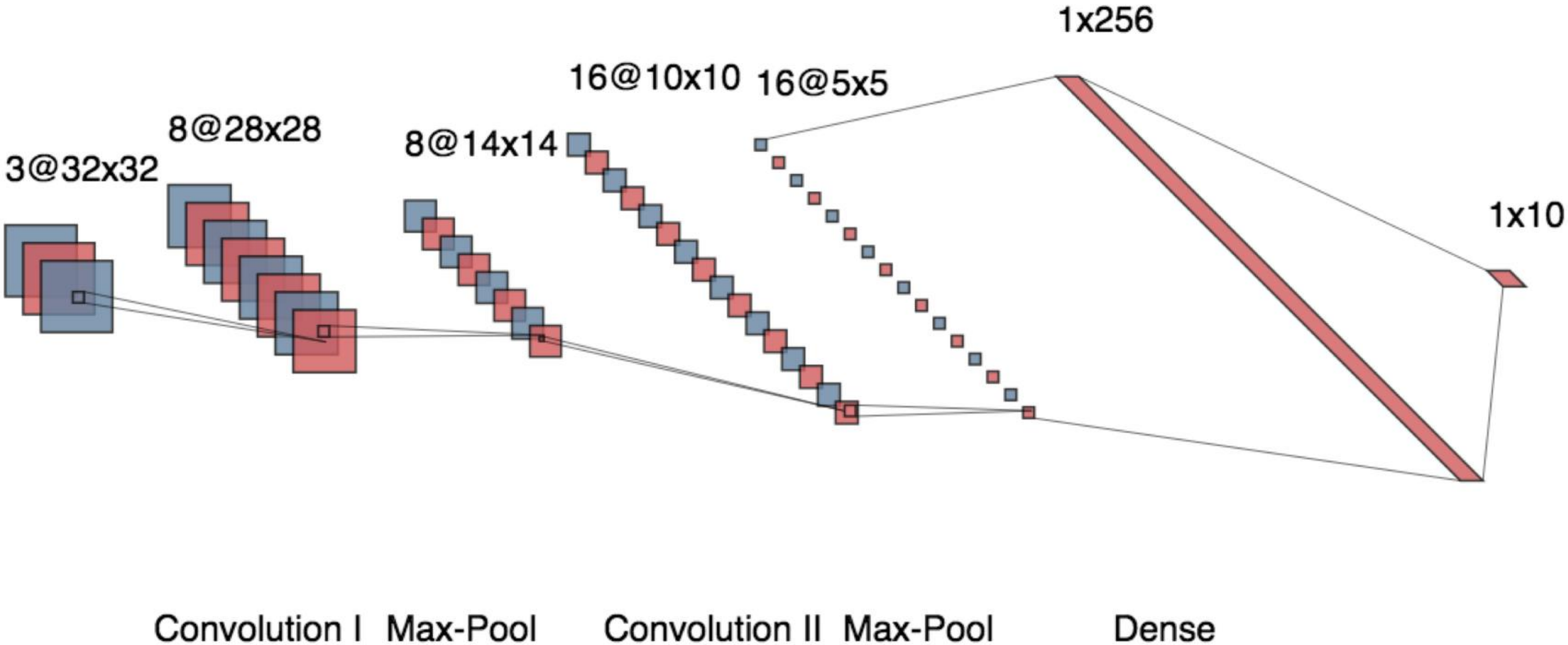
COMP90051
Statistical Machine Learning

Workshop Week 8

Xudong Han

https://github.com/HanXudong/COMP90051_Workshops

Calculate the number of parameters

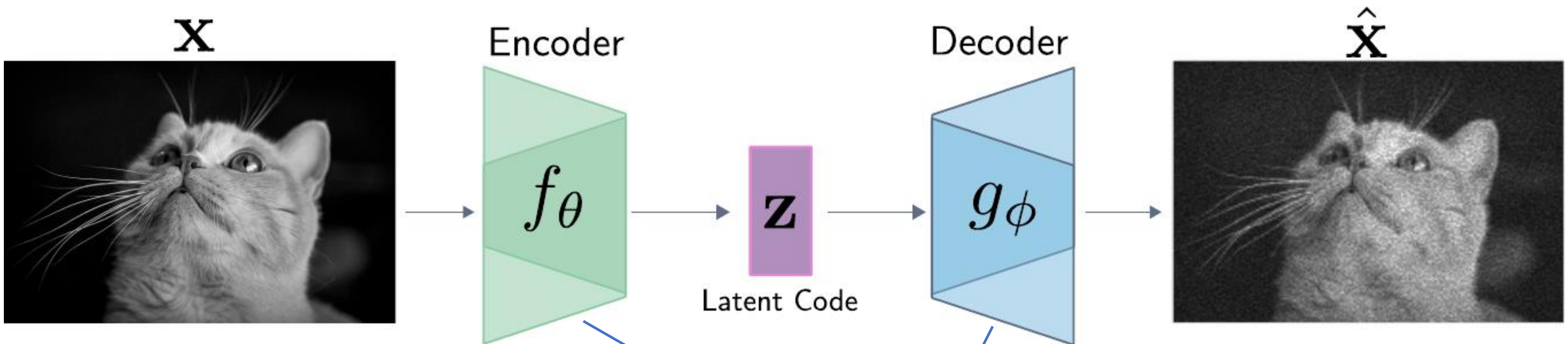


Convolution I: $3 \times 8 \times 5 \times 5 + 8$

https://pytorch.org/docs/stable/nn.html#convolution-layers

Dense I: $16 \times 5 \times 5 \times 256 + 256$

Autoencoders



$$\min_{f,g} \sum_k \|\mathbf{x}_k - g \circ f(\mathbf{x}_k)\|^2$$

To do

- SVM hyperparameters: we explore the effect of the penalty parameter and kernel.
- Primal vs. dual: we examine the computational efficiency of the primal and dual formulations in two different scenarios.
- Practice questions.

Kernel Exercises

- Mercer's Theorem
- Positive Semidefinite/ Positive Definite

  a symmetric $n \times n$ matrix M is said to be positive semidefinite if for any n nonzero dim vector z, the scalar $z^{\top} M z \geq 0$.
- Eigenvalue

  $$Mv = \lambda v$$

Eigenvalue

- $Mv = \lambda v$
- Suppose $M = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$
- How could we compute eigenvalues and eigenvectors?

Positive Semidefinite/Definite

