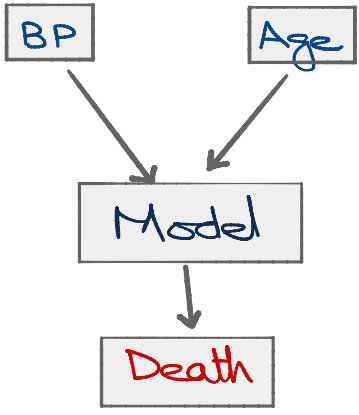
Interpret the AI models:

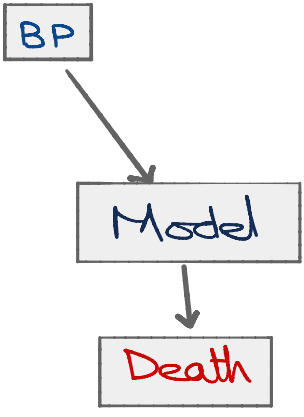
- Drop Column Method: It allows us to interpret a model by finding out how much feature contributed to the model.

Let's say we have a prognostic model;

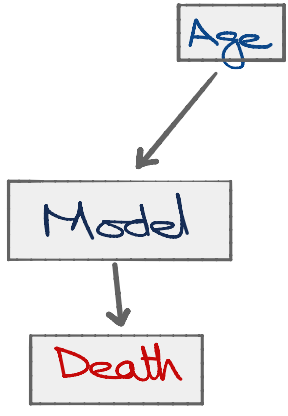


Use blood pressure & Age to get the risk of death.

Importance of each of these features to the model.
In Drop Column method, we train separate model for each feature.



To get the risk of Death.



We can evaluate each of these models on test set using a metric such as the C-index.

Model features	Test Performance
{BP, Age}	0.90
{Age}	0.85
{BP}	0.82

Now, we can determine how important a feature was by looking at the difference with and without that feature.
↳ in C-index

Feature	Importance
Age	perf({BP, Age}) − perf({Age}) = 0.9 − 0.82 ⇒ 0.08
BP	perf({BP, Age}) − perf({BP}) = 0.9 − 0.85 ⇒ 0.05

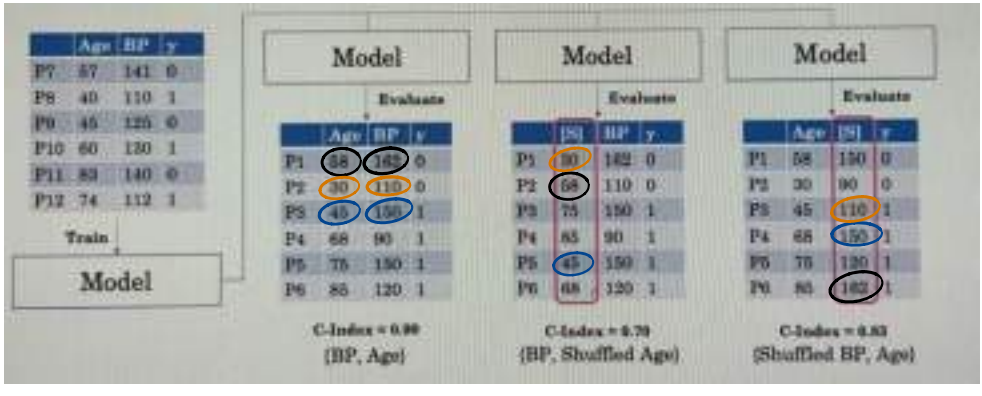
Hence, we realize that Age has a higher feature importance than BP

- Drop column method is computationally expensive. As the number of features increases, the number of models to train increases.

To tackle the challange of training multiple models, we look into another technique called **Permutation method.**

- In this method, we focus on the evaluation of the model.

The idea is before evaluating a model, we shuffle a feature column in our test set.



- Due to the shuffle in the feature data, we observe a drop in the performance.
- The key idea now is to look at the drop in performance between our full model, and our model with that particular column shuffled.

Feature	Importance
Age	perf({BP, Age}) − perf({BP, Shuffled Age}) = 0.9 − 0.7 = 0.2
BP	perf({BP, Age}) − perf({Shuffled BP, Age}) = 0.9 − 0.83 = 0.07

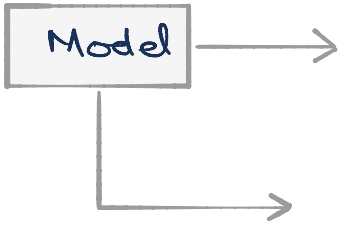
- Hence, it shows that Age is more important feature than blood pressure. The cool thing is that we can use this method without retraining a model.

