# Support Vector Clustering

Grant Baker, Matt Maierhofer

University of Colorado

December 10, 2018

# Overview

1. Support Vector Clustering
2. Fast Support Vector Clustering
3. Scalable Support Vector Clustering
4. Results

# Support Vector Clustering

- Introduced by Ben-Hur et al. [BHHSV01]
- Projects data to infinite dimensional space
- Finds the smallest bounding hypersphere on the data, allowing penalized outliers
- Sphere is projected back to the data space, forming clusters on the data

# The Primal Problem

$$
\begin{aligned}
\min_{R,a,\xi} \quad & R^2 + C\sum_j \xi_j \\
& ||\phi(X_i) - a||^2 \leq R^2 + \xi_i \\
& \xi_i \geq 0
\end{aligned}
$$

$$X \in \mathbb{R}^{m \times n},\ \xi \in \mathbb{R}^n$$

# The Lagrangian

$$
\begin{aligned}
\mathcal{L}(R, a, \xi, \beta, \mu) =& R^2 + C \sum_{j=1}^{n} \xi_j \\
& - \sum_{j=1}^{n} \beta_j (R^2 + \xi_j - ||\phi(X_j) - a||^2) - \sum_{j=1}^{n} \xi_j \mu_j
\end{aligned}
$$

# KKT Conditions - Stationarity

$$\nabla \mathcal{L} = 0$$

---

$$\sum_{j=1}^{n} \beta_j = 1$$

$$a = \sum_{j=1}^{n} \beta_j \phi(X_j)$$

$$\beta_j = C - \mu_j$$

# KKT Conditions - Complementary Slackness

$$\beta_j(R^2 + \xi_j - ||\phi(X_j) - a||^2) = 0$$
$$\xi_j \mu_j = 0$$

# The Dual Problem

$$\max_{\mu,\beta,R,a,\xi} \mathcal{L}(R,a,\xi,\beta,\mu)$$

$$\nabla_{R,a,\xi}\mathcal{L} = 0$$

# Using Stationarity

Let $\hat{L}(\beta) = \mathcal{L}$ under our stationarity conditions.

$$\begin{aligned}
\hat{L}(\beta) &= R^2 + C\sum_j \xi_j - \sum_j \beta_j(R^2 + \xi_j - ||\phi(X_j) - a||^2) - \sum_j \xi_j\mu_j \\
&= R^2(1 - \sum_j \beta_j) + \sum_j \xi_j(C - \beta_j - \mu_j) + \sum_j \beta_j(||\phi(X_j) - a||^2) \\
&= \sum_j \beta_j||\phi(X_j) - a||^2 \\
&= \sum_j \beta_j(||\phi(X_j)||^2 - 2a \cdot \phi(X_j) + ||a||^2) \\
&= \sum_j \beta_j\Big(||\phi(X_j)||^2 - 2\sum_i \beta_i\phi(X_i)\cdot\phi(X_j)\Big) + ||\sum_i \beta_i\phi(X_i)||^2 \\
&= \sum_j \beta_j\phi(X_j)\cdot\phi(X_j) - \sum_{i,j} \beta_i\beta_j\phi(X_i)\cdot\phi(X_j)
\end{aligned}$$

# What is $\phi$?

- $\phi(x)$ maps $x$ from an $n$-dimensional data space to an infinite dimensional feature space
- We don't actually ever compute $\phi(x)$, instead we use the "Kernel Trick"
- Define $K(x_1, x_2) = \phi(x_1) \cdot \phi(x_2) = e^{-q||x_1 - x_2||_2^2}$
- Note $K(x, x) = 1$

# The Dual - Reworked

$$\max_{\beta \geq 0} \left( 1 - \sum_{i,j} \beta_i \beta_j K(X_i, X_j) \right)$$

# Defining the Radius

Note that if $0 < \beta_j < C$, $\phi(X_j)$ is on the surface of the hypersphere, and we define

$$R^2(x) = ||\phi(x) - a||^2 = K(x, x) - 2\sum_j \beta_j K(x, X_j) + \sum_{i,j} \beta_i \beta_j K(X_i, X_j)$$

as the distance from the center of the sphere, $R = R(X_j)$ is the radius of our hypersphere.

