

Modeliranje 1-D porazdelitve: razpadi Higgsovega bozona

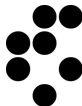
1. naloga: Praktikum strojnega učenja v fiziki

predavatelj: prof. dr. Borut Paul Kerševan

asistent: Jan Gavranovič

Institut Jožef Stefan (F9)

13. oktober 2023



**Institut
"Jožef Stefan"
Ljubljana, Slovenija**



Univerza v Ljubljani
Fakulteta *za matematiko
in fiziko*

2. vaje

Pregled 2. vaje

1. Ocenjevanje parametrov (ponovitev predavanj):
 - Maximum likelihood estimation (MLE),
 - Maximum a posteriori estimation (MAP).
2. Različne metode (unbinned) fitanja in knjižnica `scikit-learn`:
 - Kernel Ridge Regression (KRR),
 - Gaussian Process Regression (GPR).
3. Podrobnejša navodila za reševanje naloge.
4. Zaključek 1. naloge.

Ocenjevanje parametrov

(*Parameter estimation*)

Maximum likelihood estimation - MLE

- Želimo poskati parametre θ , ki dajo največjo verjetnost podatkom \mathcal{D} .
- MLE definicija:

$$\hat{\theta}_{\text{MLE}} \equiv \underset{\theta}{\operatorname{argmax}} p(\mathcal{D}|\theta).$$

- Predpostavimo, da so podatki *independent and identically distributed (iid)*:

$$p(\mathcal{D}|\theta) = \prod_{n=1}^N p(\mathbf{y}_n|\mathbf{x}_n, \theta), \quad \text{kjer je } p(\mathbf{y}_n|\mathbf{x}_n, \theta) \text{ (pogojni) likelihood.}$$

- Ponavadi delamo z **log-likelihood**:

$$\mathcal{L}(\theta) \equiv \log p(\mathcal{D}|\theta) = \sum_{n=1}^N \log p(\mathbf{y}_n|\mathbf{x}_n, \theta).$$

- Glavna ideja strojnega učenja:

optimalni parametri: $\hat{\theta} = \underset{\theta}{\operatorname{argmin}} \mathcal{L}(\theta)$, kjer je $\mathcal{L}(\theta)$ funkcija izgube/cene.

- MLE je tako podan z

$$\hat{\theta}_{\text{MLE}} = \underset{\theta}{\operatorname{argmax}} \sum_{n=1}^N \log p(\mathbf{y}_n | \mathbf{x}_n, \theta).$$

- Optimizacijski algoritmi so narejeni za minimizacijo funkcij \Rightarrow uporabimo **negative log-likelihood**:

$$\mathcal{L}(\theta) = \text{NLL}(\theta) = -\log p(\mathcal{D} | \theta) = -\sum_{n=1}^N \log p(\mathbf{y}_n | \mathbf{x}_n, \theta).$$

- Za modele brez oznak \mathbf{y} (*labels*) je MLE (*unsupervised*):

$$\hat{\theta}_{\text{MLE}} = \underset{\theta}{\operatorname{argmax}} \left[-\sum_{n=1}^N \log p(\mathbf{x}_n | \theta) \right]$$

ali zapisano s pričakovano vrednostjo:

$$\hat{\theta}_{\text{MLE}} = \underset{\theta}{\operatorname{argmax}} [-\mathbb{E}_{\mathbf{x} \sim p_{\mathcal{D}}} \log p(\mathbf{x} | \theta)],$$

kjer je $p_{\mathcal{D}}$ empirična/podatkovna porazdelitev.

Maximum a posteriori estimation - MAP

- Ponovimo Bayesov izrek:

$$p(\boldsymbol{\theta}|\mathcal{D}) = \frac{p(\mathcal{D}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{p(\mathcal{D})} \propto p(\mathcal{D}|\boldsymbol{\theta})p(\boldsymbol{\theta}).$$

- Količine zgoraj predstavljajo:
 - $p(\boldsymbol{\theta}|\mathcal{D})$ - aposteriorna porazdelitev $\boldsymbol{\theta}$ (znanje o parametrih po opazovanju podatkov),
 - $p(\mathcal{D}|\boldsymbol{\theta})$ - verjetje (likelihood) podatkov pri danih parametrih (kaj nam povejo podatki),
 - $p(\boldsymbol{\theta})$ - prior (kaj vemo o parametrih pred opazovanjem podatkov),
 - $p(\mathcal{D})$ - porazdelitev podatkov (normalizacijski faktor neodvisen od parametrov).
- V tem primeru iščemo najboljše parametre logaritma aposteriorne porazdelitve:

$$\begin{aligned}\hat{\boldsymbol{\theta}}_{\text{MAP}} &= \operatorname{argmax}_{\boldsymbol{\theta}} p(\boldsymbol{\theta}|\mathcal{D}) = \operatorname{argmax}_{\boldsymbol{\theta}} [\log p(\mathcal{D}|\boldsymbol{\theta}) + \log p(\boldsymbol{\theta})] \\ &= \operatorname{argmax}_{\boldsymbol{\theta}} \left[\sum_i \log p(\mathcal{D}_i|\boldsymbol{\theta}) + \log p(\boldsymbol{\theta}) \right].\end{aligned}$$

- MAP je ekvivalenten MLE, če je prior enak 1 (enakomerna porazdelitev).

