

Week 10 – Support Vector Machines (SVM)

Regression

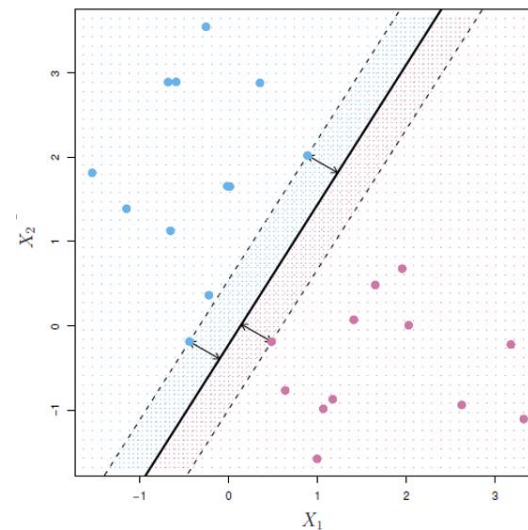
Pedram Jahangiry

JON M.
HUNTSMAN
SCHOOL OF BUSINESS

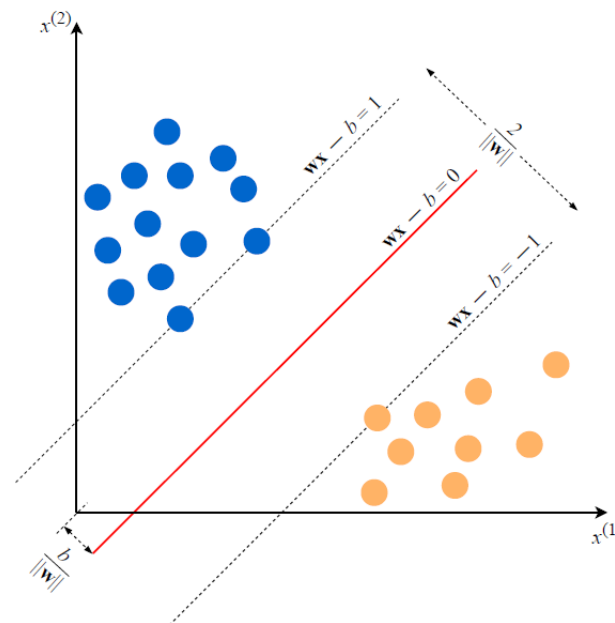
UtahStateUniversity

Support Vector Classification (SVC)

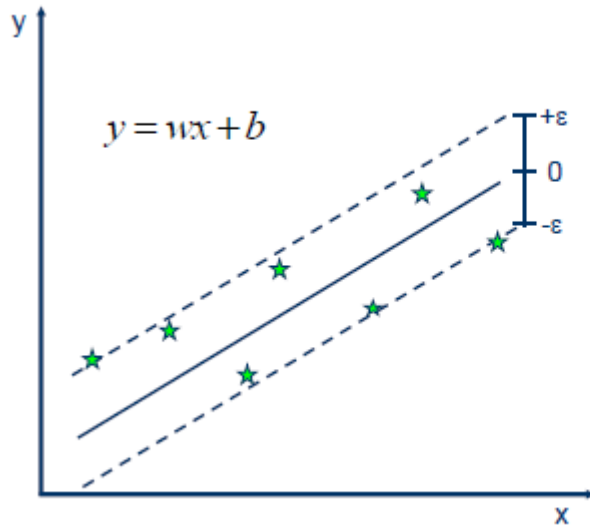
$$\begin{aligned}
 & \underset{\beta_0, \beta_1, \dots, \beta_p, \epsilon_1, \dots, \epsilon_n, M}{\text{maximize}} && M \\
 & \text{subject to} && \sum_{j=1}^p \beta_j^2 = 1, \\
 & && y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \dots + \beta_p x_{ip}) \geq M(1 - \epsilon_i) \\
 & && \epsilon_i \geq 0, \quad \sum_{i=1}^n \epsilon_i \leq C,
 \end{aligned}$$



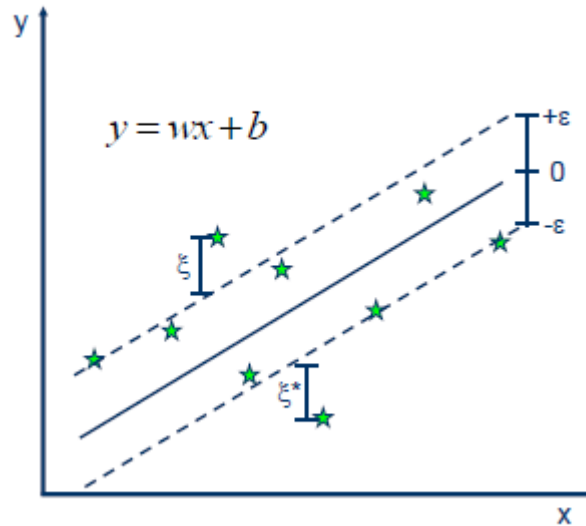
$$\begin{aligned}
 & \underset{w, b, \zeta}{\min} && \frac{1}{2} w^T w + C \sum_{i=1}^n \zeta_i \\
 & \text{subject to} && y_i(w^T \phi(x_i) + b) \geq 1 - \zeta_i, \\
 & && \zeta_i \geq 0, i = 1, \dots, n
 \end{aligned}$$