* Permutation method works for any model that uses data structured in tables. Also there will be slight differences in the resulting c-index depending on how the values are shuffled. To take these variations into account we can take the mean of these different results to have a single value of feature importance.

$$perf_x = \frac{1}{n} \cdot \sum_{i=1}^{n} perf_i^{sx}$$

What if you are not interested in the importance of a feature on the overall model, but on a particular individual prediction?

For example:



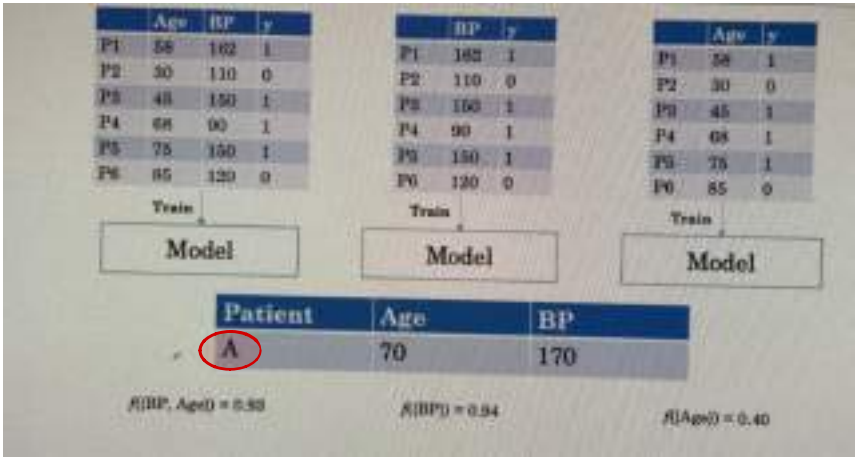
Feature	Importance
Age	0.20
BP	0.07

Patient	Age	BP
A	70	170

- How important the blood pressure was for the prediction of the model on patient A?

Although importance of Age feature is high but we might expect blood pressure would be driving risk because of its high value.

- So, first we train a model to predict outcome y using age and blood pressure. Now we use it to predict the risk for an event. Similarly, we can also train a model to predict the outcome y just using age and blood pressure.



Prediction made just using the BP is same as made by the full model.

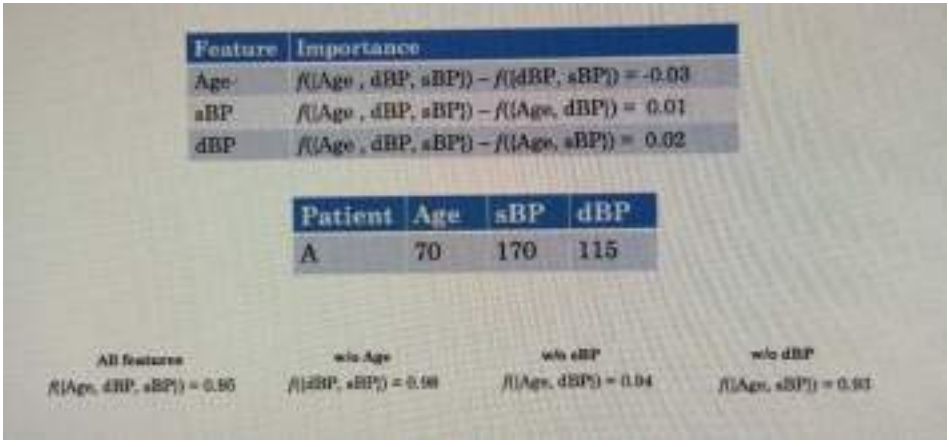
We could compute the importance of a feature for patient A;

Feature	Importance
Age	$f(\{BP, Age\}) - f(\{BP\}) = -0.01$
BP	$f(\{BP, Age\}) - f(\{Age\}) = 0.55$

When we remove the BP feature then the difference is high. Thus the importance of BP is much higher for this patient than the importance of Age.

* It is same as the column method but we are looking at the output of the model, f, not the performance of the models.

- However, this method can fail to recognize important features when there are correlated features.



- We are not able to recognize the importance of the high sBP and the high dBP for this patient.
Let's find out how can we fix this.