Kernel ridge regression (KRR)

Kernalizacija in ridge regression

- Poglejmo si enostaven primer **linearne regresije**: $y = \mathbf{w}^\top \mathbf{x}$, kjer je $\mathbf{x}, \mathbf{w} \in \mathbb{R}^D$ in $y \in \mathbb{R}$.
- Uvedemo **nelinerano bazno funkcijo** $\phi(\mathbf{x}) : \mathbb{R}^D \rightarrow \mathbb{R}^M$, ki preslika vhodne podatke $\mathbf{x} \in \mathbb{R}^D$ v prostor lastnosti (*feature space*) \mathbb{R}^M in doda kompleksnost modelu: $y = \mathbf{w}^\top \phi(\mathbf{x})$.
- Primer je **polinomska regresija**: $\mathbf{x} : \phi(\mathbf{x}) = (1, \mathbf{x}, \mathbf{x}^2, \mathbf{x}^3, \dots)^\top$, kjer so ϕ_i fiksne funkcije.
- Funkcija cene za **ridge regression** je MSE + regularizacijski člen:

$$\mathcal{L}(\mathbf{w}) = \frac{1}{2} \sum_{i=1}^N (y_i - \mathbf{w}^\top \phi(\mathbf{x}_i))^2 + \frac{\lambda}{2} \sum_{i=1}^N w_i^2 = \frac{1}{2} (\mathbf{y} - \Phi \mathbf{w})^\top (\mathbf{y} - \Phi \mathbf{w}) + \frac{\lambda}{2} \mathbf{w}^\top \mathbf{w}.$$

- Rešitev zapišemo s pomočjo gradienta $\nabla_{\mathbf{w}} \mathcal{L}(\mathbf{w})$ kot implicitno shemo:

$$\hat{\mathbf{w}} = \frac{1}{\lambda} \sum_{i=1}^N (y_i - \mathbf{w}^\top \phi(\mathbf{x}_i)) \phi(\mathbf{x}_i) = \sum_{i=1}^N \alpha_i \phi(\mathbf{x}_i) = \Phi^\top \boldsymbol{\alpha}.$$

- V $\mathcal{L}(\mathbf{w})$ vstavimo $\hat{\mathbf{w}}$ in dobimo:

$$\mathcal{L}(\boldsymbol{\alpha}) = \mathbf{y}^\top \mathbf{y} - \mathbf{y}^\top \Phi \Phi^\top \boldsymbol{\alpha} - \boldsymbol{\alpha}^\top \Phi \Phi^\top \mathbf{y} + \boldsymbol{\alpha}^\top \Phi \Phi^\top \Phi \Phi^\top \boldsymbol{\alpha} + \frac{\lambda}{2} \boldsymbol{\alpha}^\top \Phi \Phi^\top \boldsymbol{\alpha}.$$

Matrični zapis

- Vhodni podatki (linerno) in funkcije $\phi(\mathbf{x}_i)$ (nelinerano):

$$\mathbf{X}^\top = \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_N \end{bmatrix} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1D} \\ x_{21} & x_{22} & \cdots & x_{2D} \\ \vdots & \vdots & \ddots & \vdots \\ x_{N1} & x_{N2} & \cdots & x_{ND} \end{bmatrix} \quad \text{in} \quad \Phi(\mathbf{X})^\top = \begin{bmatrix} \phi(\mathbf{x}_1) \\ \phi(\mathbf{x}_2) \\ \vdots \\ \phi(\mathbf{x}_N) \end{bmatrix} = \begin{bmatrix} \phi_1(\mathbf{x}_1) & \phi_2(\mathbf{x}_1) & \cdots & \phi_M(\mathbf{x}_1) \\ \phi_1(\mathbf{x}_2) & \phi_2(\mathbf{x}_2) & \cdots & \phi_M(\mathbf{x}_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_1(\mathbf{x}_N) & \phi_2(\mathbf{x}_N) & \cdots & \phi_M(\mathbf{x}_N) \end{bmatrix}$$

- Uvedemo **kernel** $K = \Phi\Phi^\top$, ki je *simetrična in pozitivno semi-definitna matrika skalarnih produktov vektorjev \mathbf{x}_i* :

$$K = \Phi\Phi^\top = \phi(\mathbf{x}_i)\phi(\mathbf{x}_j)^\top = \begin{bmatrix} \phi(\mathbf{x}_1)^\top \phi(\mathbf{x}_1) & \phi(\mathbf{x}_1)^\top \phi(\mathbf{x}_2) & \cdots & \phi(\mathbf{x}_1)^\top \phi(\mathbf{x}_N) \\ \phi(\mathbf{x}_2)^\top \phi(\mathbf{x}_1) & \phi(\mathbf{x}_2)^\top \phi(\mathbf{x}_2) & \cdots & \phi(\mathbf{x}_2)^\top \phi(\mathbf{x}_N) \\ \vdots & \vdots & \ddots & \vdots \\ \phi(\mathbf{x}_N)^\top \phi(\mathbf{x}_1) & \phi(\mathbf{x}_N)^\top \phi(\mathbf{x}_2) & \cdots & \phi(\mathbf{x}_N)^\top \phi(\mathbf{x}_N) \end{bmatrix} \in \mathbb{R}^{N \times N}$$

Nov zapis optimalnih uteži za ridge regression s pomočjo K

- S tem lahko zapišemo

$$\mathcal{L}(\boldsymbol{\alpha}) = \mathbf{y}^\top \mathbf{y} - \mathbf{y}^\top \mathbf{K} \boldsymbol{\alpha} - \boldsymbol{\alpha}^\top \mathbf{K} \mathbf{y} + \boldsymbol{\alpha}^\top \mathbf{K}^2 \boldsymbol{\alpha} + \frac{\lambda}{2} \boldsymbol{\alpha}^\top \mathbf{K} \boldsymbol{\alpha}.$$

- Z minimizacijo $\nabla_{\boldsymbol{\alpha}} \mathcal{L}(\boldsymbol{\alpha}) = 0$ in nekaj premetavanja dobimo:

$$\hat{\boldsymbol{\alpha}} = (\mathbf{K} + \lambda' \mathbf{I})^{-1} \mathbf{y}.$$

- Uteži lahko sedaj zapišemo kot (vstavimo $\hat{\boldsymbol{\alpha}}$):

$$\hat{\mathbf{w}} = \boldsymbol{\Phi}^\top \hat{\boldsymbol{\alpha}} = \boldsymbol{\Phi}^\top (\mathbf{K} + \lambda' \mathbf{I})^{-1} \mathbf{y}.$$

- Prej smo imeli (normalna enačba):

$$\hat{\mathbf{w}} = (\boldsymbol{\Phi}^\top \boldsymbol{\Phi} + \lambda \mathbf{I})^{-1} \boldsymbol{\Phi}^\top \mathbf{y}.$$

