Principles of Machine Learning: Exercise 5

Alina Pollehn (3197257), Julian Litz (3362592), Manuel Hinz (3334548)
Felix Göhde (3336445), Felix Lehmann (3177181), Caspar Wiswesser (3221493)
Adrian Köring (3347785), Greta Günther (3326765), Linus Mallwitz (3327653)
Niklas Mueller-Goldingen (3363219), Jennifer Kroppen (2783393)

11.01.2024

Exercise 4.1: Overview

1. Goal: Fitting a polynomial to noisy data via polynomial regression
2. First step: Compute a feature map, where the ith column is composed of the powers $x_i^0, \ldots, x_i^d$
3. Second step: Compute the weights: $\hat{w} = [\Phi\Phi^\intercal]^{-1}\Phi y$ using a numerically stable inversion algorithm (i.e. using QR decomposition)
4. Third step: Compute the fitted model: $\hat{f}(x) = \varphi(x)^\intercal \hat{w}$

Results: Plots

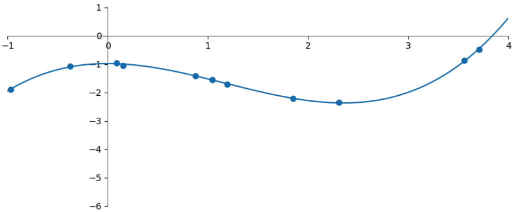


Figure: Polynomial fit for $d = 3$

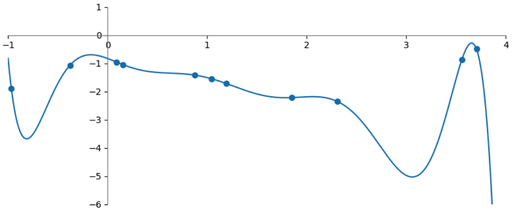


Figure: Polynomial fit for $d = 9$

Results: Discussion

- While $d = 9$ naturally gives a much better MSE, we not only know (because our data was generated by adding noise to a third degree polynomial), but can clearly see that we are overfitting!
- Therefore we would prefer our cubic fit, even if it isn't a perfect

Adding regularization

We now compute the solution to regularized least squares in the same way: $\hat{w} = [\Phi\Phi^\intercal + \lambda I]^{-1}\Phi y$ for different lambda:

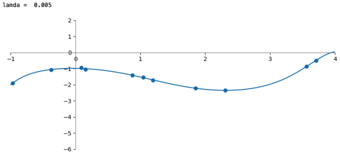


Figure: Polynomial fit for $\lambda = 0.005$

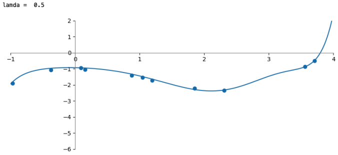


Figure: Polynomial fit for $\lambda = 0.5$

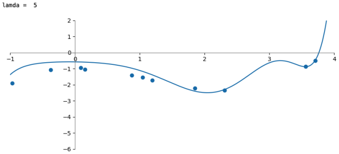


Figure: Polynomial fit for $\lambda = 5$

Regularized least squares

- Great results for smaller $\lambda = 0.5, 0.005$
- $\lambda = 5$ gives a worse result, which intuitively makes sense, because we are adding a larger value, decreasing the impact of our gram matrix before inverting!

Setting

- Lecture 07 showed that the dual least squares solution is given by $\hat{w} = \Phi[\Phi\Phi^\intercal]^{-1}y$
- After regularization this becomes $\hat{w} = \Phi[\Phi\Phi^\intercal + \lambda I]^{-1}y$
- We kernelize the expression $\hat{f}(x) = \phi^\intercal(x)\Phi[\Phi\Phi^\intercal + \lambda I]^{-1}y$ to get

$$\hat{f}(x) = k(x)^\intercal[K + \lambda I]^{-1}y$$

- Good choices for the model parameters where given: $\lambda = 0.5, b = 1, d = 3$

Results

- b tranlates the result,
- d is the degree of the polynomial, therefore influencing the shape of our model in the usual ways
- This, similarly to the previous task, leads to better results for $d = 9$, again because of the regularization

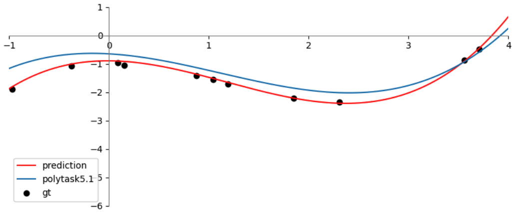


Figure: Polynomial fit for the given parameters

Least squares SVMs for regression

- Adding to the lecture we can use SVMs for regression as well
- We want to build a least squares SVM regression model

$$\hat{f}(x) = \varphi(x)^\intercal \Phi \hat{\lambda} + \hat{b}$$

- Where

$$\begin{bmatrix} \hat{\lambda} \\ \hat{b} \end{bmatrix} = \begin{bmatrix} \Phi^\intercal \Phi + \frac{1}{C} I & 1 \\ 1^\intercal & 0 \end{bmatrix} \begin{bmatrix} y \\ 0 \end{bmatrix}$$