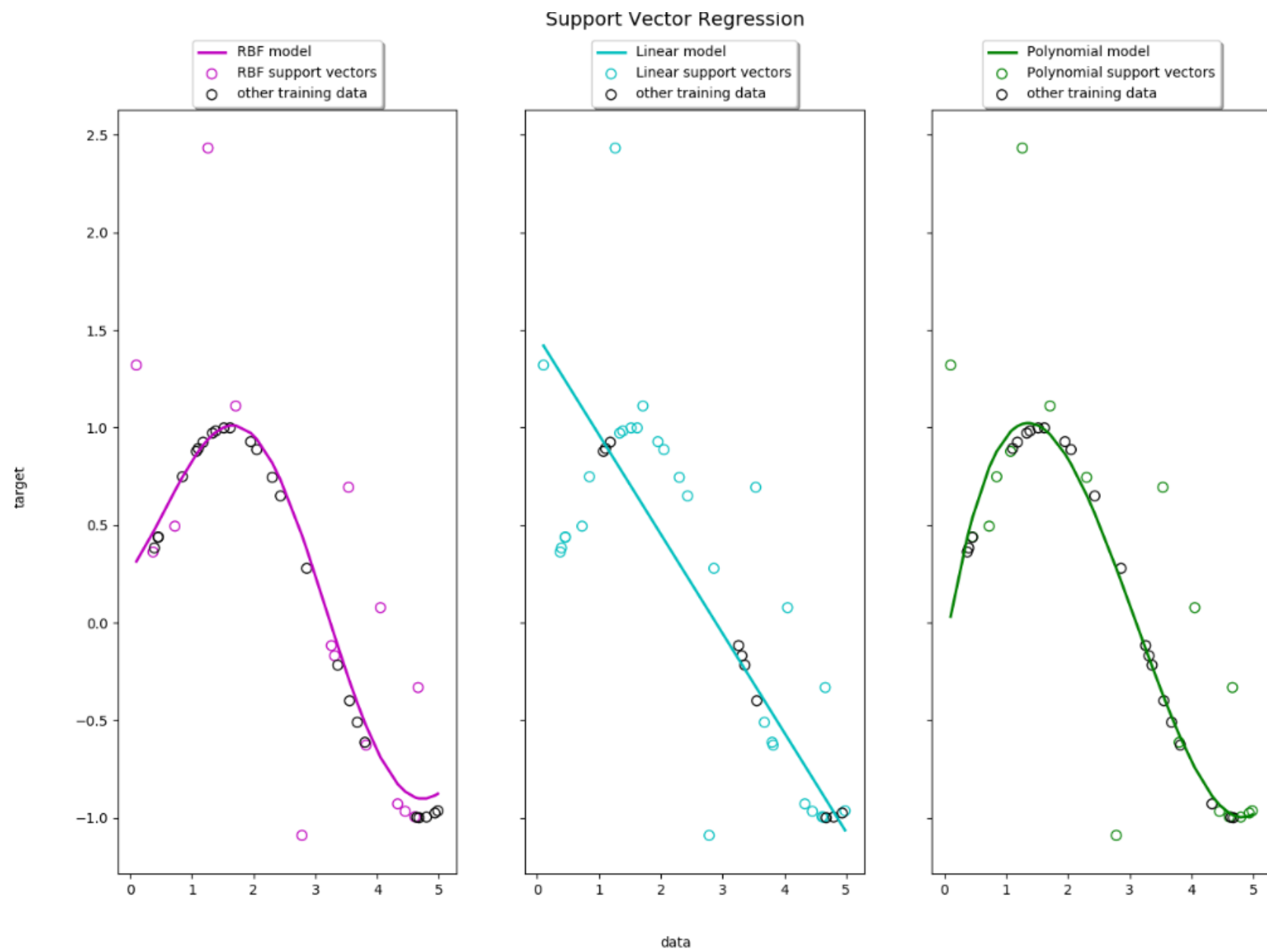
Support Vector Regression (SVR)



- Minimize:
$$\min \frac{1}{2} \|w\|^2$$
- Constraints:
$$y_i - wx_i - b \leq \varepsilon$$
$$wx_i + b - y_i \leq \varepsilon$$



