

ORES Custom Documentation III

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

Goals

1. Check out IRC Channel ✓
2. Read towardsdatascience article ✓
3. Research the model item quality ✓
4. Study pipeline definition ×
5. Take a look at LIME + Machine Learning ✓
6. Understand and describe all parameters ~ mostly
 - Get in touch on IRC and ask if there is a list of all parameters (and what match_rate means?) ✓
7. Open repository on Github to document the research ✓
8. Research ~
 - (a) In what other cases than confusion matrices are those parameters explained?
 - (b) Are there already visualizations of some of these parameters in any contexts?
 - (c) Are there any applications, where I can filter for these parameters → visualizations or just about anything?

1 IRC Channel

Link: IRC Freenode #wikimedia-ai

2 Beyond Accuracy: Precision and Recall

Link: Towards Data Science

- Classification problems: is passenger a terrorist?
- Positive class is way smaller than negative → accuracy is not a good measure for model performance (for example 99%) models aren't good enough
- Focus on identifying positive cases:

- **Recall**: Ability of model to find all relevant cases within dataset

$$\text{Recall} = \frac{\text{true positives}}{\text{true positives} + \text{false negatives}}$$
$$= \frac{\text{terrorists correctly ident.}}{\text{terr.corr.ident.} + \text{terrorists incorrectly labeled as not terrorists}}$$

Labeling everyone as terrorist → **recall** = 1.0

But low **precision**!

- **Precision**: Ability of model to find only relevant cases within dataset

$$\text{Precision} = \frac{\text{true positives}}{\text{true positives} + \text{false positives}}$$
$$= \frac{\text{terrorists correctly ident.}}{\text{terr.corr.ident.} + \text{individuals incorrectly labeled as terrorists}}$$

- Other hypoth. model: only identify one single terrorist correctly

Recall really low but:

Precision = 1.0

- **Key learning**: Increasing **recall** decreases **precision** and vice versa

2.1 Combining Precision and Recall

- There are cases in which we want to maximize one of those 2 parameters
- But if we want an optimal combination: **F1-score**, the harmonic mean of precision and recall

- $$F1 = 2 * \frac{\text{precision} * \text{recall}}{\text{precision} + \text{recall}}$$

- Better than using simple average because it punishes extreme values:

Classifier with precision 1.0 and recall 0.0 has average of 0.5, but **F1** = 0.0

Optimal balance of **precision** and **recall** \Rightarrow maximize **F1**

2.2 Visualizing Precision and Recall

- Confusion Matrix:

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

- ROC curve (Receiver Operating Characteristic):

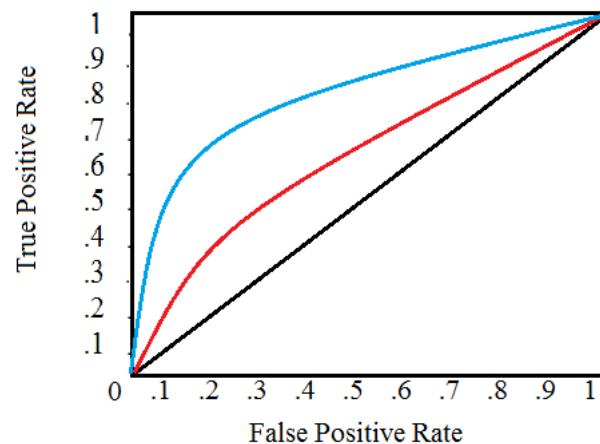
- Definition: plots the true positive rate (TPR) versus the false positive rate (FPR) as a function of the model's threshold for classifying a positive

- True Positive Rate (=Recall): $\frac{TP}{TP+FN}$

- False Positive Rate: $\frac{FP}{FP+TN}$

tpr: the portion, of all positive data, that was predicted as positive

fpr: the portion, of all negative data, that was predicted as positive



Bottom left: Decision threshold = 1.0 \Rightarrow all data is predicted as false \rightarrow tpr = fpr = 0.0

Top right: Decision threshold = 0.0 \Rightarrow all data is predicted as positive \rightarrow tpr = fpr = 1.0

Question: tpr = 1.0 also works if all positive data is predicted as positive and all the negative data is predicted as negative, but then tpr = 1.0 \neq fpr = 0.0... (How is this graphic still correct?)