- Sedaj imamo z uvedbo matrike K :

$$\hat{\mathbf{w}} = \boldsymbol{\Phi}^\top (\boldsymbol{\Phi} \boldsymbol{\Phi}^\top + \lambda' \mathbf{I})^{-1} \mathbf{y}.$$

Mercerjev teorem in kernel trik

- Za vsako pozitivno semi-definitno izbiro kernel matrike $K(\mathbf{x}_i, \mathbf{x}_j)$ je mogoče najti preslikavo $\phi(\mathbf{x}) : \mathbb{R}^D \rightarrow \mathbb{R}^M$, da je $K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)\phi(\mathbf{x}_j)^\top$.
- Zapišemo lahko pozitivno-semi definitno matriko K kot:

$$K(\mathbf{X}, \mathbf{X}') = \begin{bmatrix} K(\mathbf{x}_1, \mathbf{x}_1) & K(\mathbf{x}_1, \mathbf{x}_2) & \cdots & K(\mathbf{x}_1, \mathbf{x}_N) \\ K(\mathbf{x}_2, \mathbf{x}_1) & K(\mathbf{x}_2, \mathbf{x}_2) & \cdots & K(\mathbf{x}_2, \mathbf{x}_N) \\ \vdots & \vdots & \ddots & \vdots \\ K(\mathbf{x}_N, \mathbf{x}_1) & K(\mathbf{x}_N, \mathbf{x}_2) & \cdots & K(\mathbf{x}_N, \mathbf{x}_N) \end{bmatrix}$$

za vhodne podatke $\mathbf{x} \in \mathbb{R}^D$ (je funkcija samo teh in ne ϕ).

- Mercerjev teorem pove, da lahko iz pozitivno semi-definitnega jedra K pridemo do preslikave $\phi(\mathbf{x})$, ki je ustvarila ta kernel \Rightarrow **ne rabimo poznati oblike ϕ ampak samo matriko K .**

Primer: polinomska regresija

- Zamislimo si polinomsko nelinearno bazno funkcijo za D vhodnih dimenzij (dimenzija podatkov), $M = 3$ (dimenzija prostora lastnosti) in 2 vhodna podatka ($N = 2$):

$$\Phi : \mathbf{X} \rightarrow \Phi(\mathbf{X}) = \begin{bmatrix} \phi(\mathbf{x}_1) \\ \phi(\mathbf{x}_2) \end{bmatrix} \in \mathbb{R}^{2 \times 3}, \quad \text{kjer je} \quad \mathbf{X} = [\mathbf{x}_1 \ \mathbf{x}_2]^\top \in \mathbb{R}^{2 \times D}.$$

- Kernel matrika za primer polinomske preslikave $\mathbf{x} : \phi(\mathbf{x}) = (\mathbf{x}_1, \sqrt{2}\mathbf{x}_1\mathbf{x}_2, \mathbf{x}_2^2)$ je

$$K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)\phi(\mathbf{x}_j)^\top = \begin{bmatrix} x_{1i} \\ \sqrt{2}x_{1i}x_{2i} \\ x_{2i}^2 \end{bmatrix} \begin{bmatrix} x_{1j} \\ \sqrt{2}x_{1j}x_{2j} \\ x_{2j}^2 \end{bmatrix}^\top = x_{1i}^2x_{1j}^2 + 2x_{1i}x_{1j}x_{2i}x_{2j} + x_{2i}^2x_{2j}^2.$$

oziroma:

$$K(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i^\top \mathbf{x}_j)^2 = (x_{1i}x_{1j} + x_{2i}x_{2j})^2.$$

- To je primer polinomskega jedra: $K(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i^\top \mathbf{x}_j + \mathbf{r})^d$.
- Izračunali smo matriko jedra K ne da bi poznali naravo ϕ .

Kako napovemo nove vrednosti?

- Napovedujemo vrednosti y za nove podatke \mathbf{x} .
- Uporabimo $y = \mathbf{w}^\top \phi(\mathbf{x})$ z optimalnimi utežmi $\hat{\mathbf{w}} = \Phi^\top (\Phi \Phi^\top + \lambda' \mathbf{I})^{-1} \mathbf{y}$, kar nam da

$$\mathbf{y}_{\text{predict}} = \hat{\mathbf{w}}^\top \Phi = \mathbf{y} (\mathbf{K} + \lambda' \mathbf{I})^{-1} \Phi^\top \phi(\mathbf{x}) = \mathbf{y} (\mathbf{K} + \lambda' \mathbf{I})^{-1} \mathbf{k}_x,$$

kjer je \mathbf{k}_x (za napoved v točki \mathbf{x}):

$$\mathbf{k}_x = \Phi^\top \phi(\mathbf{x}) = \begin{bmatrix} K(\mathbf{x}_1, \mathbf{x}) \\ K(\mathbf{x}_2, \mathbf{x}) \\ \vdots \\ K(\mathbf{x}_N, \mathbf{x}) \end{bmatrix} = \begin{bmatrix} \phi(\mathbf{x}_1)^\top \phi(\mathbf{x}) \\ \phi(\mathbf{x}_2)^\top \phi(\mathbf{x}) \\ \vdots \\ \phi(\mathbf{x}_N)^\top \phi(\mathbf{x}) \end{bmatrix} \in \mathbb{R}^{N \times N}$$

neodvisen od ϕ .