- Minimize:
$$\frac{1}{2} \|w\|^2 + C \sum_{i=1}^N (\xi_i + \xi_i^*)$$
- Constraints:
$$y_i - wx_i - b \leq \varepsilon + \xi_i$$
$$wx_i + b - y_i \leq \varepsilon + \xi_i^*$$
$$\xi_i, \xi_i^* \geq 0$$



SVR in Python

- Find the SVR Sklearn documentation [here](#)
- Blackbox version of SVR in python:

```
from sklearn import svm
X = [[0, 0], [2, 2]]
y = [0.5, 2.5]
clf = svm.SVR()
clf.fit(X, y)

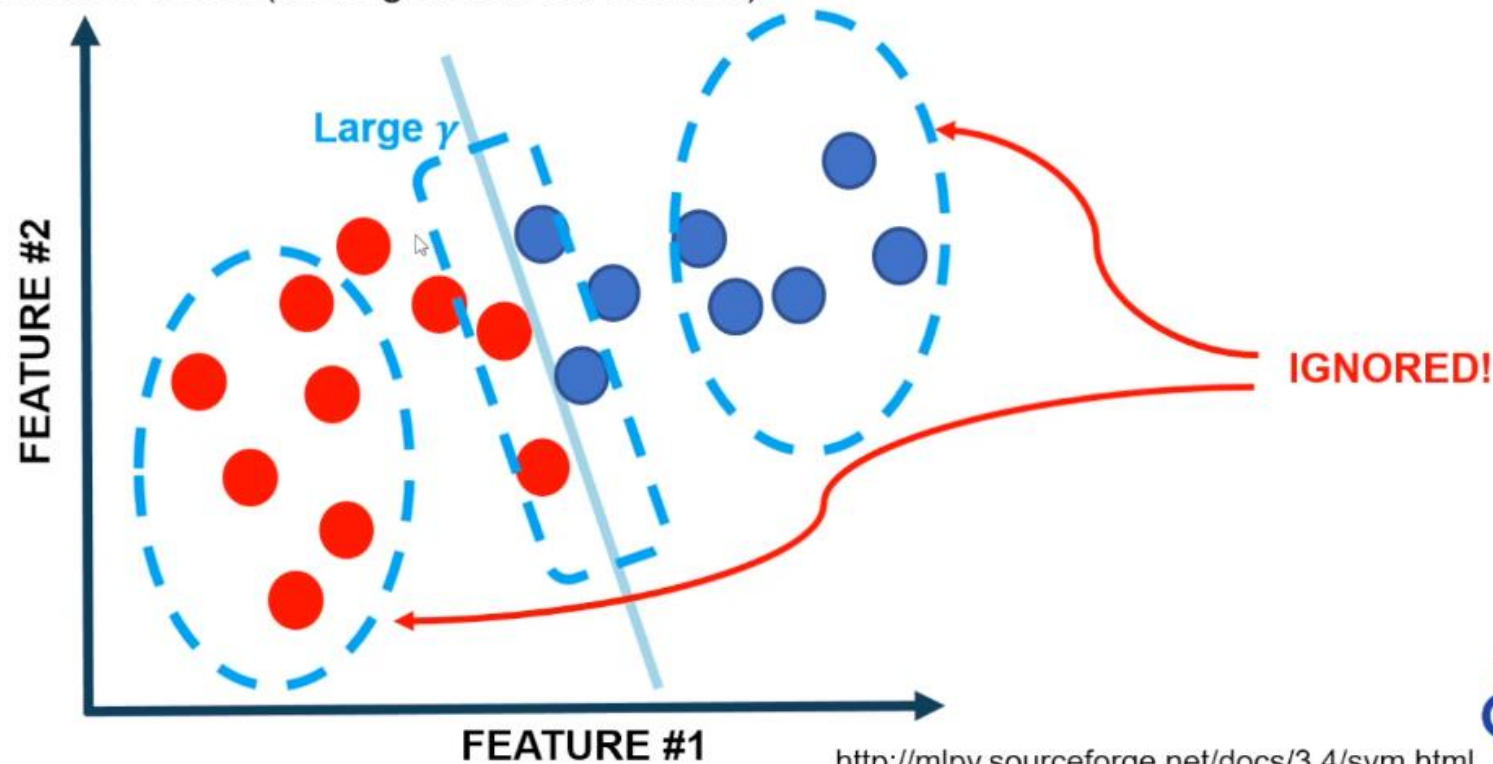
clf.predict([[1, 1]])
```

```
SVR(C=1.0, cache_size=200, coef0=0.0, degree=3, epsilon=0.1,
    gamma='auto_deprecated', kernel='rbf', max_iter=-1, shrinking=True,
    tol=0.001, verbose=False)
>>> clf.predict([[1, 1]])
array([1.5])
```

