AutoPrognosis

Ahmed M. Alaa
Mihaela van der Schaar

June, 2019

Method

- Imputation algorithms $\mathcal{A}_d$
  Hyperparameters Θ_d
- Feature process. algorithms $\mathcal{A}_f$
  Hyperparameters Θ_f
- Classification algorithms $\mathcal{A}_c$
  Hyperparameters Θ_c
- Calibration algorithms $\mathcal{A}_a$
  Hyperparameters Θ_a

- Set of all pipelines $\mathcal{P} = \mathcal{A}_d \times \mathcal{A}_f \times \mathcal{A}_c \times \mathcal{A}_a$
- Set of all hyperparameters $\Theta = \Theta_d \times \Theta_f \times \Theta_c \times \Theta_a$
- Set of all pipeline configurations $\mathcal{P}_\Theta$

- Combined Pipeline Selection and Hyperparameter optimization problem (CPSH)

$$P^*_{\theta^*} \in \arg\max_{P_\theta \in \mathcal{P}_\Theta} \frac{1}{K} \sum_{i=1}^{K} \mathcal{L}(P_\theta; \mathcal{D}^{(i)}_{\text{train}}, \mathcal{D}^{(i)}_{\text{valid}})$$

Automated Pipeline Configuration

- **The CPSH problem** $\arg\max_{P_\theta \in \mathcal{P}_\Theta} \frac{1}{K}\sum_{i=1}^{K} \mathcal{L}(P_\theta; \mathcal{D}_{\text{train}}^{(i)}, \mathcal{D}_{\text{valid}}^{(i)})$
- **Bayesian optimization**

$$f(P_\theta) = \frac{1}{K}\sum_{i=1}^{K} \mathcal{L}(P_\theta; \mathcal{D}_{\text{train}}^{(i)}, \mathcal{D}_{\text{valid}}^{(i)}) + \varepsilon$$

Gaussian process prior

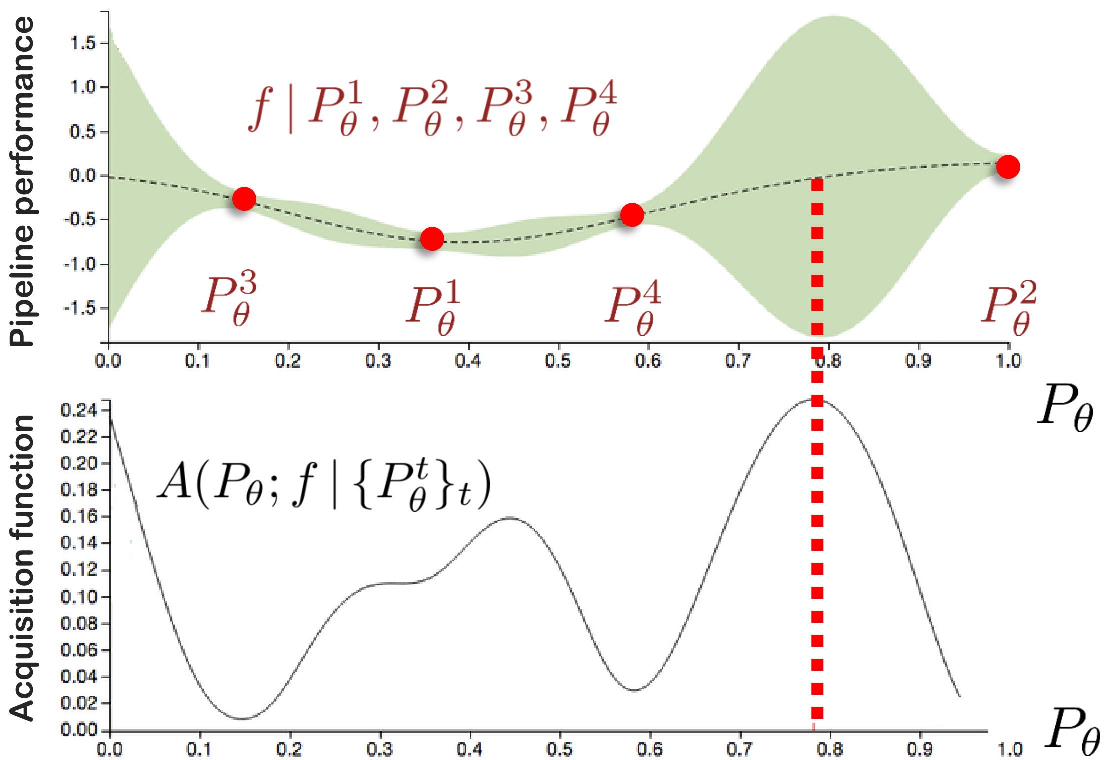
$$f \sim \mathcal{GP}(\mu(P_\theta), k(P_\theta, P'_\theta))$$