Gaussian process regression (GPR)

Gaussovski procesi

- Gaussian process is a collection of random variables, any finite number of which have a joint Gaussian distribution.
- Gaussovski proces $f(\mathbf{x}) \sim GP(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}'))$ je popolnoma določen z njegovim povprečnjem in kovariančno funkcijo:

$$m(\mathbf{x}) = \mathbb{E}[f(\mathbf{x})], \quad k(\mathbf{x}, \mathbf{x}') = \mathbb{E}[(f(\mathbf{x}) - m(\mathbf{x}))(f(\mathbf{x}') - m(\mathbf{x}'))].$$

- Bayesovska linerna regresija $f(\mathbf{x}) = \phi(\mathbf{x})^\top \mathbf{w} + \epsilon$ z apriorno porazdelitvijo $\mathbf{w} \sim \mathcal{N}(0, \Sigma)$:

$$\mathbb{E}[f(\mathbf{x})] = 0 \quad \text{in} \quad \text{cov}(f(\mathbf{x}), f(\mathbf{x}')) = k(\mathbf{x}, \mathbf{x}') = \mathbb{E}[f(\mathbf{x})f(\mathbf{x}')] = \phi(\mathbf{x})^\top \Sigma \phi(\mathbf{x}').$$

- Primer jedra je **Radial Basis Function (RBF)** in je kvadrat eksponenta:

$$k(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{1}{2}|\mathbf{x} - \mathbf{x}'|^2\right),$$

ki je funkcija vhodnih in izhodnih (napovedanih) podatkovnih točk.

Kako napovemo nove vrednosti?

- Napovedana porazdelitev je v \mathbf{x}_* :

$$\mathbf{y}_* \sim \mathcal{N}(K(\mathbf{X}_*, \mathbf{X})[K(\mathbf{X}, \mathbf{X}) + \sigma^2 \mathbf{I}]^{-1} \mathbf{y}, \\ K(\mathbf{X}_*, \mathbf{X}_*) - K(\mathbf{X}_*, \mathbf{X})[K(\mathbf{X}, \mathbf{X}) + \sigma^2 \mathbf{I}]^{-1} K(\mathbf{X}, \mathbf{X}_*)) .$$

- Napoved je tako enaka

$$\mathbf{f}_* = K(\mathbf{X}_*, \mathbf{X})[K(\mathbf{X}, \mathbf{X}) + \sigma^2 \mathbf{I}]^{-1} \mathbf{y} ,$$

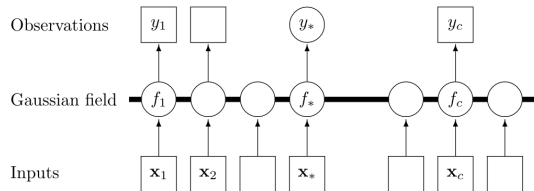
s pripadajočo napako

$$\text{cov}(\mathbf{f}_*) = K(\mathbf{X}_*, \mathbf{X}_*) - K(\mathbf{X}_*, \mathbf{X})[K(\mathbf{X}, \mathbf{X}) + \sigma^2 \mathbf{I}]^{-1} K(\mathbf{X}, \mathbf{X}_*) .$$

- Za kovariančno funkcijo (kernel) vzamemo RBF s **hiperparametri**:

$$k(\mathbf{x}, \mathbf{x}') = \sigma_f^2 \exp \left(-\frac{1}{2} \frac{|\mathbf{x} - \mathbf{x}'|^2}{\ell^2} \right) + \sigma^2 \mathbf{I} .$$

- Pomagaš si lahko s skeniranjem čez mrežo možnih vrednosti: [GridSearchCV](#).



 [Install](#) [User Guide](#) [API](#) [Examples](#) [Community](#) [More](#)

scikit-learn

Machine Learning in Python

[Getting Started](#) [Release Highlights for 1.3](#) [GitHub](#)

- Simple and efficient tools for predictive data analysis
- Accessible to everybody, and reusable in various contexts
- Built on NumPy, SciPy, and matplotlib
- Open source, commercially usable - BSD license

Classification

Identifying which category an object belongs to.

Applications: Spam detection, image recognition.

Algorithms: Gradient boosting, nearest neighbors, random forest, logistic regression, and more...



Examples

Regression

Predicting a continuous-valued attribute associated with an object.

Applications: Drug response, Stock prices.

Algorithms: Gradient boosting, nearest neighbors, random forest, ridge, and more...



Examples

Clustering

Automatic grouping of similar objects into sets.

Applications: Customer segmentation, Grouping experiment outcomes

Algorithms: k-Means, DBSCAN, hierarchical clustering, and more...



Examples

Dimensionality reduction

Reducing the number of random variables to consider.

Applications: Visualization, Increased efficiency

Algorithms: PCA, feature selection, non-negative matrix factorization, and more...



Examples

Model selection

Comparing, validating and choosing parameters and models.

Applications: Improved accuracy via parameter tuning

Algorithms: grid search, cross validation, metrics, and more...



Examples

Preprocessing

Feature extraction and normalization.

Applications: Transforming input data such as text for use with machine learning algorithms.

Algorithms: preprocessing, feature extraction, and more...



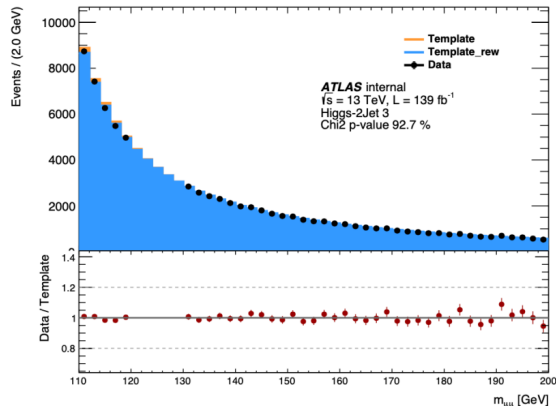
Examples

Reševanje naloge

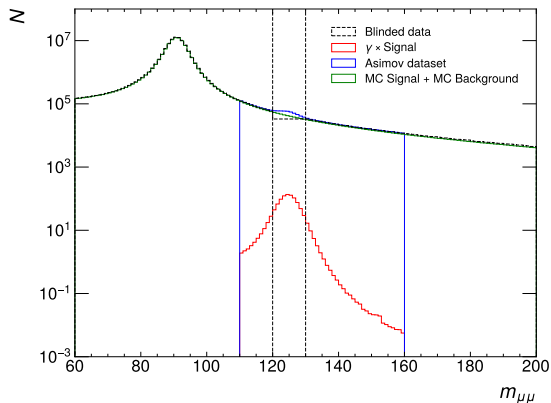
Blinding

- Simulacija ozadja ni vedno najboljša.
- Za oceno ozadja vzamemo izmerjene podatke, pri tem je potrebno izključiti območje, kjer pričakujemo signal \Rightarrow **blinding**.
- Izključimo npr. interval $[120, 130]$ GeV, kjer pričakujemo signal.
- Naredimo Numpy masko:

```
1 mask = (bin_centers < 119) | (bin_centers > 131)
2 bin_centers_masked = bin_centers[mask]
3 bin_values_masked = bin_values[mask]
4 bin_errors_masked = bin_errors[mask]
5
```
- Na tem ozadju (“background from data”) naredimo fit in dobimo $b(x_k)$.



