}}{\mathtt{TP}+\mathtt{FP}}$
2. Ability of the model to find **only** relevant cases within the dataset
3. Again, we take a look at the relevant cases, the infected people: . This time around though, we are **not** interested in the ratio of correctly predicted to be infected people to **all** infected people. Instead we want to know how good the model is at only predicting those to be infected, that actually are. Therefore, we want the ratio of all to all those predicted to be infected: $+$

   $$= \frac{\sum}{\sum + \sum} = \frac{30}{30+10} = 0.75$$

4. 

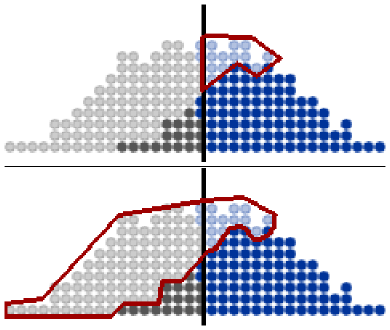
1.3 f1

1. -
2. f1-Score, the harmonic mean of recall and precision, a metric from **0** (worst) to **1** (best), used to evaluate the accuracy of a model by taking into account recall and precision: $= 2 * \frac{\texttt{precision}*\texttt{recall}}{\texttt{precision}+\texttt{recall}}$
3. For our example model, that would result in $= 2 * \frac{0.75*\frac{30}{35}}{0.75+\frac{30}{35}} = 0.8$
4. -
5. <u>Additional information</u>: Compared to the simple average (of recall and precision), the harmonic mean punishes extreme values (e.g. precision 1.0 and recall 0.0 $\rightarrow$ average = 0.5, but f1 = 0)

1.4 fpr

1. $\frac{\texttt{FP}}{\texttt{FP}+\texttt{TN}}$
2. The false positive rate is the probability of a false alarm.
3. In our example, a false alarm would obviously be labeling someone as infected, who isn't: . Now we just have to ask ourselves what portion of those, that, if they **were** incorrectly predicted as infected (because they **are not** infected:), **are** incorrectly predicted as infected? Hence, we are looking for the ratio of to all non infected people: + .

   $$= \frac{\sum}{\sum + \sum} = \frac{10}{10+55} \approx 0.15$$

4. 

1.5 roc_auc

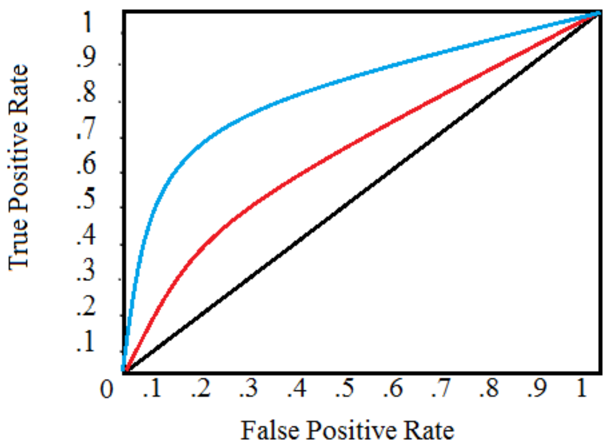
1. -

2. The **area under** the **curve** of the **ROC**-curve, a measure between 0.5 (worthless) and 1.0 (perfect: getting no FPs), rates the ability of a model to achieve a blend of recall and precision.

3. In our example, we haven't used the notion of threshold yet. For classifying people as infected or not, the classifier will evaluate multiple criteria and calculate the probability that a patient is infected. Many binary classifiers have the threshold at 0.5, meaning that, if the probability of a true outcome is higher than 50%, it is classified as a positive; or in our case as an infected patient. Depending on the situation however, it can be useful to move that threshold.

The receiver operating characteristic (ROC) curve is used to visualize the performance of a classifier

The ROC curve plots the TPR versus FPR as a function of the model's threshold for classifying a positive.



Increasing the threshold → moving up a curve (≡ model) to the top right corner, where all data is predicted as positive (threshold = 1.0) and vice versa

Our best case is a curve hugging the top left corner, because that means a low **fpr** and high **tpr**, which, again, means a lot of positives and few negatives on the right of the thresholds, looking at class curves like the **Loan Threshold** one.