Gaussian process posterior

$$f \mid \{P_\theta^t\}_t$$

Select new pipeline via acquisition function

$$P_\theta^{t+1} = \arg\max_{P_\theta} A(P_\theta; f \mid \{P_\theta^t\}_t)$$



The Curse of Dimensionality

- **Statistical and computational complexity of the CPSH problem**
- **High-dimensional pipeline space (*D*)**

Gaussian process prior

$$f \sim \mathcal{GP}(\mu(P_\theta), k(P_\theta, P'_\theta))$$

Sample complexity for non-parametric estimation of α-smooth functions [Stone, 1982]

$$\Theta\left(t^{-\frac{\alpha}{2\alpha+D}}\right)$$

Exponentially many iterations!

Gaussian process posterior

$$f \,|\, \{P_\theta^t\}_t$$

Computational complexity of GP posterior After *t* iterations [Rasmussen & Williams, 2006]

$$\mathcal{O}(t^3)$$

Select new pipeline via acquisition function

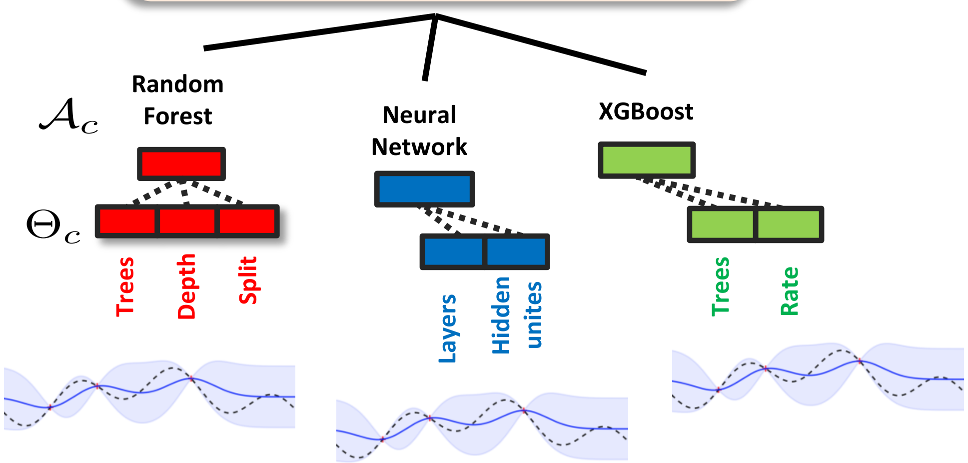
$$P_\theta^{t+1} = \arg\max_{P_\theta} A(P_\theta; f \,|\, \{P_\theta^t\}_t)$$

Computational complexity of maximizing acquisition [Snoek, 2015]

$$\mathcal{O}(n^D)$$

Two Extremes

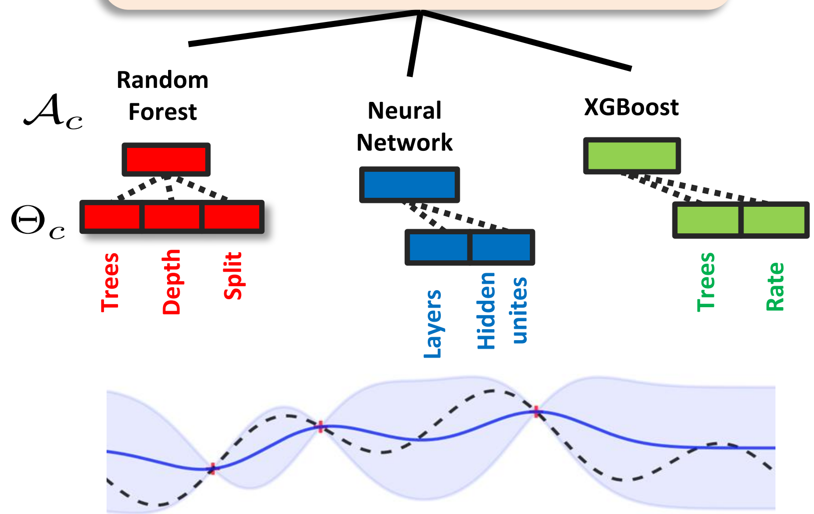
One Gaussian process per algorithm



- Low-dimensional
- No information sharing

Almost manual tuning!

