Beyond prediction

Explainability and confidence

Let's talk about last lab a bit

The image feature actually helps with some tweaks

- Remove weak features
- Do it by price section
- Treat different types differently

"[...] For a bank to predict people who would default we build two models. One that was very simple, is actually linear prediction of some kind or a local linear prediction [...] that economist could interpret [...], and then we built the second model, big neural net with some sort of latent variables [...] very sophisticated, worked very well, way better than the other one.

When presented to them, ***"which one you want, the one that is explainable, or the one that actually works"***

Every single time I've talked to people who wanted to use a machine learning system, they look at you in the eye and say *"you know, is very important to have an explanation that is used for making decisions, [...] we aren't interested in anything unless you have an explanation[...]"*

And then you show them two models: one that has an explanation (kind of) [...]

Yann LeCun
The Great AI Debate - NIPS2017

"Every single time they'll take the second one! **Which means they don't really care about explainability, it's a way for them to be reassured, and most people are more reassured big a good AUC curve than the possible explanations of the decision a model internally makes.**

Also NNnets are not black boxes, you can do sensitivity analysis to evaluate how each value impacts the score"

https://www.slideshare.net/keepler/keepler-understanding-your-own-predictive-models-136332955

ML products

Customer facing

Recommendation systems, maps (traffic estimate), speech2text

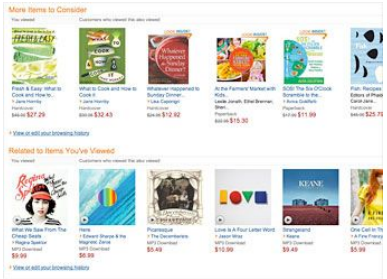
Best guess by the model
Mostly automatic (check deposit by app)

Internal facing

Loan applications, demand forecasting

Can say "I'm not sure"
Human in the loop
Machine-assisted

Need to say why or give confidence level



Slides courtesy of Phiradet Bangcharoensap

Two levels of understanding

Model level

- Describes the model tendency
- Talks about the behavior on training data

Output level

- Attributes model decision for a given test sample to different features

Model level: Simple example

Logistic regression

$$g(z') = \phi_0 + \sum_{i=1}^{M} \phi_i z'_i,$$

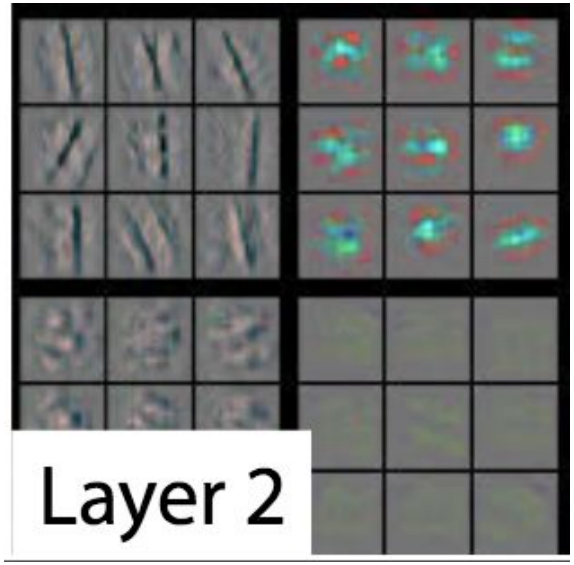
You can say which feature is important from size of the coefficients.

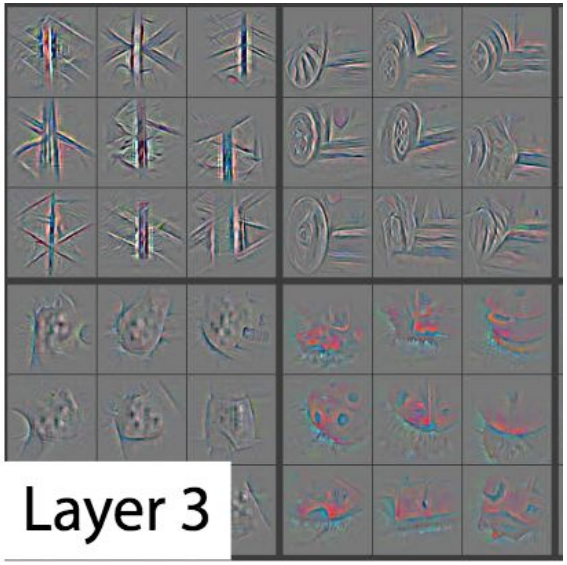
Pros: Simple, easy to understand
Cons: Only model linear effects