- $[a \quad b]\begin{bmatrix}1 & 1\\ 1 & 1\end{bmatrix}\begin{bmatrix}a\\ b\end{bmatrix} = [a+b \quad a+b]\begin{bmatrix}a\\ b\end{bmatrix}$
  $= a^2 + ab + ab + b^2 = (a+b)^2$
- $[a \quad b]\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}\begin{bmatrix}a\\ b\end{bmatrix} = a^2 + b^2$
- For a kernel $k(x_i, x_j)$, the full Gram matrix

$$\begin{bmatrix} k(x_1, x_1) & \cdots & k(x_1, x_n) \\ \vdots & \ddots & \vdots \\ k(x_n, x_1) & \cdots & k(x_n, x_n) \end{bmatrix}$$

Mercer's Theorem – Positive Semidefinite

- Given any $C \in \mathbb{R}^n$
- $\begin{bmatrix} c_1 & \cdots & c_n \end{bmatrix} \begin{bmatrix} k(x_1, x_1) & \cdots & k(x_1, x_n) \\ \vdots & \ddots & \vdots \\ k(x_n, x_1) & \cdots & k(x_n, x_n) \end{bmatrix} \begin{bmatrix} c_1 \\ \vdots \\ c_n \end{bmatrix}$
- Since we assume that the kernel $k(x_i, x_j)$ is valid kernel

$$C^T K C = \sum_{i,j}^{n} c_i c_j k(x_i, x_j) \geq 0$$

$$k'(x_i, x_j) = \lambda k(x_i, x_j) \text{ for } \lambda > 0$$

- For a kernel $k'(x_i, x_j)$, the full Gram matrix

$$K' = \begin{bmatrix} \lambda k(x_1, x_1) & \cdots & \lambda k(x_1, x_n) \\ \vdots & \ddots & \vdots \\ \lambda k(x_n, x_1) & \cdots & \lambda k(x_n, x_n) \end{bmatrix}$$

