MTTTS16 Learning from Multiple Sources

5 ECTS credits

Autumn 2019, University of Tampere

Lecturer: Jaakko Peltonen

Lecture 6: Multitask learning with kernel methods and nonparametric models

On this lecture:

- Multi-task learning methods that use a “kernel trick” to create nonlinear regression / classification methods

- Multi-task learning methods that do not require the predictor /classifier to come from a simple parametric family

Part 1: Nonlinear Multitask learning with a “kernel trick”

Multitask learning with a kernel trick, intro

- On lecture 4 we saw how canonical correlation analysis could be done in a nonlinear way by a "**kernel trick**"
- The idea was: **transform** the data with nonlinear transformations into a new space, and perform linear canonical correlation analysis in that space
- Instead of needing to know a parametric transformation into the new space, it was possible to perform all calculations using only an inner product or "kernel" defined between data vectors.
- A kernel is essentially a **similarity measure**
- Simple kernel between two vectors: traditional inner product. Corresponds to no transformation.
- Many other possible kernels, for example **Gaussian function**
- Each kernel corresponds to an inner product after an (unknown) nonlinear transformation. It is enough to know the kernel.
- We now follow a similar approach to **multi-task learning**.

Multitask learning with a kernel trick, intro

- Task relationships have modeled by assuming that error terms (noise) in different regression tasks are correlated
- Extensions of various regularization methods (priors) to multi-task learning have been proposed
- Relations between tasks have been modeled as post–processing after learning
- The paper of Evgeniou et al. claims it does not “follow a Bayesian or a statistical approach” but instead a regularization approach
- In practice this can be seen as a particular kind of model and prior

Following the approach from T. Evgeniou, C. A. Micchelli, and M. Pontil. Learning Multiple Tasks with Kernel Methods. Journal of Machine Learning Research 6: 615-637, 2005. Images from that paper.

Multitask learning with a kernel trick, single-task case

- **Single-task learning notation**:
  data set $\{(x_i, y_i) : i \in \mathbb{N}_m\} \subset \mathcal{X} \times \mathcal{Y}$
  where $\mathbb{N}_m$ denotes $\{1, \ldots, m\}$, $\mathcal{X}$ is usually part of $\mathbb{R}^d$ and $\mathcal{Y}$ is $\{-1, 1\}$ for binary classification
- Task: learn a function f to get small expected error $E[L(y, f(x))]$
  where L is a **loss function** such as squared error $(y - f(x))^2$
- Learn parameters to minimize **loss + regularization**:

$$\frac{1}{m} \sum_{j \in \mathbb{N}_m} L(y_j, f(x_j)) + \gamma \|f\|_K^2$$

regularization parameter

  where $\|f\|_K^2$ is the norm of f in a space $\mathcal{H}_K$ of functions
- **The solution** has the form:

$$f(x) = \sum_{j \in \mathbb{N}_m} c_j K(x_j, x)$$

(the fact that this happens is called the "representer theorem")

  so the task is to find the weights c_j

Following the approach from T. Evgeniou, C. A. Micchelli, and M. Pontil. Learning Multiple Tasks with Kernel Methods. Journal of Machine Learning Research 6: 615-637, 2005. Images from that paper.

Multitask learning with a kernel trick, single-task case

- **Statistical interpretation**:
  - f defines a regression model,
  - squared error corresponds to log-likelihood of Gaussian noise around predicted outputs,
  - $\|f\|_K^2$ is log-probability of f in a prior distribution
  - minimization = finding the **maximum a posteriori** model fit (maximizing posterior probability given data)

- Now we treat the multi-task setting with a similar setup

Following the approach from T. Evgeniou, C. A. Micchelli, and M. Pontil. Learning Multiple Tasks with Kernel Methods. Journal of Machine Learning Research 6: 615-637, 2005. Images from that paper.

Multitask learning with a kernel trick, linear case

- **Multi-task learning notation**:
  n tasks, each task *l* comes with *m* examples $\{(x_{i\ell}, y_{i\ell}) : i \in \mathbb{N}_m\}$.
  All data together marked as $\{(x_{i\ell}, y_{i\ell}) : i \in \mathbb{N}_m, \ell \in \mathbb{N}_n\}$
- Assume the tasks have a common input space, $X_\ell = X$
- For each task, learn a function $f_\ell : \mathcal{X}_\ell \to \mathcal{Y}_\ell$
- At first we will assume the functions are linear, $f_\ell(x_{j\ell}) = u'_\ell x_{j\ell}$
- Minimize a loss term + regularization term:

$$R(u) := \frac{1}{nm} \sum_{\ell \in \mathbb{N}_n} \sum_{j \in \mathbb{N}_m} L(y_{j\ell}, u'_\ell x_{j\ell}) + \gamma J(u)$$

regularization term

  where the regularization term is over parameters *u* of all tasks

- **Statistical interpretation:** similar to the single-task case, but now there is a model for each task, and the prior is a **joint prior** (log-likelihood from a **joint distribution**) for parameters of all * tasks

Following the approach from T. Evgeniou, C. A. Micchelli, and M. Pontil. Learning Multiple Tasks with Kernel Methods. Journal of Machine Learning Research 6: 615-637, 2005. Images from that paper.

Multitask learning with a kernel trick, linear case

- Form of the prior: collect parameters of all tasks into a long vector $u = (u_\ell : \ell \in \mathbb{N}_n) \in \mathbb{R}^{nd}$ and set $J(u) = u'Eu$ where *E* is a matrix that represents known relationships between tasks. We assume *E* is symmetric and positive definite.
- In the case where *E* is diagonal the learning reduces to the single-task case, each task is learned independently (for example when *E* is an identity matrix, $J(u) = \sum_{\ell \in \mathbb{N}_n} \|u_\ell\|^2$ which has separate terms for each task)
- For a suitable choice of E the regularization becomes $\sum_{\ell, q \in \mathbb{N}_n} \|u_\ell - u_q\|^2$ which forces all tasks to use similar parameters.
- Assume the input-output functions are of the form $u_\ell = B'_\ell w, \;\; \ell \in \mathbb{N}_n$ so that $f_\ell(x) = w' B_\ell x$ where *w* is common to all tasks
- Denote the concatenation of the feature matrices by $B := [B_\ell : \ell \in \mathbb{N}_n]$

Following the approach from T. Evgeniou, C. A. Micchelli, and M. Pontil. Learning Multiple Tasks with Kernel Methods. Journal of Machine Learning Research 6: 615-637, 2005. Images from that paper.

Multitask learning with a kernel trick, linear case

- We wish to convert the loss function+regularization to a form

$$S(w) := \frac{1}{nm} \sum_{\ell \in N_n} \sum_{j \in N_m} L(y_{j\ell}, w' B_\ell x_{j\ell}) + \gamma w' w$$

  where the regularization (prior) term **does not contain a task relationship matrix**

- It turn out this is possible. If we know *B* and set $E = (B'B)^{-1}$, then the new form is related to the old form by $S(w) = R(B'w)$. (or if we know *E* and set $B = T'E^{-1}$)

- Because the multi-task case is now in a similar form as the single-task case, the **representer theorem** applies and the solution has the form $w^* = \sum_{j \in \mathbb{N}_m} \sum_{\ell \in \mathbb{N}_n} \underbrace{c_{j\ell}}_{\text{weights to be solved}} B_\ell x_{j\ell}$

- This form of w can be inserted to the previous equations to solve the task functions

Following the approach from T. Evgeniou, C. A. Micchelli, and M. Pontil. Learning Multiple Tasks with Kernel Methods. Journal of Machine Learning Research 6: 615-637, 2005. Images from that paper.

