Reinforcement learning

Week 5

Bandits, exploration, production hacks

Previously

- Value-based methods

  We know Q(s,a) → we know $\pi^*(a|s)$

- Model-free setting: learn from trajectories

  Q-learning, SARSA, expected value SARSA

- Too many states: approximate RL

  deep q-learning, duct tape, more duct tape

Today: exploration

how to pick actions to learn π(a|s) faster?*

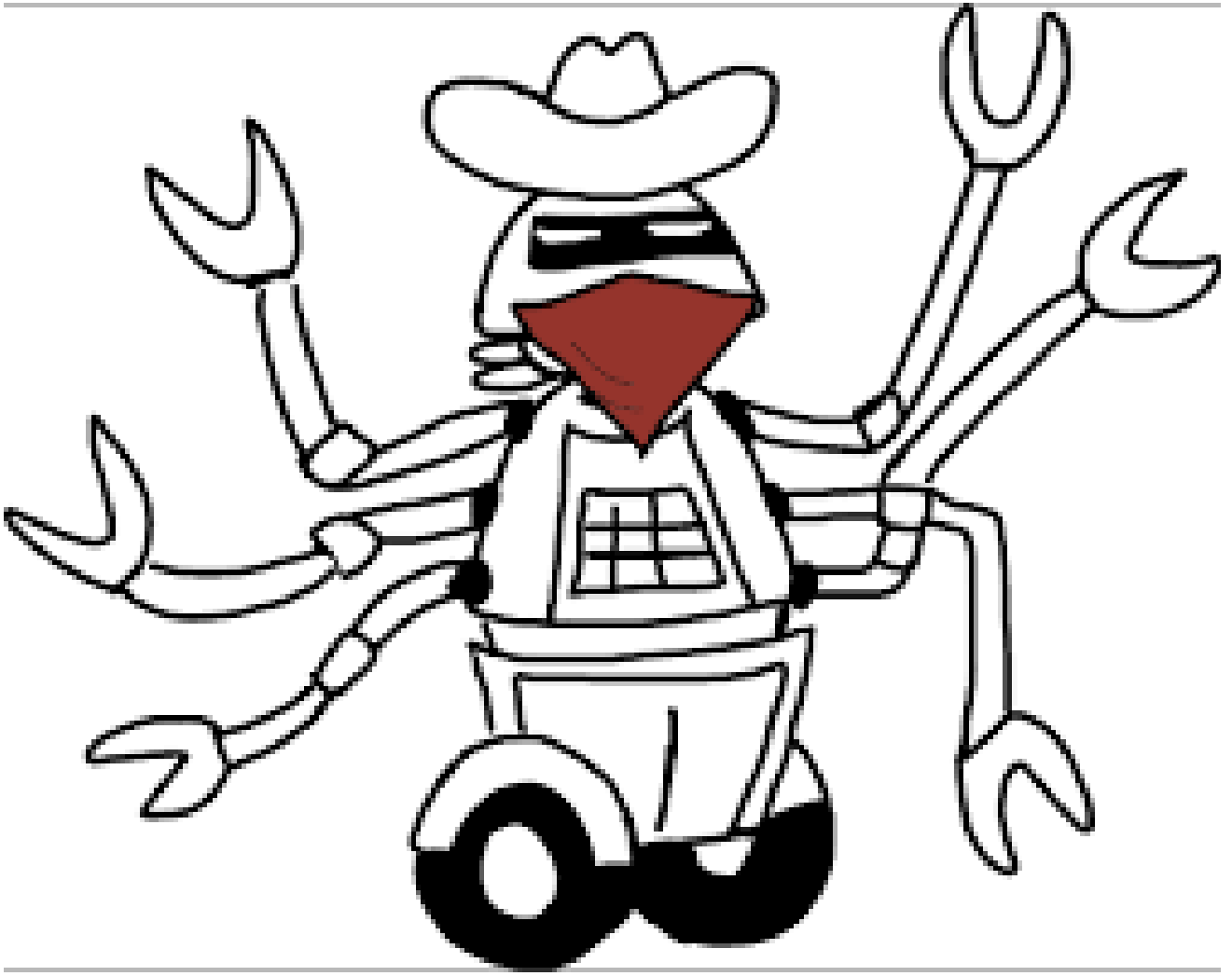
Multi-armed bandits

A simplified MDP with only one step

Why: it's simpler to explain exploration methods,
Formulae are shorter
(we can generalize to MDP if you wish)

Multi-armed bandits

Multi-armed bandits



What is: contextual bandit

Examples:

- banner ads (RTB)
- recommendations
- medical treatment

Basicaly it's 1-step MDP where

- G(s,a) = r(s,a)
- Q(s,a) = E r(s,a)
- All formulae are 50% shorter

How to measure exploration

You own a website that sells banner ads.

There are two ways to learn optimal ad placement:
“method A” vs “method B”
Both are black boxes

How do you choose one?

How to measure exploration

Bad idea: by the sound of their name
Good idea: with $$$ it brought/lost you

How to measure exploration

Bad idea: by the sound of their name
Good idea: with $$$ it brought/lost you

Regret of policy π(a|s)**:**

Consider an optimal policy, π*(a|s)

Regret per tick = optimal – yours

$$\eta = \sum_{t} \underset{s,a\sim\pi^{*}}{E} r(s,a) - \underset{s,a\sim\pi_{t}}{E} r(s,a)$$