- $C^T K' C = \lambda \sum_{i,j}^{n} c_i c_j k(x_i, x_j) \geq 0$
- $k'(x_i, x_j)$ is a valid kernel

Extension

- $k'(x_i, x_j) = k_\alpha(x_i, x_j)k_\beta(x_i, x_j)$ is a valid kernel
- can be used for question 2-b

- $k'(x_i, x_j) = \exp(k(x_i, x_j))$

- Taylor series

The Taylor series of a real or complex-valued function $f(x)$ that is infinitely differentiable at a number a is the power series

Kernel Exercises

$$f(x) = \sum_{n=0}^{\infty} \frac{f^{(n)}(a)}{n!}(x-a)^n$$

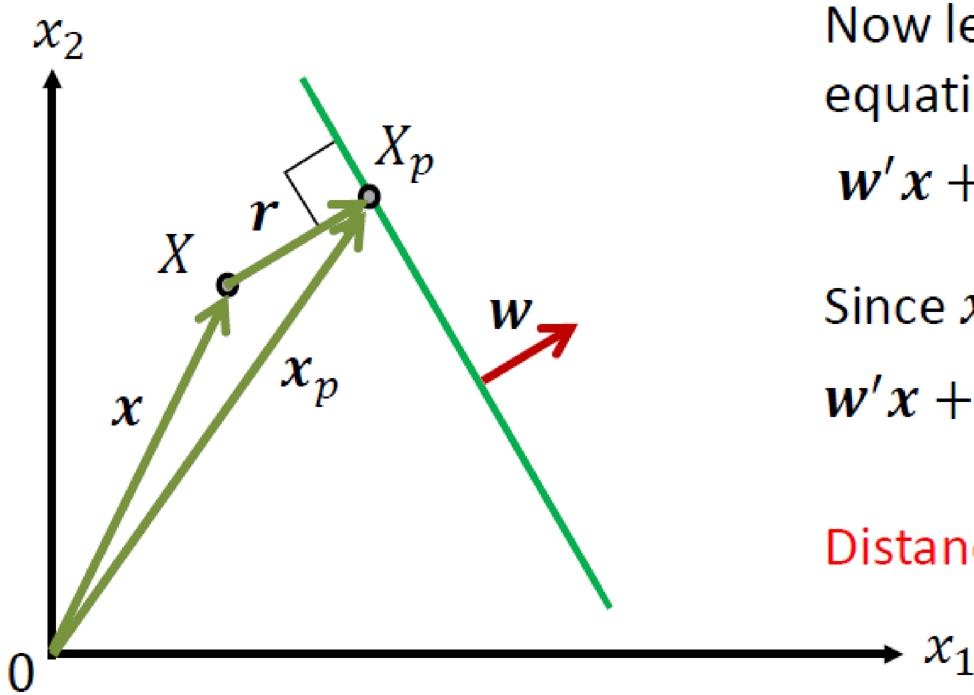
Given $f(x) = \exp(\mathrm{x})$ and $\mathrm{a} = 0$, by applying Taylor series:

$$f(x) = \sum_{n=0}^{\infty} \frac{\exp(0)}{n!}(x)^n$$

Thus, $\exp(k(x_i, x_j)) = \sum_{n=0}^{\infty} \frac{\left(k(x_i, x_j)\right)^n}{n!}$

SVM

- Vectors $\boldsymbol{r}$ and $\boldsymbol{w}$ are parallel, but not generally of the same length. Trivially, $\boldsymbol{r} = \boldsymbol{w}\frac{\|\boldsymbol{r}\|}{\|\boldsymbol{w}\|}$
- Next, points X and X_p can be viewed as vectors $\boldsymbol{x}$ and $\boldsymbol{x}_p$. By vector addition, we have that $\boldsymbol{x} + \boldsymbol{r} = \boldsymbol{x}_p$ or $\boldsymbol{x} + \boldsymbol{w}\frac{\|\boldsymbol{r}\|}{\|\boldsymbol{w}\|} = \boldsymbol{x}_p$