One Gaussian process for *all* algorithms



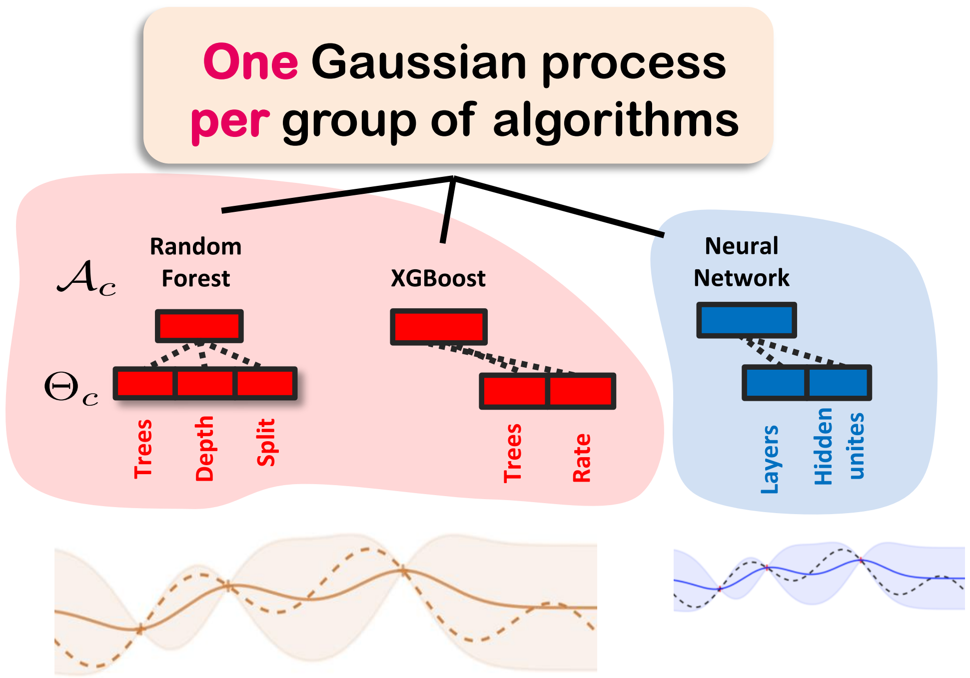
- High-dimensional
- Full information sharing

Optimal, fully automated!

Bayesian Optimization with Structured Kernel Learning

- **Main idea:** Not ALL algorithms need to share information! Correlation?
- **Example:** XGBoost correlated with Random forest, but not with neural networks!

- **<u>Learn</u> a structured kernel that groups <u>correlated</u> algorithms together:**
  - **Low dimensionality for every group**
  - **Relevant information sharing within a group**