Multitask learning with a kernel trick, linear case

- The optimal function of each task *q* has the form

$$f_q^*(x) = \sum_{j \in \mathbb{N}_m} \sum_{\ell \in \mathbb{N}_n} c_{j\ell} K((x_{j\ell}, \ell), (x, q)), \;\; x \in \mathbb{R}^d, q \in \mathbb{N}_n$$

where the kernel function is a kernel between any two samples (*x* and *t)* from **any two tasks** (*x* from task *l*, *t* from task *q*) :

$$K((x, \ell), (t, q)) = x' B'_\ell B_q t, \;\; x, t \in \mathbb{R}^d, \; \ell, q \in \mathbb{N}_n$$

- This kernel is a **linear multi-task kernel**
- The kernel is computes based both on the input features of the two samples, and **which tasks they come from**
- When the kernel is known, the **optimal functions** can be found by minimizing the loss+regularized with respect to the weights $c_{j\ell}$ (for example by gradient descent, or by more advanced methods if the equations have a suitable form)

Following the approach from T. Evgeniou, C. A. Micchelli, and M. Pontil. Learning Multiple Tasks with Kernel Methods. Journal of Machine Learning Research 6: 615-637, 2005. Images from that paper.

Multitask learning with a kernel trick, linear case

- **Different choices of the kernel** lead to different learning problems

- The regularizer $J(u) = \sum_{\ell,q \in \mathbb{N}_n} u'_\ell u_q G_{\ell q}$ where $G = (G_{\ell,q} : \ell, q \in \mathbb{N}_n)$ is a positive definite matrix corresponds to the kernel $K((x,\ell),(t,q)) = x' t \, G^{-1}_{\ell q}$. It emphasizes similarity of tasks *l*, *q* by weight $G_{\ell q}$

- The kernel $K((x,\ell),(t,q)) = (1-\lambda+\lambda n \delta_{\ell q}) x' t, \;\; \ell, q \in \mathbb{N}_n, \; x, t \in \mathbb{R}^n$ corresponds to the regularizer

$$J(u) = \frac{1}{n}\left(\sum_{\ell \in \mathbb{N}_n} \|u_\ell\|^2 + \frac{1-\lambda}{\lambda} \sum_{\ell \in \mathbb{N}_n} \|u_\ell - \frac{1}{n} \sum_{q \in \mathbb{N}_n} u_q\|^2 \right)$$

which has a trade-off (controlled by λ) between keeping parameters of individual tasks small, and keeping parameters of each task similar to the average parameter value over tasks.

Following the approach from T. Evgeniou, C. A. Micchelli, and M. Pontil. Learning Multiple Tasks with Kernel Methods. Journal of Machine Learning Research 6: 615-637, 2005. Images from that paper.

Multitask learning with a kernel trick, linear case

- Task-clustering regularizer:

$$J(u) = \min \left\{ \sum_{k \in \mathbb{N}_c} \left(\sum_{\ell \in \mathbb{N}_n} \rho_k^{(\ell)} \|u_\ell - u_{0k}\|^2 + \rho \|u_{0k}\|^2 \right) : u_{0k} \in \mathbb{R}^d, k \in \mathbb{N}_c \right\}$$

where k indexes the c tasks, $\rho_k^{(\ell)} \geq 0, \rho > 0,$ u_{0k} are cluster probabilities, corresponds to

$$J(u) = \sum_{\ell, q \in \mathbb{N}_n} u'_\ell u_q G_{\ell q} \quad \text{where} \quad G_{\ell q} = \sum_{k \in \mathbb{N}_c} \left(\rho_k^{(\ell)} \delta_{\ell q} - \frac{\rho_k^{(\ell)} \rho_k^{(q)}}{\rho + \sum_{r \in \mathbb{N}_n} \rho_k^{(r)}} \right)$$

The corresponding kernel is $K((x, \ell), (t, q)) = G_{\ell q}^{-1} x' t$

Following the approach from T. Evgeniou, C. A. Micchelli, and M. Pontil. Learning Multiple Tasks with Kernel Methods. Journal of Machine Learning Research 6: 615-637, 2005. Images from that paper.

Multitask learning with a kernel trick, nonlinear case

- The previous kernels can be used with nonlinear relationships, just replace x't with a nonlinear similarity function like exp(-||x-t||^2)
- Generally: for each task *l* we use nonlinear functions of the form

  $$f_\ell(x) = \langle w, \Phi_\ell(x) \rangle, \ \ x \in \mathcal{X}, \ \ \ell \in \mathbb{N}_n$$

  where $\Phi_\ell(x)$ is a task-dependent nonlinear transformation and *w* creates a linear projection to the output.
- We again minimize a sum of losses+regularization

$$S(w) := \frac{1}{nm} \sum_{\ell \in N_n} \sum_{j \in N_m} L(y_{j\ell}, \langle w, \Phi_\ell(x_{j\ell}) \rangle) + \gamma \langle w, w \rangle$$

- By the representer theorem the solution again has the form

$$f_q^*(x) = \sum_{j \in \mathbb{N}_m} \sum_{\ell \in \mathbb{N}_n} c_{j\ell} K((x_{j\ell}, \ell), (x, q)), \ \ x \in \mathbb{R}^d, q \in \mathbb{N}_n$$

where the kernel is $K((x,\ell),(t,q)) = \langle \Phi_\ell(x), \Phi_q(t) \rangle \ \ x, t \in \mathcal{X}, \ \ell, q \in \mathbb{N}_n$

Following the approach from T. Evgeniou, C. A. Micchelli, and M. Pontil. Learning Multiple Tasks with Kernel Methods. Journal of Machine Learning Research 6: 615-637, 2005. Images from that paper.

Multitask learning with a kernel trick, nonlinear case

- It turns out that $K((x,\ell),(t,q)) = G(z_\ell(x), z_q(t))$ where $z_\ell(x)$ and $z_q(t)$ are some task-dependent transformations, is a valid multi-task kernel.
- For example, when G is Gaussian and transformations are linear, this becomes

$$K((x,\ell),(t,q)) = \exp(-\beta \|B_\ell x - B_q t\|^2)$$

where B_ℓ and B_q are task-dependent transformations

Part 2: Nonlinear Multitask learning with nonparametric methods

Multitask learning with nonparametric methods

- Most methods restrict input-output functions to a particular parametric family
- Even kernel methods use a parametric family: typically linear projection after a nonlinear transformation.
- Can we built multi-task methods that do not require a parametric family? Yes.
- Here we consider multi-task learning with **nonparametric methods**, in particular with **Gaussian processes**

Following the approach from Kai Yu, Volker Tresp, and Anton Schwaighofer. Learning Gaussian Processes from Multiple Tasks. In Proceedings of the 22nd International Conference on Machine Learning (ICML 2005), 2005.. Images from that paper.

Multitask learning with nonparametric methods

- A **Gaussian process** is a prior over input-output functions
- A Gaussian process prior does not specify any parametric family for the functions, it only specifies how output values for two different input points are likely to be related.
- A Gaussian process is specified by a mean function and a covariance function
- Idea: for any two input points x, x', a Gaussian process prior says the output values f(x), f(x') jointly have a Gaussian distribution,
  whose mean is given by the mean function $\mu(x)=E[f(x)]$,
  and covariance is given by the covariance function
  $k(x,x')=E[(f(x)-\mu(x))(f(x')-\mu(x'))]$
- If the likelihood function is also Gaussian, then the posterior distribution over the input-output functions (after seeing the observations) is also a Gaussian process!

Following the approach from Kai Yu, Volker Tresp, and Anton Schwaighofer. Learning Gaussian Processes from Multiple Tasks. In Proceedings of the 22nd International Conference on Machine Learning (ICML 2005), 2005.. Images from that paper.