- Which can easily be kernelized:

$$\begin{bmatrix} \hat{\lambda} \\ \hat{b} \end{bmatrix} = \begin{bmatrix} K + \frac{1}{C} I & 1 \\ 1^\intercal & 0 \end{bmatrix} \begin{bmatrix} y \\ 0 \end{bmatrix}$$

and

$$\hat{f}(x) = k(x)^\intercal \hat{\lambda} + \hat{b}$$

- Good results for a wider range of parameters (keeping the d fixed)
- Results differ more when changing the degree d

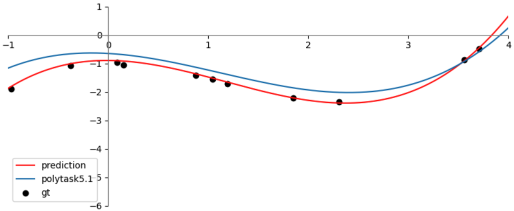


Figure: Polynomial fit for the given parameters

Other degrees

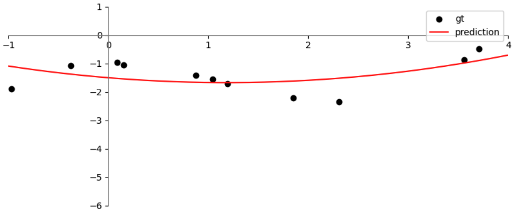


Figure: Polynomial fit for the given parameters $d = 2$

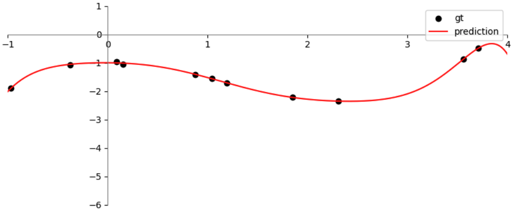


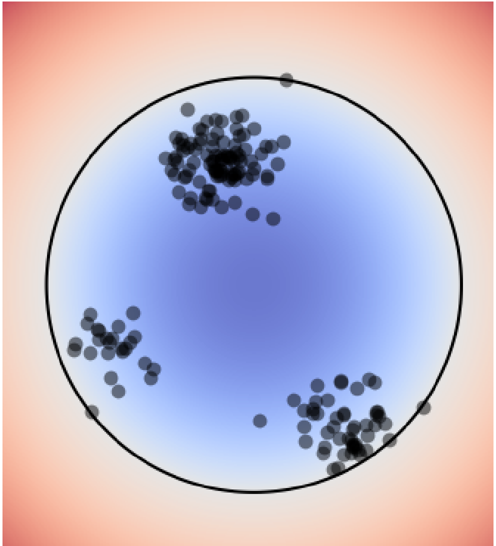
Figure: Polynomial fit for the given parameters $d = 9$

Minimum enclosing balls

- We already saw how to compute the minimal enclosing ball for a given dataset in lecture 08:
  - Using Frank-Wolfe solve
$$\underset{\mu}{\operatorname{argmin}}\, \mu^\intercal X^\intercal X \mu - \mu^\intercal z$$
  - where X denotes the dataset $X = [x_1, \ldots, x_n] \in \mathbb{R}^{m \times n}$ and $z = \operatorname{diag}[X^\intercal X]$
  - under the constraints $1^\intercal \mu = 1$ and $\mu \geq 0$
- given $\hat{\mu}$ we can than either compute the radius and the center of the ball
$$\hat{c} = X\hat{\mu} \text{ and } \hat{r} = \sqrt{\hat{\mu}^\intercal z - \hat{\mu}^\intercal X^\intercal X \hat{\mu}}$$
- which leads to a function
$$\chi_B(x) = \|x - \hat{c}\|^2 - \hat{r}^2$$
which is negative for $x \notin B \cup \partial B$!
- we can also rewrite
$$\chi_B(x) = x^\intercal x - 2x^\intercal X\hat{\mu} + \hat{\mu}^\intercal X^\intercal X\hat{\mu} - \hat{\mu}^\intercal z + \hat{\mu}^\intercal X^\intercal X\hat{\mu}$$

Result and Frank-Wolfe-Implementation

```
def fwDualMEB(matX,vecZ,T=100):
    m, n = matX.shape
    vecM = np.ones(n)/n
    for t in range(T):
        beta = 2 / (t+2)
        vecG = 2 * matX.T @ \
            matX @ vecM - vecZ
        imin = np.argmin(vecG)
        vecM *= (1-beta)
        vecM[imin] += beta
    return vecM
```



Kernel minimum enclosing balls

- Using our second formulation

$$\chi_B(x) = x^\intercal x - 2x^\intercal X\hat{\mu} + \hat{\mu}^\intercal X^\intercal X\hat{\mu} - \hat{\mu}^\intercal z + \hat{\mu}^\intercal X^\intercal X + \hat{\mu}$$

  we can kernalize everything:
- We get:

$$\chi_B(x) = K(x,x) - 2\kappa^\intercal\hat{\mu} - \hat{\mu}^\intercal k + 2\hat{\mu}^\intercal K\hat{\mu}$$