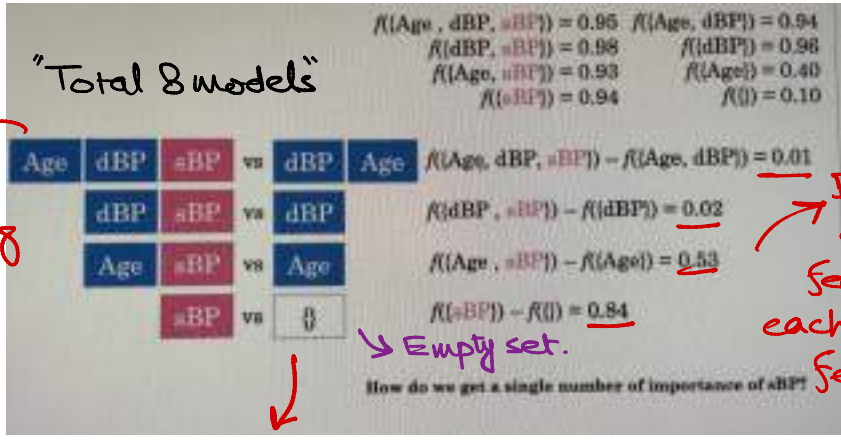
Shapley Values:

- The Shapley values method can assign feature importances correctly even in the presence of correlated features.

Earlier we saw:

$f(\{Age, dBP, sBP\}) - f(\{Age, dBP\}) = 0.01$ (0.95 − 0.94)

With Shapley values we are going to consider all feature sets that contain sBP.



Models containing sBP.

Importance of sBP feature to each of these feature subsets.

Models not containing sBP.

{} - Empty set: We are going to define this model output equal to the expected value of the outcome in the dataset. In this case, 10% of the patient population actually

have the event.

How do we combine these four numbers to get a single number that represents the importance of sBP to patient A?

- We first imagine all possible ways the feature set might be formed.

{Age, dBP, sBP}, {Age, sBP, dBP}, {sBP, Age, dBP}*
= 0.01 = 0.53 = 0.84

{dBP, Age, sBP}, {dBP, sBP, Age}, {sBP, dBP, Age}*
= 0.01 = 0.02 = 0.84

- Now find out the contribution of sBP when it joined the feature set by computing the prediction of the model with the three features - the prediction with the features which came before sBP into the feature set.

"-" sign

e.g. $f(\{Age, dBP, sBP\}) - f(\{Age, dBP\})$
$= 0.01$

* $f(\{sBP\}) - f(\{\})$
$= 0.84$

- Finally we average all of these feature importance:

Importance of sBP for Patient A
$= (0.01 + 0.01 + 0.53 + 0.02 + 0.84 + 0.84)/6$
$= 0.38$

⇒ This importance value is called the Shapley Value and is represented by "I".

- Similarly we can compute the shapley values for the other features.

Patient	Age	sBP	dBP
A	70	170	115

Importance for Patient A	Shapley Value (I)
Age	0.10
sBP	0.38
dBP	0.37

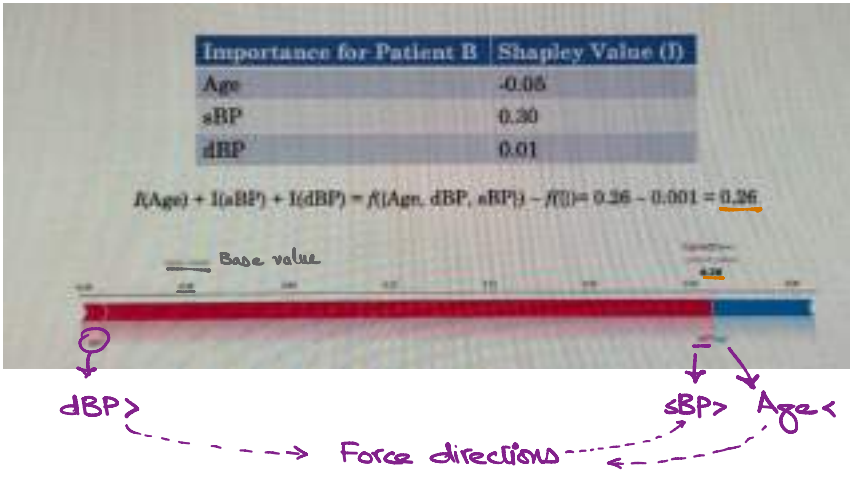
Now we can notice that the contribution of sBP and dBP look much higher relative to age.

- Another cool property of Shapley Values:

$$I(Age) + I(sBP) + I(dBP) = f(\{Age, dBP, sBP\}) - f(\{\})$$
$$= 0.95 - 0.1$$
$$= \underline{0.85}$$

It is telling us how much each feature contributes to the additional risk over the baseline risk of the population.