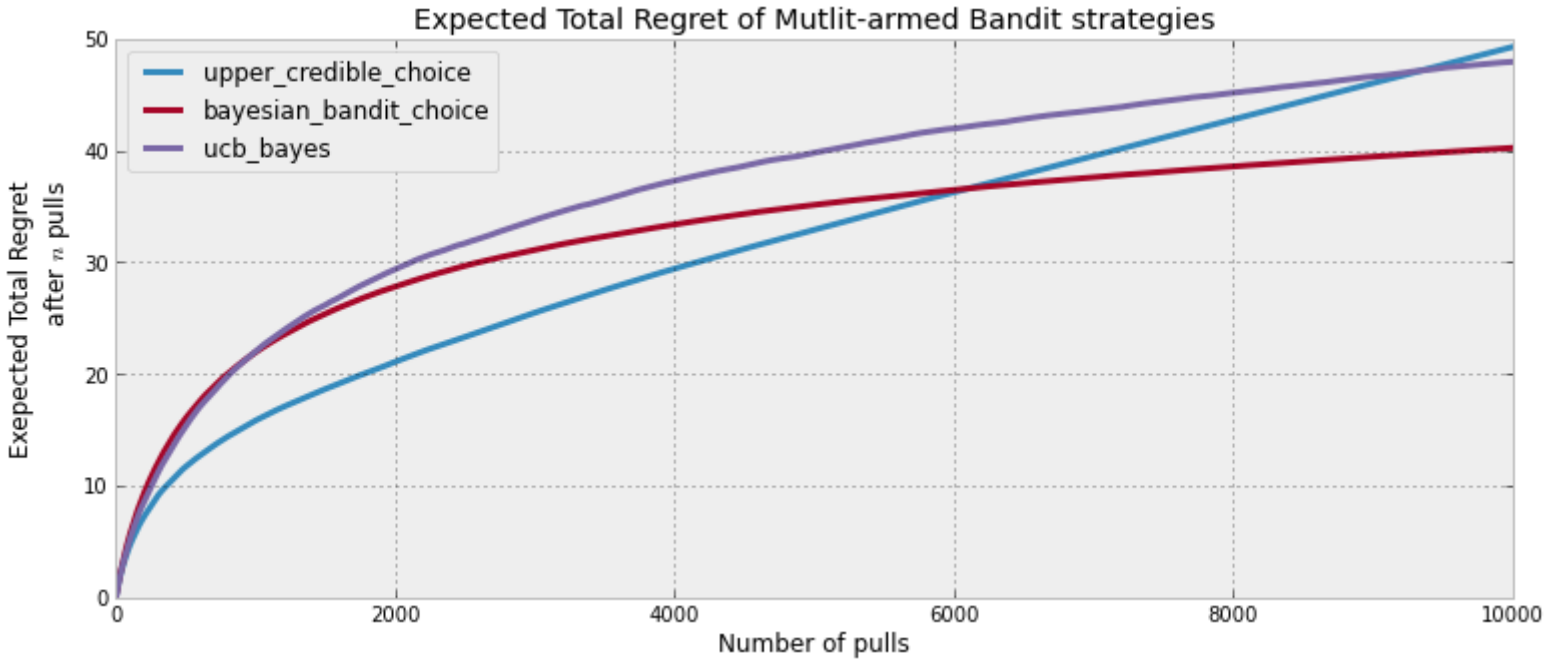
Finite horizon: t < max_t Infinite horizon: t → inf

How to measure exploration

Bad idea: by the sound of their name
Good idea: with $$$ it brought/lost you

Regret of policy π(a|s)**:**

Regret per tick = optimal – yours



Quiz time

Strategies:

- **ε-greedy**
  - With probability ε take a uniformly random action;
  - Otherwise take optimal action.
- **Boltzman**
  - Pick action proportionally to transformed Qvalues

$$P(a) = softmax\left(\frac{Q(s,a)}{std}\right)$$

- Policy based: add entropy

Quiz time

Your agent uses ε-greedy strategy with **ε=0.1**

What about his regret?

$$\eta = \sum_{t} \underset{s,a\sim\pi^{*}}{E} r(s,a) - \underset{s,a\sim\pi_{t}}{E} r(s,a)$$

Quiz time

Your agent uses ε-greedy strategy with **ε=0.1**

What about his regret?

$$\eta = \sum_t \underset{s,a\sim\pi^*}{E} r(s,a) - \underset{s,a\sim\pi_t}{E} r(s,a)$$

It grows linearly with time!
We never reach optimal policy cuz of **ε**
Q: How do we fix it?

Greedy in the limit

If exploration is asymptotically zero, $\lim_{t \to inf} \epsilon = 0$

we will reach π* in the limit

Practical: multiply by 0.99 every K steps :)

How many lucky random actions it takes to

- Apply medical treatment
- Control robots
- Invent efficient VAE training

Except humans can learn these in less than a lifetime

How many lucky random actions it takes to

- Apply medical treatment
- Control robots
- Invent efficient VAE training

We humans explore not with e-greedy policy!

BTW how humans explore?

