ARTICLE

Machine Learning Model Metrics - Can I Trust Them?

My machine learning model performs at 95% accuracy. Should I be worried?

Yes. You should be worried, and you should review your model development and testing process thoroughly.

Often, if metrics appear too good to be true, they probably are. In fact, it's not uncommon for machine learning ("ML") model performance metrics to be inflated even if they seem average. The inflation can be for a number of reasons, and the consequences can be dangerous and expensive.

It has always been essential to ensure that ML models perform within expected limits. However, in recent years this challenge has become even more relevant, due to the democratisation of ML methodologies by no code platforms and coding frameworks like Hugging Face.[1]

The number of articles that focus on the pitfalls of ML metrics is a fraction of those that cover the power of ML-driven methodologies. But the more we progress in the field of ML and Artificial Intelligence (AI), the more we deal with models that are astonishingly good on paper and disappointingly average when faced with unseen data.

Below we describe common pitfalls to avoid when training a model based on our day-to-day experiences of helping clients avoid mistakes. Mistakes which should not be accepted or tolerated anymore if the data science community has the ambition to be recognised as experts, trusted, and followed.

What do we mean by metrics?

In machine learning, accuracy, precision, recall and F1-score are common metrics to measure model performance for classification problems. The equations for these metrics are:

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

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F_1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$$

They are all based on how a model trained to separate a particular category of interest (positive category) from the remaining data (negative category). There are two ways the model can be right:

- True Positive (TP): the model correctly identifies data in the positive category.
- True Negative (TN): the model correctly identifies data in the negative category.

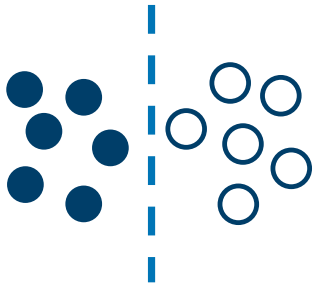
And there are two ways the model can be wrong:

- False Positive (FP): the model incorrectly labels negative category data as positive.
- False Negative (FN): the model incorrectly labels positive category data as negative.

Let's look at an example. Suppose we train a model to look at a photograph of black and white dots and draw the line which best separates the black dots (our positive category) from the white dots (our negative category).

In Figure 1, we see our trained model - the blue dotted line - applied to some new data. Dots to the left of the line are labelled "black", dots to the right of the line are labelled "white".

Figure 1– A trained model to separate black and white dots.

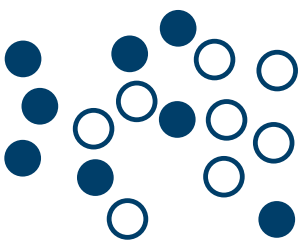


Our model metrics for this dataset would be:

- Accuracy is 100%: we perfectly separated our dots.
- Precision is 100%: all the dots we labelled as black were correctly labelled.
- Recall is 100%: our model doesn't exclude any black dots.
- F1-score is 100%: our harmonic mean, which seeks to balance our precision and recall, is perfect because our precision and recall scores were perfect.

Unfortunately, the real world is never this clean. A more realistic example is shown in Figure 2.

Figure 2 – In real-world datasets the different classes are often inter-mixed, making separation more challenging.

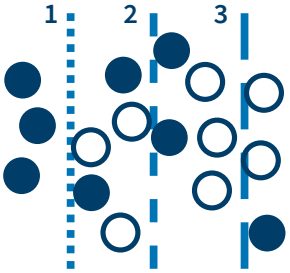


Let's keep things simple with our 'vertical line' model. Where would you draw the line to separate these dots? We can't avoid trade-offs as shown in Figure 3. Model 1 gives us high precision. We can be confident that the dots labelled as 'black' by the model are indeed black. The trouble is, we don't keep that many black dots. Model 3 gives us high recall. Our model correctly identifies most of the black dots, but it also includes a lot of white dots which pollutes our sample. Model 2 is a balance between models 1 and 3.

In the real world, the choice depends on what's important to us and the consequences of the model getting it wrong. For example, a company with a small marketing budget might use the model 1 approach, prioritising precision, to target customers likely to respond to marketing. On the other hand, a company investigating serious fraud might use model 3, prioritising recall, to make sure it doesn't miss serious cases.

Figure 3 – A collection of simple, linear models, resembling rules cannot possibly separate the different classes.*

* (In practice, different algorithms allowing for non-linear separation would be used. However, even then, trade-offs are almost always inevitable.)



The metrics for our three models are shown in table 1.

Table 1 – Models performance metrics

Model	(%) Accuracy	(%) Precision	(%) Recall	(%) F1-score
1	69	100	38	55
2	63	63	63	63
3	56	54	88	67

This is why, when evaluating the performance of an algorithm, it is important to consider multiple metrics as a single metric can be misleading on its own. If the performance of model 1 was described only by "This model has 100% precision" and put into production, the results would be very disappointing.

A confusion matrix can help us intuitively explore trade-offs. The structure of a confusion matrix is shown in Table 2.

Table 2 – Structure of a confusion matrix.*

* The numbers in parentheses are for model line 2 of the black and white dots example.

		Actual values	
		Positive	Negative
Predicted values	Positive	TP (5)	FP (3)
	Negative	FN (3)	TN (5)

Relying on a single metric can create confusion for stakeholders. So, it is crucial to understand what accuracy and the other metrics mean, as well as to have a sense of their shortcomings.

For example, consider a classification model where we are predicting whether a particular news article is relevant to a given topic. Assume that half the articles are relevant to the topic and half are not. A recall of 100% can be achieved by labelling all articles as relevant, but the precision will only be 50%. Although the F1-score provides a balanced view of recall and precision, it is still best practice to provide the breakdown of other metrics to provide a comprehensive view of performance; a particular F1-score can be generated by multiple precision/recall combinations.