Multitask learning with nonparametric methods

• In Gaussian process based inference, the task is to compute the mean function and covariance function of the posterior distribution, given the prior and the observations.

• When the posterior distribution has been computed, it can be used to predict values of the output at new input points, as an expectation over the posterior.

• Gaussian process prediction gives both the prediction at the new point (= mean of the function value over the posterior) and the uncertainty about the prediction (= variance of the function value over the posterior)

• Gaussian process computation can be done in closed form if the prior and likelihood are simple.

Following the approach from Kai Yu, Volker Tresp, and Anton Schwaighofer. Learning Gaussian Processes from Multiple Tasks. In Proceedings of the 22nd International Conference on Machine Learning (ICML 2005), 2005.. Images from that paper.

Multitask learning with nonparametric methods

- If the prediction model is noise-free, $y=f(x)+e$,
- Then the posterior given a data set D={(x,y)} is

$$p(y|x,D)=\int_f p(y|x,f)\,p(f|D)\,df$$

$$\sim\frac{1}{Z}\exp\left(-(y-\hat{y}_{x,D})^2/2\sigma^2_{x,D}\right)$$

- where $\hat{y}_{x,D}=\boldsymbol{k}^T_{x,D}\boldsymbol{K}_D^{-1}\boldsymbol{y}_D$, $\boldsymbol{K}_D$ is the covariance matrix of the observed data D (evaluated by computing the covariance function between all pairs of the observed data), $\boldsymbol{k}_{x,D}=[k(x,x_1),\ldots,k(x,x_N)]^T$ is the covariance function computed between the new input point and all observed input points, and $\boldsymbol{y}_D=[y_{1,}\ldots,y_N]^T$ is the set of observed output values
- Similarly, the variance (uncertainty) at the new input point is $\sigma^2{}_{x,D}=k(x,x)-\boldsymbol{k}^T_{x,D}\boldsymbol{K}_D^{-1}\boldsymbol{k}_{x,D}$
- The noisy predictions are slightly more complicated, details added later

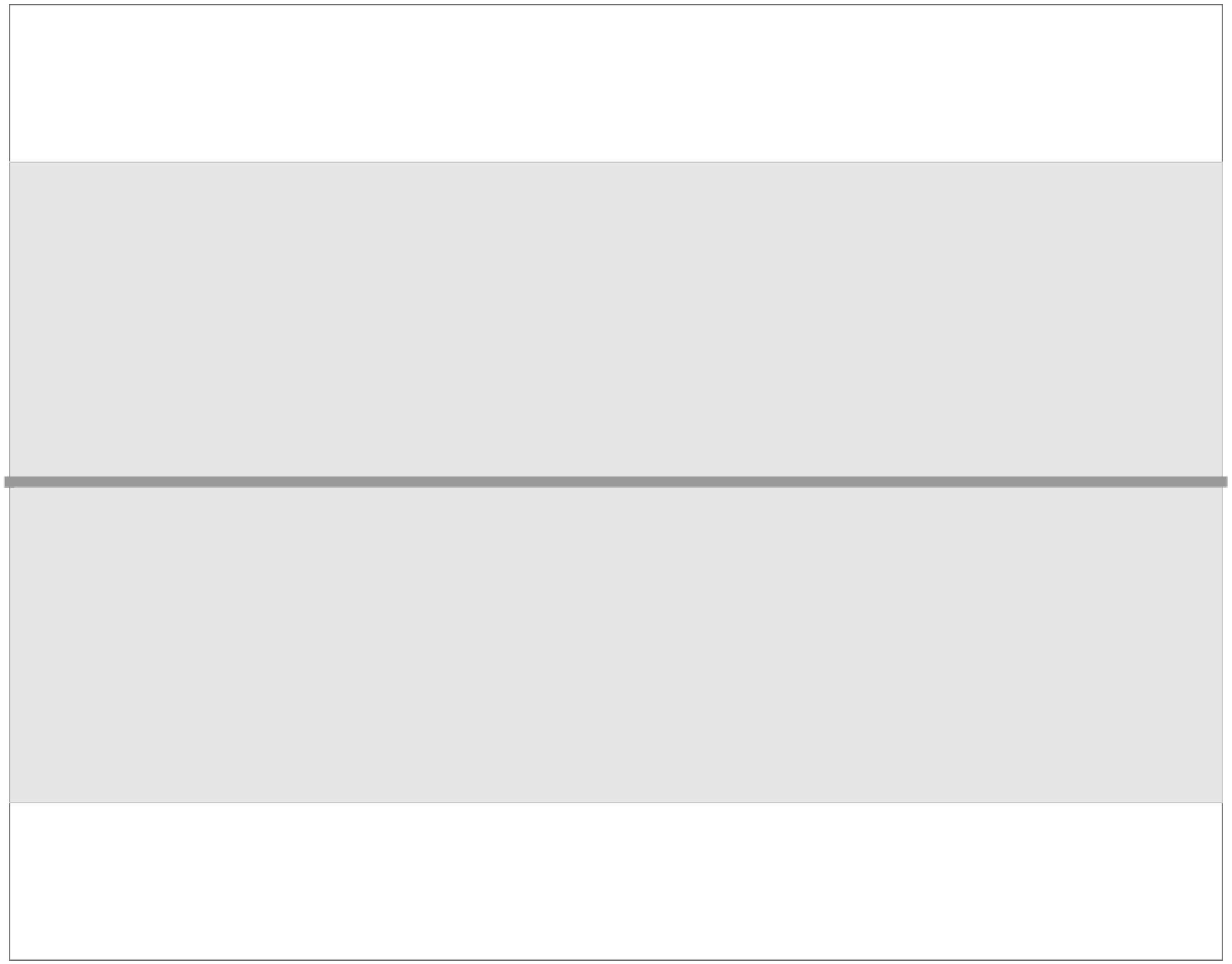
Following the approach from Kai Yu, Volker Tresp, and Anton Schwaighofer. Learning Gaussian Processes from Multiple Tasks. In Proceedings of the 22nd International Conference on Machine Learning (ICML 2005), 2005.. Images from that paper.

Gaussian Processes (GPs)

The gray line shows the mean function.

The light gray area shows the standard deviation (uncertainty).

This is the GP prior over functions before seeing any data.