Whether some new particles violate physics

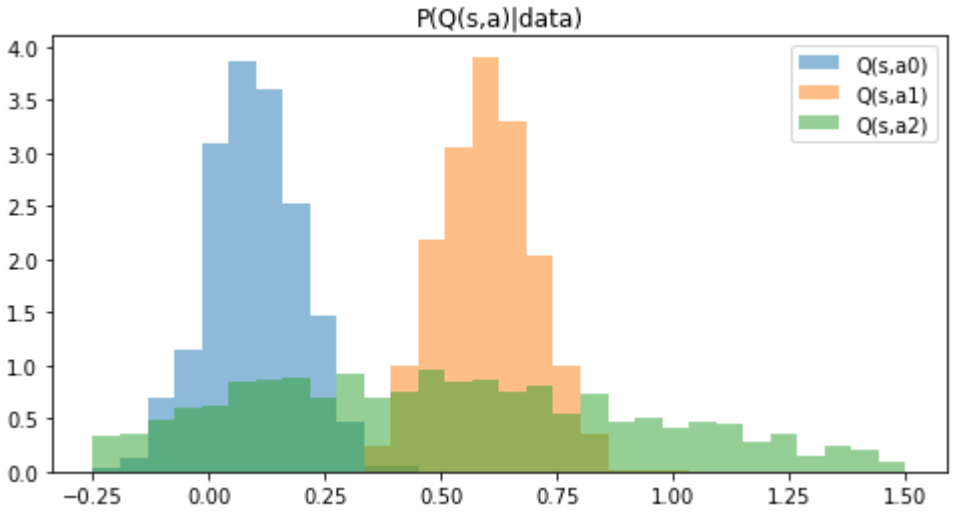
Vs

Whether you still can't fly by pulling your hair up

Uncertainty in returns

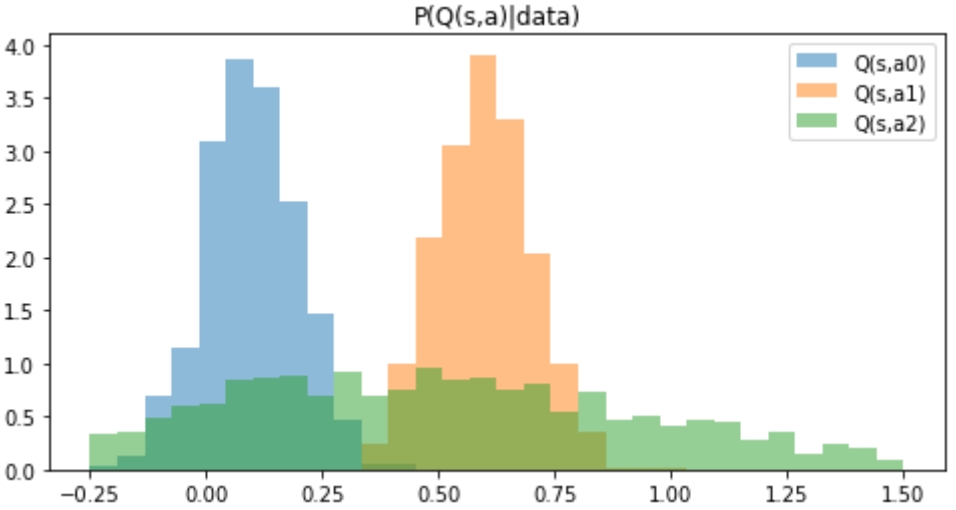
We want to try actions if we believe there's a chance they turn out optimal.

Idea: let's model how certain we are that Q(s,a) is what we predicted



Thompson sampling

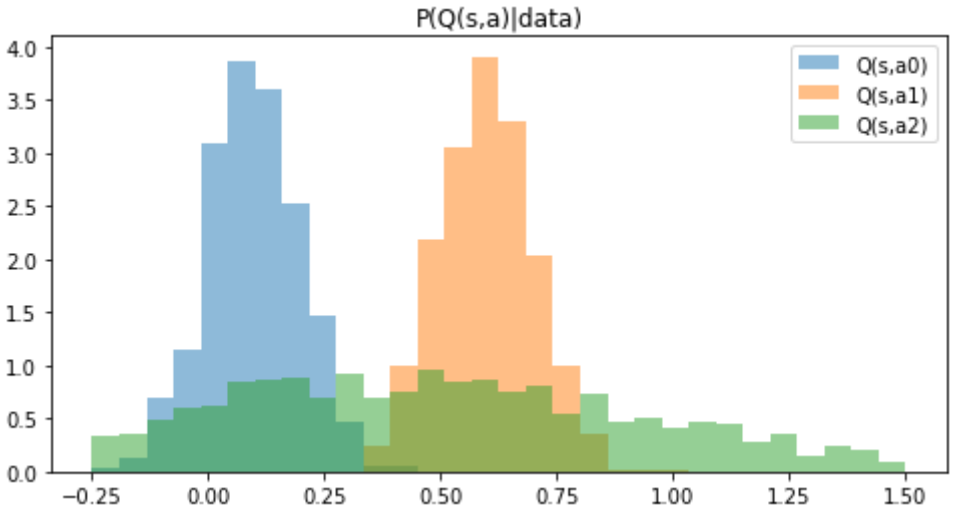
- Policy:
  - sample **once** from each Q distribution
  - take argmax over samples
  - which actions will be taken?



Thompson sampling

- Policy:
  - sample **once** from each Q distribution
  - take argmax over samples
  - which actions will be taken?

Takes a1 with p ~ 0.65, a2 with p ~ 0.35, a0 ~ never



Optimism in face of uncertainty

Idea:

Prioritize actions with uncertain outcomes!

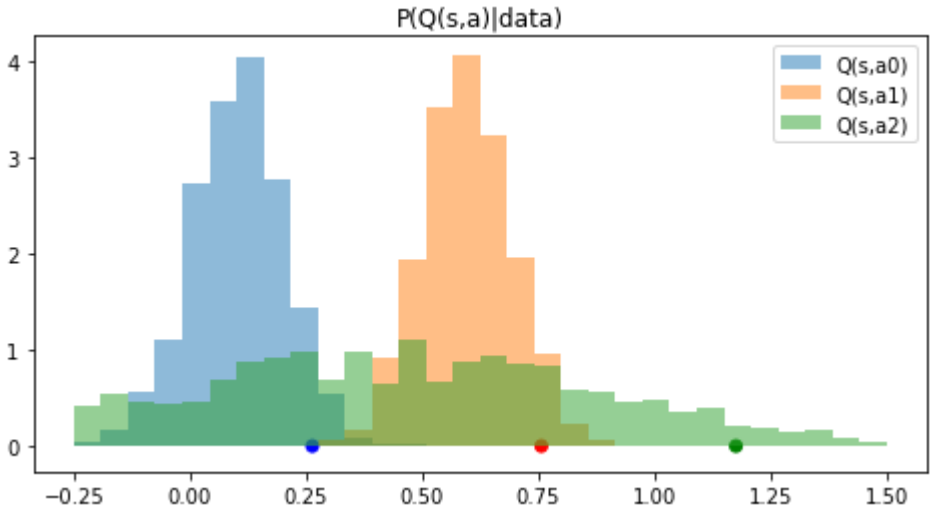
More uncertain = better.

Greater expected value = better

Math: try until upper confidence bound is small enough.