- We can quantify ROC curve by calculating the area under the curve (AUC): metric between 0 and 1

AUC of blue curve > AUC of red curve \Rightarrow blue classifier is better at achieving blend of precision and recall

Random classifier (black line) has AUC of 0.5

2.3 Example Application

Definitely worth checking out in the article itself. Amongst other things: how to calculate F1 from confusion matrix

2.4 Conclusion

Accuracy is not always the most useful metric. Instead, **recall**, **precision**, **F1** and **ROC** help us assess classifications in smarter ways in some contexts.

3 Itemquality

- We are looking into the *itemquality* model in the wikidatawiki context, containing the *damaging-*, *goodfaith-* and *itemquality-*model.

- Also, Wikidata is the central storage for structured data of many Wikimedia projects (including Wikipedia).
- Wikidata Wiki link: “The Wikidata repository consists mainly of items, each one having a label, a description and any number of aliases. Items are uniquely identified by a Q followed by a number, such as Douglas Adams”: 42
- Now the purpose of the itemquality model becomes clear: rate the quality of said items

4 Pipeline definition ?

- Github Link
- This line in particular determines which features are used: Github Link
- Constructing the feature list, culminating in: Github Link

5 LIME and Machine Learning

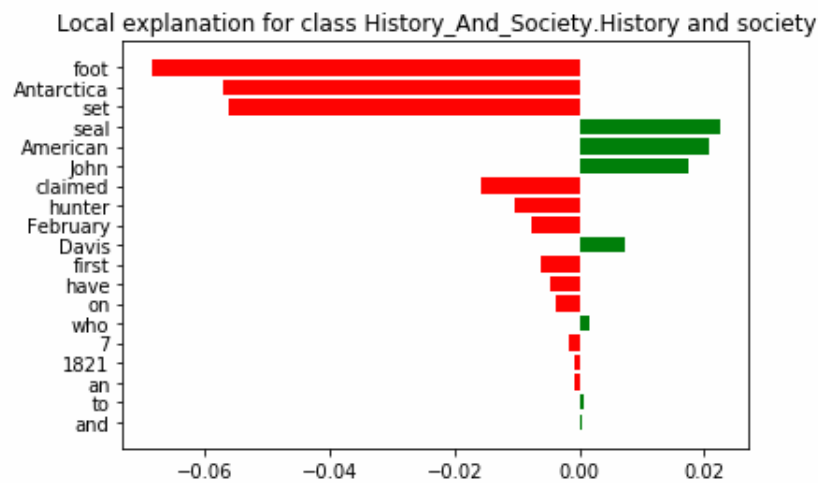
- Link: **lime**, a unit testing and functional testing framework
- Adam Wight’s Lime Sandbox: Github link

→ Take a look at “Explain draft topic.ipynb”:

Use Lime to explain the scores a drafttopic (see: <https://en.wikipedia.org/?oldid=42344094>) got

The result will be “[('Culture.Language and literature', 0.5791824638170249), ('Geography.Countries', 0.3864267805975978), ...]”

After specifying a few parameters for the Lime explainer, it can be used to explain the scoring; results will look like this:



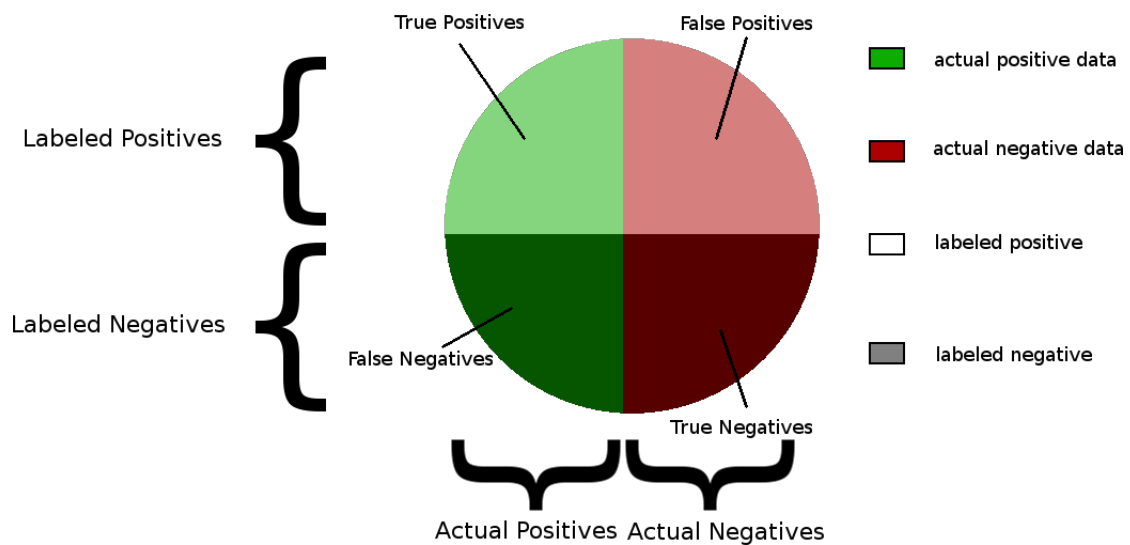
6 Crucial Metrics

As described in this article on *Towards Data Science* ([Link](#)), the right metrics for classification tasks heavily depend on the context. Let's take a look at possible parameters/metrics to use in ORES API queries.

Keep in mind the confusion matrix:

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

Or in another format:



Note that the colors of all parts of the circle are always a mix of two colors from the legend on the right.

The following subsections describe interesting metrics, notably in the wiki-data/wiki context and, more precisely, itemquality model, and will be defined and described with the help of the already mentioned sources, not only from this file, but from other files within the repository as well.

6.1 Recall

- Recall (\equiv True Positive Rate) is defined as the ability of a model to find all relevant cases within the dataset.
- Or: the portion, of all actual positive data, that was labeled as positive
- In other words $\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$
- Again in other words = $\frac{\text{correctly_labeled_as_positives}}{\text{actual_positives}}$

- =

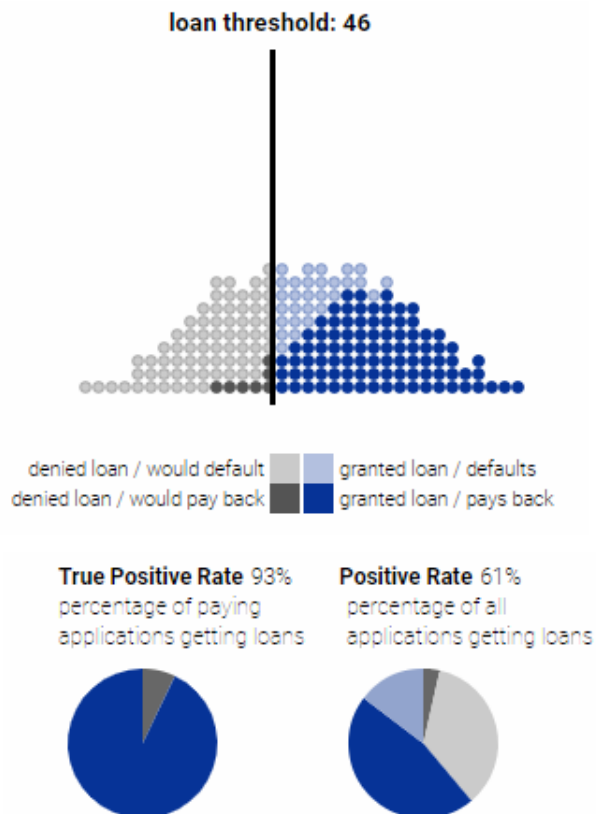
6.2 Precision

- Ability of the model to find only relevant cases within the dataset

- The portion, of all as “positive” labeled data, that is actually positive
- $= \frac{TP}{TP+FP}$
- $= \frac{\text{correctly_labeled_as_positives}}{\text{all_labeled_as_positive}}$
- $= \frac{\text{green}}{\text{green} + \text{red}}$

Recall vs Precision

When increasing one of these two, the other one naturally decreases. For an intuitive example, let's take a look at Google's Loan Threshold Simulation:



The dark grey / dark blue dots, representing clients that would actually pay back their loan, are more and more included (\rightarrow given loans) if we move the threshold further to the left.