One-dimensional input values shown on the horizontal axis

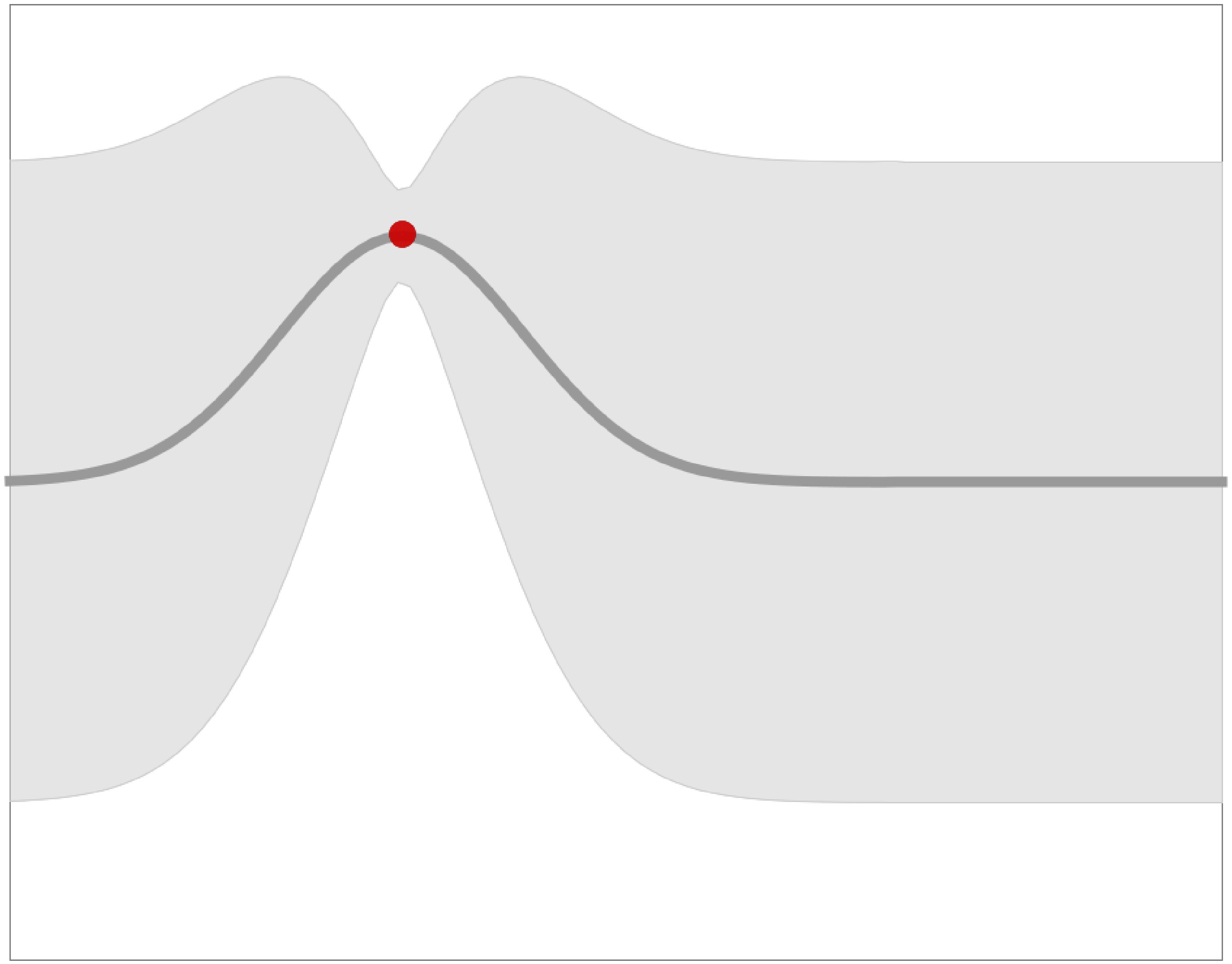
Distribution of output values (distribution of input-output functions) shown on the vertical axis

Gaussian Processes (GPs)

The gray line shows the mean function.

The light gray area shows the standard deviation (uncertainty).

This is the GP prior over functions before seeing any data.

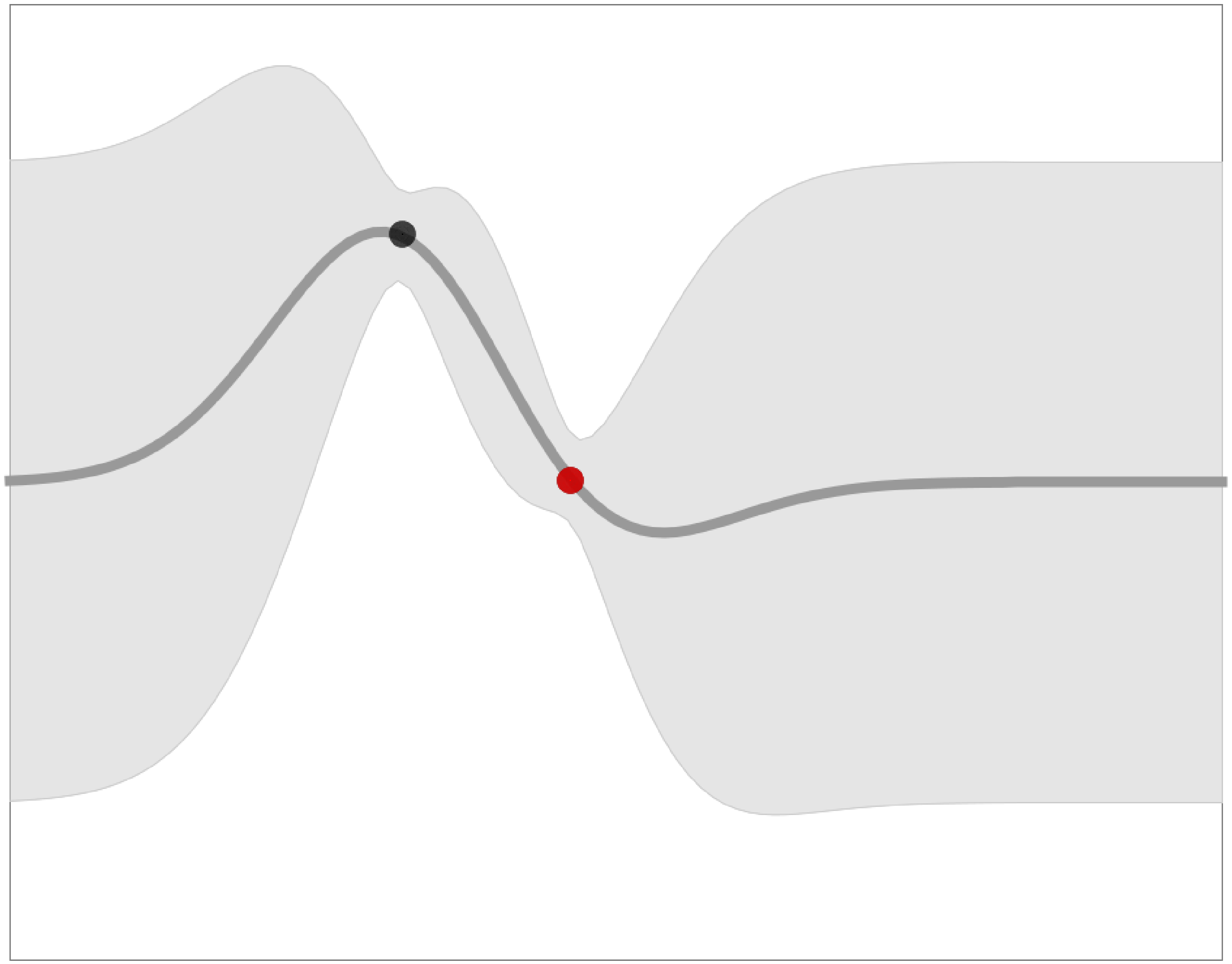


Gaussian Processes (GPs)

The gray line shows the mean function.

The light gray area shows the standard deviation (uncertainty).