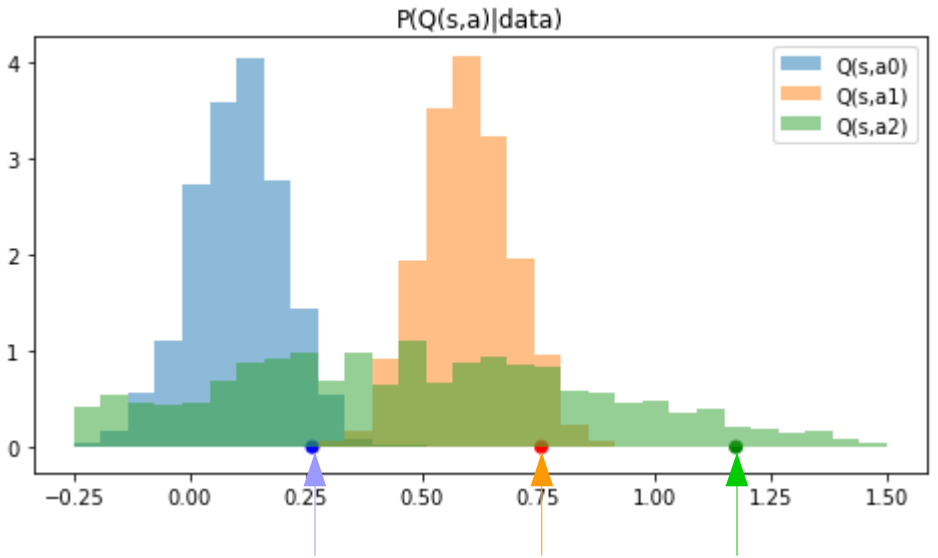
Optimism in face of uncertainty

- Policy:
  - Compute 95% upper confidence bound *for each a*
  - Take action with highest confidence bound
  - What can we tune here to explore more/less?



Optimism in face of uncertainty

- Policy:
  - Compute 95% upper confidence bound *for each a*
  - Take action with highest confidence bound
  - Adjust: change 95% to more/less



Frequentist approach

There's a number of inequalities that bound

P(x>t) < something

- E.g. Hoeffding inequality (arbitrary x in [0,1])

$$P\left(x - Ex \geq t\right) = e^{-2nt^2}$$

- Remember any others?

Frequentist approach

There's a number of inequalities that bound

P(x>t) < something

- E.g. Hoeffding inequality (arbitrary x in [0,1])

$$P\left(x - Ex \geq t\right) = e^{-2nt^2}$$

- Remember any others?

(Chernoff, Chebyshev, over9000)

Count-based exploration

UCB-1 for bandits

Take actions in in proportion to $\tilde{v}_a$

$$\tilde{v}_a = v_a + \sqrt{\frac{2 \log N}{n_a}}$$

- N number of time-steps so far
- n_a times action **a** is taken

Count-based exploration

UCB-1 for bandits

Take actions in in proportion to $\tilde{v}_a$

$$\tilde{v}_a = v_a + \sqrt{\frac{2 \log N}{n_a}}$$

Upper conf. bound for r in [0,1]

- N number of time-steps so far
- n_a times action **a** is taken

If not?

Count-based exploration

UCB-1 for bandits

Take actions in in proportion to $\tilde{v}_a$

$$\tilde{v}_a = v_a + \sqrt{\frac{2 \log N}{n_a}}$$

Upper conf. bound for r in [0,1]

- N number of time-steps so far
- n_a times action **a** is taken

If not – divide by r_max

Count-based exploration

UCB generalized for multiple states

$$\tilde{Q}(s,a) = Q(s,a) + \alpha \cdot \sqrt{\frac{2 \log N_s}{n_{s,a}}}$$

where

- N_s visits to state **s**
- $n_{s,a}$ times action **a** is taken from state **s**

Bayesian UCB

The usual way:

- Start from prior P(Q)
- Learn posterior P(Q|data)
- Take q-th percentile

What models can learn that?

Bayesian UCB

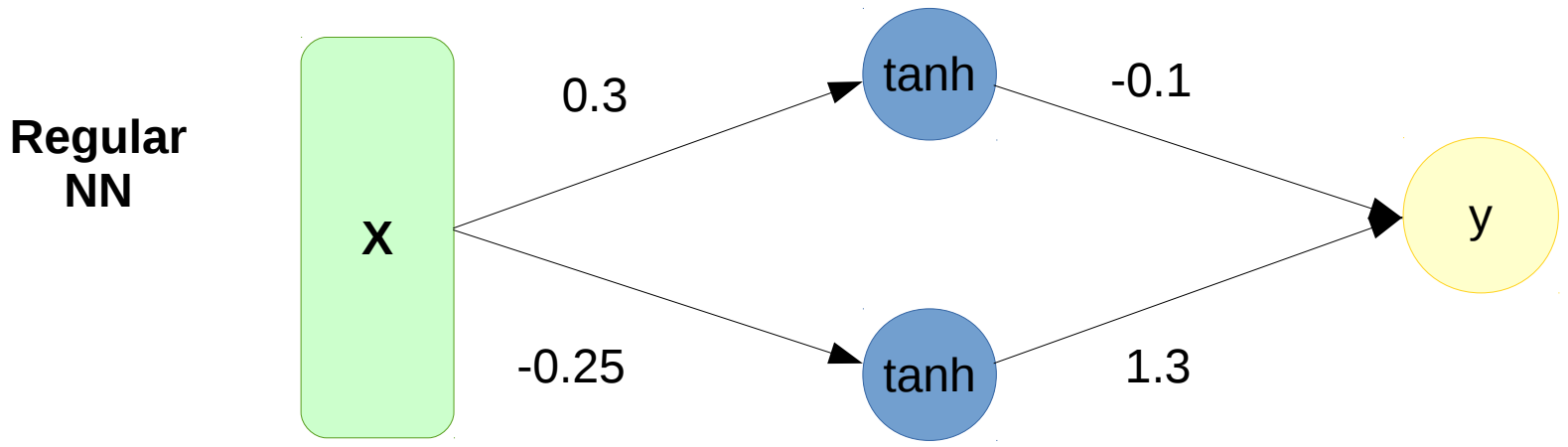
The usual way:

- Start from prior P(Q)
- Learn posterior P(Q|data)
- Take q-th percentile

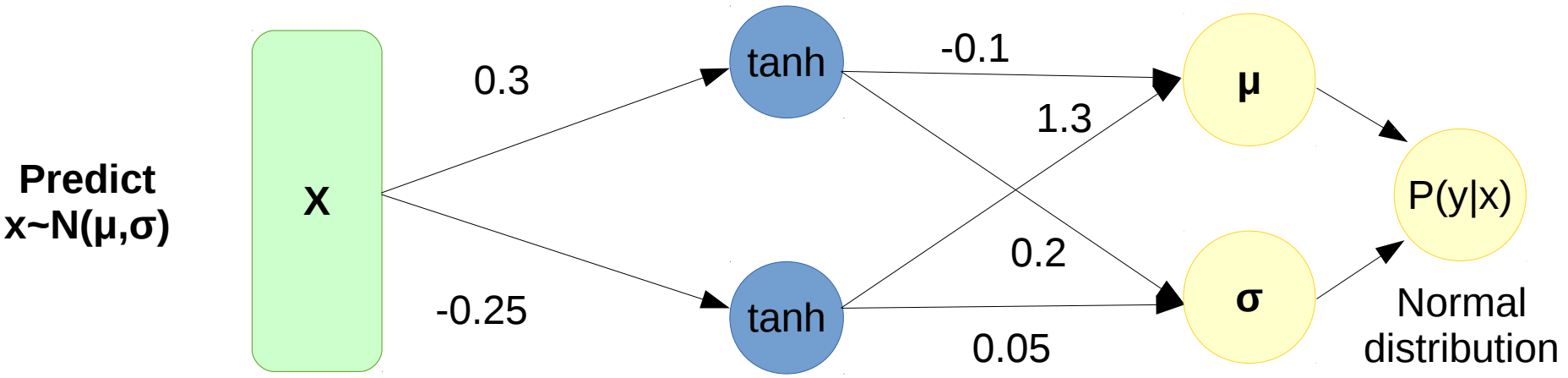
Approach 1: learn parametric P(Q), *e.g. normal*

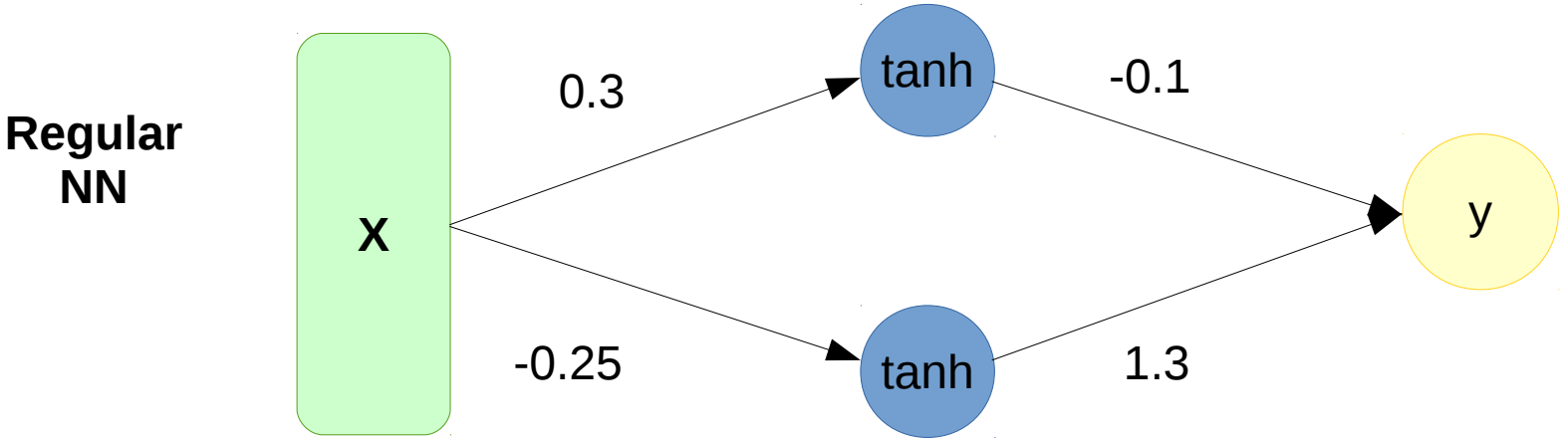
Approach 2: use bayesian neural networks

Parametric



Parametric

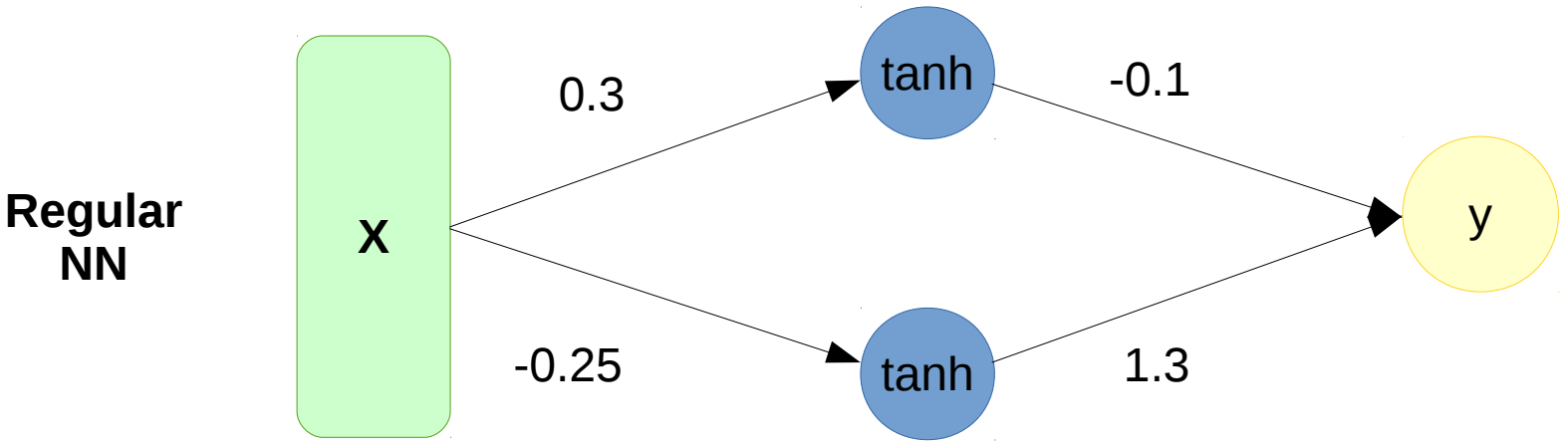

Predict
x~N(μ,σ)
x
0.3
-0.25
tanh
tanh
-0.1
1.3
0.2
0.05
μ
σ
P(y|x)
Normal
distribution