Assuming that we have had a threshold of 0.5 all along to get the previous results, one point on our ROC curve would be: $(0.15, 0.86) = (\texttt{fpr}, \texttt{tpr})$.

We would now have to change the thresholds, look at the (probably) changed resulting data (new numbers in terms of positives and negatives, as well as **TP**, **FN**, **TN** and **FP**) and then plot every resulting $(\texttt{fpr}, \texttt{tpr})$-point in order to plot the full ROC curve.

We would most certainly like to quantify this visualized performance of our binary classifier, which is why we calculate the area under the curve (**auc**) of the ROC curve.

4. -

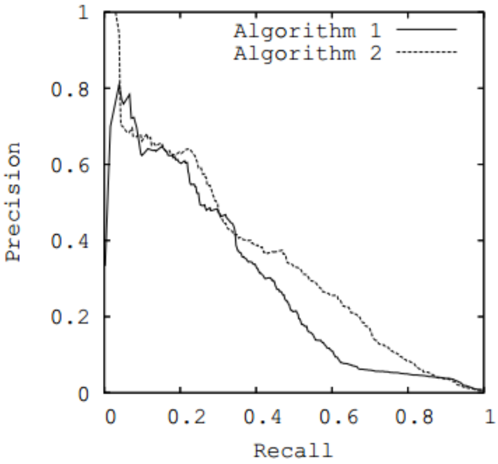
5. Additional information: We can think of AUC as representing the probability that a classifier will rank a randomly chosen positive observation higher than a randomly chosen negative observation. That's why **roc_auc** is a useful metric even for datasets with highly unbalanced classes. (Source)

1.6 pr_auc

1. -

2. Similarly to the **roc_auc**, the area under the precision recall curve **pr_auc** evaluates a classifiers performance. The main difference, however, is that the **pr_auc** does not make use of **true negatives**. It is therefore favourable to use **pr_auc** over **roc_auc** if true negatives are unimportant to the general problem or if there are a lot more negatives than positives.

   The PR-curve plots the Precision versus the Recall:



Instead of the top left corner for the ROC-curve, here, we want our curve to reach the top right corner for our classifier to be perfect.

The following scenario provides a good example (Source 1 and Source 2) of a case with a lot more negatives than positives and comparing ROC to PR:

- Out of 1 million documents, we want to find the 100 relevant ones.
- The task is accomplished by two different algorithms:
  (a) 100 retrieved documents, 90 relevant
  (b) 2000 retrieved documents, 90 relevant
- Algorithm (a) is obviously preferable.
- We know that ROC- and PR-curves both plot coordinates with `tpr` = `recall` as one dimension. Now the question is: how do they differ in the other dimension, when plotting both algorithms?
- In all cases `tpr` = `recall` = 0.9. We also have:
  (a) `TN` = 999890 and `FP` = 10
  (b) `TN` = 997990 and `FP` = 1910
- ROC:
  (a) $\texttt{fpr} = \frac{\texttt{FP}}{\texttt{FP+TN}} = \frac{10}{10+999890} = 0.00001$
  (b) $\texttt{fpr} = \frac{1910}{1910+997990} = 0.00191$

  Having retrieved many more documents, and therefore having many more **false positives**, algorithm (b) has a higher **fpr** than algorithm (a).

The **fpr** also takes into account the vast amount of **true negatives** though, which is why the difference between the two **fpr**s is still quite small: 0.0019.

- PR:
  (a) $\texttt{precision} = \frac{\texttt{TP}}{\texttt{TP+FP}} = \frac{90}{90+10} = 0.9$
  (b) $\texttt{precision} = \frac{90}{90+1910} = 0.045$

  Not accounting for **true negatives**, the **precision** is not affected by the relative imbalance.

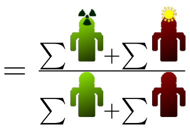
  We are presented a remarkable difference of 0.855.
- To close on this topic, not the following (by Randy C): "Obviously, those are just single points in ROC and PR space, but if these differences persist across various scoring thresholds, using ROC AUC, we'd see a very small difference between the two algorithms, whereas PR AUC would show quite a large difference."