Now let's multiply both sides of this equation by $\boldsymbol{w}$ and also add b:

$$\boldsymbol{w}'\boldsymbol{x} + b + \boldsymbol{w}'\boldsymbol{w}\frac{\|\boldsymbol{r}\|}{\|\boldsymbol{w}\|} = \boldsymbol{w}'\boldsymbol{x}_p + b$$

Since $\boldsymbol{x}_p$ lies on the boundary, we have

$$\boldsymbol{w}'\boldsymbol{x} + b + \|\boldsymbol{w}\|^2\frac{\|\boldsymbol{r}\|}{\|\boldsymbol{w}\|} = 0$$

Distance is $\|\boldsymbol{r}\| = -\frac{\boldsymbol{w}'\boldsymbol{x}+b}{\|\boldsymbol{w}\|}$

SVM

- Training data is a collection $\{\boldsymbol{x}_i, y_i\}, i = 1, \ldots, n,$ where each $\boldsymbol{x}_i$ is an m-dimensional instance and y_i is the corresponding binary label encoded as -1 or 1
- Given a perfect separation boundary, y_i encode the side of the boundary each $\boldsymbol{x}_i$ is on
- Thus the distance from the i-th point to a perfect boundary can be encoded as

$$\boxed{\|\boldsymbol{r}_i\| = \frac{y_i(\boldsymbol{w}'\boldsymbol{x}_i + b)}{\|\boldsymbol{w}\|}}$$

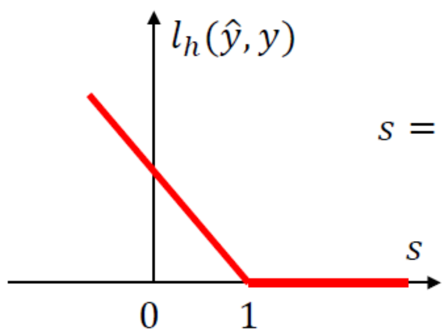
Soft-margin

- Hard-margin SVM loss

$$l_\infty = \begin{cases} 0 & 1 - y(\boldsymbol{w}'\boldsymbol{x} + b) \leq 0 \\ \infty & otherwise \end{cases}$$

- Soft-margin SVM loss (hinge loss)

$$l_h = \begin{cases} 0 & 1 - y(\boldsymbol{w}'\boldsymbol{x} + b) \leq 0 \\ 1 - y\underbrace{(\boldsymbol{w}'\boldsymbol{x} + b)}_{\hat{y}} & otherwise \end{cases}$$