Regular
NN
x
0.3
-0.25
tanh
tanh
-0.1
1.3
y

BNNs

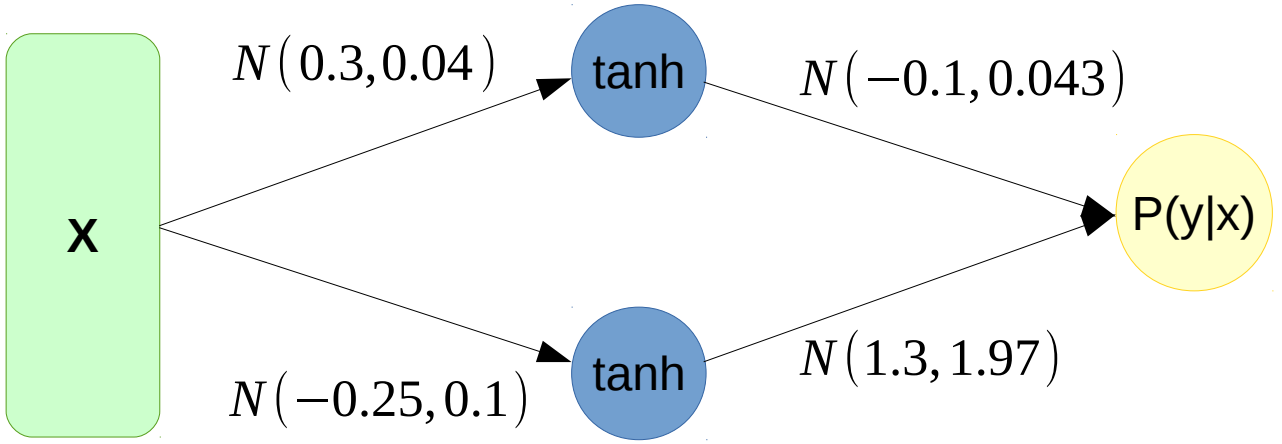
Disclaimer: this is a **hacker's guide** to BNNs!

It does not cover all the philosophy and general cases.