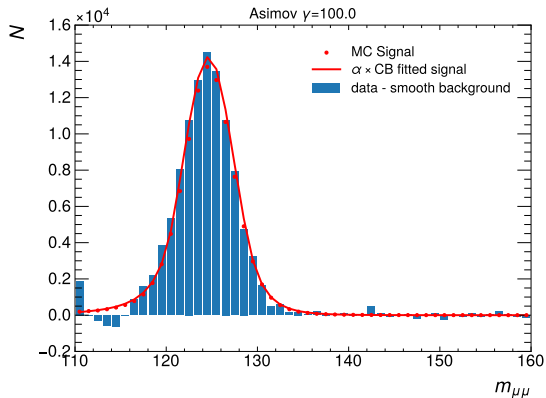
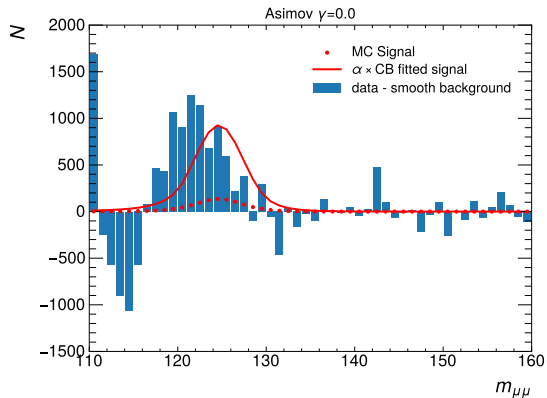
Asimov podatkovni set



- Da dobimo signal $y(x_k)$, odštejemo podatke od fitanega ozadja:
 $y(x_k) = d(x_k) - b(x_k)$.
- Na ta signal $y(x_k)$ prilagodimo funkcijo $\alpha \cdot \text{CB}$ (proste parametre smo dobili že na simulaciji signala $s(x_k)$).
- Izmerjenega signala je v podatkih zelo malo \Rightarrow postopek preizkusimo z napihnjenim signalom \Rightarrow **Asimov dataset**.
- Podatkom prištejemo MC signal:

$$d_A(x_k) = d(x_k) + \gamma \cdot s(x_k) .$$

Fitanje na odšteto ozadje



Predprocesiranje podatkov za ML algoritme

- ML algoritmi ne delujejo dobro, če so vhodne številske vrednosti na različnih skalah.
- Pred učenjem algoritmov vedno naredimo normalizacijo podatkov (*feature scaling*).
- Najbolj pogosti sta:

1. min-max normalizacija:

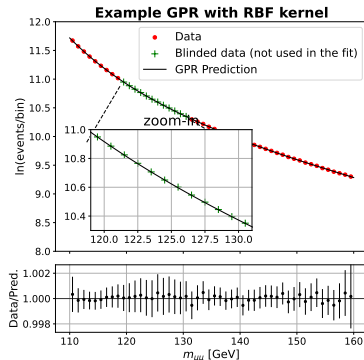
$$x_{\text{norm}} = \frac{x - x_{\min}}{x_{\max} - x_{\min}},$$

2. standardizacija:

$$x_{\text{norm}} = \frac{x - \mu}{\sigma}.$$

- Transformacije morajo imeti tudi inverz!

- Več primerov v [scikit-learn](#).
- Razmisli kaj je z napakami, pomagaš si lahko s knjižnico [uncertainties](#).



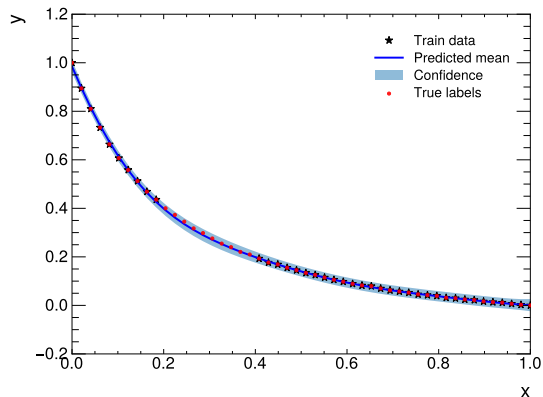
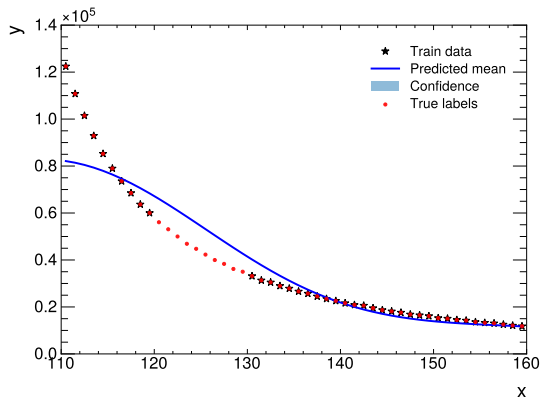
Uporaba GPR

- Uvoziti je potrebno relevantne knjižnice in izbrati poljubna jedra - kernele.
- Na voljo je več različnih jedrnih funkcij - lahko uporabiš predpripravljene (npr. Gibbs), lahko pa jih tudi kombiniraš.
- Vsako jedro ima svoj specifičen parameter ter spodnjo in zgornjo mejo.
- Primer:

```
1 # define kernel: ConstantKernel * RBF:
2 const_kernel = ConstantKernel(const0, constant_value_bounds=(const_low, const_hi))
3 rbf_kernel = RBF(RBF0, length_scale_bounds=(RBF_low, RBF_high))
4 kernel = const_kernel * rbf_kernel
5
6 # transform x data into 2d vector
7 X = np.atleast_2d(bin_centers_masked).T # true datapoints
8 X_to_predict = np.linspace(110, 160, 1000).T # what to predict
9 y = bin_values_masked # labels / targets
10
11 # define Regressor and add errors as alpha
12 gp = GaussianProcessRegressor(kernel=kernel, alpha=bin_errors_masked**2)
13
14 # fit on X with values y
15 gp.fit(X, y)
16
17 # predict
18 y_pred, sigma = gp.predict(X_to_predict, return_std=True)
19
```


Fitanje na blinded podatke

- Enostaven primer z uporabo RBF jedra (pazi skaliranje količin!).