SVM PARAMETERS OPTIMIZATION

Gamma parameter: controls how far the influence of a single training set reaches

- **Large gamma:** close reach (closer data points have high weight)
- **Small gamma:** far reach (more generalized solution)



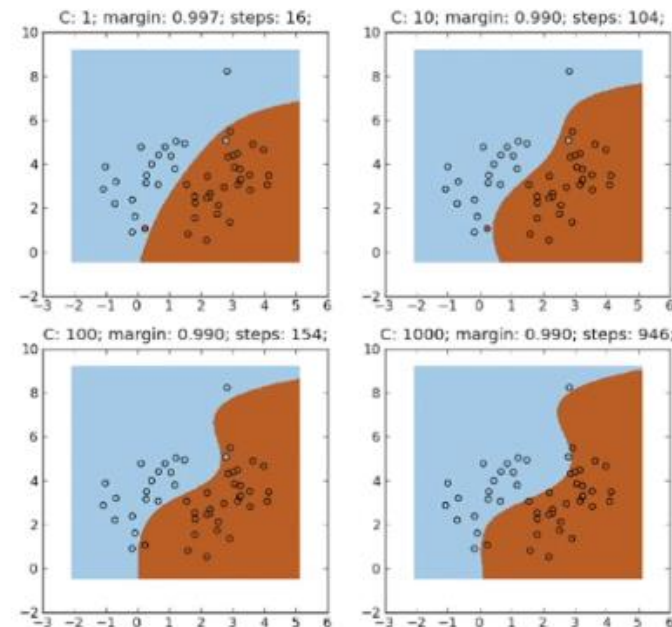
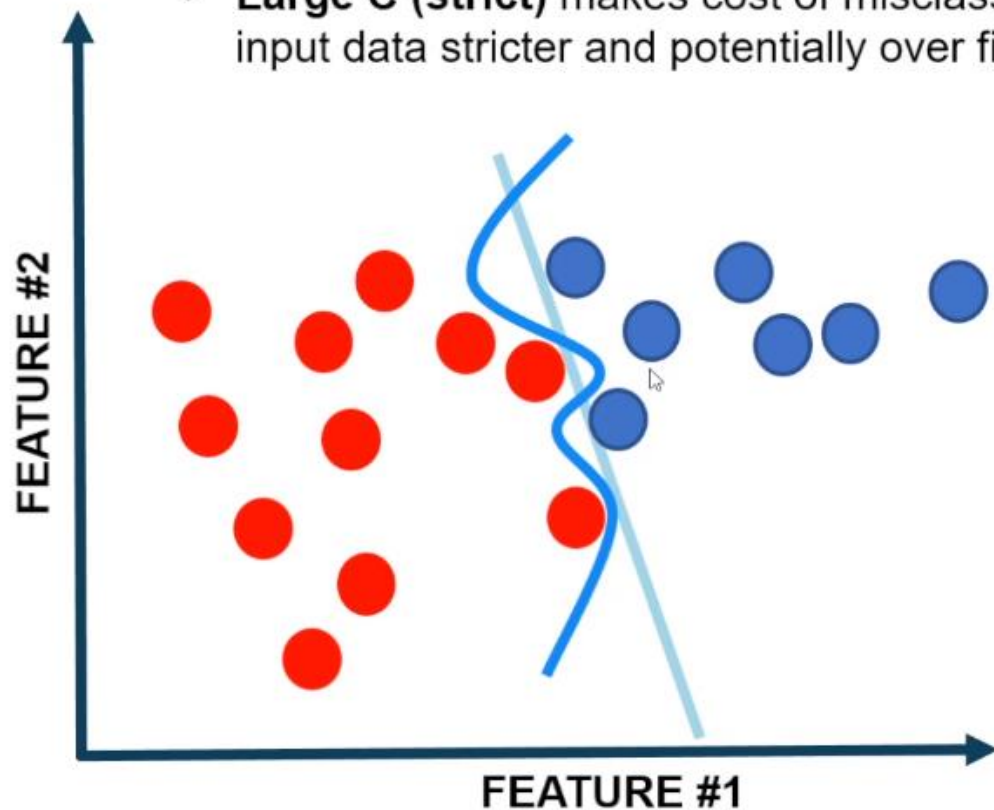
<http://mlpy.sourceforge.net/docs/3.4/svm.html>



SVM PARAMETERS OPTIMIZATION

C parameter: Controls trade-off between classifying training points correctly and having a smooth decision boundary

- **Small C (loose)** makes cost (penalty) of misclassification low (soft margin)
- **Large C (strict)** makes cost of misclassification high (hard margin), forcing the model to explain input data stricter and potentially over fit.



<http://mlpy.sourceforge.net/docs/3.4/svm.html>