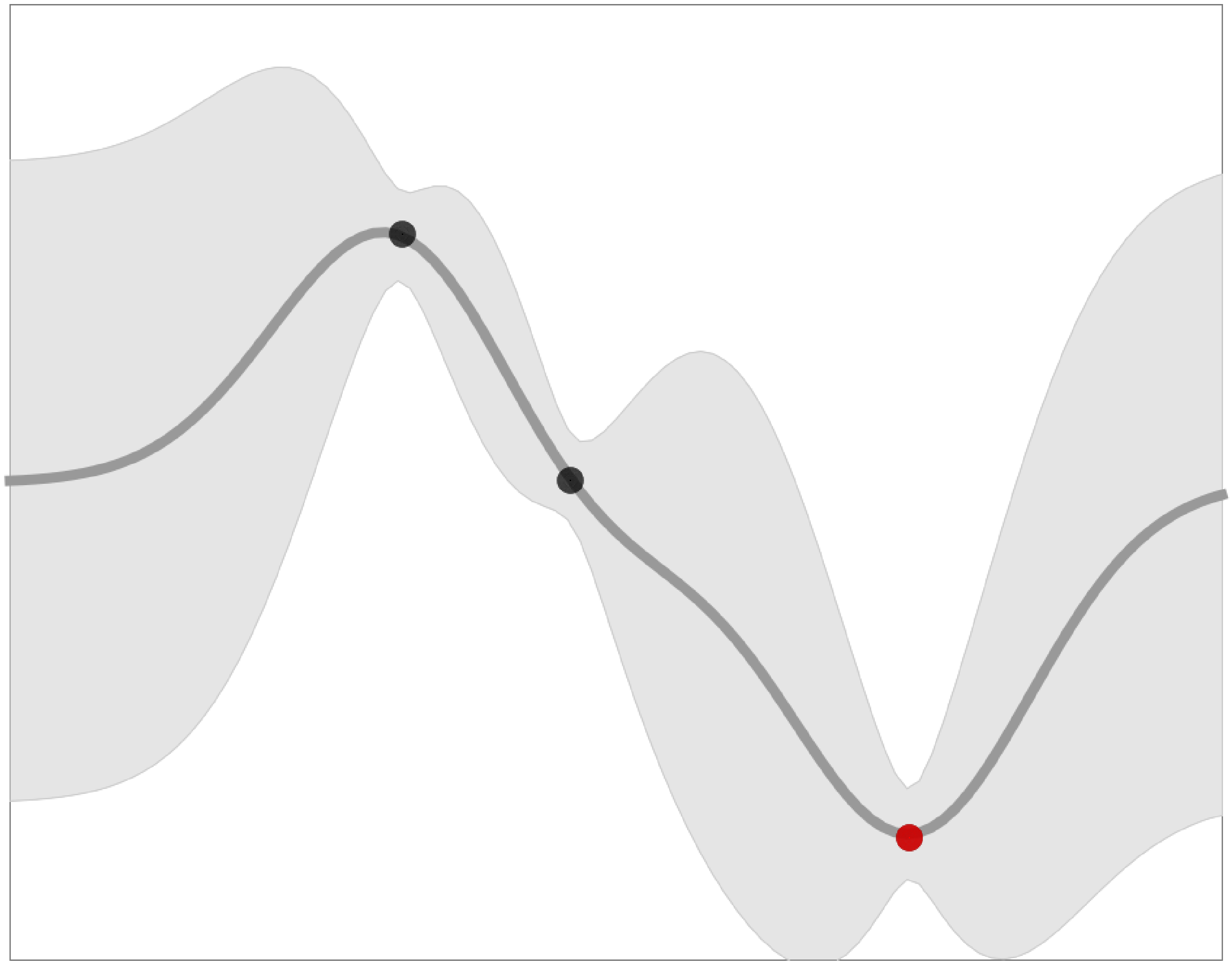
This is the GP prior over functions before seeing any data.



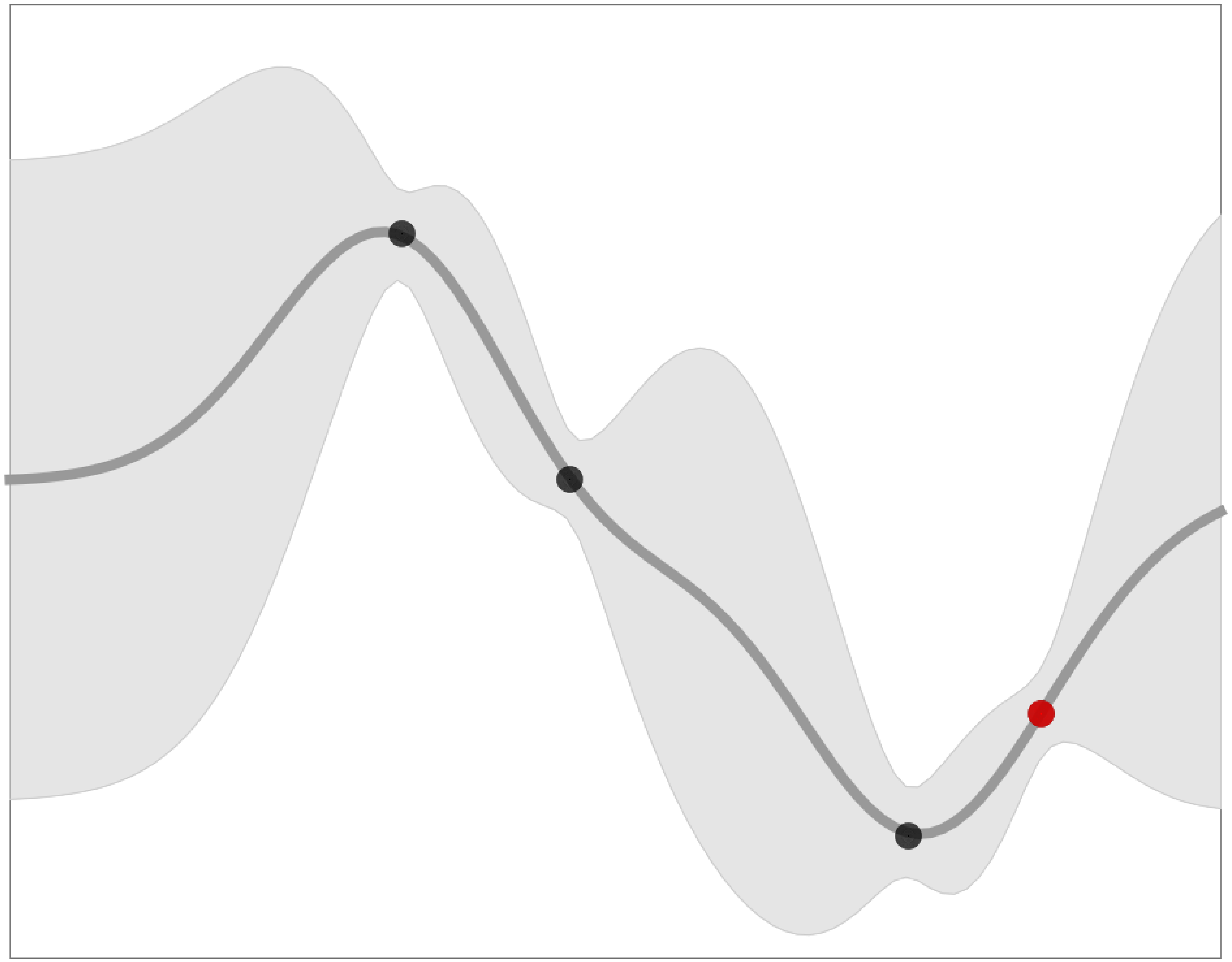
Gaussian Processes (GPs)

After seeing observations, posterior **uncertainty** about the function decreases at observations, and at **inputs correlated with the observation points**.