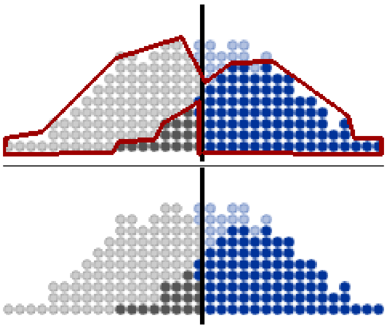
3. Similarly to the **roc_auc**, the point on the PR curve of our example for the standard threshold of 0.5 would be: $(\texttt{precision}, \texttt{recall}) = (0.75.0.86)$.

4. -

1.7 accuracy

1. $\frac{\texttt{TP+TN}}{\texttt{Total}}$
2. Accuracy measures the portion of correctly predicted data
3. In our example scenario, this is equal to asking ourselves *out of all patients, what's the portion of correctly predicted cases?* The correctly predicted cases are infected patients, predicted to be infected (), and non infected patients, predicted not to be infected (). This wanted proportion results in:

    $= \frac{30+55}{35+65} = 0.85$

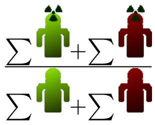
4. 

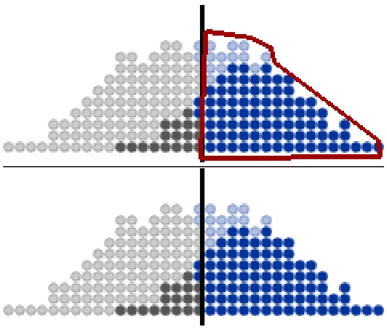
1.8 match_rate

1. $\frac{\texttt{TP+FP}}{\texttt{Total}}$

2. The match rate is the proportion of observations matched/not-matched, meaning the ratio of observations predicted to be positive to the total of observations.

3. Concerning our example, this would be equal to wanting to know what portion of the population was predicted to be infected. Those groups are:  and .

=  $= \frac{30+10}{35+65} = 0.4$

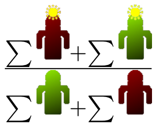
4. 

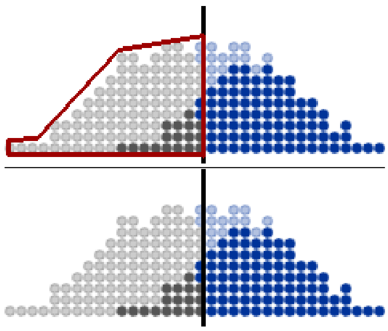
1.9 filter_rate

1. $1 - \texttt{match\_rate} = \frac{\texttt{TN+FN}}{\texttt{Total}}$

2. The filter rate is the proportion of observations filtered/not-filtered, meaning the ratio of observations predicted to be negative to the total of observations. This is the complement to the match rate.

3. In our example scenario, this would be equal to wanting to know what portion of the population was predicted not to be infected.

$=$  $= \frac{55+5}{35+65} = 0.6 = 1 - \texttt{match\_rate}$

4. 

1.10 counts

1.
   - **labels**:
     - false: $\texttt{TN} + \texttt{FP}$
     - true: $\texttt{TP} + \texttt{FN}$
   - **n**: `Total`
   - **predictions**:
     - false:
       * false: `TN`
       * true: `FP`
     - true:
       * false: `FN`
       * true: `TP`

2.
   - **labels**: The number of edits (*manually*) labeled as *false* and *true*: these values represent the **actual** positives and negatives.
   - **n**: The sample size; total number of edits taken into account
   - **predictions**: edits ...
     - false: ... actually being false ...
       * false: ... and predicted to be false
       * true: ... but predicted to be true
     - true: ... actually being true ...

    * false: ... but predicted to be false
    * true: ... and predicted to be true

3. Concerning our example:
   - **labels**:
     - false (non infected people):  +  = 55 + 10 = 65
     - true (infected people):  +  = 30 + 5 = 35
   - **n** (Total): 100
   - **predictions**:
     - false (non infected people ...)
       * false (... and predicted not to be infected):  = 55
       * true (... but predicted to be infected):  = 10
     - true (infected people ...)
       * false (... predicted not to be infected):  = 5
       * true (... predicted to be infected):  = 30
4. -
5. <u>Additional information</u>:

   When calling the enwiki damaging model for example (Link), you will get the following output for counts:

```
"counts": {
  "labels": {
    "false": 18677,
    "true": 751
  },
  "n": 19428,
  "predictions": {
    "false": {
      "false": 17958,
      "true": 719
    },
    "true": {
      "false": 320,
      "true": 431
    }
  }
},
```

   ⇒ e.g. out of 18677 edits that were labeled as *false*, 719 were false positives

1.11 rates

1.
    - false: $\frac{\texttt{counts\_labels\_false}}{\texttt{counts\_n}}$
    - true: $\frac{\texttt{counts\_labels\_true}}{\texttt{counts\_n}}$

2. The rates simply give us the proportion of edits labeled as false or true to the total number of edits taken into account.

