Stanford MEDICINE

Stanford | Center for Health Education

FUNDAMENTALS OF MACHINE LEARNING FOR HEALTHCARE

MODULE 5 - STRATEGIES AND CHALLENGES IN MACHINE LEARNING IN HEALTHCARE

LEARNING OBJECTIVES

- Recognize the pitfalls and utility of correlative and causative machine learning models in healthcare
- Describe the importance of missing and subclass variables in healthcare applications
- Discuss the important concepts and principles behind model interpretability and performance in medicine including approaches for demystifying the "black box"
- Learn the best approach for handling data in clinical machine learning applications including common challenges like missing data and class imbalance
- Understand how dynamic medical practice and discontinuous timelines impact clinical machine learning application development and deployment
- Become familiar with the relationship of data quantity and error or noise and how it can impact clinical machine learning

CHALLENGES AND STRATEGIES FOR CLINICAL MACHINE LEARNING

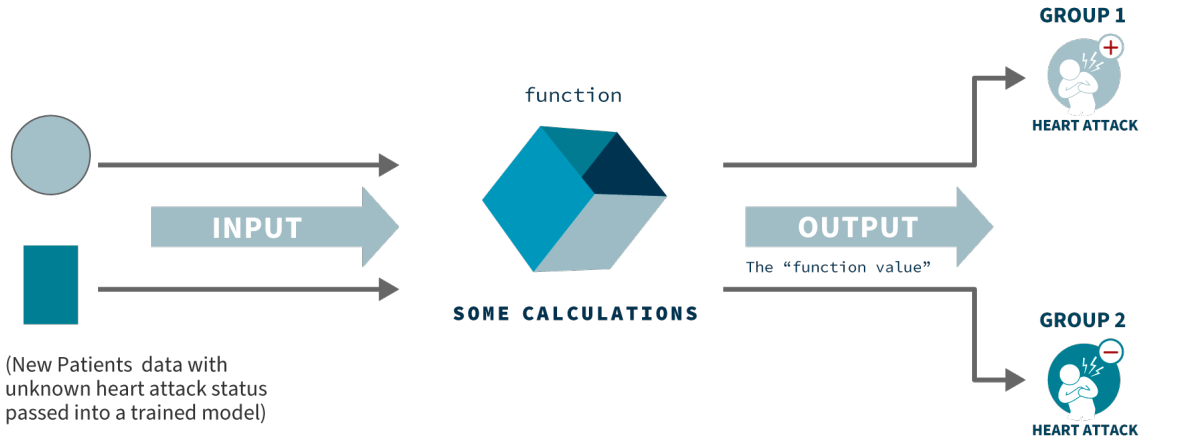
Correlation vs Causation:

- Machine learning methods such as neural networks work by learning associations between inputs and outputs based on whatever signal it can extract from the data
- It is often difficult, or even impossible, to know if the patterns your model exploits when drawing these associations are the result of correlations in the data rather than causative truths
- Lurking variables: Unforeseen variable which cause a model to fit the data based on useless correlations. Known as common response variables, confounding factors

- A serious issue in machine learning more broadly - when a model exploits unexpected or even unknown confounders that have no relevance to the task, it can severely impair or invalidate the model's ability to generalize to new datasets
- Example 1: "Russian tank problem:" An AI model for identifying tanks recognized the weather in the pictures and not the tanks
- Example 2: Chest Xray image: Model which was thought to be accurate was found to be focusing on non-medical cues in the image to draw conclusions
- Example 3: Pneumonia death risk: A high performing model used patient information to identify their risk of dying from pneumonia. It was found that the model was heavily indexing on a correlation between asthma and good patient prognosis in the data
    - In this case the model was not wrong in identifying the correlation between asthma and good patient outcomes
    - However, upon inspection doctors realized that the correlation between asthma and good patient prognosis was the direct result of a hospital policy to admit and aggressively treat asthmatic patients with pneumonia
    - The mistake would be to assume the model's prediction meant that having asthma *causes* a good outcome for pneumonia patients

Sometimes medically irrelevant correlations can still be useful - the best way for this is to **reconfigure the model's application context**, and framing the context of our model applications is something we will spend a lot more time on.

Scenario - Heart Attack Risk. You have one year of retrospective data on 1 million people, labeled with heart attack or no heart attack. You use this labeled data to train a supervised machine learning model to predict heart attack risk within 12 months.