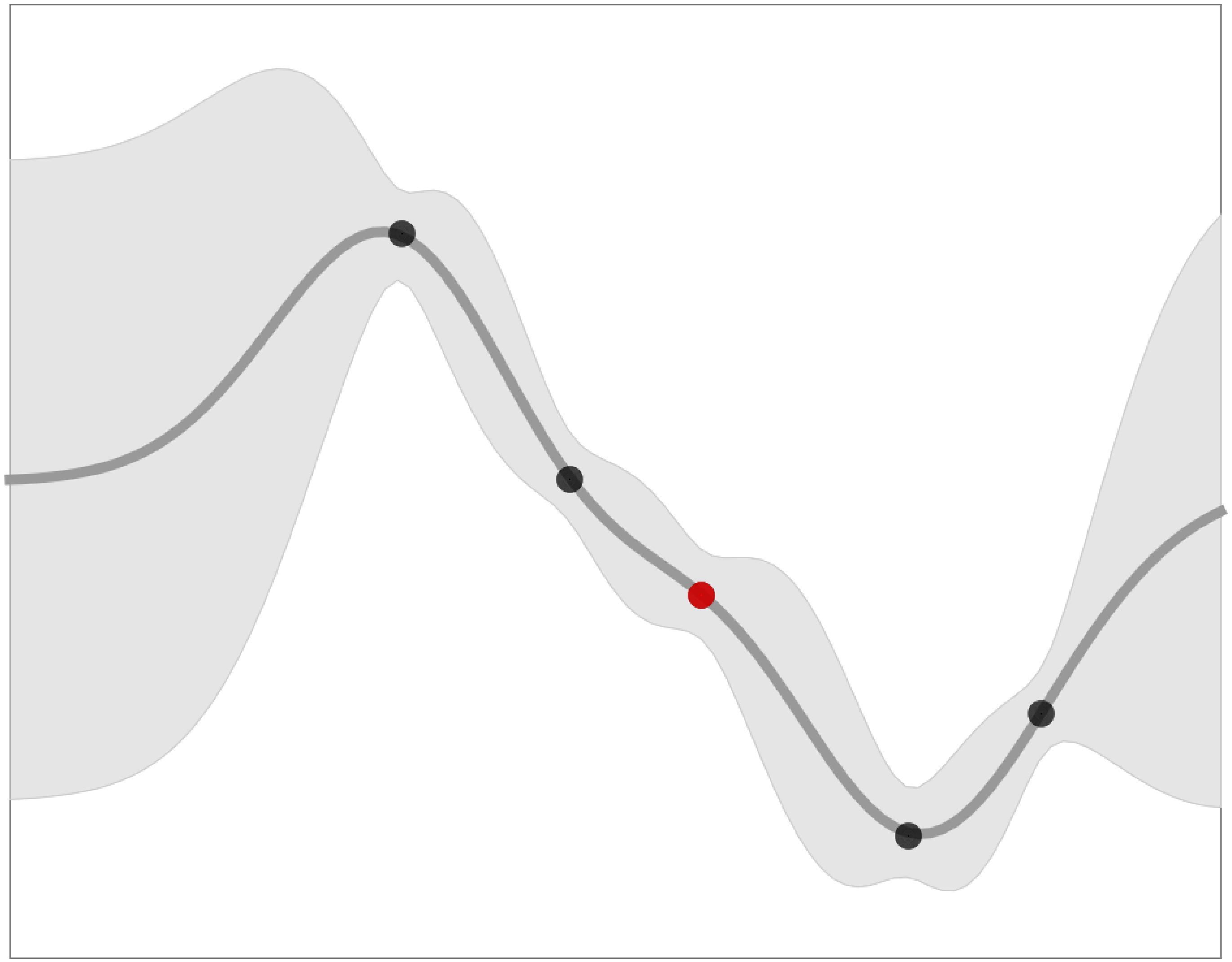
(Covariance function tells how much each pair of inputs is correlated over the possible functions)



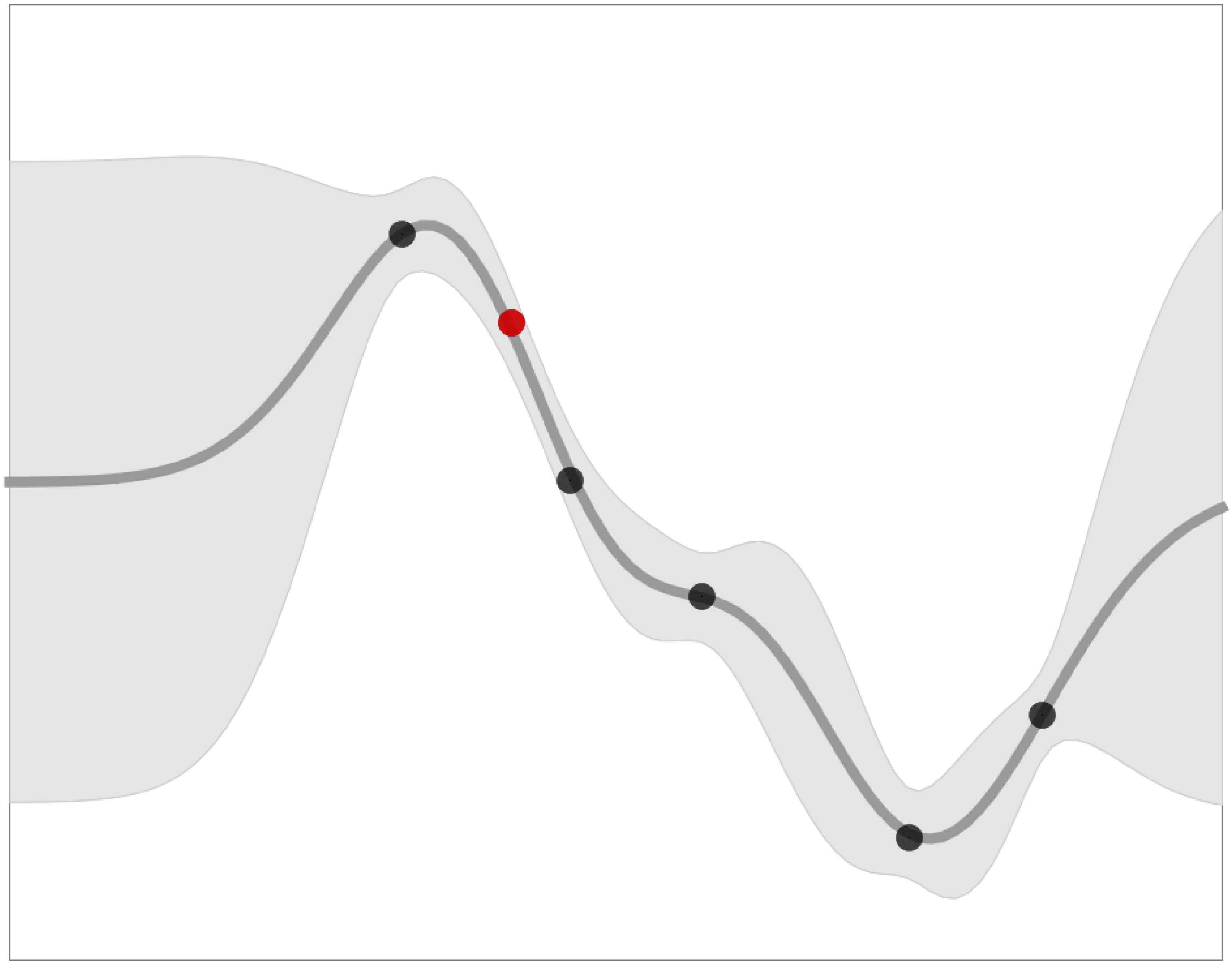
Gaussian Processes (GPs)