Sparse Additive Gaussian Processes

- **Decompose high-dimensional GP into sum of low-dimensional components**

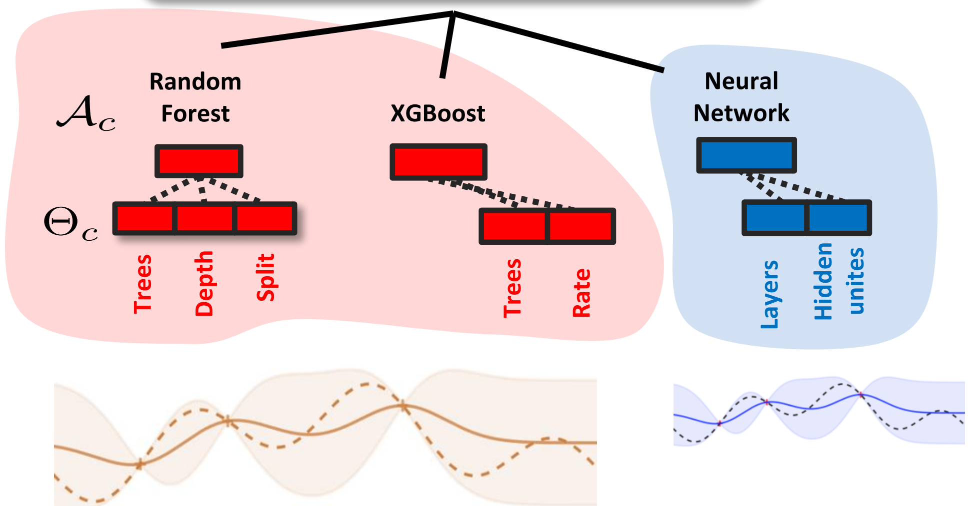
$$\boldsymbol{\Lambda}$$

Space of all pipelines

$$\{\boldsymbol{\Lambda}^{(m)}\}_m$$

Partitions of $\boldsymbol{\Lambda}$

One Gaussian process per group of algorithms



$k_1(\Lambda^{(1)}, \Lambda'^{(1)})$ $k_2(\Lambda^{(2)}, \Lambda'^{(2)})$

Sparse additive GPs:

$$f(\Lambda) = \sum_{m=1}^{M} f_m(\Lambda^{(m)})$$

D-dimensional GP **Low-dimensional GPs**

Structured kernel: $k(\Lambda, \Lambda') = \sum_{m=1}^{M} k_m(\Lambda^{(m)}, \Lambda'^{(m)})$

Big reduction in complexity!

$$\dim(\mathbf{\Lambda}^{(m)}) = d_m$$

Gaussian process prior

$$f \sim \mathcal{GP}(\mu(P_\theta), k(P_\theta, P_\theta'))$$

Sample complexity for non-parametric estimation of additive sparse α-smooth functions [Raskutti, 2009]

$$\Theta\left(\sum_m t^{-\frac{\alpha}{2\alpha+d_m}}\right)$$

Exponential only in dimension of 1 subgroup !

Gaussian process posterior

$$f \mid \{P_\theta^t\}_t$$

Computational complexity of GP posterior

Fewer iterations needed,
fewer data points in every subgroup!

Select new pipeline via acquisition function

$$P_\theta^{t+1} = \arg\max_{P_\theta} A(P_\theta; f \mid \{P_\theta^t\}_t)$$

Computational complexity of maximizing acquisition

$$\mathcal{O}\left(n^{d_m}\right)$$

Exponential improvement in statistical and computational efficiency!

...but the structure of the kernel is unknown!

- **Need to learn the subspace decomposition** $\{\mathbf{\Lambda}^{(m)}\}_m$ **from the data**