- If you the training data had only medically relevant features, then you may be able to train a medically accurate model which predicts **causal** relationships
- If your model ends up using **correlations** in the data (e.g. between grey hair and incidents of heart attack) it may have different, but still important, use cases
    - If your application context was not for treating patients in a clinic, but instead a model applied for financial, population health or medical practice management, you could actually be pretty okay if you built an accurate model based only on correlative features in your prediction model
- Model which identifies correlation as well as those which make inferences which are plausibly causal can be useful - the trick is to identify what factors the model is indexing on and to consider the relevance of those factors for a given use case

Two reasons are supervised ML models are prone to solving the wrong problem:

1. By design, they develop their own ways of problem solving independent of the programmer
2. Models lacks contextual knowledge and general common sense - this is why we need multi-disciplinary domain experts to help develop, evaluate and deploy models

A tension between **"black box"** and **"interpretable"** algorithms:

- **Black boxes**: Complex models that can make it difficult to understand exactly how the model made a given decision or prediction
- "Interpretable" model algorithms, often more linear models or models with fewer features, make predictions that are more "explainable"
- Tradeoffs then need to be made as deeper networks with more features are often better predictors, while models with fewer features are easier to visualize, understand and explain

Approaches for increasing interpretability of complex models:

- Leveraging multi-disciplinary teams to review false positive and false negative cases predicted by the model
- Testing the model on external datasets, to try to gain insight into causal vs. correlative features learned by the model.
- Focus on developing computational methods to interpret neural network prediction. One example of this is building "saliency maps"
    - Saliency: The part of an input that matters the most to the model in making its prediction

Different ways to produce saliency maps:

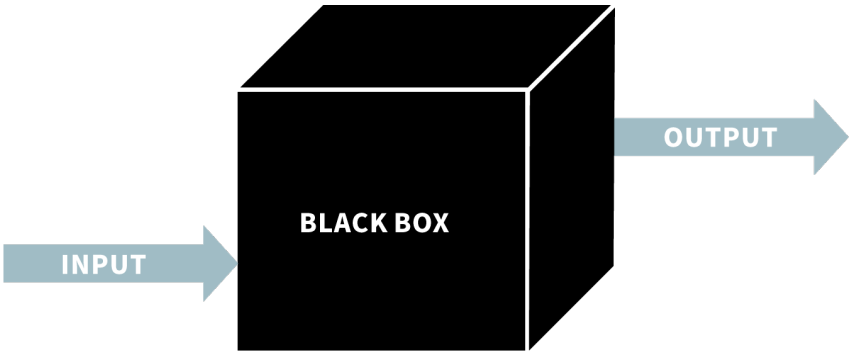
- Class activation maps (CAM): Analyzing the neurons in the final layer of some types of neural networks, to compute how much the neurons that are important for any particular class are firing at every spatial location in an image
- To visualize the relative importance of spatial locations for predicting a particular class, we can plot a weighted sum of the neuron firing in heatmaps. Intuitively, heatmaps show the spatial regions that most strongly trigger the firing of neurons important for predicting the class of interest

Another way of codifying saliency:

- We can compute the change in prediction score that would result from a small change in the pixel value at a particular location of the input. Input locations where a change would greatly affect the prediction score can be interpreted as "salient" for the model. Mathematically, we can compute this since this is just the gradient of the classifier score with respect to the pixel values. And we can also compute and plot a spatial heatmap of these gradient-based saliencies

Frequently used vocabulary for talking about the concept of interpretability: **Transparency, explainability, and inspectability**

- Transparent model: Allows us to easily understand how it works
- Explainable model: Should easily communicate why any particular output is produced
- Inspectable model: Should allow us to probe and inspect the functioning of any part of the model
- The terms' more or less mean what they sound like, and are not generally super technical. They are often used in overlapping and overloaded ways, and many people use them more generally to get at the overarching idea of model interpretability. In other words, they are all ways to get at the notion of opening up the black box of complex models such as deep neural networks.



Internal behavior of the code is unknown

It has become quite common these days to hear people refer to machine learning systems as "black boxes". The black box metaphor refers to a system for which we can only observe the inputs and outputs, but not the internal workings. There have been discussions and debates on the topic of interpretability, and referential metaphors about the black box.