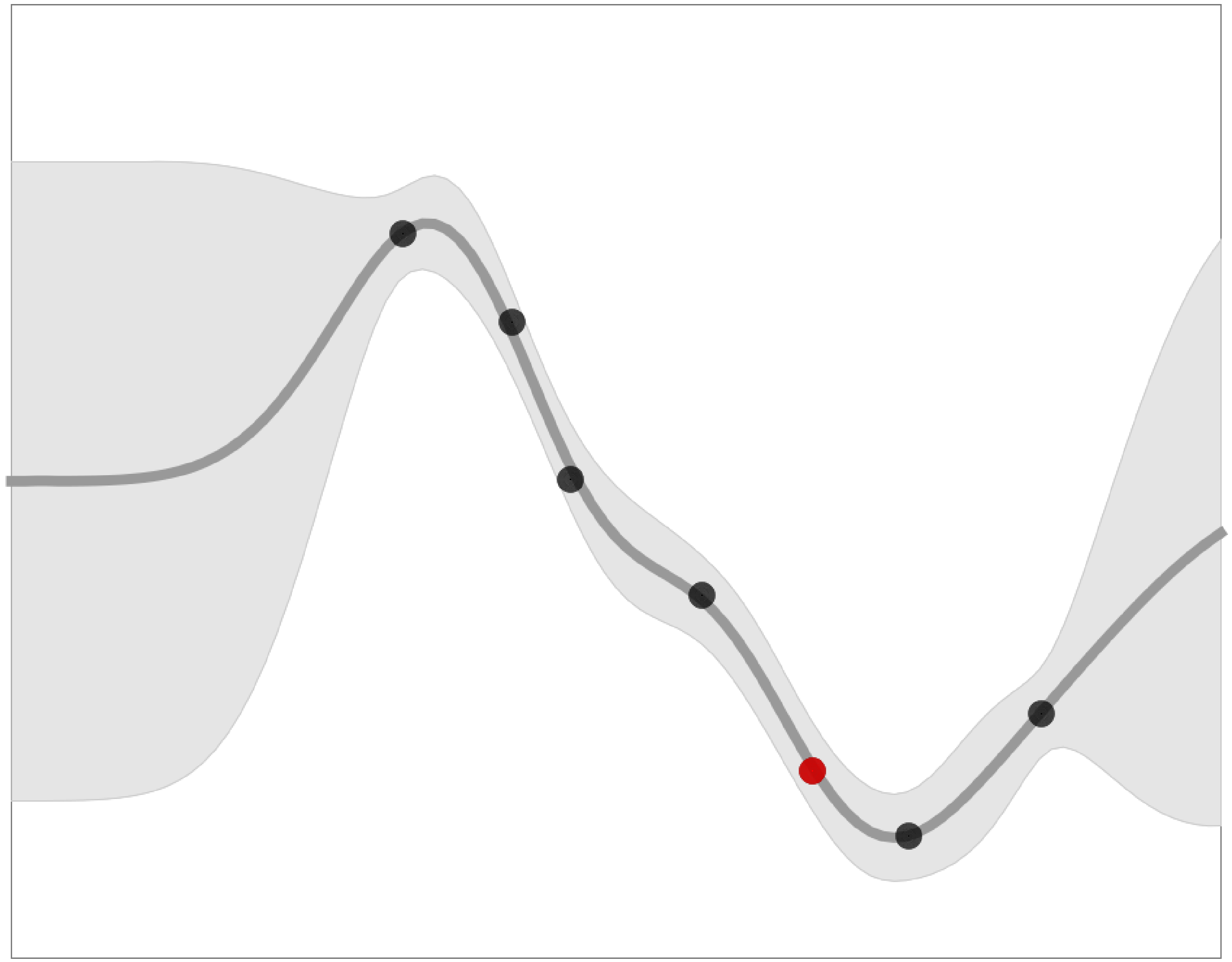
Gaussian Processes (GPs)



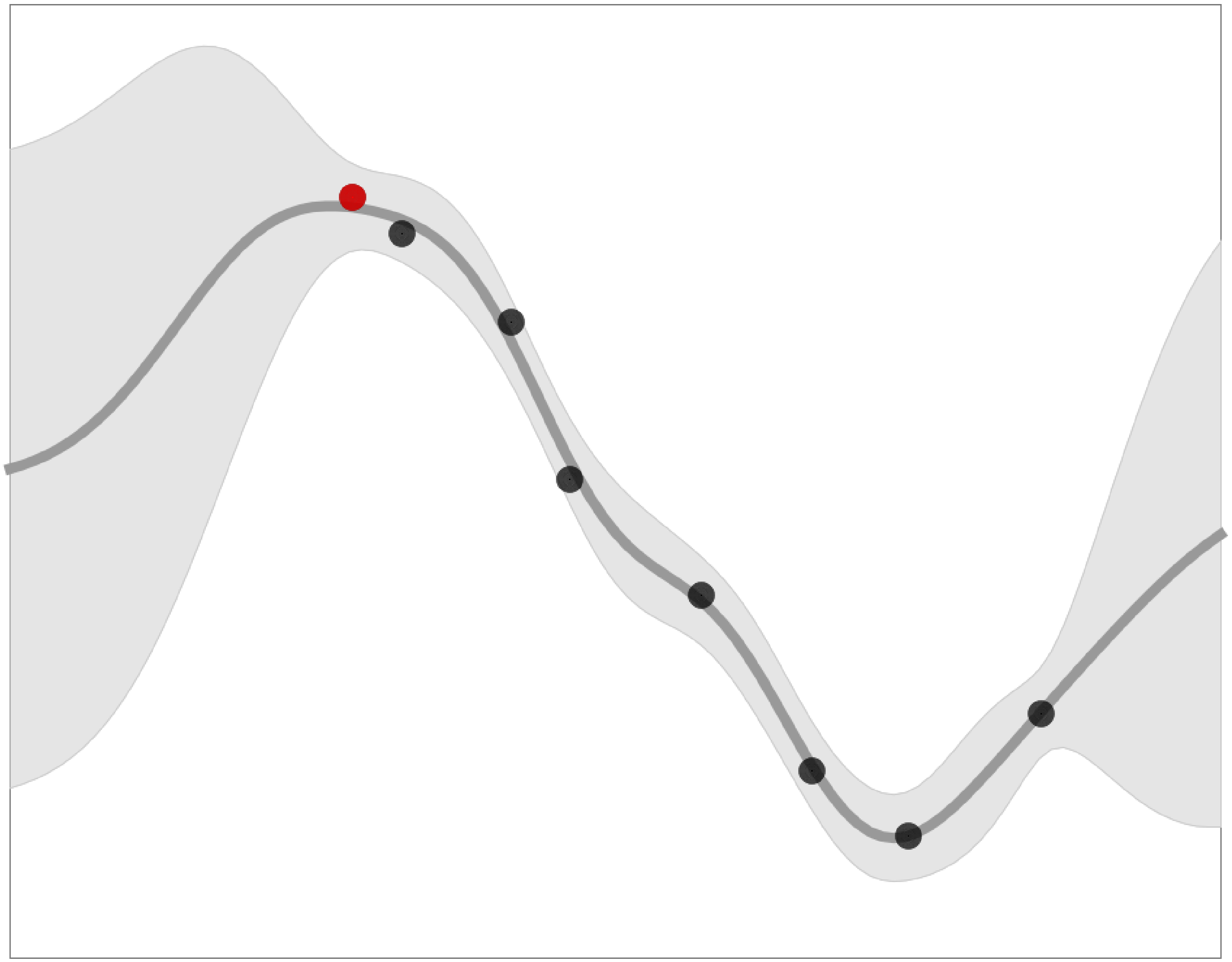
Gaussian Processes (GPs)