Similarly, accuracy can be a misleading metric for imbalanced datasets. For example, if in the last example we have 100 articles, 99 out of which are not relevant to the topic, then predicting all documents to be not relevant would achieve 99% accuracy. However, this model would have no real value.

Am I evaluating my model correctly?

Even if all metrics are provided, it does not necessarily mean that they reflect the performance of the model on unseen data. A machine learning model is typically trained on three data sets: training data, validation data, and testing data.

According to the European Commission's proposed regulation 'Artificial Intelligence Act' (21.4.2021), there is a clear distinction between the data used for different steps of model development:

- "(29) 'training data' means data used for training an AI system through fitting its learnable parameters, including the weights of a neural network;
- (30) 'validation data' means data used for providing an evaluation of the trained AI system and for tuning its non-learnable parameters and its learning process, among other things, in order to prevent overfitting; whereas the validation dataset can be a separate dataset or part of the training dataset, either as a fixed or variable split;
- (31) 'testing data' means data used for providing an independent evaluation of the trained and validated AI system in order to confirm the expected performance of that system before its placing on the market or putting into service;"[2]

The model metrics on the testing data, also known as the 'test set' or 'hold out set' are meant to reflect the model's performance on unseen data. The other two sets are used to train and fine tune the model and hence may overstate the metrics as they are not 'unseen' data.

My metrics are measured on the testing data, is that enough?

Sometimes even with metrics calculated on the testing data, it is still possible to record inflated metrics.

One common cause is data leakage. Data leakage occurs when the model relies on information that will not be available on new data. The most intuitive form of data leakage is when one of the features is only populated after the outcome of the model is produced. Imagine trying to predict if a patient has cancer. If one of the features is if the patient is currently undergoing chemotherapy, then we have data leakage as new patients before diagnosis are unlikely to be receiving chemotherapy.

Many times, when receiving a dataset from a client, the data dictionary and documentation on the collection process is sparse or non-existent. As such, understanding how the data is collected is essential to understanding if the metrics are accurate.

Data leakage is not always as obvious as the example above. Another example of data leakage is duplication in the data. If the original dataset contains duplicate rows, then there could be overlap between the testing data and the validation or training data. This means that the testing data does not measure how the model will perform on new unseen data. This is not always as straight forward as you would expect. Within Natural Language Processing (NLP), processing and analysing human language, near deduplication should sometimes be considered. Take the scenario of supervised topic classification in news articles mentioned earlier. If you have duplicate articles with

different headers or footers in your training and testing sets, this could cause the metrics to be overestimated.

In the case of NLP tasks, it is always good to cross-reference with benchmarks as this allows a sanity check of your results. It gives you the performance in the ideal scenario. In practice you would expect metrics below the benchmark performances as often data processed is of lower quality and there is likely to be some inconsistency and variance in people's labels that are used to measure the performance of the model. If the labels that a model is fed for training and evaluation are not reliable, then the model results will likely suffer.

Finally, it is important to check whether the workflow of a model evaluation reflects the actual problem being solved. For instance, if the data collection step is automated and only works 50% of the time, then this will need to be communicated in the model metrics. For instance, imagine the case where the workflow consists of calling an API which returns a document to perform a classification task to determine whether the document needs to be reviewed. Recall is incredibly important, as the document may contain essential information for that particular use case. However, if the API only successfully returns the correct document 50% of the time due to incorrect information passed to the API, regardless of how good the recall is on the standalone classification model, the whole workflow will likely have a recall less than 50%.

My model is audited, is my job as a data scientist done?

Unfortunately, even if the metrics are audited and found to be reliable, this does not guarantee that the model will continue to perform similarly on unseen data. Data drift is a term used to describe when the data changes from the data used to train the model. This can be for two main reasons: the first is that the original sample data is not representative of the population of data. For instance, take the case of monitoring a topic on social media and news articles. If the model is trained on social media data, then it is unlikely to perform as well on news articles. The second is that data can structurally change over time. Going back to the example of topic monitoring, the trends in discussion will move day-to-day depending on what is occurring in the world and language may change to include new terminology. At a certain point the language may differ enough from the training data, that the model performance starts to dip significantly. Similarly, if the model is trying to predict fraudulent or money laundering behaviour in transactional banking data, we would expect criminals to adapt their behaviour over time to avoid detection. Hence the model would need to be retrained regularly to keep up with new behaviour. In order to ensure that the model is still performing well on new data, it is advisable to have ongoing monitoring in place.

So, can I trust any machine learning metrics?

Yes. If a model is trained and productionised in a way that ensures sources of inflation in model metrics, including the more common ones mentioned above, are accounted for, then a model's metrics should be trusted. There may be a dip in performance as the model hits new unseen data, but with monitoring a new model can be re-trained when performance dips too far.

As a rule of thumb, one should always be suspicious when an ML model's metrics, or the promise of them, are incomplete or high without a detailed introduction to its unavoidable shortcomings.

The views expressed in this article are those of the author(s) and not necessarily the views of FTI Consulting, its management, its subsidiaries, its affiliates, or its other professionals.

[1] https://huggingface.co/

[2] https://eur-lex.europa.eu/legal-content/EN/TXT/HTML/?uri=CELEX:52021PC0206&from=EN

NATHALIE BAKER
Senior Director
+44 7971 246 705
nathalie.baker@fticonsulting.com

GRAHAM JACKSON, PhD
Senior Director
+44 20 3077 0168
graham.jackson@fticonsulting.com

DIMITRIS KORRES, PhD
Managing Director
+44 7790 982 019
dimitris.korres@fticonsulting.com

CLAUDIO CALVINO, PhD
Senior Managing Director
+44 20 3727 1761
claudio.calvino@fticonsulting.com