Model level: visualizing filters

Plot out the weights of the CNN filters...not so useful in practice

Layer 1





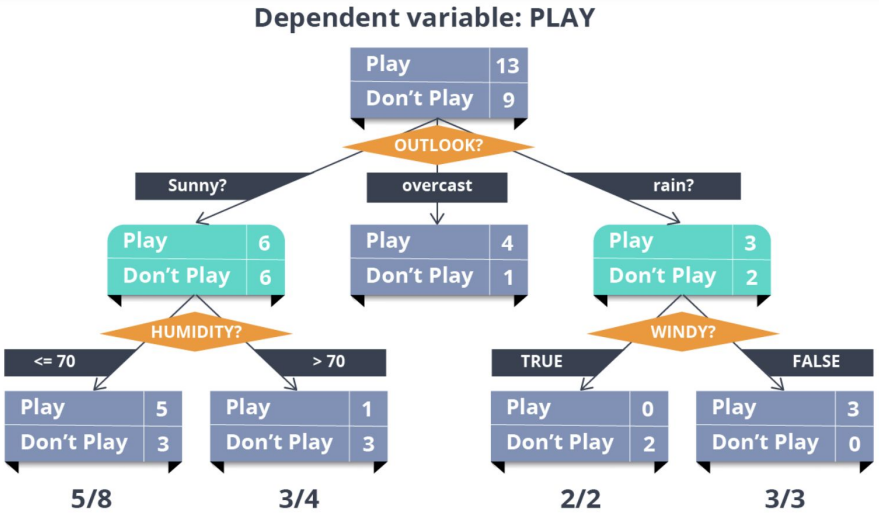
https://arxiv.org/pdf/1311.2901.pdf

Model level: feature importance in random forests

Assign scores based on how the features are use

A node that splits better (better purity at children): high score

A node with more training samples: high weight



https://www.edureka.co/blog/decision-trees/

Usefulness of model level

Feature selection

Gives you confidence that the model is learning reasonable things

Two levels of understanding

Model level

- Describes the model tendency
- Talks about the behavior on training data

Output level

- Attributes model decision for a given test sample to different features

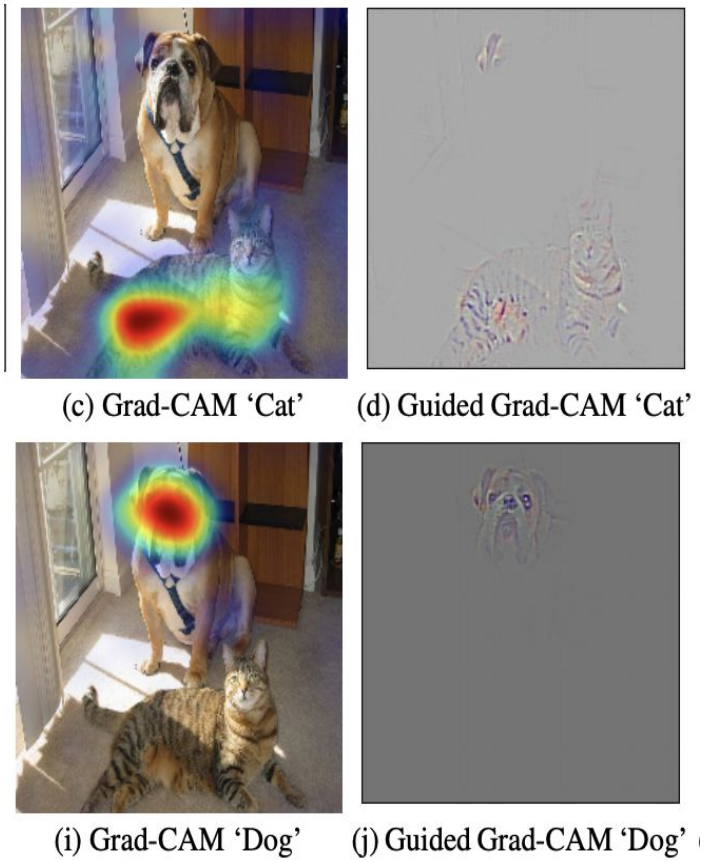
Output level: key ideas

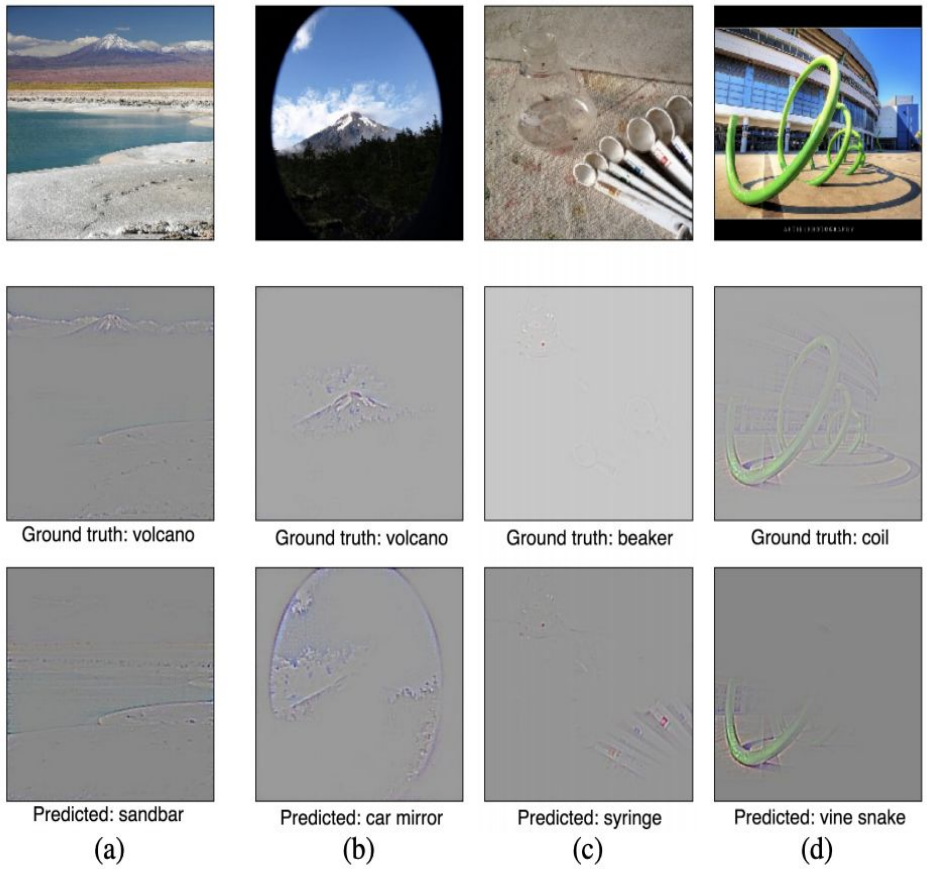
A feature is important if I tweak the feature and the output change a lot.

If I tweak the output, how does the gradient flows

Gradient descent onto the features!

Output level: Gradient-weighted Class Activation Mapping (Grad-CAM)





https://arxiv.org/pdf/1610.02391.pdf
https://github.com/jacobgil/keras-grad-cam

Output level: key ideas

A feature is important if I tweak the feature and the output change a lot.

Mostly useful for images types

Have a simpler model (**surrogate model**) explains the complicated model.

Converting things to logistic regression

Additive feature attribution

$$g(z') = \phi_0 + \sum_{i=1}^{M} \phi_i z'_i,$$

Simplified input features z' have binary values

z' can recover original features x, $x = h_x(z')$ (This mapping depends on x)

Goal: make $g(z') = f(h_x(z'))$. Then we can explain f in terms of simplified features. (**Solved by optimization**)