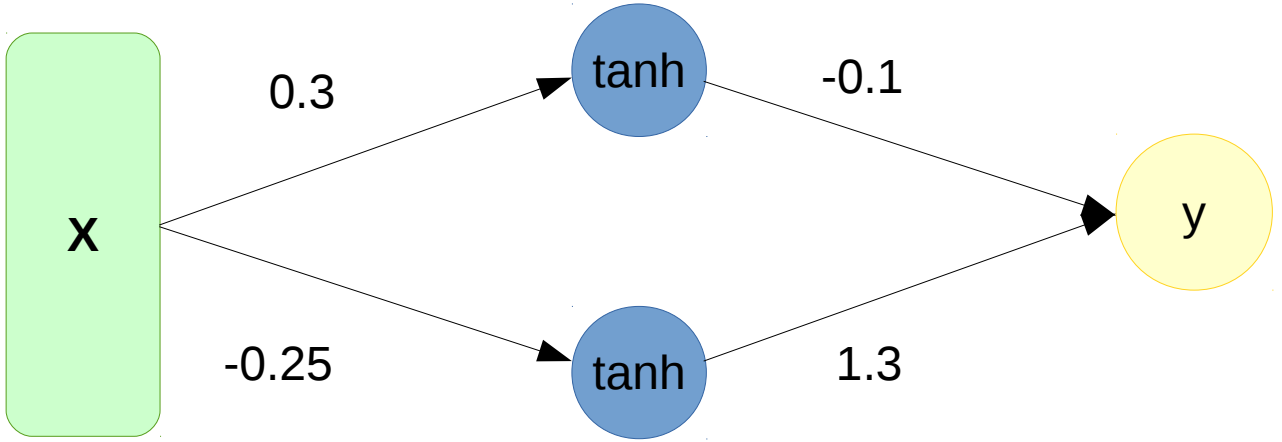


BNNs

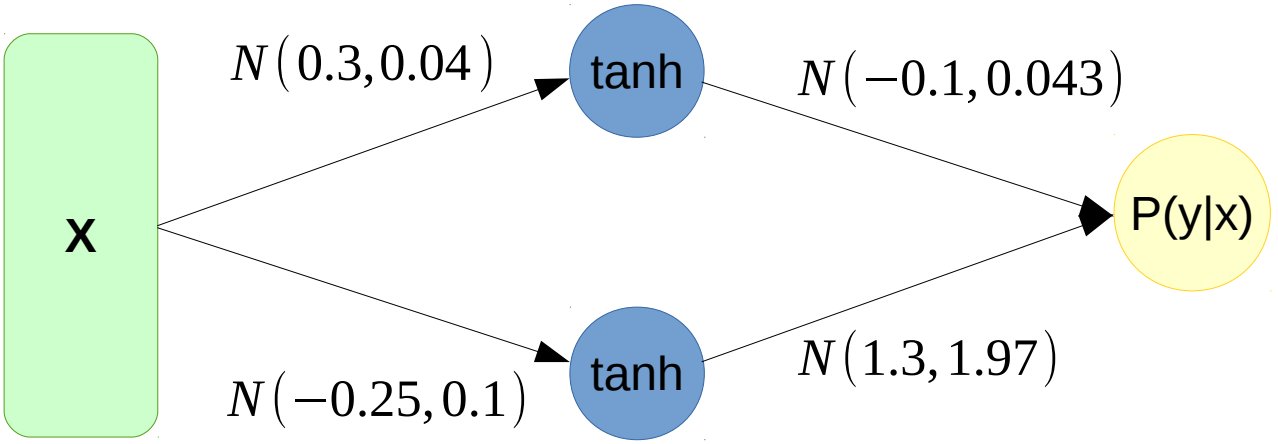
Bayesian NN



Regular NN



BNNs



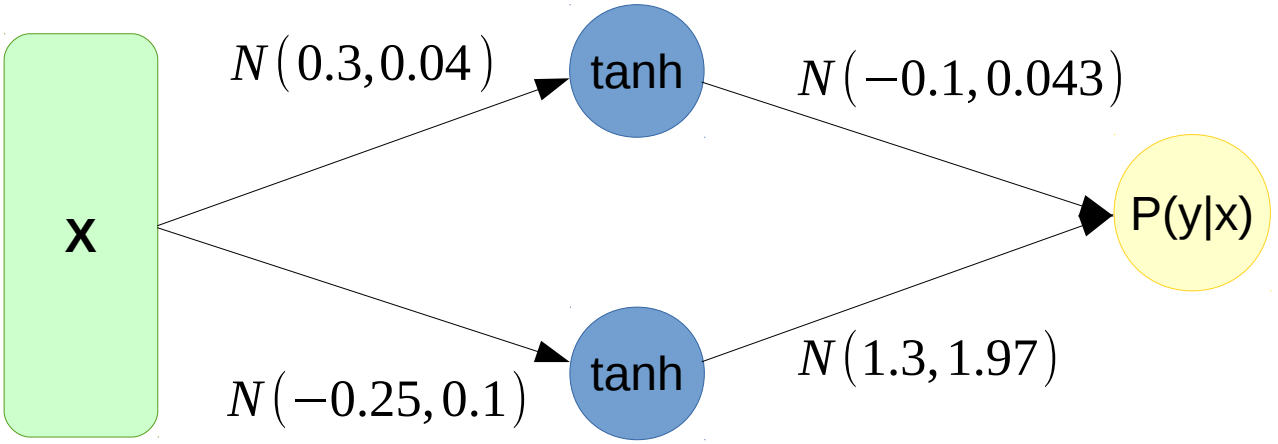
Idea:

- No explicit weights
  - Maintain parametric distribution on them instead!
    - Practical: fully-factorized normal or similar

$$q(\theta|\varphi:[\mu,\sigma]) = \prod_i N(\theta_i|\mu_i,\sigma_i)$$

$$P(y|x) = E_{\theta\sim q(\theta|\varphi)} P(y|x,\theta)$$

BNNs



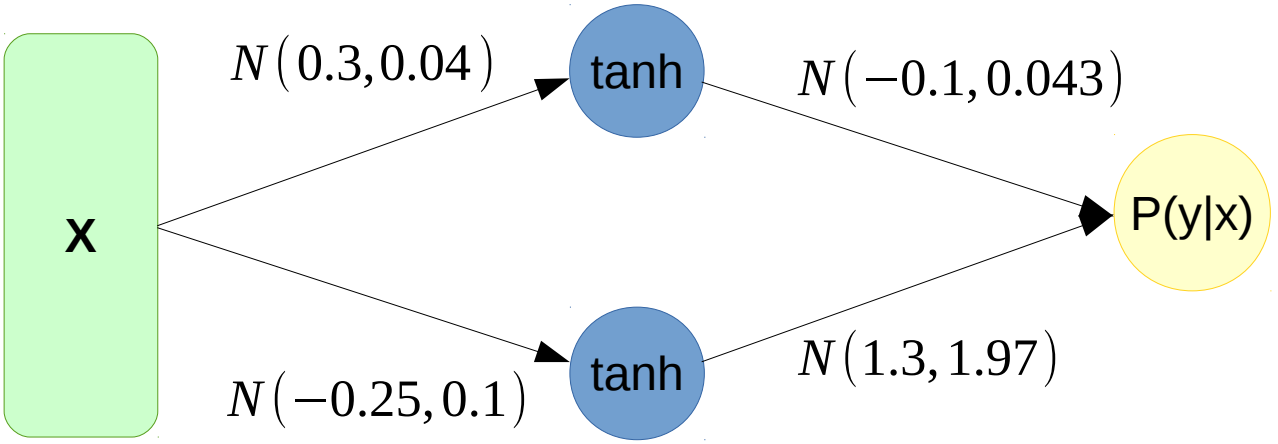
Idea:

- No explicit weights
  - Maintain parametric distribution on them instead!
    - Practical: fully-factorized normal or similar

$$q(\theta|\varphi:[\mu,\sigma]) = \prod_i N(\theta_i|\mu_i,\sigma_i)$$

$$P(y|x) = E_{\theta\sim q(\theta|\varphi)} P(y|x,\theta)$$

BNNs



Idea:

- No explicit weights
- Inference: sample from weight distributions, predict 1 point
- To get distribution, aggregate K samples (e.g. with histogram)
  - Yes, it means running network **multiple times per one X**

$$P(y|x) = E_{\theta \sim q(\theta|\varphi)} P(y|x, \theta)$$

BNNs

Idea:

- No explicit weights
  - Maintain parametric distribution on them instead!
    - Practical: fully-factorized normal or similar

$$q\left(\theta|\varphi:[\mu,\sigma]\right)=\prod_i N\left(\theta_i|\mu_i,\sigma_i\right)$$

$$P\left(y|x\right)=E_{\theta\sim q(\theta|\varphi)}P\left(y|x,\theta\right)$$

- Learn parameters of that distribution (reparameterization trick)
  - Less variance: local reparameterization trick.

$$\mathring{\varphi}=argmax_{\varphi}E_{x_i,y_i\sim d}E_{\theta\sim q(\theta|\varphi)}P\left(y_i|x_i,\theta\right)$$

wanna explicit formulae? *d = dataset*

Lower bound

d = dataset

$$-KL(q(\theta|\varphi)||p(\theta|d)) = -\int_{\theta} q(\theta|\varphi)\cdot\log\frac{q(\theta|\varphi)}{p(\theta|d)}$$

$$-\int_{\theta} q(\theta|\varphi)\cdot\log\frac{q(\theta|\varphi)}{\left[\frac{p(d|\theta)\cdot p(\theta)}{p(d)}\right]} = -\int_{\theta} q(\theta|\varphi)\cdot\log\frac{q(\theta|\varphi)\cdot p(d)}{p(d|\theta)\cdot p(\theta)}$$

$$-\int_{\theta} q(\theta|\varphi)\cdot[\log\frac{q(\theta|\varphi)}{p(\theta)} - \log p(d|\theta) + \log p(d)]$$

$$[E_{\theta\sim q(\theta|\varphi)}\log p(d|\theta)] - KL(q(\theta|\varphi)||p(\theta)) + \log p(d)$$

loglikelihood **-distance to prior** **+const**

Lower bound

$$\varphi = \underset{\varphi}{argmax}\left(-KL\left(q\left(\theta|\varphi\right)||p\left(\theta|d\right)\right)\right)$$

$$\underset{\varphi}{argmax}\left(\left[E_{\theta\sim q\left(\theta|\varphi\right)}\log p\left(d|\theta\right)\right]-KL\left(q\left(\theta|\varphi\right)||p\left(\theta\right)\right)\right)$$

Can we perform gradient ascent directly?