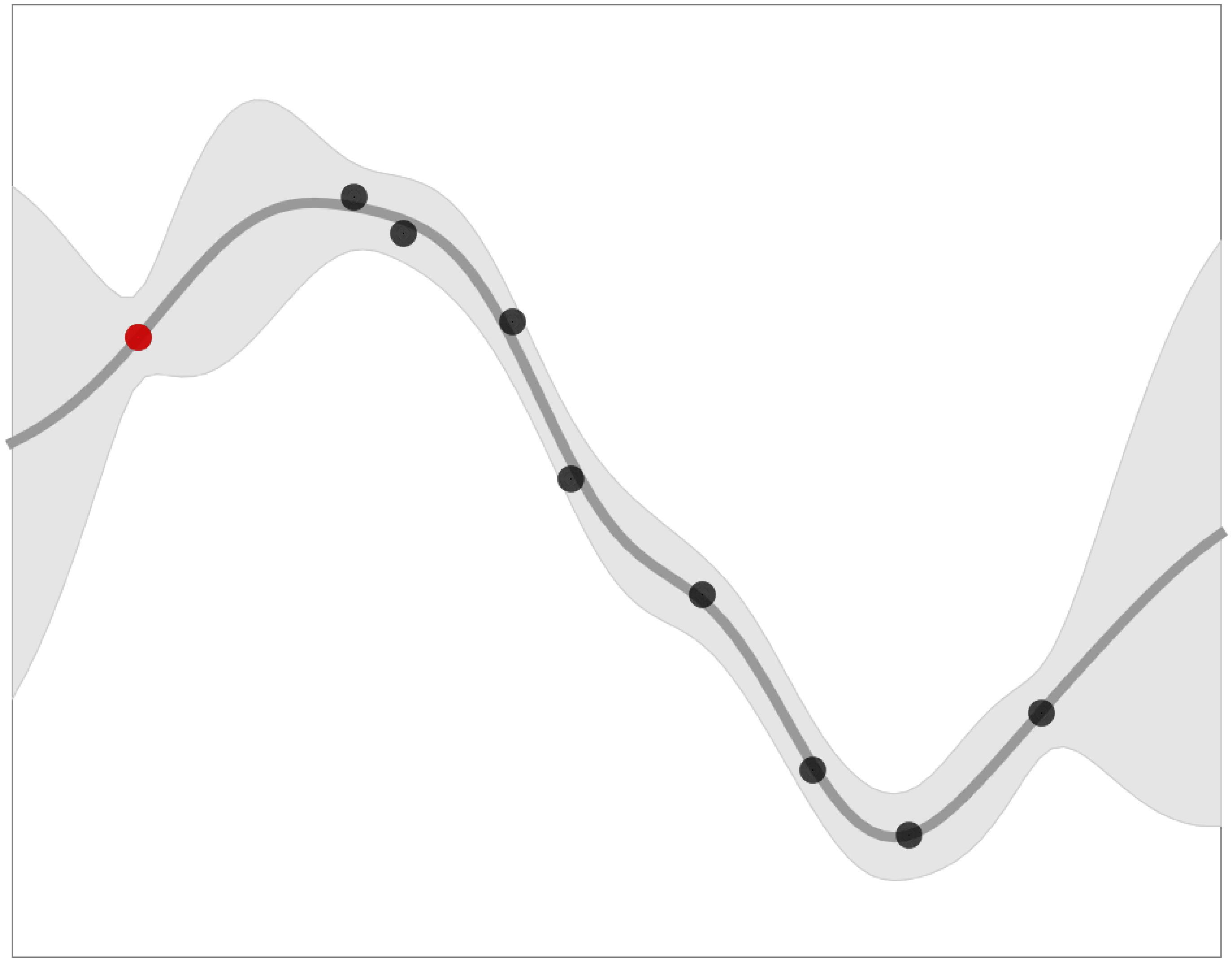
Gaussian Processes (GPs)



Gaussian Processes (GPs)



Gaussian Processes (GPs)



Multitask learning with nonparametric methods

- Multitask case: create the covariance function between input points from two tasks *l* and *k*, to depend on the task as well as the input location

$$\langle f_l(\mathbf{x}) f_k(\mathbf{x}') \rangle = \underbrace{K^f_{lk}}_{\text{kernel between task identifiers}} \underbrace{k^x(\mathbf{x}, \mathbf{x}')}_{\text{kernel between input locations}} \qquad y_{il} \sim \mathcal{N}(f_l(\mathbf{x}_i), \sigma_l^2),$$

- The resulting equation for posterior prediction at a new point $\mathbf{x}_*$ becomes

$$\bar{f}_l(\mathbf{x}_*) = (\mathbf{k}^f_l \otimes \mathbf{k}^x_*)^T \Sigma^{-1} \underbrace{\mathbf{y}}_{\text{observed output values}}$$

$$\Sigma = K^f \otimes K^x + D \underbrace{\otimes}_{\text{Kronecker product}} I$$

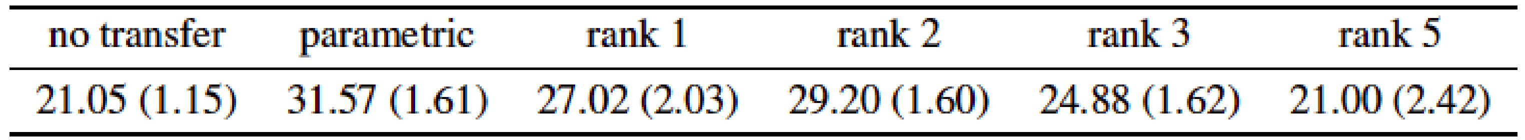
- Hyperparameters can be learned by maximizing the likelihood (probability of observations and parameters)

$$L_{\text{comp}} = -\frac{N}{2} \log |K^f| - \frac{M}{2} \log |K^x| - \frac{1}{2} \operatorname{tr}\left[\left(K^f\right)^{-1} F^{\mathrm{T}} \left(K^x\right)^{-1} F \right]$$
$$- \frac{N}{2} \sum_{l=1}^{M} \log \sigma_l^2 - \frac{1}{2} \operatorname{tr}\left[(Y - F) D^{-1} (Y - F)^{\mathrm{T}} \right] - \frac{MN}{2} \log 2\pi$$

Following the approach from Kai Yu, Volker Tresp, and Anton Schwaighofer. Learning Gaussian Processes from Multiple Tasks. In Proceedings of the 22nd International Conference on Machine Learning (ICML 2005), 2005.. Images from that paper.

Multitask learning with nonparametric methods

- Example: school data. http://www.cmm.bristol.ac.uk/learning-training/multilevel-m-support/datasets.shtml.
- Examination records from 139 secondary schools in years 1985, 1986 and 1987. A random 50% sample with 15362 students.
- Task: predict exam score of a student belonging to a specific school, based on four student-dependent features (year of the exam, gender, VR band and ethnic group) and four school-dependent features (percentage of students eligible for free school meals, percentage of students in VR band 1 , school gender and school denomination).
- Result:



no transfer	parametric	rank 1	rank 2	rank 3	rank 5
21.05 (1.15)	31.57 (1.61)	27.02 (2.03)	29.20 (1.60)	24.88 (1.62)	21.00 (2.42)

Table 1: Percentage variance explained on the school dataset for various situations. The figures in brackets are standard deviations obtained from the ten replications.

Following the approach from Kai Yu, Volker Tresp, and Anton Schwaighofer. Learning Gaussian Processes from Multiple Tasks. In Proceedings of the 22nd International Conference on Machine Learning (ICML 2005), 2005.. Images from that paper.

References

- T. Evgeniou, C. A. Micchelli, and M. Pontil. **Learning Multiple Tasks with Kernel Methods**. Journal of Machine Learning Research 6: 615-637, 2005.
- Bonilla, E. V., Chai, K. M. A., and Williams, C. K. I. 2008. **Multitask Gaussian Process Prediction**. Advances in Neural Information Processing Systems (NIPS). http://papers.nips.cc/paper/3189-multi-task-gaussian-process-prediction
- Kai Yu, Volker Tresp, and Anton Schwaighofer. **Learning Gaussian Processes from Multiple Tasks**. In Proceedings of the 22nd International Conference on Machine Learning (ICML 2005), 2005. http://www.tresp.org/papers/multitaskGP_final.pdf
- Mauricio A. Álvarez, Neil D. Lawrence; **Computationally Efficient Convolved Multiple Output Gaussian Processes**. Journal of Machine Learning Research, 12(May):1459−1500, 2011. http://www.jmlr.org/papers/volume12/alvarez11a/alvarez11a.pdf