Example of simplified inputs

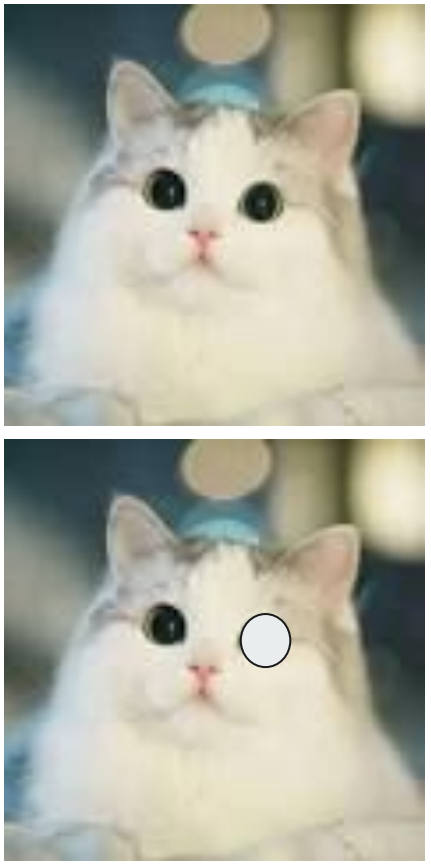
Original x = image

z = 0 if patch is not present

z = 1 if patch is present

$h_x(z) = x$ if, $z=1$

$h_x(z) = x$ with missing patch, if $z=0$

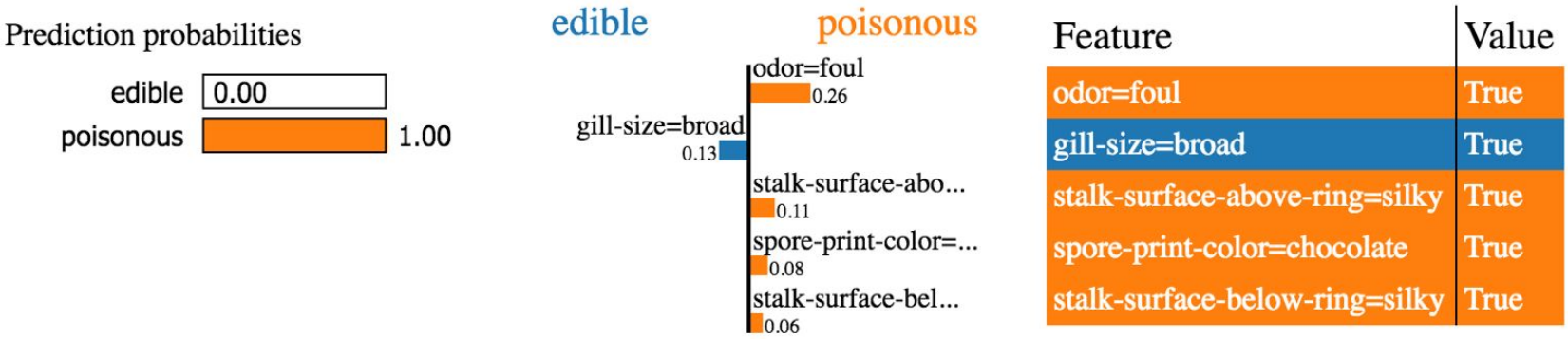


Additive feature attribution

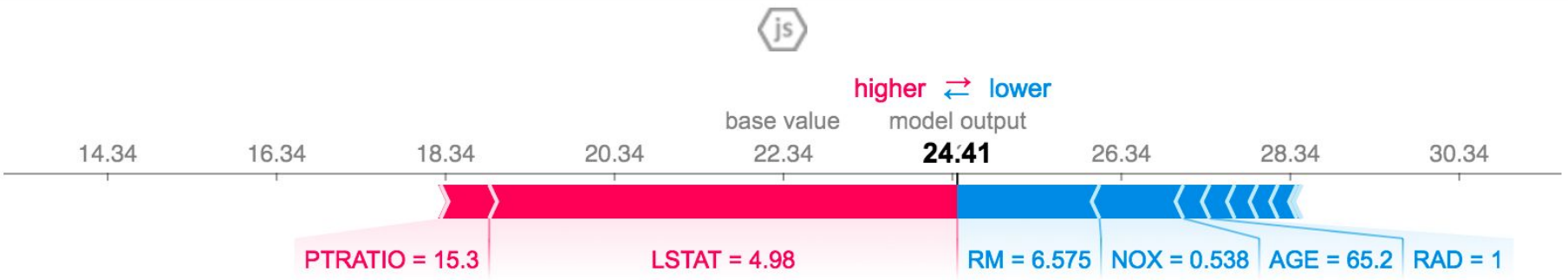
$$g(z') = \phi_0 + \sum_{i=1}^{M} \phi_i z'_i,$$

Many methods can be considered as additive feature attribution. For example LIME and SHAP

LIME

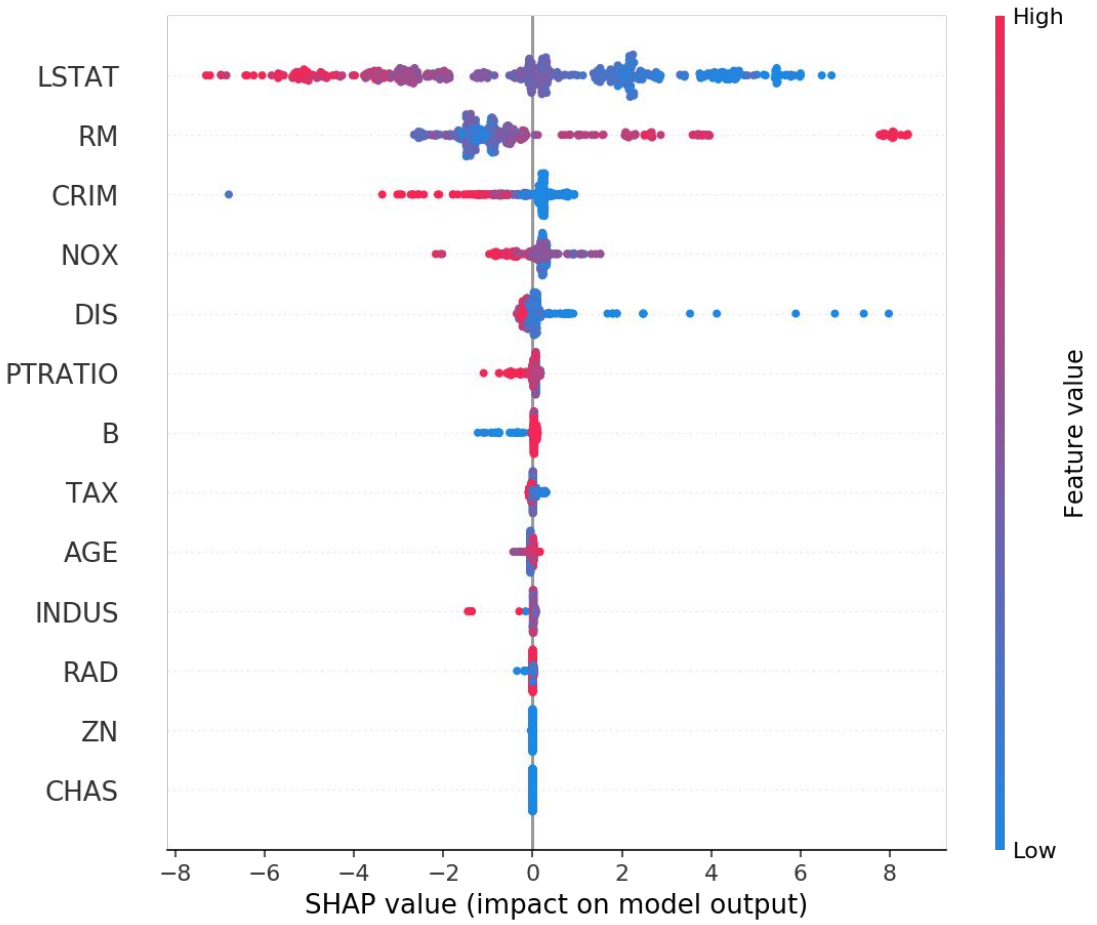


SHAP example



https://github.com/slundberg/shap

SHAP example



Notes on additive attribution

If the features are correlated, it's hard to divide the attribution

F_1 = temperature

F_2 = wind speed

F_3 = wind speed/2

PM2.5level = $200 - 5*F_1 - 4*F_2 - 3*F_3$

Notes on attributions

It does not necessary tell how to improve.