But so are clients that would not. Thus moving the threshold to the left increases the **recall (tpr)** but decreases the **precision** and vice versa when moving to the right.

6.3 f1

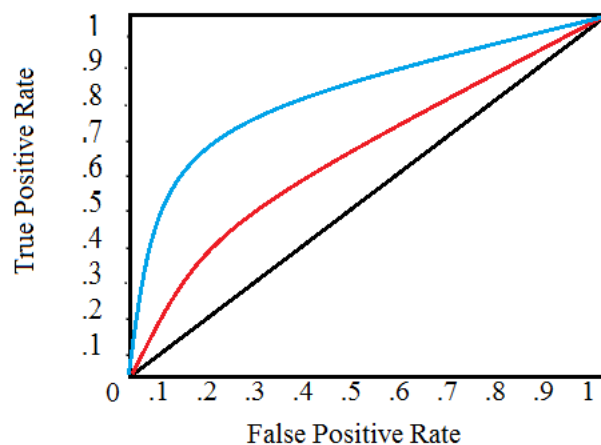
- F1-Score: The optimal combination of recall and precision: the harmonic mean of the two
- $$f1 = 2 * \frac{\text{precision} * \text{recall}}{\text{precision} + \text{recall}}$$
- Note: harmonic mean is used instead of average to punish extreme values (e.g. precision 1.0 and recall 0.0 \rightarrow average 0.5, but F1 = 0)

6.4 fpr

- We have already mentioned the **TPR**, now, the false positive rate (**FPR**) is the probability of a false alarm
- The portion, of all negatives, that were labeled as positives (\rightarrow false positives):
- $$fpr = \frac{FP}{FP+TN}$$

6.5 roc_auc

- Summarize the performance of a classifier over all possible thresholds
- The receiver operating characteristic (ROC) curve plots the TPR versus FPR as a function of the model's threshold for classifying a positive



Two classifiers are shown, one in blue, one in red. The black line is a random classifier.

Keep in mind that on the **X-axis** and **Y-axis** we have the portion, of all **negative** and of all **positive** data, respectively, that was predicted as positive

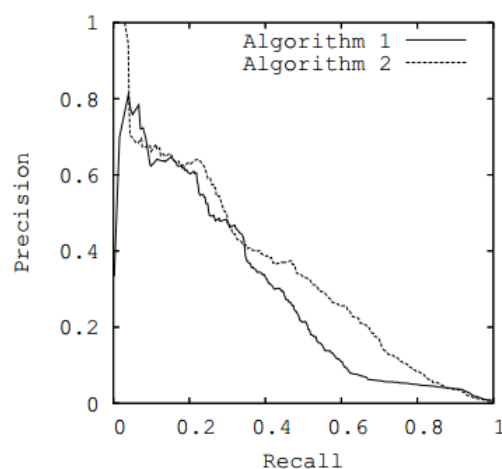
As we increase the threshold we include fewer and fewer data (also data labeled as positive), thus moving down to the bottom left (\rightarrow decision threshold = 1.0) on the corresponding curve; top right \rightarrow threshold = 0.0

That means in the top right corner all data is predicted as positive and in the bottom left corner all data is predicted as negative

- Now, to evaluate / quantify a given ROC, we calculate the area under the curve (**AUC**)
- The AUC is a metric between 0 and 1, where higher values signify better capability of achieving a blend of precision and recall (since a higher peaking curve means higher **tpr**)
- Note: the random classifier will achieve an AUC of 0.5.