- where $K(x,x) = \exp(0) = 1 \in \mathbb{R}$ and $k = 1 \in \mathbb{R}^n$, because $k_j = K(x_j, x_j) = \exp(0) = 1$.
- Using a Gaussian kernel:

$$k(u,v) = \exp(-\frac{1}{2\sigma^2}\|u-v\|^2)$$

  and the following Frank-Wolfe-Algorithm:

Frank-Wolfe-Algorithm for kernel minimum enclosing balls

- Solving the minimization problem problem

$$\begin{aligned}
&\operatorname*{argmin}_{\mu} \mu^\intercal K \mu - \mu^\intercal k \\
= &\operatorname*{argmin}_{\mu} \mu^\intercal K \mu - \mu^\intercal 1
\end{aligned}$$

- under the constraints $1^\intercal \mu = 1$ and $\mu \geq 0$

```python
def fwDualMEB2(K,k, T=100):
    m, n = K.shape
    vecM = np.ones(n)/n
    for t in range(T):
        beta = 2 / (t+2)
        vecG = K @ vecM - k
        imin = np.argmin(vecG)
        vecM *= (1-beta)
        vecM[imin] += beta
    return vecM
```

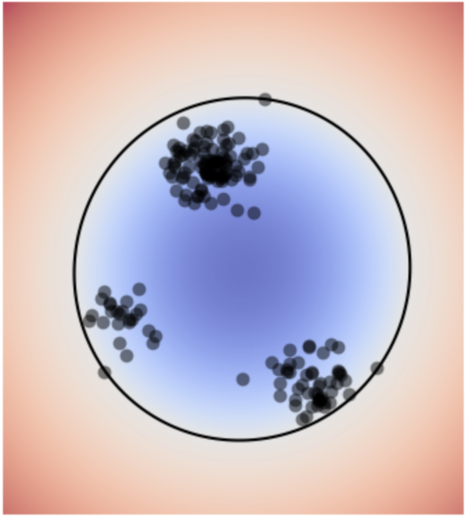


Figure: $\sigma = 4$

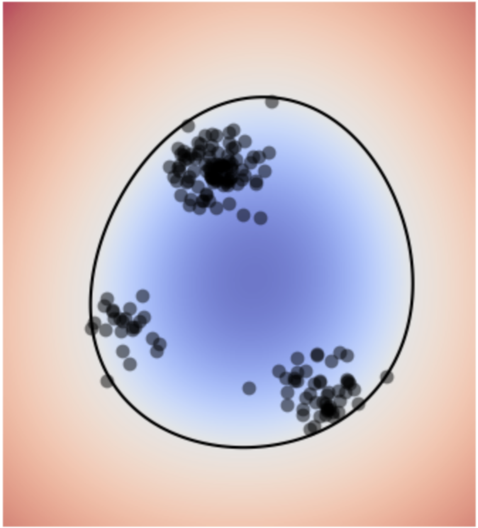


Figure: $\sigma = 2$

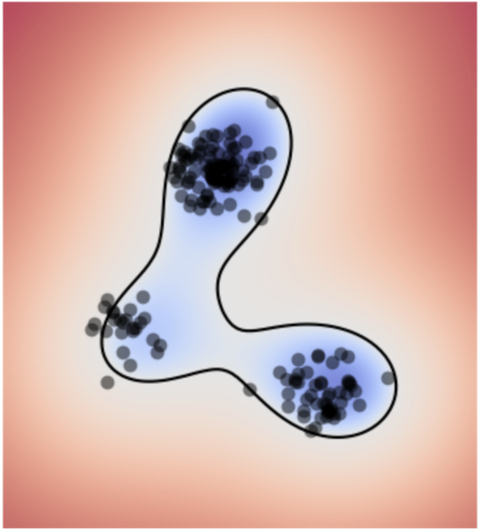


Figure: $\sigma = 1$

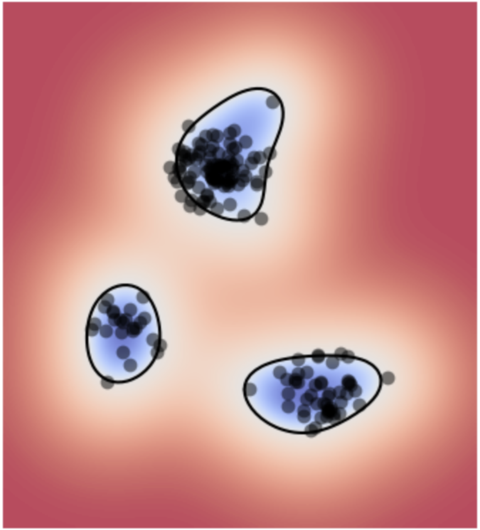


Figure: $\sigma = 0.5$