If the explainer says the sales is low because of weak marketing

Does not mean increasing marketing will improve sales.

See Causal Inference

Lab 1

Electricity usage prediction using deep learning

Part1: Predict next day usage from day of week, previous day usage, etc.

1.1 Predict usage

1.2 Predict difference in usage

Part2: Use SHAP to study the effect of features

Confidence

ML products

Customer facing

Recommendation systems, maps (traffic estimate), speech2text

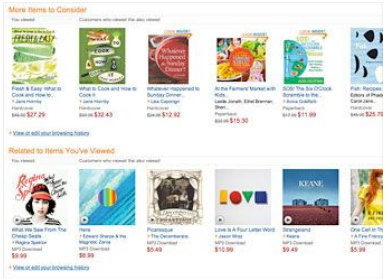
Best guess by the model
Mostly automatic (check deposit by app)

Internal facing

Loan applications, demand forecasting

Can say "I'm not sure"
Human in the loop
Machine-assisted

Need to say why or give confidence level

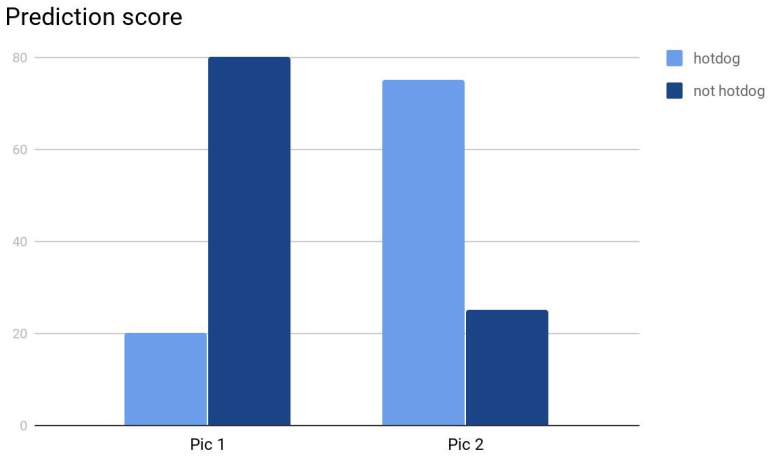


Slides courtesy of Phiradet Bangcharoensap

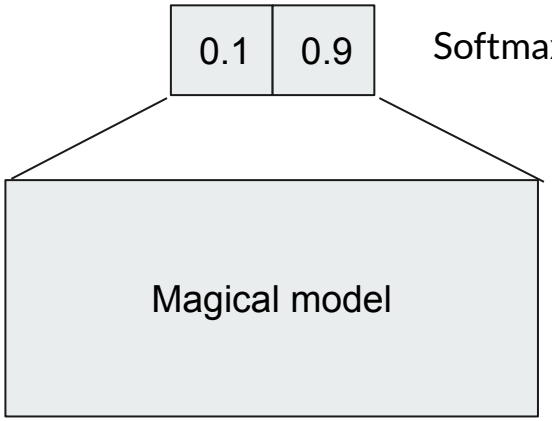
Confidence score

Practical models are not only accurate, but need to be able to state its confidence

Confidence = probability of being correct

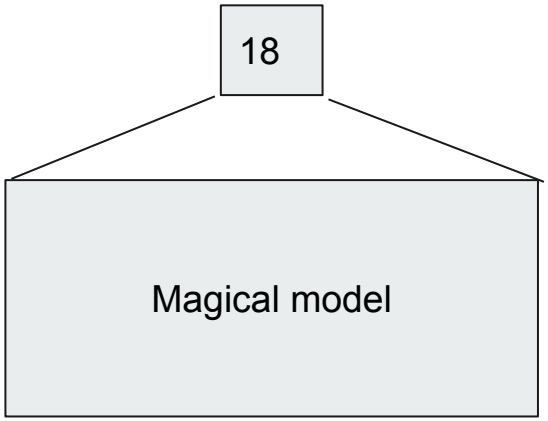


Naive way for confidence



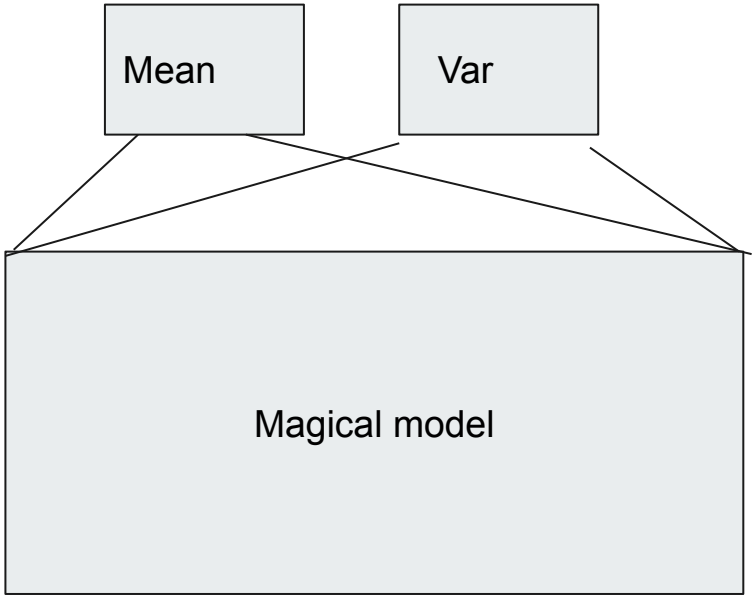
Confidence level = 0.9?

What about regression?