GPR: pripravljena (CERN) koda

- Cilj: preizkusi `GaussianProcessRegressor` za glajenje ozadja iz podatkov.
- Preberi dokumentacijo in preizkusi kodo, poskusi razumeti posamezne dele.
- Prepričaj se, da je dodajanje svojih kernelov v `sklearn` preprosto (RBF, Matern, ...).
- Uporabiš lahko tudi *Gibbsov kernel*, ki ga ni v `sklearn` knjižnici.
- Gibbsov kernel je v svojem bistvu RBF kernel, le da se `length_scale` lahko spreminja z linearno funkcijo:

$$L(x) = L_{\text{slope}}x + c .$$

- Pazi kako pripraviš podatke (logaritem, minmax, ...) in kako se pri tem spremenijo napake.
- **Dodatno** si lahko pogledaš še knjižnico `GPYtorch` ter RBF jedro z učljivimi (*learnable*) hiperparametri ℓ in σ^2 .

Uporaba KRR

- Cilj: prilagodi algoritem za naše namene fitanja ozadja.

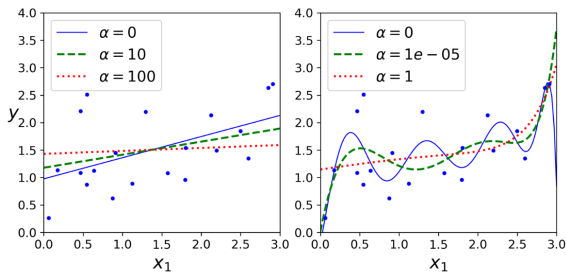


Figure 4-17. A linear model (left) and a polynomial model (right), both with various levels of Ridge regularization

- Kako se tukaj upoštevajo napake meritev? fit metoda dodatno sprejme argument `sample_weight` (velika napaka pomeni majhno utež).

- Uporaba ridge regresije je enostavna (glej poglavje 4):

As with Linear Regression, we can perform Ridge Regression either by computing a closed-form equation or by performing Gradient Descent. The pros and cons are the same. Equation 4-9 shows the closed-form solution, where \mathbf{A} is the $(n+1) \times (n+1)$ identity matrix,¹¹ except with a 0 in the top-left cell, corresponding to the bias term.

Equation 4-9. Ridge Regression closed-form solution

$$\hat{\theta} = (\mathbf{X}^T \mathbf{X} + \alpha \mathbf{A})^{-1} \mathbf{X}^T \mathbf{y}$$

Here is how to perform Ridge Regression with Scikit-Learn using a closed-form solution (a variant of Equation 4-9 that uses a matrix factorization technique by André-Louis Cholesky):

```
>>> from sklearn.linear_model import Ridge
>>> ridge_reg = Ridge(alpha=1, solver="cholesky")
>>> ridge_reg.fit(X, y)
>>> ridge_reg.predict([[1.5]])
array([[1.55071465]])
```

- Kernel je dodan v razredu `KernelRidge`.

Zaključek 1. naloge

Povzetek 2. dela 1. naloge

- Prejšnji teden ste fitali s preprostim `curve_fit`, ta teden je cilj uporabiti metode strojnega učenja, predvsem GPR.
- Poskusi tudi z uporabo KRR in še kakšne podobne metode (npr. SVM o kateri ste govorili na predavanjih).
- GPR najprej poizkusite na preprosti način → spoznaj se s knjižnico `scikit-learn`.
- GPR - CERN skripta → preberi kodo, uporabi Gibbsov kernel, pazi na napake.
- KRR preberi dokumentacijo → poskusi sam zgladiti ozadje.
- Preveri različna jedra, kaj je prednost GPR?
- Dokončaj nalogo → spoznal si različne metode fitanja, najdi Higgso!

Še enkrat po korakih

1. Pofitaj histogram simulacije signala (simulated Higgs signal):
 - Uporabi nastavek za CB funkcijo.
 - Dobiš funkcijo $s(x_k)$ s parametri za MC signal.
2. Pofitaj histogram simulacije ozadja (simulated background):
 - Uporabi različne možnosti in intervale, ni ti treba izključiti območja signala (ker ga tu ni).
 - Dobiš funkcijo $m(x_k)$.
3. Pofitaj histogram podatkov, da dobiš dobro oceno za ozadje (background from data):
 - Moraš pa izključiti interval, kjer je v podatkih signal \Rightarrow okno okrog $[120, 130]$ GeV.
 - Uporabi različne možnosti, podobno kot na simulaciji ozadja \Rightarrow razni funkcijski nastavki in ML metode.
 - Lahko vzameš tudi nastavek $q(x_k)m(x_k)$, kjer je $q(x_k)$ preprosta funkcija za fit in $m(x_k)$ pofitana na simulaciji ozadja.
 - Dobiš funkcijo $b(x_k)$.
4. Odštej od podatkov čim bolj zglajeno ozadje.
 - Če so podatki v binih $d_k(x)$, potem narediš $y(x_k) = d(x_k) - b(x_k)$.
5. Na kar ostane (torej $y(x_k)$) nalepi parametrizacijo signala.
 - Na funkcijo $s(x_k)$ dodaj parameter normalizacije/skaliranja α , da dobiš $\alpha \cdot s(x_k)$.
 - Naredi fit na $y(x_k)$, da dobiš najboljšo oceno $\hat{\alpha}$ in njeno napako $\sigma_{\hat{\alpha}}$.
6. Ponovi postopek za različne vrednosti napihnjenega signala γ : $d_A(x_k) = d(x_k) + \gamma \cdot s(x_k)$.