- There remain concerns about black box models, even if they have been properly vetted and can reliably achieve high performance. We have seen how things can go wrong when models learn spurious correlations in the data
- "clinicians and regulators should not insist on explainability because people can't explain how they work for most of the things they do" - Geoff Hinton

There are two distinct "flavors" of machine learning model explainability: **intrinsic** and **post-hoc** interpretability.

- **Intrinsic interpretability** is simply referring to models, often simple models, that are self-explanatory from the start
- **Post-hoc interpretability** is used to understand decisions by complex models that do not have prescriptive declarative knowledge representations or features

INTERPRETABILITY AND PERFORMANCE OF MACHINE LEARNING MODELS IN HEALTHCARE

Intrinsic interpretability is often advantageous in healthcare. It allows doctors to more easily adjust and add to their interpretation of model predictions

- Example: The LACE index predicts 30-day hospital readmission risk and is calculated using 4 intuitive and transparent feature inputs: Length of current admission, admission acuity, patient comorbidities, and number of emergency department visits in the past 6 months
- Systems like LACE allow clinicians to add their own assessment of the relevant factors. It also allows them to consider other features not included in the LACE model and decide their relative importance

With complex machine learning models which consider multitudes more features than LACE, it is practically impossible to apply intrinsic interpretability. Instead we consider post hoc interpretability on a case-by-case basis.

- In such cases it is be harder to adjust the use of the model based on clinical judgement because it would not be possible to know which features, and combinations of features, contributed to the model's the recommendation

Choosing between performance and interpretability is not easy, and often the choice comes down to trust.

- Trust in the development methods, data, and metrics as well as, when available, data about outcomes when using the model are all important.

Use cases suited to black box solutions:

- Text summarization
- Hospital resource triaging
- Pathology slide quantification
- Medical image reconstruction

MEDICAL DATA FOR MACHINE LEARNING

Data types and sources include:

- EHR data
- Both structured and unstructured data types
- Imaging and other pixel based diagnostics
- Genomics
- Peripheral sources of data

In general, it is a good idea to start with a small sample dataset that represents the type of data that you expect will be used in the model. Sample and preliminary analysis and discussion should be done **before** investing a ton of time and resources.

Important things to consider, or to try and glean from your sample data before investing too much time and resources in the project:

- How is the sample data generated?
- How might it fit into a ML workflow?
- What metrics might be useful in evaluating the data?
- How much data might be needed for the project to be successful based on the use case?

- What are the potential use cases in the context of clinical care?
- What is the project's timeline? When and how will data come in?
- Is the data idea needed actually available in the real-world?
- if you are building an application that is expected to produce real-time results, how will the real-time data be sent to the model?
- What preprocessing for feature engineering of the data will be needed in order to run the model?

Note that using clean, pre-processed historical data is likely to give an overly optimistic view of models' performance and an unrealistic impression of their potential value. This is one of the strongest arguments to have domain experts and stakeholders involved early on in development.

When evaluating your data, look out for heterogeneous, incomplete, and sometimes duplicative data types that are created in the routine practice of medicine.

- The heterogeneity of data sources and types can complicate the problems of maintaining data provenance, timely availability of data, and knowing what data will be available for which patient at what time

You can run into problems when your dataset includes a relatively small number of examples of one of the labeled output classes, for instance when you are trying to identify a rare event. This is called a **class imbalance:**

- Class imbalance: Refers to the output labels in the data, and where there is a lot more of one label and much less of another label. This is extremely common in many medical datasets

Having imbalance data does not mean you cannot get good results; it effects how much data you may need.

In evaluation, it is important to look out for the accuracy paradox problem.

- The accuracy paradox: Where your model accuracy turns out to be outstanding, but you have a very imbalanced dataset
- If you have a very imbalanced dataset with - say a data set with a ratio of 1:100 abnormal to normal scans - a model may be able to get very high accuracy in predictions by classifying all the scans as normal, as it will be right 99% of the time. Thus, you may have a very high prediction accuracy while having a totally useless, and thus functionally inaccurate model.