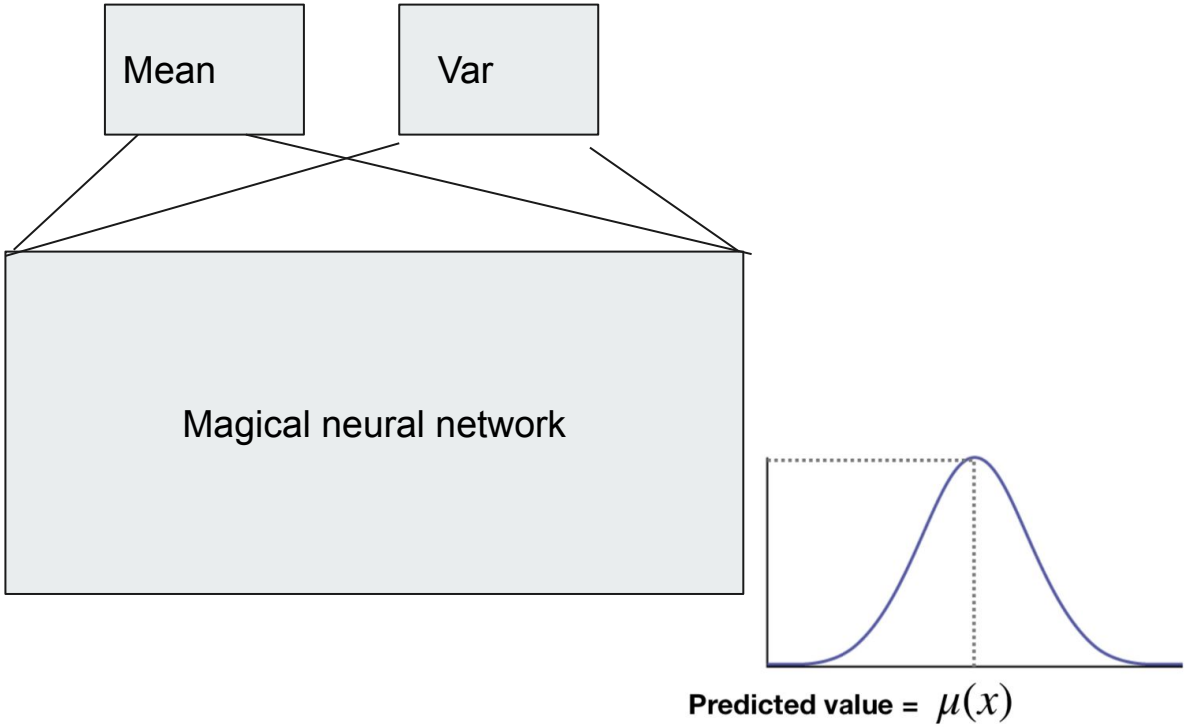
Confidence level = ???

A Naive way for regression (1994!)



Estimating the mean and variance of the target probability distribution. IEEE 1994

A Naive way for regression (1994!)

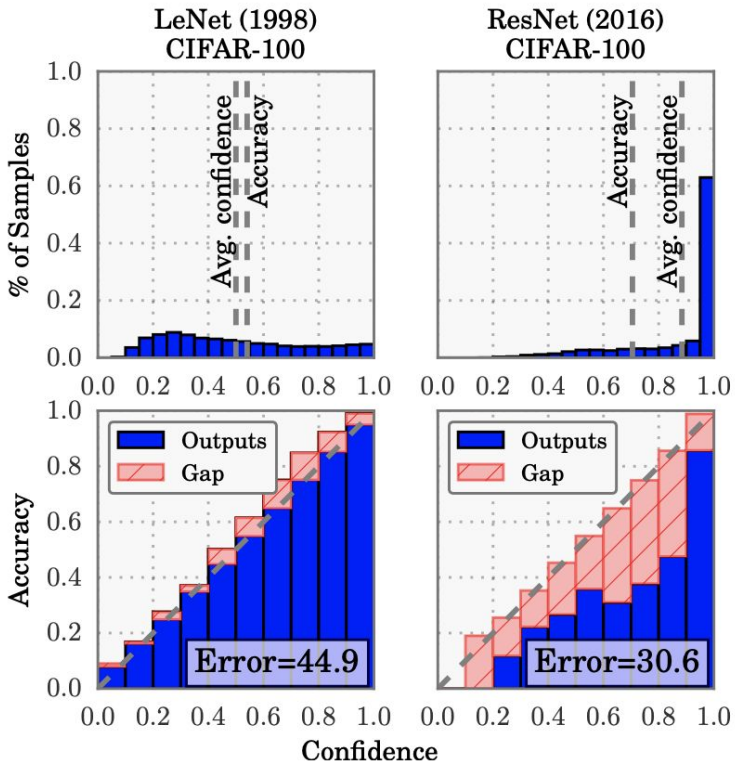


Estimating the mean and variance of the target probability distribution. IEEE 1994

Poorly calibrated confidence

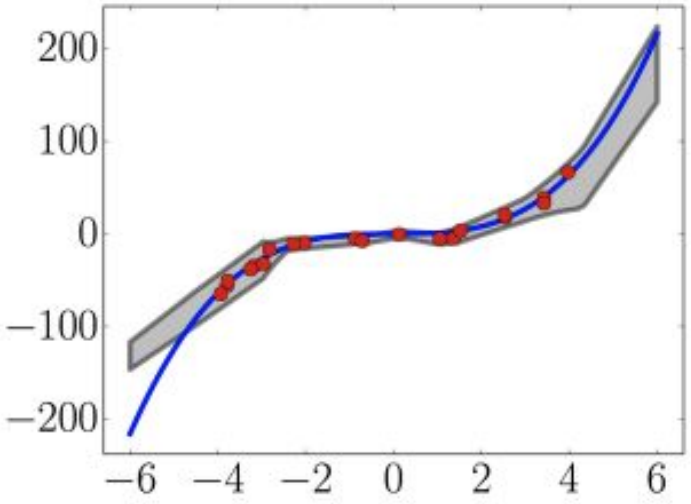
Deep Neural Networks are always overconfident!

Confidence = Probability of being correct

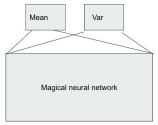


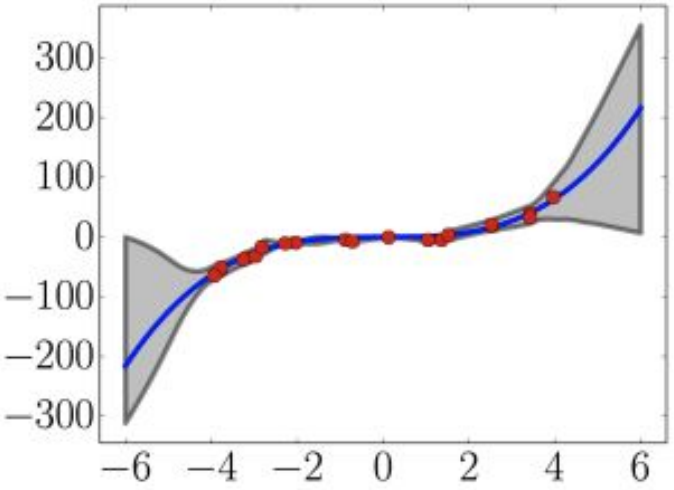
On Calibration of Modern Neural Networks JMLR 2017