- **Bayesian approach:**

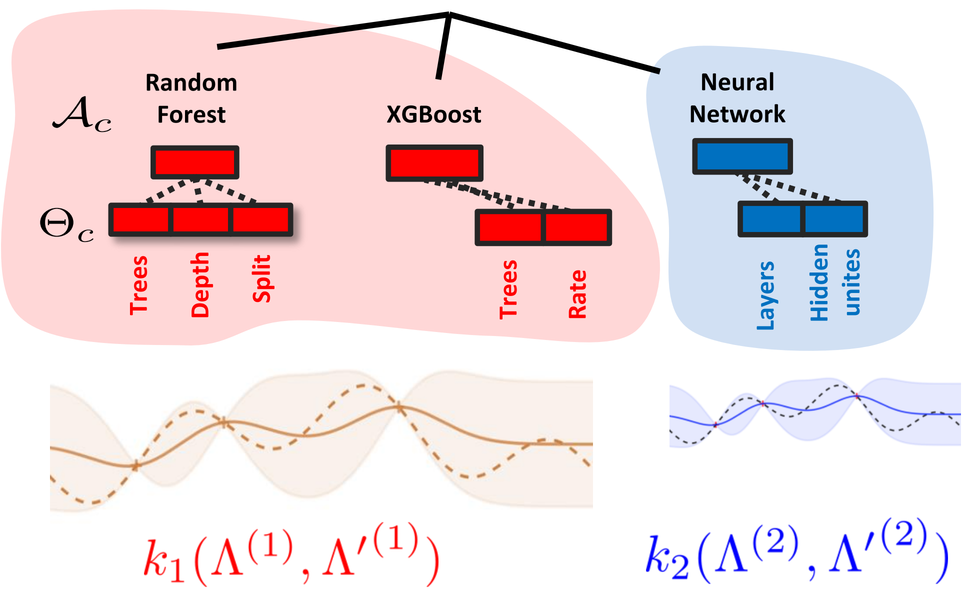
Prior on decompositions

$$\{\mathbf{\Lambda}^{(m)}\}_m \sim \Pi(\gamma)$$

Compute posterior in concurrence with BO

$$\{\mathbf{\Lambda}^{(m)}\}_m \,|\, \{P_\theta^t\}_t$$

One Gaussian process per group of algorithms



Structured Kernel Learning (I)

- **Define the variable** $z_{v,i} \in \{1, \ldots, M\}$**: indicator for the subspace allocation for algorithm** i **in** $\mathcal{A}_v$

Prior on $z_{v,i}$ **= Prior on** $\{\Lambda^{(m)}\}_m$

- **Bayesian inference:**

Prior on decompositions

$$\alpha \sim \text{Dirichlet}(M, \gamma)$$
$$z_{v,i} \sim \text{Multinomial}(\alpha)$$

Compute posterior in concurrence with BO

$$\mathbb{P}(z, \alpha \,|\, \{f(P_\theta^t)\}_t, \gamma) \propto \mathbb{P}(\{f(P_\theta^t)\}_t \,|\, z)\, \mathbb{P}(z \,|\alpha)\, \mathbb{P}(\alpha, \gamma)$$

Gibbs Sampling

$$\mathbb{P}(z_{v,i} = m \,|\, z/\{z_{v,i}\}, \mathcal{H}_t) \propto \mathbb{P}(\mathcal{H}_t \,|\, z)\,(|\mathcal{A}_v^{(m)}| + \gamma_m)$$

Gumbel-Max Sampler

$$\omega_m \overset{\text{i.i.d}}{\sim} \text{Gumbel}(0, 1),\; m \in \{1, \ldots, M\},$$
$$z_{v,i} \sim \arg\max_m \mathbb{P}(\mathcal{H}_t \,|\, z, z_{v,i} = m)(|\mathcal{A}_v^{(m)}| + \gamma_m) + \omega_m.$$