There are alternative accuracy metrics and methods of sampling data which help avoid falling into this problem

- Remember that in a classifier model, if you are simply randomly subsampling your total data for the test set to derive your metrics, then your test data will also have the same class imbalance which can skew some metrics that are tied to prevalence like PPV and NPV

Dealing with the accuracy paradox:

- Choose the proper metrics and re-evaluate performance of your classifier
- Artificially sample a small hold out test set from your data that contains more of a balance of classes. Where possible, simply collect more data, especially instances of the minority class
- Resample your dataset
    - You may try over-sampling (or more formally known as sampling with replacement)
        - This is best for situations where you do not have a lot of data in the first place and your data is also imbalanced
    - You may try under-sampling (remove instances from the over-represented class)
        - This works best when you have sufficient data for the smaller class
- Adjustment your model to account for the imbalance
    - E.g. Train your machine learning systems in a setting that includes "rewarding" (via math!) the model more for correctly classifying an important rare class than for the more common or prevalent label.
- Think about metrics
    - Pay special attention to Precision-Recall curves when there is a moderate to large class imbalance
    - Rely more on ROC curves if there is even class distribution
- In serious cases, consider sticking with algorithmic approaches that tend to tolerate these imbalances
    - I.e. decision trees (and related algorithms that extend upon them such as random forests!)

It is important to consider how much data you will need to train your model. Often how much data you can get will come down to how expensive/ cumbersome it is to acquire, curate, clean, label etc a good dataset (personnel, licensing fees, equipment run time, etc.).

In most machine learning algorithms, as you increase the size of your dataset, performance grows accordingly and then reaches a plateau. This plateau can vary depending on the complexity of the algorithm.

- For regression and simpler machine learning models you may have heard of the "1 in 10" rule that suggests the need for at least 10 examples of each label class

- For neural networks and data with more complex features a rule of thumb is somewhere around 1,000 examples for each label class

General factors which impact how much data you need for your model:

- The number of features in the dataset relative to the number of uncorrelated or weakly correlated attributes in the dataset
- Whether or not model performance is up to par on any number of metrics (including but not limited to accuracy)
    - Making a more complex model and or tuning hyperparameters to increase performance can only improve the model in limited ways and can also run the risk of overfitting. So unless the performance of the model is very close to the goal, the best next step is probably still acquiring more data.

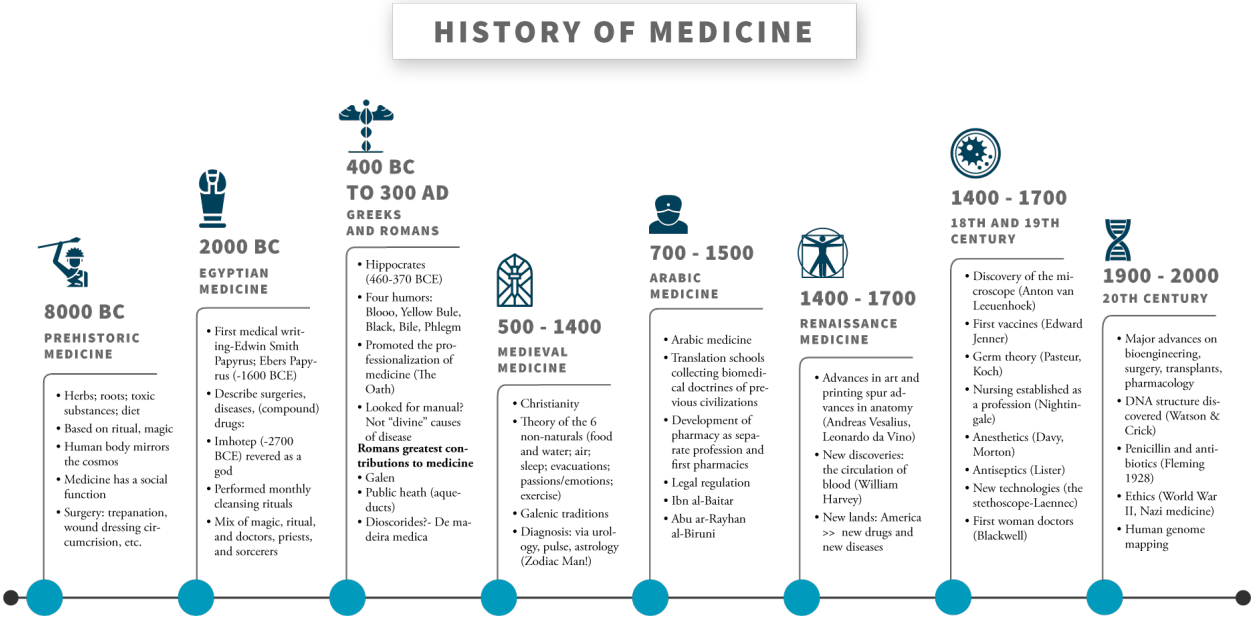
Though more data is often good, naively adding more data may not be helpful, and in some cases could decrease performance, especially in a dynamic field such as healthcare.