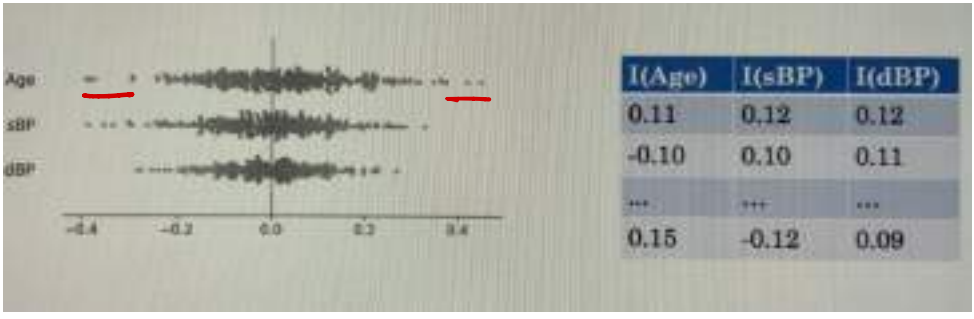
- We can visualize the Shapley values and their total using a force plot.



Shapley values for all patients:

- We can compute the shapley values for all patients in a dataset, then visualize the distribution of the shapley values for each of the features.

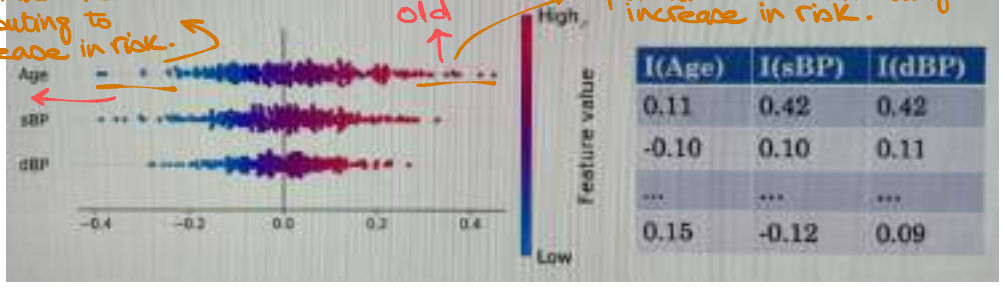
- In the plot below we can see that the age feature has large positive & negative importance values for many patients in this dataset.



I(Age)	I(sBP)	I(dBP)
0.11	0.12	0.12
-0.10	0.10	0.11
...	...	...
0.15	-0.12	0.09

One challenge here is that we can't tell whether we have high shapley values for these patients because they're very old or because they are very young or mixture of the two.

↳ To solve this problem, we can colour the points in the summary plot based on the feature value. Red is here is a high feature value, which means that when a point is red for age, this means that a patient is old. Similarly, a blue point is when the feature value is very low. For age, this means that a patient is young.



I(Age)	I(sBP)	I(dBP)
0.11	0.42	0.42
-0.10	0.10	0.11
...	...	...
0.15	-0.12	0.09

We compute the overall importance by taking the absolute value of the Shapley values for all the patients and then taking the average of those Shapley values for each of the features. So that way we are able to say that age is the most important feature overall.

- This allows to go from individual Shapley values to global importance in a population.

|I(Age)|	|I(sBP)|	|I(dBP)|
0.11	0.42	0.42
0.10	0.10	0.11
...	...	...
0.15	0.12	0.09

Mean(|I(Feature)|)

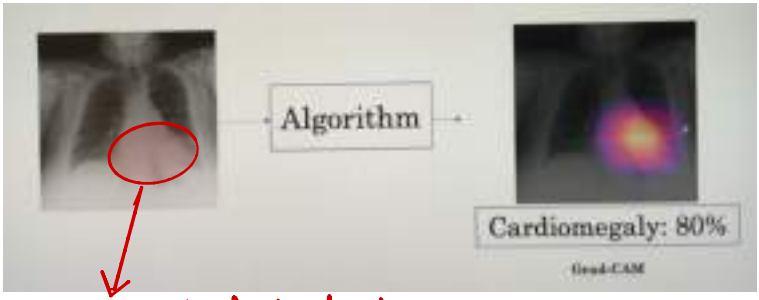
- The newer extensions such as SHAP, don't require retraining of these models and can perform these computations of the Shapley values efficiently.

Interpreting the Deep learning models

- Let's see how can we interpret a CNN model.