# Clustering

Finally, we build an adjacency matrix for the points. Two points, $(X_i, X_j)$, are considered adjacent if the line between them in the data space is within the hypersphere in the feature space, or

$$R(x) \leq R \;\; \forall x \in \{X_i + t(X_j - X_i) | t \in [0,1]\}$$

In practice, we sample points on the line to check. After we build the adjacency matrix, clusters of points are determined via a breadth first search of the matrix.

# Speeding Up the Adjacency Matrix

- Computing the Adjacency Matrix can be very costly, $O(mn^3)$, want to reduce overall cost
- Ideally want to find a subset of “important” points and cluster using these
- Fast Support Vector Clustering[PDLL17], introduced by Pham et al., helps solve this

# Fast Support Vector Clustering

- First, take the set of all points within $\epsilon$ of the boundary. This should give a solid representation of the boundary
- Next, we use a fixed point iterator to push our points to an equilibrium at a local optima of our radius function.

$$x_{k+1} = \frac{\sum_j \beta_j e^{-q||X_j - x_k||^2} X_j}{\sum_j \beta_j e^{-q||X_j - x_k||^2}}$$

- We then cluster these “equilibrium points” using the adjacency matrix method
- The boundary points are assigned to the same cluster as the corresponding equilibrium points, and the other points are assigned to the same cluster as the closest boundary point in the data space.

# Speeding Up the Training Process

- Scalable Support Vector Clustering, Pham et al. [PLD17]
- Instead of solving a quadratic program, we fit the primal problem into the stochastic gradient framework
- Introduce a budget for the maximum number of points with nonzero weight

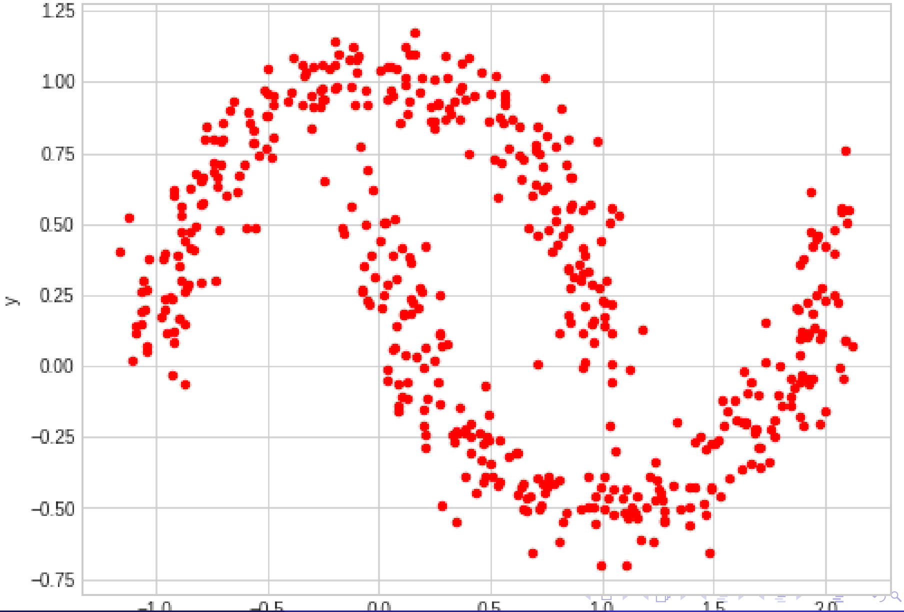
# Scalable Support Vector Clustering

1. Draw $X_i$ at random from $X$
2. Update the weights $\beta_j \leftarrow \frac{t-1}{t}\beta_j$
3. Check if $X_i$ is outside the cluster, i.e. if