Reparameterization trick

$$\varphi = \underset{\varphi}{argmax}\left(-KL\left(q\left(\theta|\varphi\right)||p\left(\theta|d\right)\right)\right)$$

$$\underset{\varphi}{argmax}\left(\underline{\left[E_{\theta\sim q(\theta|\varphi)}\log p\left(d|\theta\right)\right]} - \underline{KL\left(q\left(\theta|\varphi\right)||p\left(\theta\right)\right)}\right)$$

Use reparameterization trick

simple formula (for normal q)

What does this log P(d|...) mean?

BNN likelihood

$$E_{\theta\sim N(\theta|\mu_\varphi,\sigma_\varphi)}\log p\left(d|\theta\right) = E_{\psi\sim N(0,1)}\log p\left(d|\left(\mu_\varphi + \sigma_\varphi\cdot\psi\right)\right)$$

Better: local reparameterization trick (google it)

Reparameterization trick

$$\varphi = \underset{\varphi}{argmax}\left(-KL\left(q\left(\theta|\varphi\right)||p\left(\theta|d\right)\right)\right)$$

$$\underset{\varphi}{argmax}\left(\left[E_{\theta\sim q(\theta|\varphi)}\log p\left(d|\theta\right)\right]-KL\left(q\left(\theta|\varphi\right)||p\left(\theta\right)\right)\right)$$

In other words,
$\Sigma_{x,y\sim d}$ log p(y|x,μ+σψ)

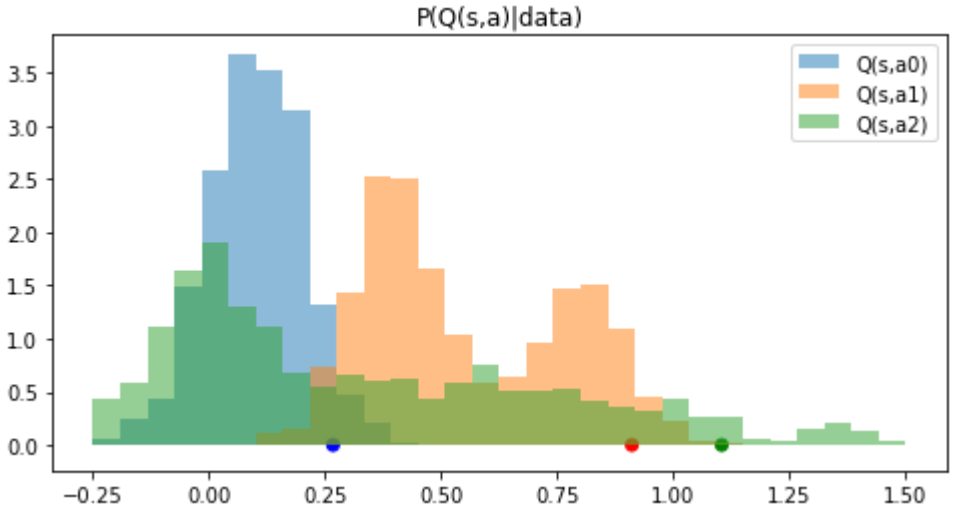
BNN likelihood

$$E_{\theta\sim N(\theta|\mu_\varphi,\sigma_\varphi)}\log p\left(d|\theta\right)=E_{\psi\sim N(0,1)}\log p\left(d|\left(\mu_\varphi+\sigma_\varphi\cdot\psi\right)\right)$$

Better: local reparameterization trick (google it)

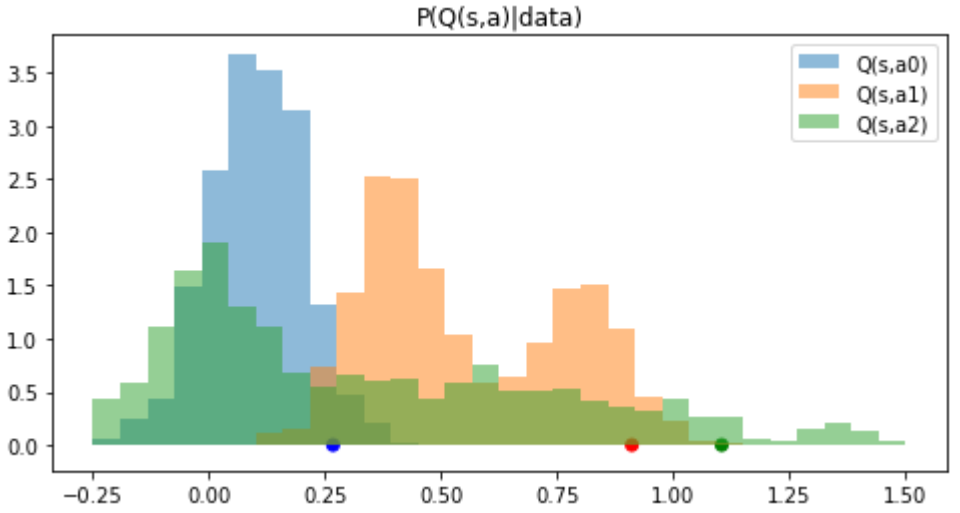
Using BNNs

- If you sample from BNNs
  - Can learn ~arbitrary distribution (e.g. multimodal)
  - But it takes running network many times
  - Use empirical percentiles for exploration priority
    - Again, 3 points on horizontal axis are percentiles



Using BNNs

- If you sample from BNNs
  - Can learn ~arbitrary distribution (e.g. multimodal)
  - **But it takes running network many times**
  - Use empirical percentiles for exploration priority
    - Again, 3 points on horizontal axis are percentiles



Practical stuff

- Approximate exploration policy

  with something cheaper

- Bayesian UCB:
  - Prior can make or break it
  - Sometimes parametric guys win (vs bnn)

- Of course, neural nets aren't

  always the best model

What we learned?

What new stuff did we learn today?

Note to self: I should stay silent.

What we learned?

- Regret: money agent lost while learning optimal policy
  - Less regret = better exploration
  - ε-greedy and boltzmann can be really bad!
- Using uncertainty:
  - Thompson, UCB, bayes-UCB
- Scaling things up: bayesian neural networks