6.6 pr_auc

- This one stands for **Precision Recall AUC** (see: <http://www.chioka.in/differences-between-roc-auc-and-pr-auc/>)
- Sample PR-curve:



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

- **pr_auc** is, as expected, the area under the PR-curve. The higher its value, the better the model

6.7 roc_auc vs pr_auc

see: <https://www.kaggle.com/general/7517>


- In short, if the class imbalance problem exists, **pr_auc** is more appropriate than **roc_auc**

If TNs are not meaningful to the problem or there are a lot more negatives than positives, **pr_auc** is the way to go (it does not account for TNs).

- Intuitive explanation:
 - If the model needs to perform equally on the positive and negative class → **roc_auc**
 - If it's not interesting how the model performs on negative class → **pr_auc** (example: detecting cancer; find all positives and make sure they're correct!)

6.8 accuracy


- $\text{accuracy} = \frac{\text{TP} + \text{TN}}{\text{Total}} = \frac{\text{Green}}{\text{Total}}$



6.9 match_rate

- “The proportion of observations matched/not-matched”

- $\text{match_rate} = \frac{\text{TP} + \text{FP}}{\text{Total}} = \frac{\text{Green}}{\text{Total}}$



6.10 filter_rate

- “The proportion of observations filtered/not-filtered” (?)
- `filter_rate = 1 - match_rate`

- $= \frac{TN+FN}{Total} = \frac{\text{Pie Chart}}{\text{Pie Chart}}$



6.11 !<metric>

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

6.11.1 Existing !<metric>s

- !f1
- !precision
- !recall

7 Github Repository

To be found at:

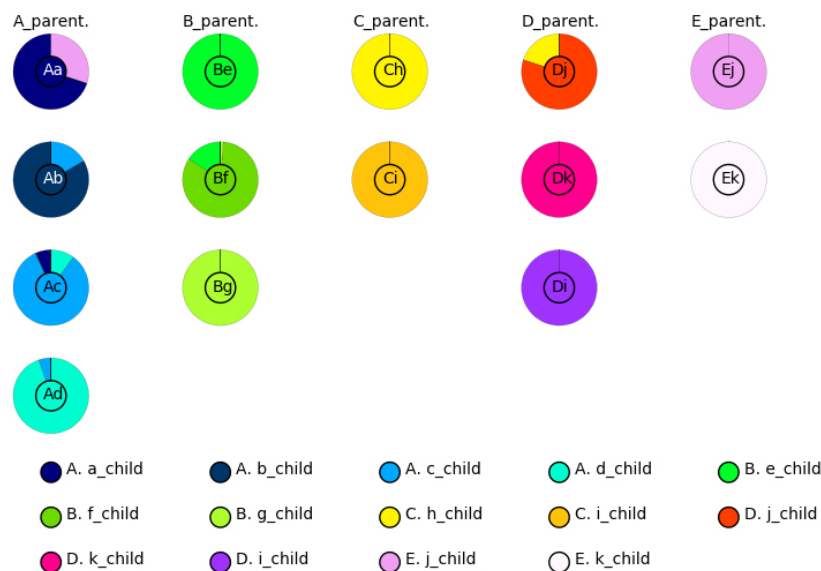
<https://github.com/tguelenman/OresCustomDocumentation>

8 Research

8.1 Confused Pie Plots

source:

<http://www.dietergalea.com/confusion-matrices-alternative/>



Good for multi-class classification with a lot of classes (or probably even good for 5 or more)

- Inner ellipse: show the expected target class
- Outer ellipse: predicted classes
- Rows here: child nodes
- Maybe applicable for itemquality → multiple classes... somehow?
- Python code available: check on website linked above or on Github:
<https://github.com/dterg/confused-pie-plots>

Questions

- **Q:** What's the relation between F1 and ROC_AUC? Article describes them as: F1: "If we want to create a balanced classification model with the optimal balance of recall and precision, then we try to maximize the F1 score."; ROC_AUC: "AUC ... will be greater ... meaning ... better at achieving a blend of precision and recall."

A: F1 is used for one threshold value, ROC_AUC summarizes the performance of a model for all thresholds