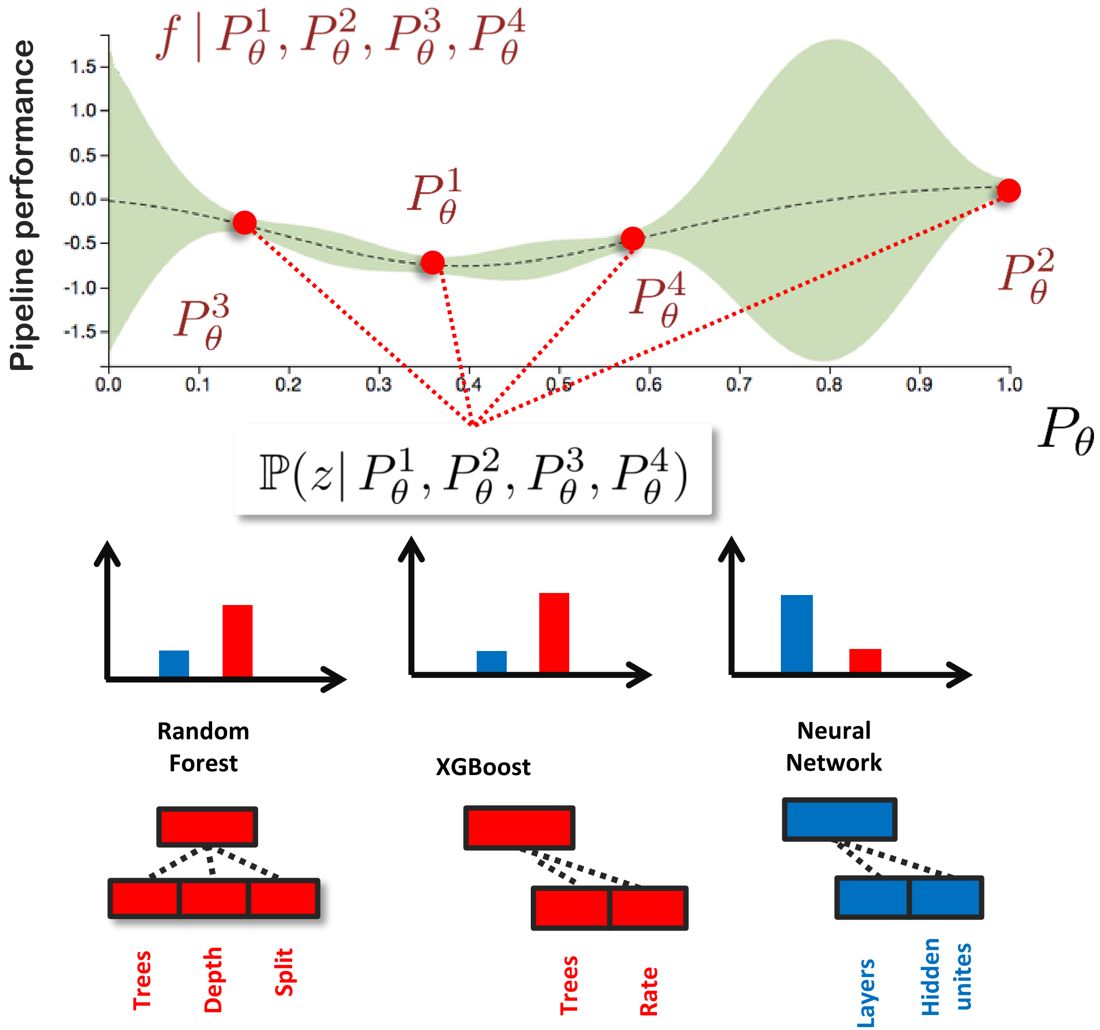
Structured Kernel Learning (II)