Extra

GPR: Bayesovska linerna regresija

- Imamo podatkovno množico (set) $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$, kjer je $\mathbf{x}_i \in \mathbb{R}^D$ in $y_i \in \mathbb{R}$.
- Linearna regresija z Gaussovskim šumom:

$$f(\mathbf{x}) = \mathbf{x}^\top \mathbf{w}, \quad y = f(\mathbf{x}) + \varepsilon \quad \text{kjer je} \quad \varepsilon \sim \mathcal{N}(0, \sigma^2).$$

- Izmerjene vrednosti $f(\mathbf{x})$ se razlikujejo od pravih vrednosti y zaradi šuma ε .
- Uvedemo bazne funkcije $\phi(\mathbf{x}) : \mathbb{R}^D \rightarrow \mathbb{R}^M$, da dobimo bolj ekspresiven model:

$$f(\mathbf{x}) = \phi(\mathbf{x})^\top \mathbf{w} + \epsilon \quad \text{kompaktni matrični zapis.}$$

- \Rightarrow uporabimo Bayesovsko linearno regresijo.

GPR: Gaussovski prior

- Kako pridemo do parametrov θ (oziroma uteži \mathbf{w})?
- Za primer GPR:

$$p(\mathbf{w}|\Phi, \mathbf{y}) = \frac{p(\mathbf{y}|\Phi, \mathbf{w})p(\mathbf{w})}{p(\mathbf{y}|\Phi)} \propto p(\mathbf{y}|\Phi, \mathbf{w})p(\mathbf{w}).$$

- Prior je Gaussovka (opiše naše neznanje o optimalnih utežeh):

$$\mathbf{w} \sim \mathcal{N}(0, \Sigma),$$

kjer je Σ kovariančna matrika.

- Likelihood za meritve pri danih parametrih je Gaussovski:

$$\begin{aligned} p(\mathbf{y}|\Phi, \mathbf{w}) &= \prod_{i=1}^N p(y_i|\phi(\mathbf{x}_i), \mathbf{w}) = \prod_{i=1}^N \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(y_i - \phi(\mathbf{x}_i)^\top \mathbf{w})^2}{2\sigma^2}\right) \\ &= \frac{1}{(2\pi)^{N/2}\sigma^N} \exp\left(-\frac{1}{2\sigma^2} \sum_{i=1}^N (y_i - \phi(\mathbf{x}_i)^\top \mathbf{w})^2\right) = \mathcal{N}(\Phi^\top \mathbf{w}, \sigma^2 \mathbf{I}). \end{aligned}$$

GPR: Ocenjevanje uteži \mathbf{w} z MAP

- Zapišemo samo člene, ki so odvisni od parametrov (uteži \mathbf{w}):

$$\begin{aligned} p(\mathbf{w}|\Phi, \mathbf{y}) &\propto p(\mathbf{y}|\Phi, \mathbf{w})p(\mathbf{w}) \propto \exp\left(-\frac{1}{2\sigma^2}(\mathbf{y} - \Phi^\top \mathbf{w})^\top (\mathbf{y} - \Phi^\top \mathbf{w})\right) \exp\left(-\frac{1}{2}\mathbf{w}^\top \Sigma^{-1} \mathbf{w}\right) \\ &\propto \exp\left(-\frac{1}{2}(\mathbf{w} - \mathbf{w}')^\top \left(\frac{1}{\sigma^2}\Phi\Phi^\top + \Sigma^{-1}\right) (\mathbf{w} - \mathbf{w}')\right) \sim \mathcal{N}(\mathbf{w}', \mathbf{A}^{-1}), \end{aligned}$$

kjer uvedemo $\mathbf{w}' = \frac{1}{\sigma^2}\mathbf{A}^{-1}\Phi\mathbf{y}$ in $\mathbf{A} = \frac{1}{\sigma^2}\Phi\Phi^\top + \Sigma^{-1}$.

- Povprečje \mathbf{w}' aposteriorne porazdelitve je enako njenemu maksimumu (MAP ocena za \mathbf{w}).

GPR: Fitanje modela

- Uporabimo MAP metodo, kar pomeni uporabo aposteriorne porazdelitve za določitev naše ocene parametrov \mathbf{w} .
- Zapišemo (vzamemo negtiven logaritem):

$$\hat{\mathbf{w}}_{\text{MAP}} = \underset{\mathbf{w}}{\operatorname{argmax}} [-\log p(\mathbf{w}|\Phi, \mathbf{y})] = \underset{\mathbf{w}}{\operatorname{argmax}} [-\log p(\mathbf{y}|\Phi, \mathbf{w}) - \log p(\mathbf{w})]$$

\Rightarrow v optimalni oceni smo dobili še dodaten člen $\log p(\mathbf{w})$ (likelihood je utežen).

- Za GPR dobimo:

$$\hat{\mathbf{w}}_{\text{MAP}} = \underset{\mathbf{w}}{\operatorname{argmax}} \left[-\log(2\pi)^{N/2} - \log \sqrt{\det(\mathbf{A}^{-1})} + \frac{1}{2}(\mathbf{w} - \mathbf{w}')^{\top} \mathbf{A}(\mathbf{w} - \mathbf{w}') \right] .$$

- Problem fitanja se prevede na maksimizacijo zgornje funkcije (aposterior).
- Izračunati moramo $\nabla_{\mathbf{w}} \mathcal{L}(\mathbf{w}) \Rightarrow$ za MAP in Gaussovko je to kar \mathbf{w}' (analitična rešitev).

GPR: Kako napovemo nove vrednosti?