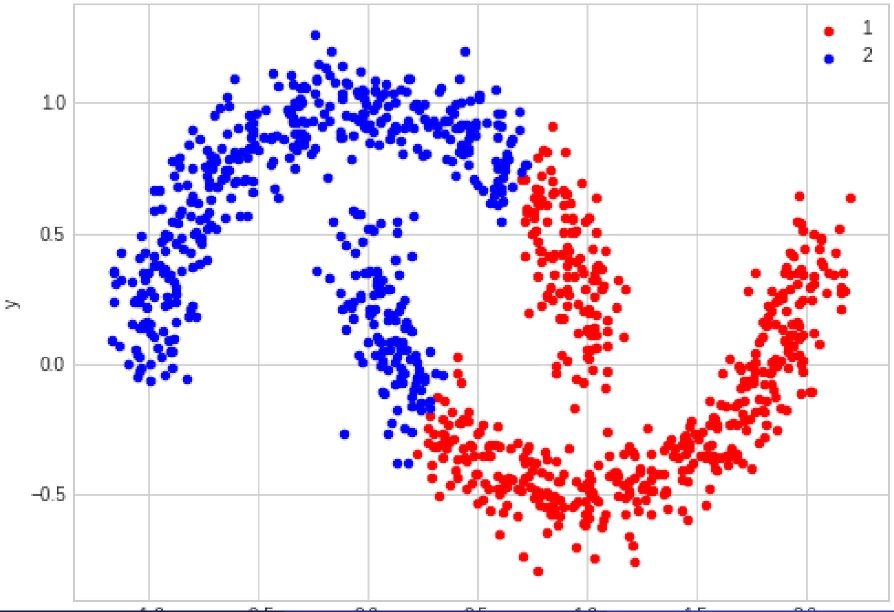
$$f(X_i) = 2\sum_j \beta_j K(X_j, X_i) - R < 0$$

4. If so, update $\beta_i \leftarrow \beta_i + 2CR\eta_t$
5. If there are more than B nonzero $\beta_j$, set the smallest one to 0