There is a crucial distinction between "adding data" and "adding information." Adding more data does not equal adding more information.

In healthcare in particular, we often find that growing datasets by adding historical data often amounts to accumulating arbitrary or outdated correlations.

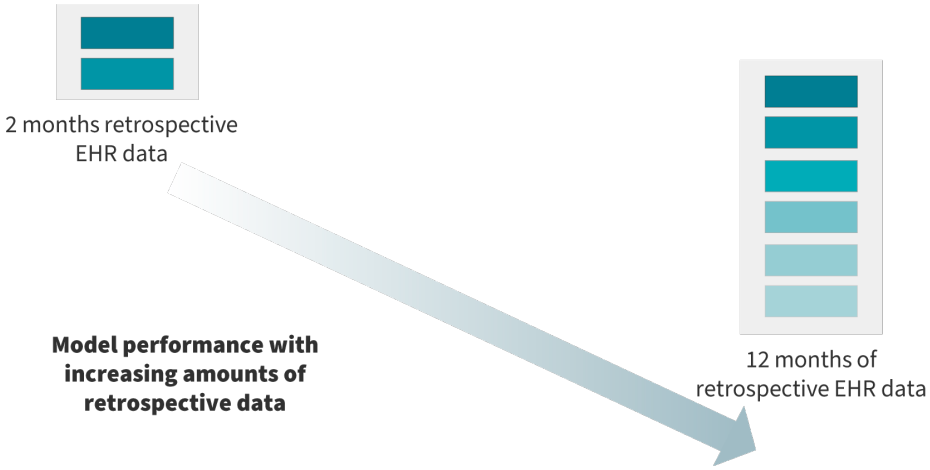
- As the number of these useless correlations in the data expands it can lead to models that learn correlations and cannot generalize and are limited in practice
- These spurious correlations can be hard to detect and can lead to medical decision making based on false correlations rather than real factors.

Conceptually, retrospective medical data has a "shelf life" or "expiration date".



- The dynamic nature of clinical practices over time challenges the presumption that learning from historical clinical data will inform current and future clinical practices

An important relationship emerges in the separation between when data is generated relative to the time learned prediction models are applied and evaluated.



- Example: For example, Stanford research that used retrospective EMR datasets of varying size found that a small dataset that covers about 2k patients and one month of the most recent data was MORE effective in the final performance of a machine learning prediction model than a much larger dataset composed data collected over a 1 year period.

While old data probably can't be used off the shelf, there are techniques we can use to adjust for the context of the time

- By considering, what were the medical practices at the time? How limited were diagnoses? Etc. it may be possible to salvage data, and may be worth doing in some circumstances, but doing so may introduce more noise than relevant information.

Trained clinical models in healthcare that are able to incorporate a continuous stream of data could allow automated methods to rapidly detect and adapt to shifting practice changes to avoid hitting an "expiration date" for effectiveness.

"Garbage in garbage out!" - bad data will result in bad models. No matter how sophisticated the machine learning algorithm is or the data engineering techniques

The choice of data and problem to solve is infinitely more important than the algorithm or the model. **We want high quality data.**

The assessment of, and methodology to create a high-quality dataset are not standardized.

- Among other things, this means that data coming from different sources may vary in its organization. In particular, phenotyping is very important for models that are expected to be deployed across hospitals.
- To help with this problem, when you choose data for any model learning exercise, the label of interest should be noted and described in your work in a *reproducible* manner.

It is also important to be clear how your data and labeling scheme relate to ground truth.

- Labels like mortality have a relatively straightforward relation to readily available determinations of ground truth
- With other labels, like pneumonia, it can be much more difficult to codify ground truth, as that truth may only be expressed in clinical and medical imaging data, which can be hard to mechanistically interpret, and can be fraught with inaccuracies as well as confounding information.
- Look out for labels (like with diabetic patients) that rely on numerical cutoffs that can changed over time in medical practice, and or those that vary by age in terms of the upper and lower bounds.
    - For these labels the consideration of data "shelf life" and treatment changes is very important.