Out of distribution problem



output from predicting variance via maximum likelihood





output from ensemble

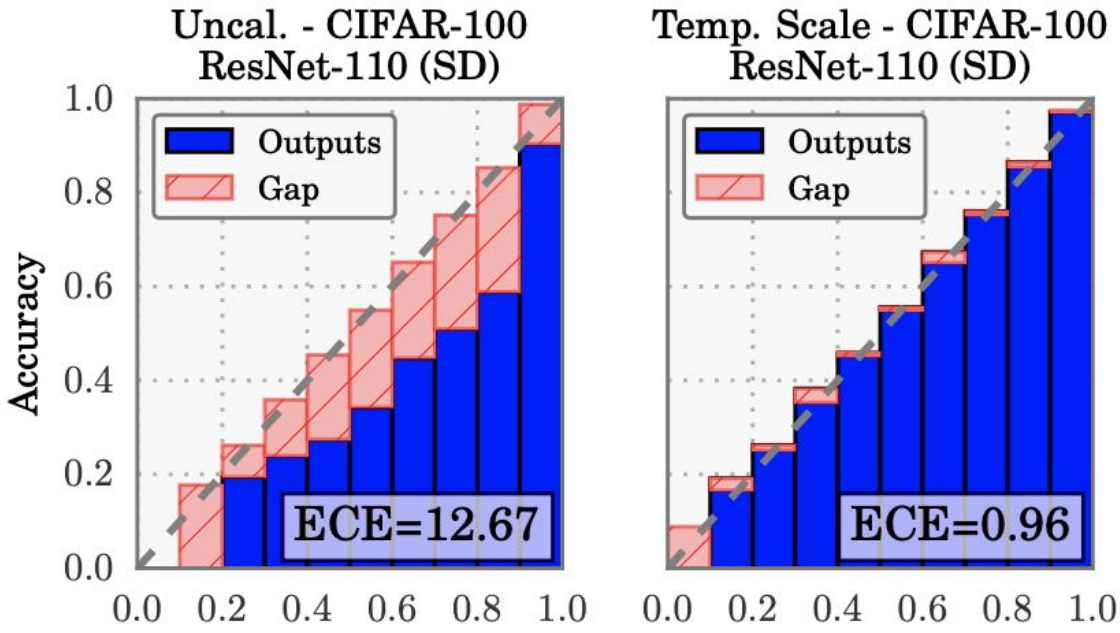
Simple and scalable predictive uncertainty estimation using deep ensembles. 2017

Neural network calibration

Make the confidence output follows the probability of being correct.

How?

Need a seperate training set to train the calibration (calibration set)



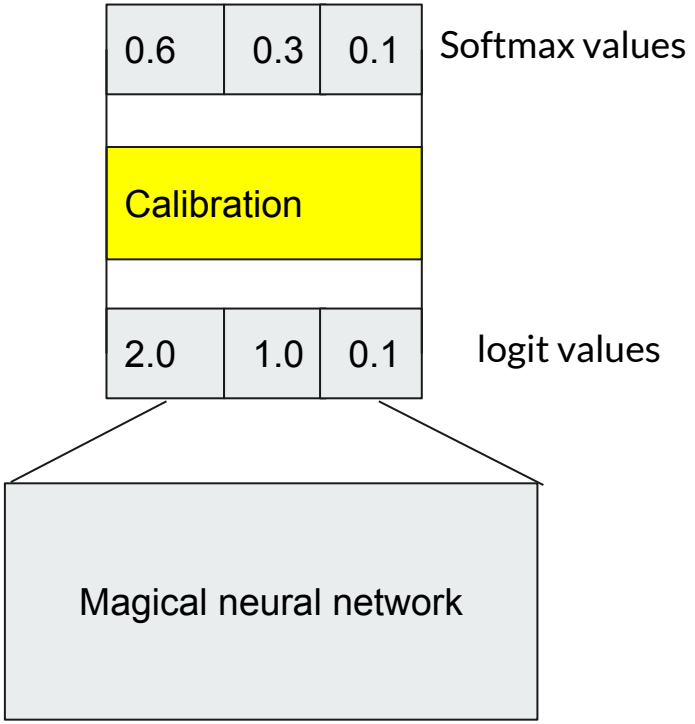
Overview of methods for calibration

1 Calibration

2 Ensemble

Calibration

Post processing after getting pre-softmax output (logit)



Calibration - Temperature scaling

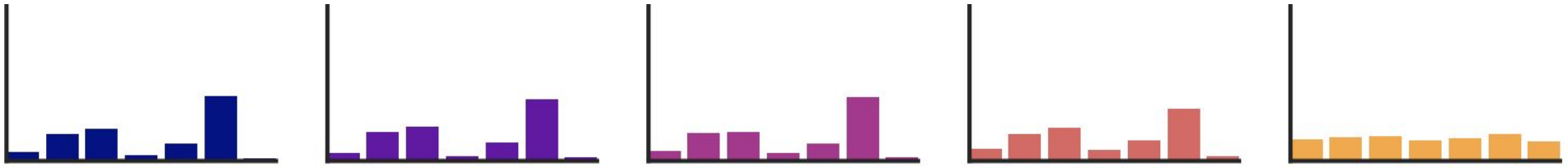
Add a temperature T to rescale the softmax

$$\hat{q}_i = \max_k \sigma_{\mathrm{SM}}(\mathbf{z}_i / T)$$

T is tuned to maximize log likelihood on the calibration set

Low T

High T



https://arxiv.org/pdf/1611.01144.pdf

Other calibration methods

Histogram binning
Bayesian binning into quartiles (BBQ)
Matrix and vector scaling (model on top of model)
Isotonic regression (model on top of model)

Try different methods on your dataset. No absolute best.

Model ensemble for variance estimation

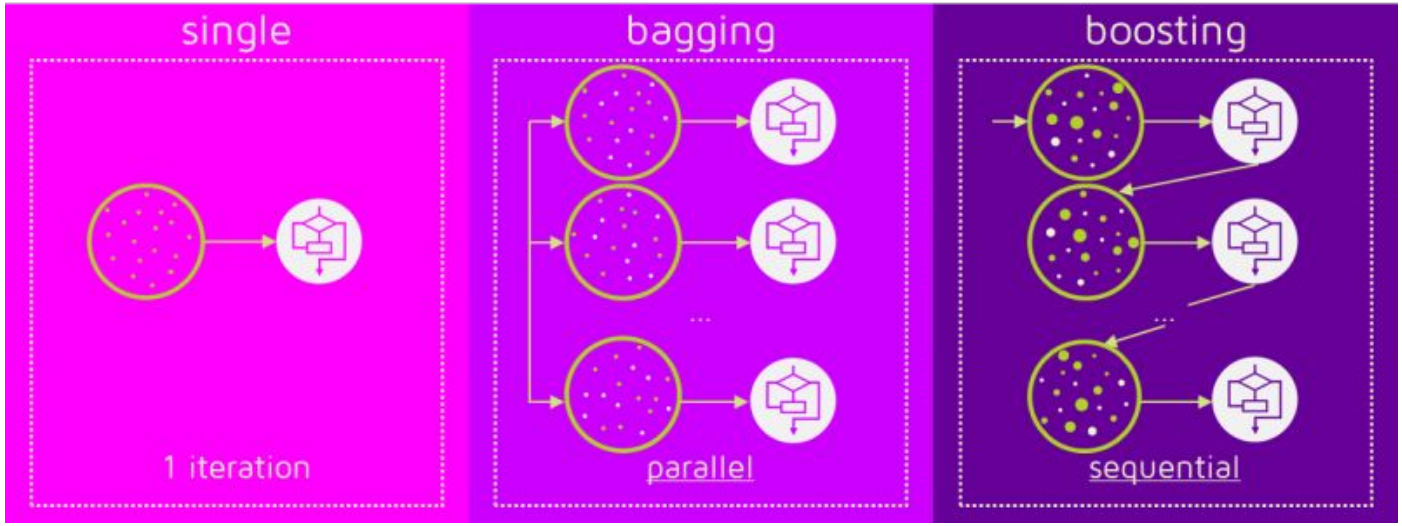
Intuition: if you have two independent experts, if they agree you can be confident about their judgement