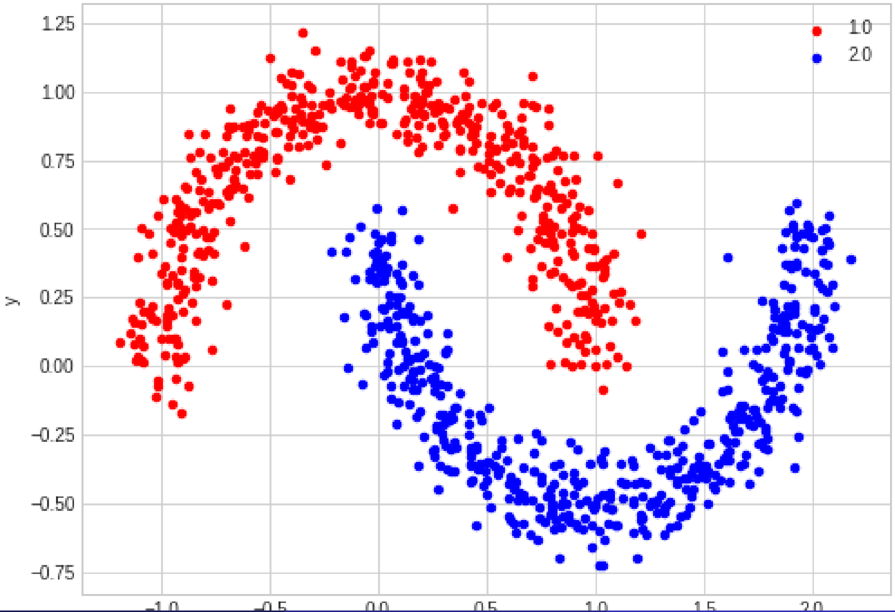
# Two Moons



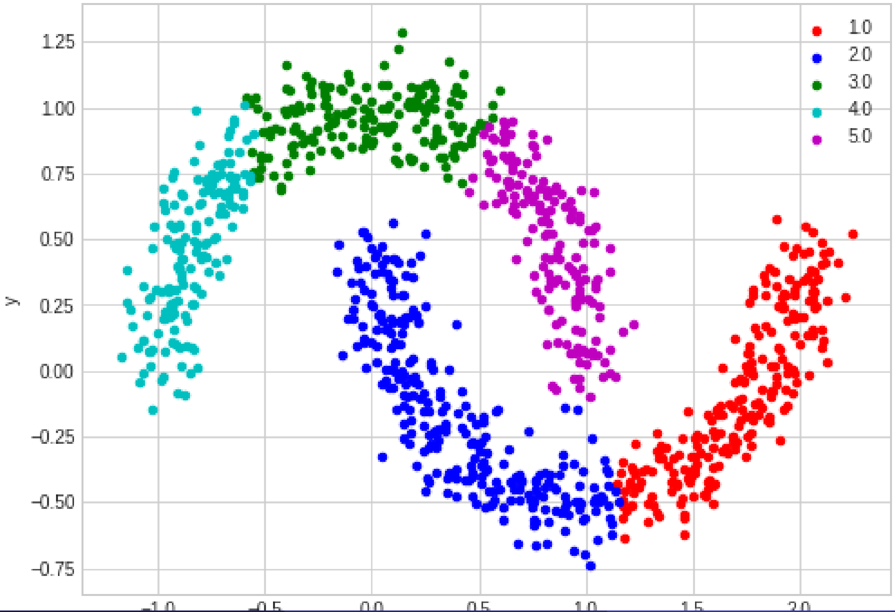
# KMeans



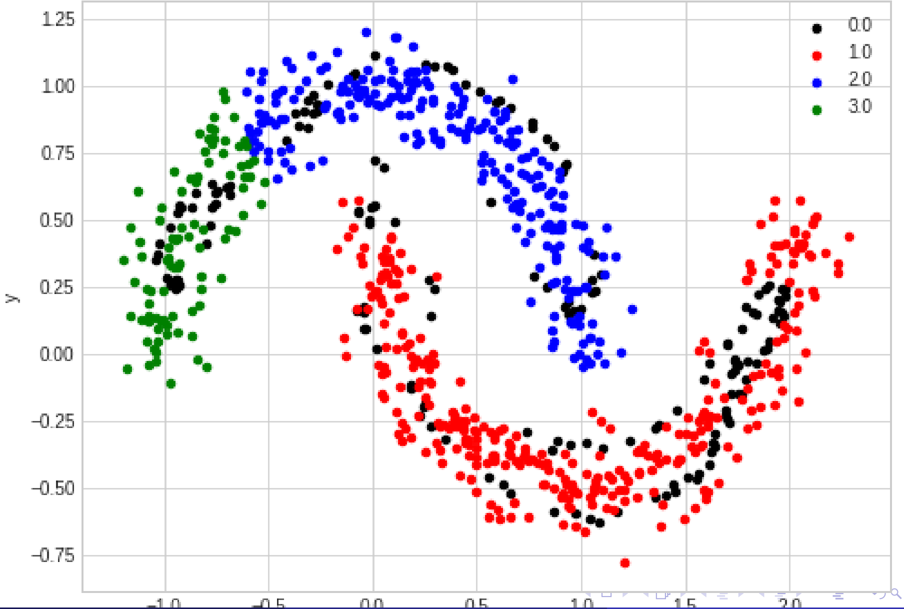
# Two Moons with SVC



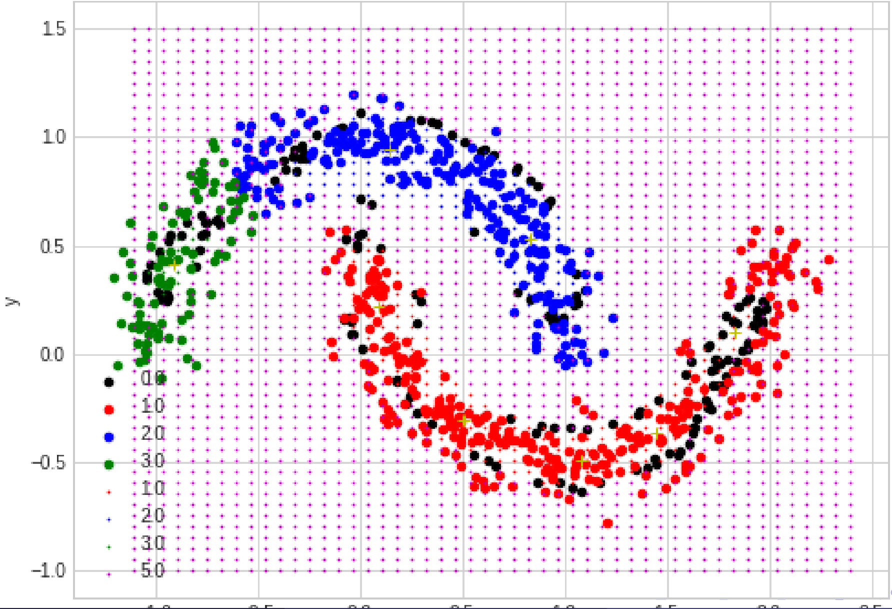
# Two Moons Failure



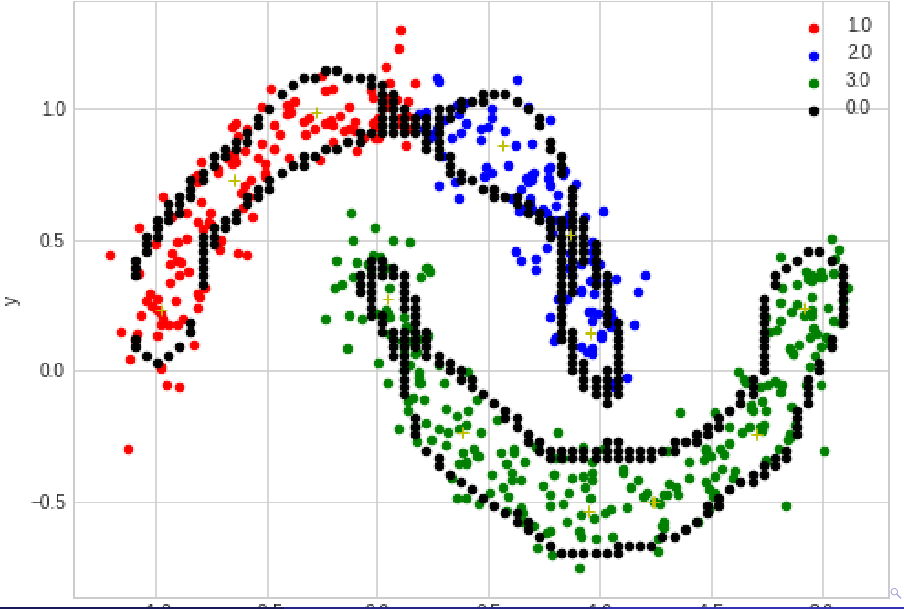
# Two Moons Failure



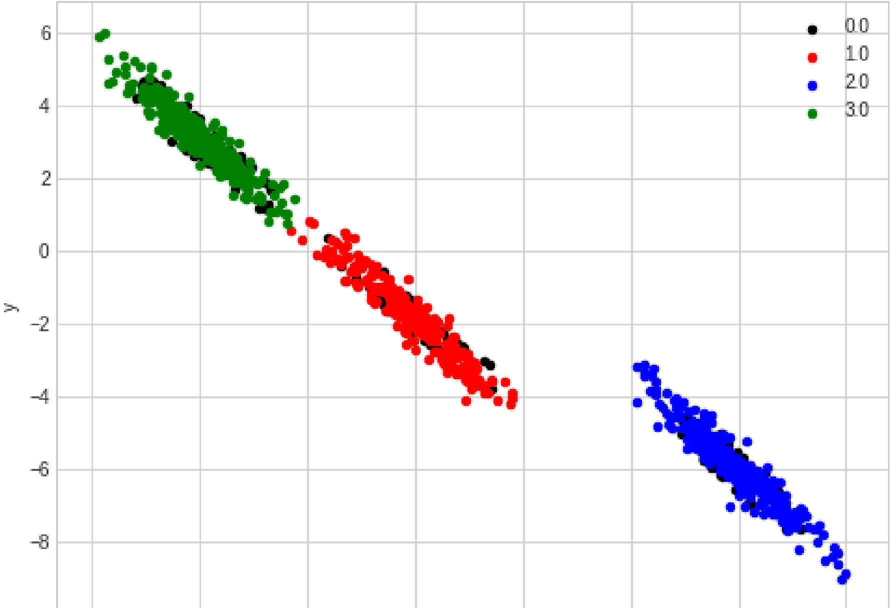
# Two Moons Failure



# Two Moons Failure



# Distorted Blobs



# References

Asa Ben-Hur, David Horn, Hava Siegelmann, and Vladimir Vapnik.
Support vector clustering, 2001.

Tung Pham, Hang Dang, Trung Le, and Thai Hoang Le.
Fast support vector clustering.
*Vietnam Journal of Computer Science*, 4(1):13–21, Feb 2017.

Tung Pham, Trung Le, and Hang Dang.
Scalable support vector clustering using budget.
*CoRR*, abs/1709.06444, 2017.