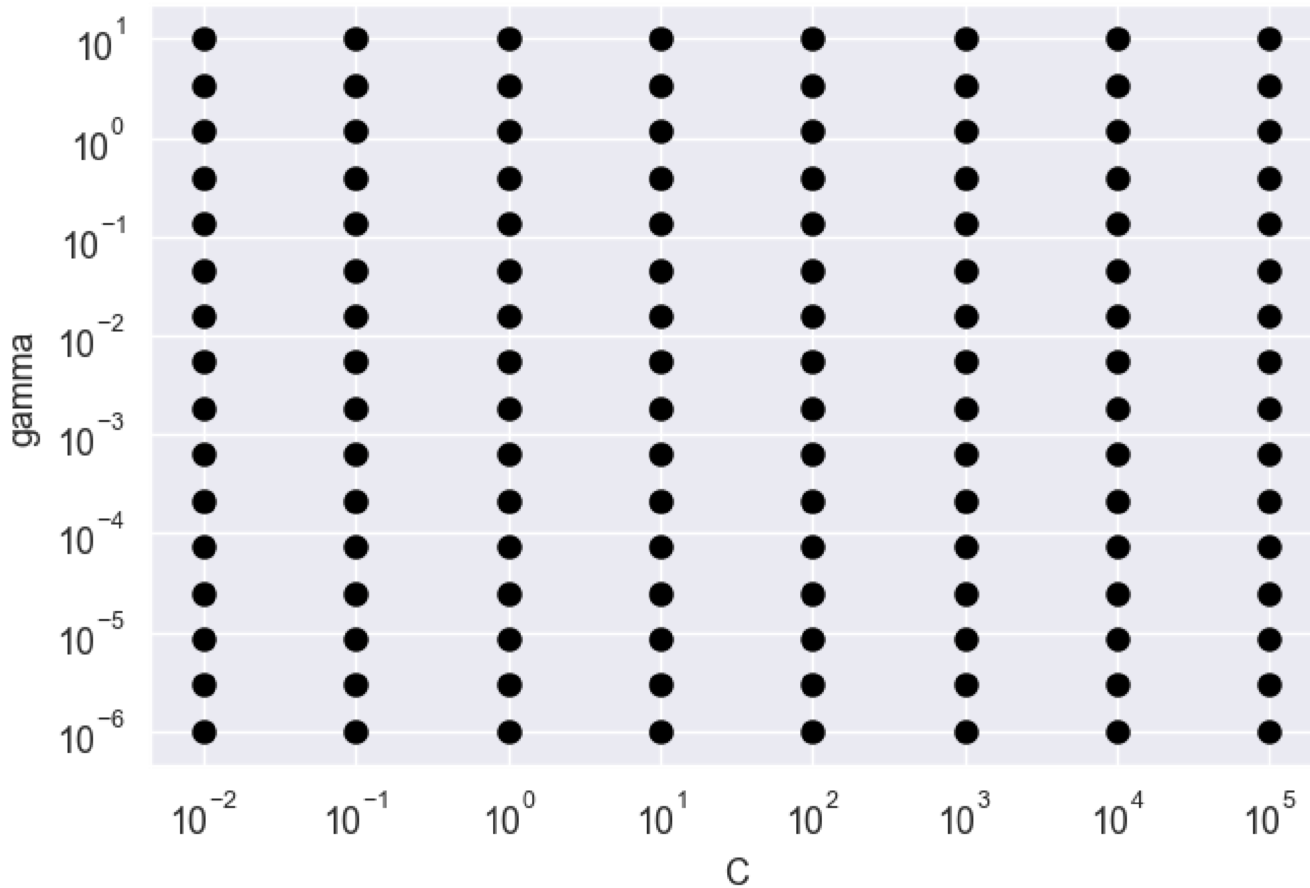
$s = y\hat{y}$

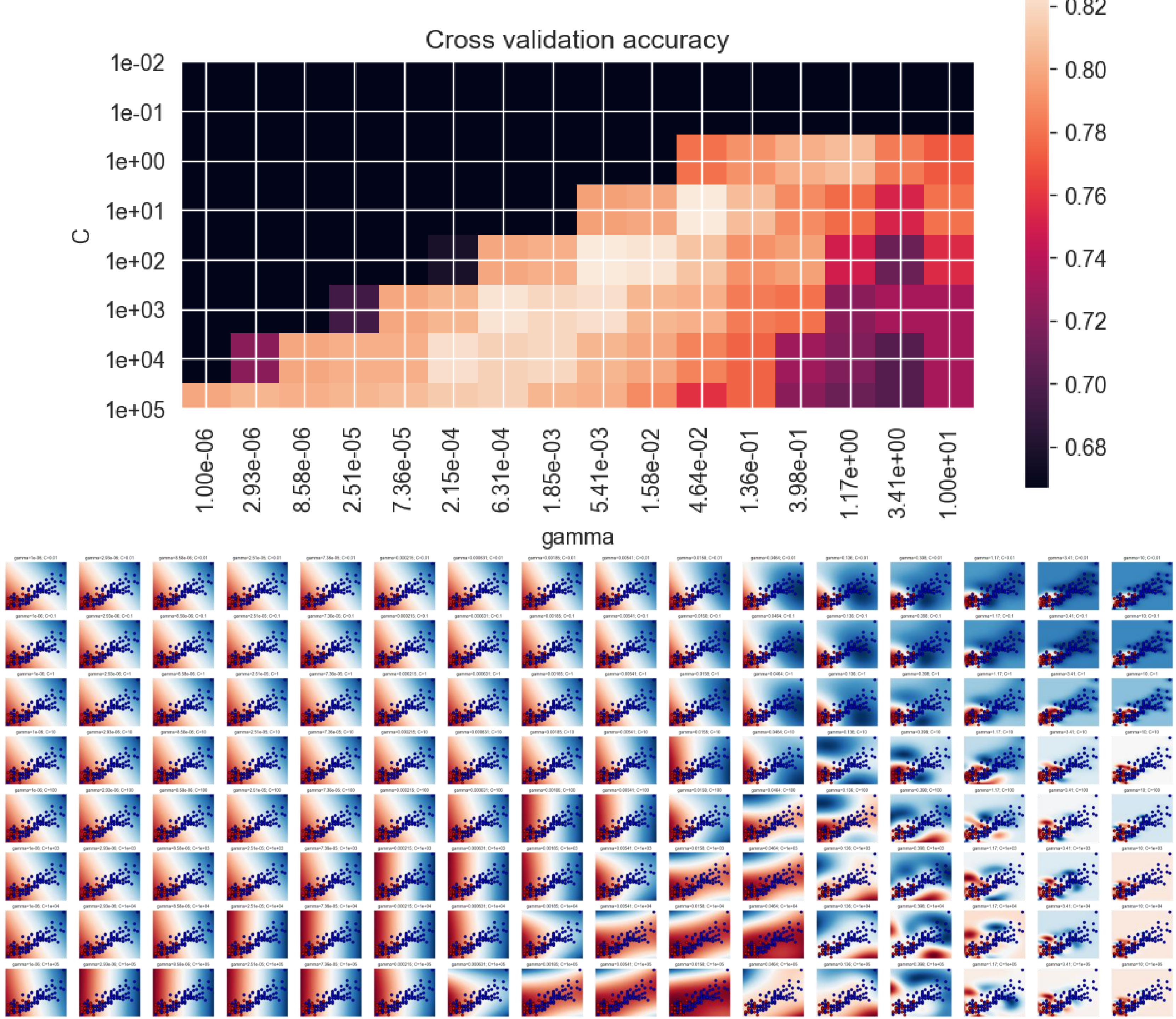
compare this with perceptron loss

SVC

- https://scikit-learn.org/stable/modules/svm.html#svc
- radial basis function (RBF) kernel
- Hyper-parameter
  C
  gamma

Parameter Grid Search




Cross validation accuracy
C
1e-02
1e-01
1e+00
1e+01
1e+02
1e+03
1e+04
1e+05
gamma
1.00e-06
2.93e-06
8.58e-06
2.51e-05
7.36e-05
2.15e-04
6.31e-04
1.85e-03
5.41e-03
1.58e-02
4.64e-02
1.36e-01
3.98e-01
1.17e+00
3.41e+00
1.00e+01
0.82
0.80
0.78
0.76
0.74
0.72
0.70
0.68

Primal vs dual

- Introduce auxiliary objective function via auxiliary variables

$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\nu}) = f(\boldsymbol{x}) + \sum_{i=1}^{n} \lambda_i g_i(\boldsymbol{x}) + \sum_{j=1}^{m} \nu_j h_j(\boldsymbol{x})$$

Primal constraints became penalties

  - Called the *Lagrangian* function
  - New $\boldsymbol{\lambda}$ and $\boldsymbol{\nu}$ are called the *Lagrange multipliers* or *dual variables*

- (Old) primal program: $\min_{\boldsymbol{x}} \max_{\boldsymbol{\lambda} \geq \mathbf{0}, \boldsymbol{\nu}} \mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\nu})$
- (New) dual program: $\max_{\boldsymbol{\lambda} \geq \mathbf{0}, \boldsymbol{\nu}} \min_{\boldsymbol{x}} \mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\nu})$

May be easier to solve, advantageous

- Duality theory relates primal/dual:
  - Weak duality: dual optimum $\leq$ primal optimum
  - For convex programs (inc. SVM!) strong duality: optima coincide!