- Povprečimo po vseh možnih vrednostih parametrov uteženih z njihovimi aposteriornimi verjetnostmi.
- Napovedano porazdelitev za \mathbf{y}_* v točki \mathbf{x}_* dobimo s povprečenjem:

$$p(f_*|\phi_*, \Phi, \mathbf{y}) = \int p(f_*|\phi_*, \mathbf{w})p(\mathbf{w}|\Phi, \mathbf{y}) d\mathbf{w} = \mathcal{N}\left(\frac{1}{\sigma^2}\phi_*^\top \mathbf{A}^{-1}\Phi\mathbf{y}, \phi_*^\top \mathbf{A}^{-1}\phi_*\right).$$

- Napovedna porazdelitev je spet Gaussovka s povprečjem aposteriorne porazdelitve pomnoženim z vhodnimi podatki.
- Napovedana varianca ima kvadratično obliko, ki je odvisna od vhodnih podatkov.
- Zgornje lahko zapišemo na naslednji način:

$$p(f_*|\phi_*, \Phi, \mathbf{y}) = \mathcal{N}\left(\phi_*^\top \Sigma \Phi (K + \sigma^2 \mathbf{I})^{-1} \mathbf{y}, \phi_*^\top \Sigma \phi_* - \phi_*^\top \Sigma \Phi (K + \sigma^2 \mathbf{I})^{-1} \Phi^\top \Sigma \phi_*\right),$$

kjer smo definirali $K = \Phi^\top \Sigma \Phi$.

GPR: Kaj pa bazne funkcije ϕ ?

- Brez ϕ imamo linearni model \rightarrow s pomočjo baznih funkcij ϕ projeciramo vhodne podatke \mathbf{x} v višje dimenzionalni prostor lastnosti (*feature space*) in uporabimo linearni model tam.
- Polinomska regresija: $\mathbf{x} : \phi(\mathbf{x}) = (1, \mathbf{x}, \mathbf{x}^2, \mathbf{x}^3, \dots)^\top$, kjer so ϕ_i fiksne funkcije.
- Prostor lastnosti je vedno v obliki: $\phi_*^\top \Sigma \phi_*$, $\phi_*^\top \Sigma \Phi$ ali $\Phi^\top \Sigma \phi_*$.
- Zapišemo v obliki:

$$k(\mathbf{x}, \mathbf{x}') = \phi(\mathbf{x})^\top \Sigma \phi(\mathbf{x}') \quad (\text{kovariančna funkcija ali kernel}),$$

kjer je \mathbf{x} vhodni podatek, \mathbf{x}' pa podatek, ki ga napovedujemo.

- Σ je pozitivno definitna matrika in jo lahko zapišemo kot $(\Sigma^{1/2})^2$.
- SVD razcep te matrike je $\Sigma = UDU^\top$, oziroma $\Sigma^{1/2} = UD^{1/2}U^\top$.
- S tem dobimo zapis s skalarnim produktom:

$$\psi(\mathbf{x}) = \Sigma^{1/2} \phi(\mathbf{x}) \Rightarrow k(\mathbf{x}, \mathbf{x}') = \psi(\mathbf{x}) \cdot \psi(\mathbf{x}').$$

GPR: Gaussovski procesi

- Gaussian process is a collection of random variables, any finite number of which have a joint Gaussian distribution.
- Gaussovski proces $f(\mathbf{x}) \sim GP(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}'))$ je popolnoma določen z njegovim povprečjem in kovariančno funkcijo:

$$m(\mathbf{x}) = \mathbb{E}[f(\mathbf{x})], \quad k(\mathbf{x}, \mathbf{x}') = \mathbb{E}[(f(\mathbf{x}) - m(\mathbf{x}))(f(\mathbf{x}') - m(\mathbf{x}'))].$$

- Bayesovska linerna regresija $f(\mathbf{x}) = \phi(\mathbf{x})^\top \mathbf{w}$ z apriorno porazdelitvijo $\mathbf{w} \sim \mathcal{N}(0, \Sigma)$:

$$\mathbb{E}[f(\mathbf{x})] = 0 \quad \text{in} \quad \text{cov}(f(\mathbf{x}), f(\mathbf{x}')) = k(\mathbf{x}, \mathbf{x}') = \mathbb{E}[f(\mathbf{x})f(\mathbf{x}')] = \phi(\mathbf{x})^\top \Sigma \phi(\mathbf{x}').$$

- Primer jedra je Radial Basis Function (RBF) in je kvadrat eksponenta:

$$k(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{1}{2}|\mathbf{x} - \mathbf{x}'|^2\right),$$

ki je funkcija vhodnih in izhodnih (napovedanih) podatkovnih točk.

GPR: Kako napovemo nove vrednosti?

- Napovedana porazdelitev je v \mathbf{x}_* :

$$\mathbf{y}_* \sim \mathcal{N}(K(\mathbf{X}_*, \mathbf{X})[K(\mathbf{X}, \mathbf{X}) + \sigma^2 \mathbf{I}]^{-1} \mathbf{y}, \\ K(\mathbf{X}_*, \mathbf{X}_*) - K(\mathbf{X}_*, \mathbf{X})[K(\mathbf{X}, \mathbf{X}) + \sigma^2 \mathbf{I}]^{-1} K(\mathbf{X}, \mathbf{X}_*)) .$$

- Napoved je tako enaka

$$\mathbf{f}_* = K(\mathbf{X}_*, \mathbf{X})[K(\mathbf{X}, \mathbf{X}) + \sigma^2 \mathbf{I}]^{-1} \mathbf{y} ,$$

s pripadajočo napako

$$\text{cov}(\mathbf{f}_*) = K(\mathbf{X}_*, \mathbf{X}_*) - K(\mathbf{X}_*, \mathbf{X})[K(\mathbf{X}, \mathbf{X}) + \sigma^2 \mathbf{I}]^{-1} K(\mathbf{X}, \mathbf{X}_*) .$$

- Za kovariančno funkcijo (kernel) vzamemo RBF s hiperparametri:

$$k(\mathbf{x}, \mathbf{x}') = \sigma_f^2 \exp \left(-\frac{1}{2} \frac{|\mathbf{x} - \mathbf{x}'|^2}{\ell^2} \right) + \sigma^2 \mathbf{I} .$$