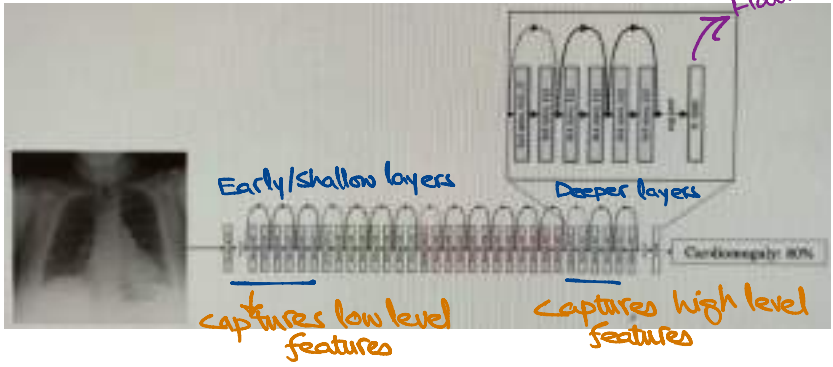
How can we produce heat maps to identify locations in the chest x-ray that contributes most to a network's classification.

In this chest x-ray the patient has an enla-rged heart also called cardiomegaly.



Our model is looking at this part to make decision but how can we get the heat map (using Grad-CAM)?

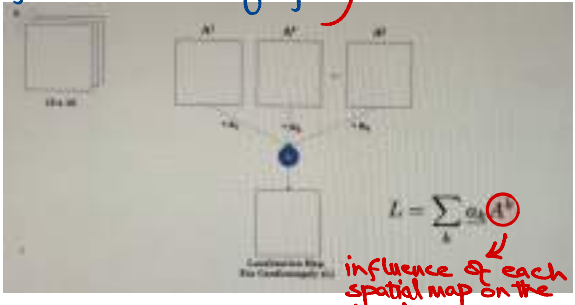
→ When an image is passed through CNN network it passes through many layers.



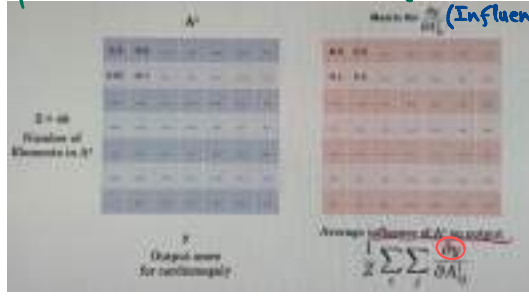
Once the features are flattened, the spatial information is lost. So, we want to visualize the features before the flattening happens.

- We can get the spatial maps from the output of the last convolutional layer when the image is passed in. These maps are typically much smaller in size than the input, depending on the choice of architecture. K is the number of spatial maps. How can we go from these spatial maps to a heat map over whole image?

- We are going to weight each of these maps with a weight given by a, and then add all those up to get a localization map, L, for cardiomegaly.



- To compute the weights a_k, we first find the influence of each feature within the A_1. The influence of each feature on y can be computed by taking partial derivative w.r.t to the feature.

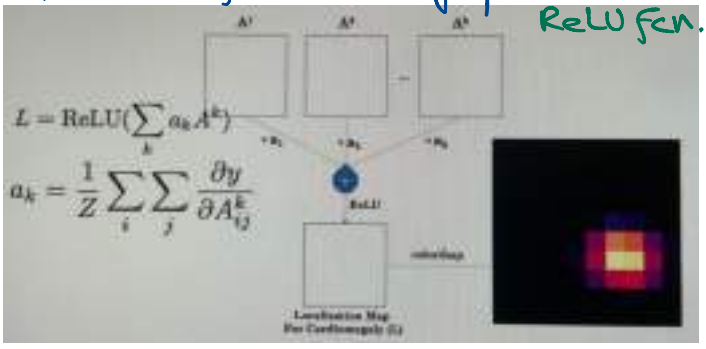


* y presents the output of the network for cardiomegaly right before the activation fn.

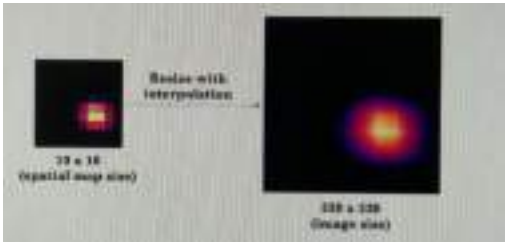
- The average influence can then be used as the weight we're seeking for the spatial map.

$$a_k = \frac{1}{Z} \sum_i \sum_j \frac{\partial y}{\partial A^k_{ij}}$$

- We are only interested in features that have a positive influence on the score for cardiomegaly. This can be done by applying ReLU fcn.



- Then we resize the heat map & overlay it on the original image.



- If the model has multiple possible output classes, the GradCAM can be extended to any class C by using the particular score for that class y_c to get weights for that class a^c_k.

THANK YOU