Model ensemble for variance estimation

1. Train multiple models and have them predict the same input.
2. Calculate the mean and variance of the prediction
3. The confidence can be from calculating the area under the Gaussian pdf

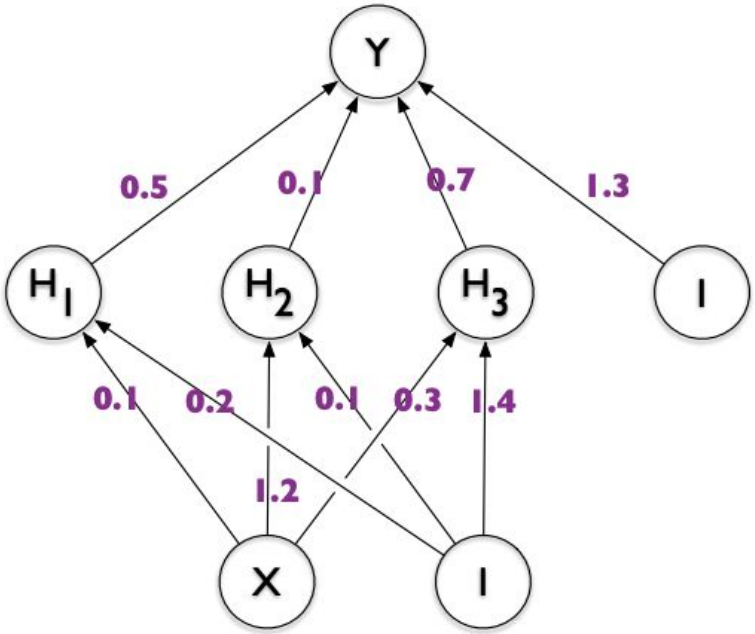
Training multiple models?

Random forest are inherently random. Training the model multiple times will give different models.

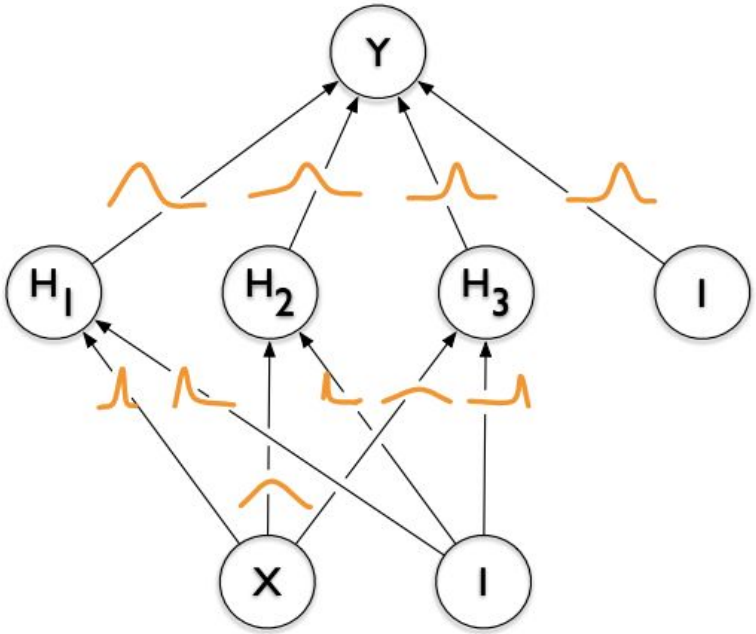


Bayesian Neural Network

Normal NN



Bayesian NN

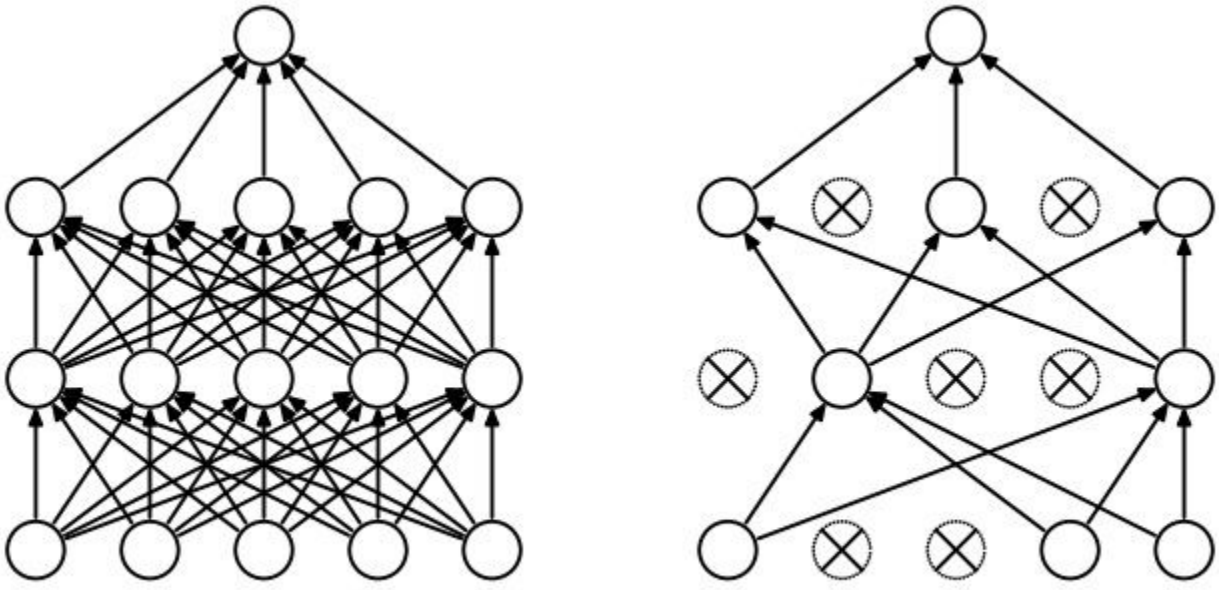


Problem: hard to do inference. Need to find $P(y \mid x, D) = \sum_{\theta} P(y \mid x, \theta) P(\theta \mid D)$

Weight Uncertainty in Neural Network. ICML 2015

Training multiple models

Dropout can be used for creating multiple prediction from one neural network model