We can expect that our labels will not be 100% accurate compared to a ground truth, thus we need to find ways to **estimate and understand our label noise.**

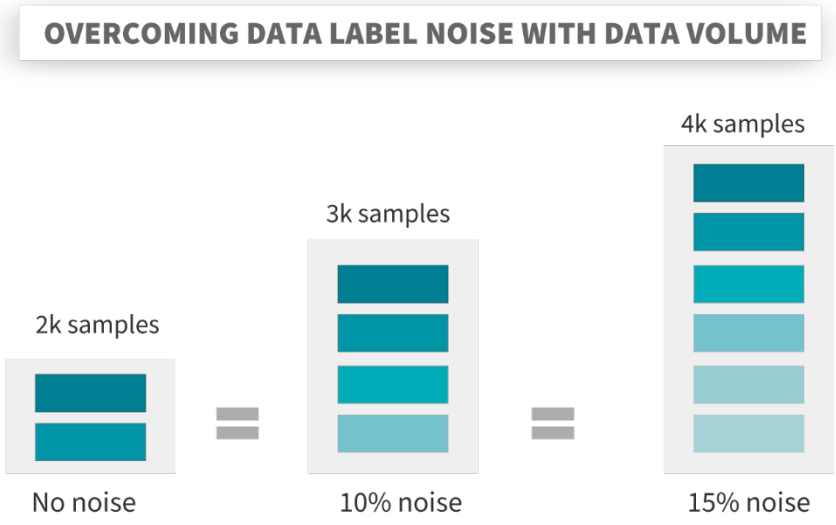
- To evaluate the noise in your labels in a large dataset, start by taking a subset of the data, then use best practices (often with domain experts) to label this subset of data using multiple reviewers. From there, compare the agreement to the original label to evaluate the accuracy of the original labels.
- Note that the label noise could be also a reflection of the difficulty of the labelling task. To investigate the difficulty of a task, try to determine if there is disagreement in the labels among experts.

There are many strategies to address label noise, many of which are outside the scope of this course, but some of the simpler approaches include triangulating multiple labels together.

- This approach works towards reducing noise by adding additional confirmation labels. Applying multiple noisy labels can help narrow down a cohort closer towards ground truth, nearly invariably at the expense of dataset volume.
- After adding the additional labels, you can compare again any change in noise with a subset labeled by hand.

It is important to note that data label noise is inevitable and even very noisy data can train a very good model.

- In fact, there are many cases in which you can overcome data label noise by increasing data volume.



- A study showed that the rule of thumb result was that with 10% noise you need 50% more data and if there is 15% noise you need to double the data.

Formally, the version of supervised learning that involves noisy, bad, or "weak" labels is called **weak supervision**.

- Labels might be considered weak if they are not very accurate or incomplete
- Note that noise, or weak labeling behaves differently in your test and train set. Though you can train on noisy data, often very effectively, if you have noisy labels and separate out a percentage of that data as a test set, you will likely underestimate the true performance of your model

In medicine the problem of getting good, sufficiently sized and well labeled data sets comes up again and again.

- Leveraging and scaling expert medical knowledge to label datasets often presents as a critical bottleneck for many supervised machine learning applications in healthcare

It is important to take the time to remove ALL label noise from the test set to arrive at a true estimation of model performance.

CITATIONS AND ADDITIONAL READINGS

Chen, J. H., Alagappan, M., Goldstein, M. K., Asch, S. M., & Altman, R. B. (2017). Decaying relevance of clinical data towards future decisions in data-driven inpatient clinical order sets. *International journal of medical informatics*, *102*, 71-79. https://www.sciencedirect.com/science/article/pii/S138650561730059X

Chen, J. H., Goldstein, M. K., Asch, S. M., & Altman, R. B. (2016). DYNAMICALLY EVOLVING CLINICAL PRACTICES AND IMPLICATIONS FOR PREDICTING MEDICAL DECISIONS. *Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing*, *21*, 195–206. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4719775

Kelly, C.J., Karthikesalingam, A., Suleyman, M. *et al.* Key challenges for delivering clinical impact with artificial intelligence. *BMC Med* **17,** 195 (2019). https://doi.org/10.1186/s12916-019-1426-2 https://bmcmedicine.biomedcentral.com/articles/10.1186/s12916-019-1426-2

Office, U. (2020, January 21). Artificial Intelligence in Health Care: Benefits and Challenges of Machine Learning in Drug Development [Reissued with revisions on Jan. 31, 2020.]. https://www.gao.gov/products/GAO-20-215SP