- **Learning structured kernel in concurrence with BO**

Posterior over performance

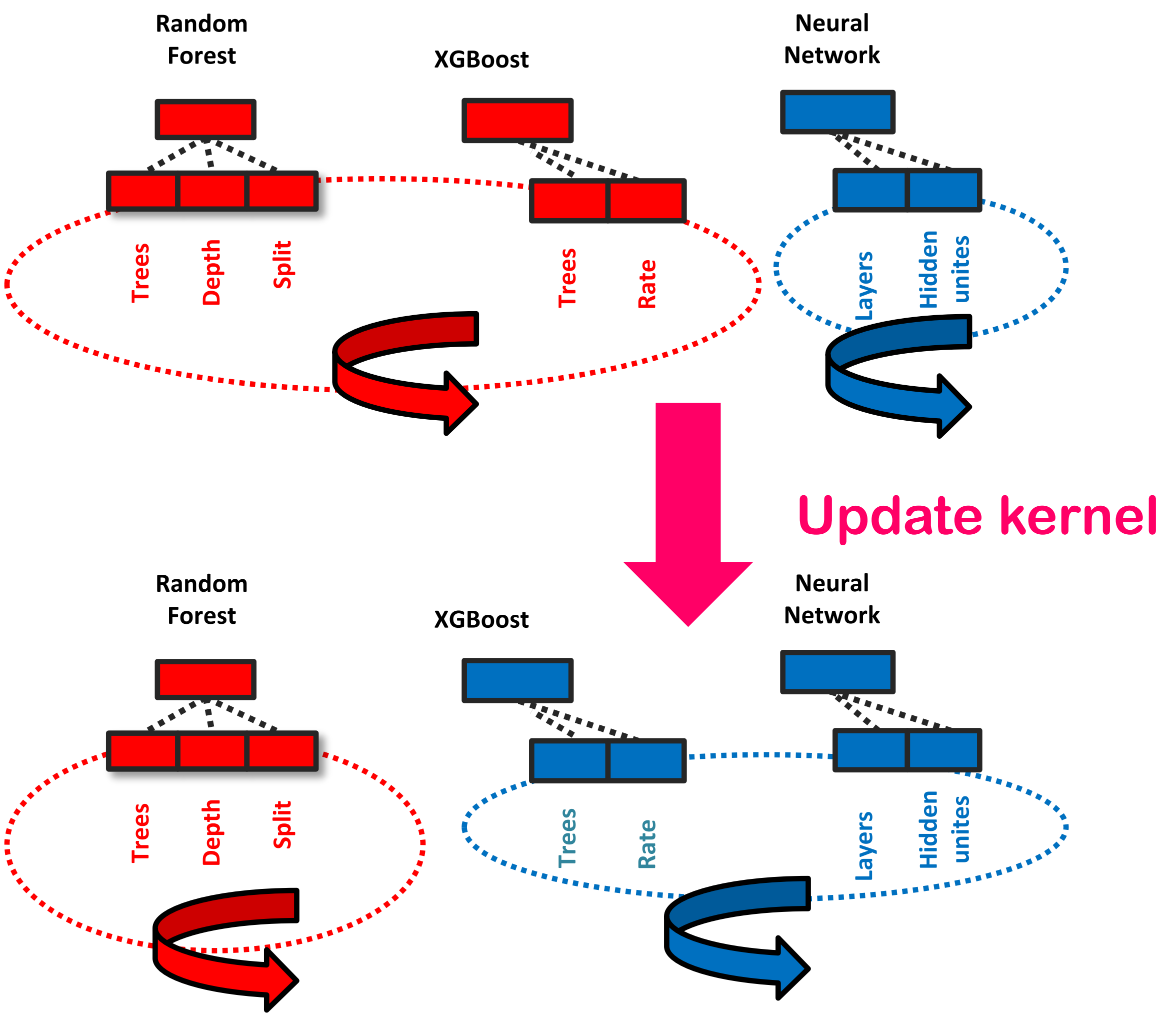
M=2

Posterior over decompositions



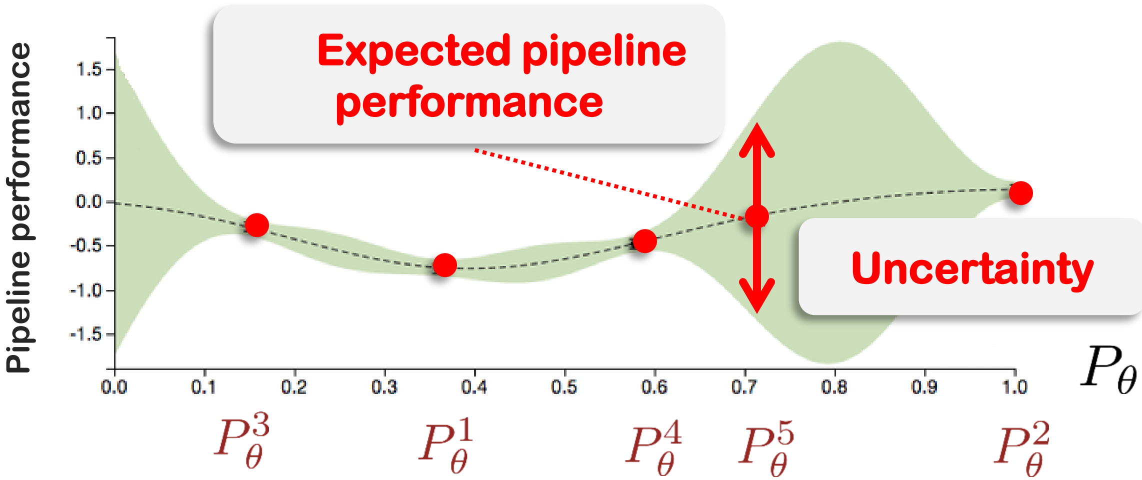
What does the algorithm do?

- ## A stochastic search path



Post-hoc Ensemble Construction

- Create an ensemble using the posterior distribution of performances



- Bayesian model averaging
- Create a linear combination of pipelines $\sum_i w_i P_\theta^i$
- Weight of every pipeline = empirical probability of it being the best!

$$
\begin{aligned}
w_i &= \mathbb{P}(P_\theta^{i^*} = P_\theta^i \,|\, \mathcal{H}_t) \\
&= \prod_{j \neq i} \Phi\left((\mu_i - \mu_j) \cdot (\sigma_i^2 + \sigma_j^2)^{-\frac{1}{2}} \right),
\end{aligned}
$$

The AP Algorithm