(a) Standard Neural Net

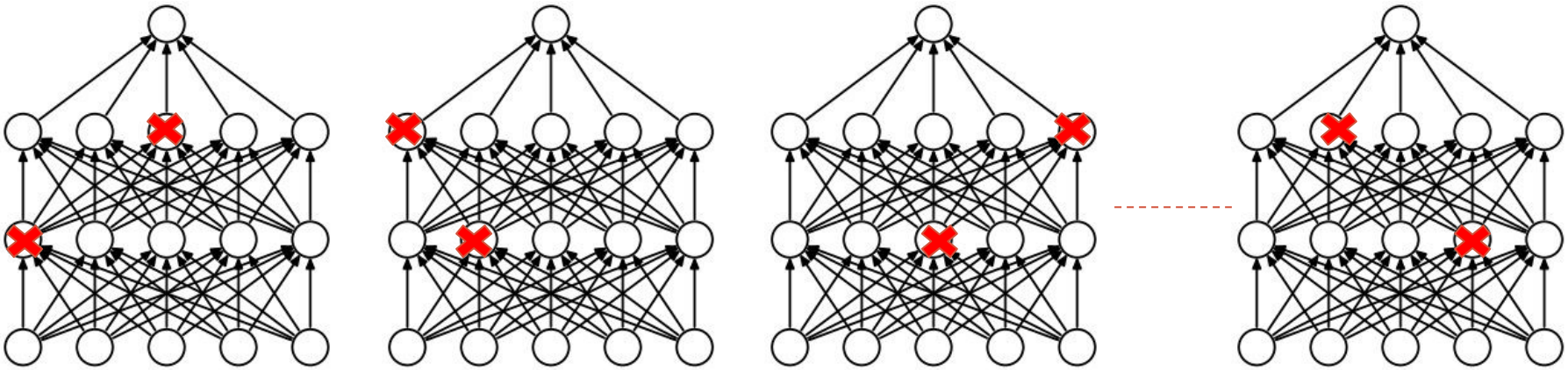
(b) After applying dropout.

Monte Carlo dropout for confidence estimation

compute mean and variance for the answer and confidence

20.2 +- 3.2

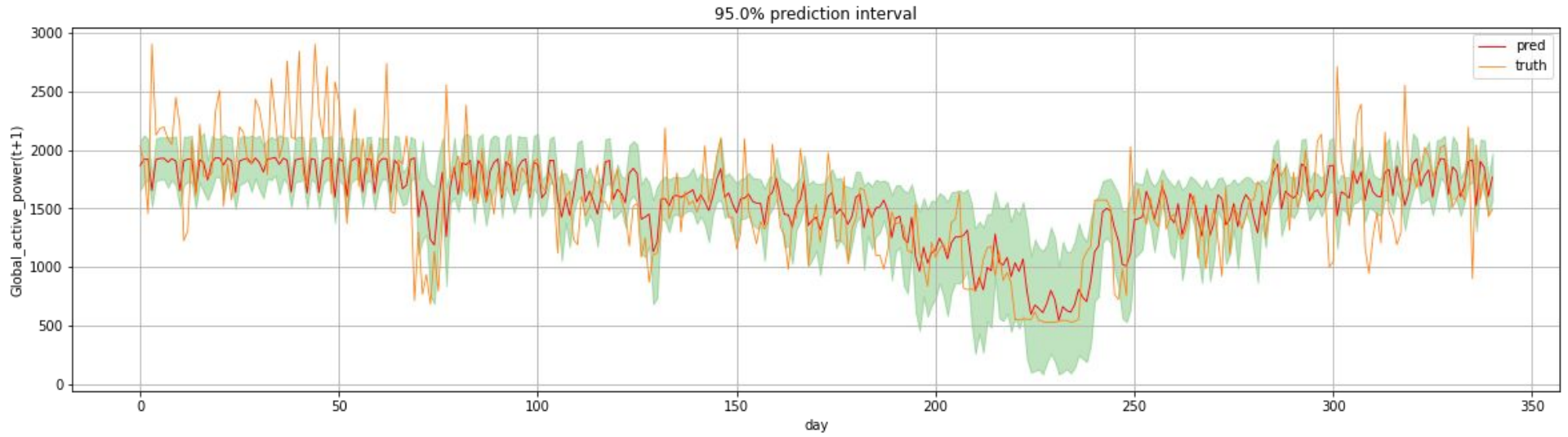
25 27 18 15



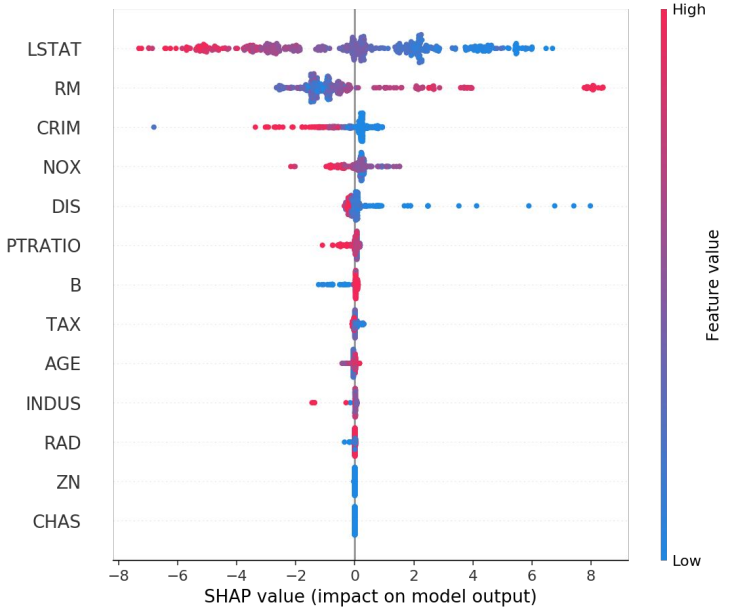
Lab 2

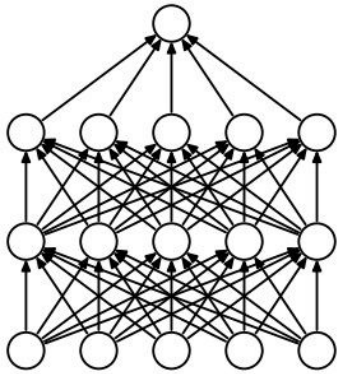
Let’s create confidence intervals for the electricity usage

- Part 1: XGBoost model
- Part 2: Deep learning model

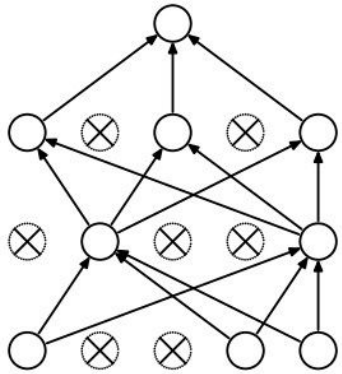


Conclusion





(a) Standard Neural Net



(b) After applying dropout.