3. For the example scenario:

    - false (proportion of infected people to the total number of people tested): $\sum$  $/ 100 = \frac{65}{100} = 0.65$
    - true (proportion of infected people to the total number of people tested): $\sum$  $/ 100 = \frac{35}{100} = 0.35$

4. -

5. <u>Additional information</u>:

   Calling the API the same way as for **counts** (Link), we get:


```
"rates": {
  "population": {
    "false": 0.966,
    "true": 0.034
  },
  "sample": {
    "false": 0.961,
    "true": 0.039
  }
},
```


   The number of edits taken into account for **sample** equals the **n** from **counts** = 19428.

   Now we see that, also looking at the output under **counts**, *rates_sample_false*: $0.961 = \frac{18677}{19428}$.

   Note that we are shown results for “population” and “sample”. There is a significant number of bot edits and edits that don’t need reviewing (admins, autopatrolled users). The **sample** of edits does not contain any of those.

1.12 !<metric>

- Any <metric> with an exclamation mark is the same metric for the negative class
- e.g. recall $= \frac{\texttt{TP}}{\texttt{TP+FN}} \Rightarrow$ **!**recall $= \frac{\texttt{TN}}{\texttt{TN+FP}}$
- Example usage: find all items that are not "E" class (itemquality model) $\rightarrow$ look at **!**recall for "E" class.

1.12.1 Existing !<metric>s

- !f1
- !precision
- !recall

2 Revision data, RecentChanges and newFilters

3 pr_auc, roc_auc and filtering the ORES UI output

3.1 pr_auc and roc_auc

Insights from the linked video and discussion under **Goals** have been added to the metrics list.

3.2 What parameters could be used to filter the output of the current UI?

4 A Review of User Interface Design for Interactive Machine Learning - Notes

- Interactive Machine Learning (**IML**) complements human percept. + int.: computat. speed + power
- Interactive process: input from user without needing knowledge of ML

  $\Rightarrow$ UI Design is fundamental!

... but no consolidated principles on implement. of such a UI

- In this paper: proposit. of model of generalised IML system and solut. principles for effective IML interf.

4.1 Introduction

- ML: application as development tool only for experts
- IML: training process as HCI task → more accessible

  for example: human input in example selection, creation and labelling

- Note: expert may still be required to underlying alg.
- Ideal case: non expert constructs own learned concepts by creating or collecting training data according to their need

  In practice: major challenge for ML practitioner (=expert) and UI designer!

  ⇒ This paper: focuses on interface design challenge

- IML: allows for application of domain knowledge (domain experts can now train models)!
- IML workflow is co-adaptive: user and target model directly influence each other's behaviour

  Again, this is done over the interface: it's a dialogue between human and machine

- Four key challenges for designing and IML interface:

  1. Users are imprecise and inconsistent. "Unless users perceive their own deficiencies, this failure [poor model resulting from poor training by user] is attributed to the system."
  2. Uncertainty relating user input and user intent. E.g. user doesn't assign an example to a concept does not imply a counter example
  3. Model is not your conventional information structure: "ML model evolves in response to user input but not necessarily in a way that is perceived as intuitive or predictable by the user"
  4. Training is open ended.

4.1.1 Guidance for the designer

Structural and behavioural breakdown of IML system will be presented.

Help designers think about the architecture and support discussion of design considerations with clear terminology

Also help with how to present information to users and what interactions to promote

- Structural: Constituent components (user, interface, data, model)

  Interface will be divided into four elements itself.

5 Possible goal of the thesis

Questions

- Q: Should I ask Aaron how he would like us to work together?

  A:

- Q: In what situations exactly do we want to optimize the threshold in the context of user centered threshold optimization?

  A: