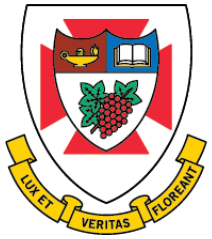
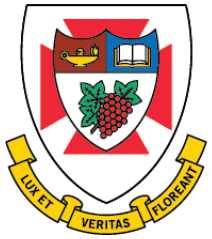


THE UNIVERSITY OF WINNIPEG | Professional, Applied and Continuing Education

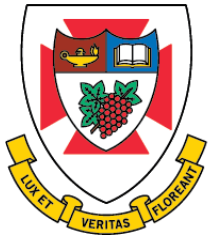
INTRODUCTION TO MACHINE LEARNING

DIT 45100



THE UNIVERSITY OF WINNIPEG | Professional, Applied and Continuing Education

Module 3
Linear Classification Techniques



THE UNIVERSITY OF WINNIPEG | Professional, Applied and Continuing Education

Multinomial Logistic Regression

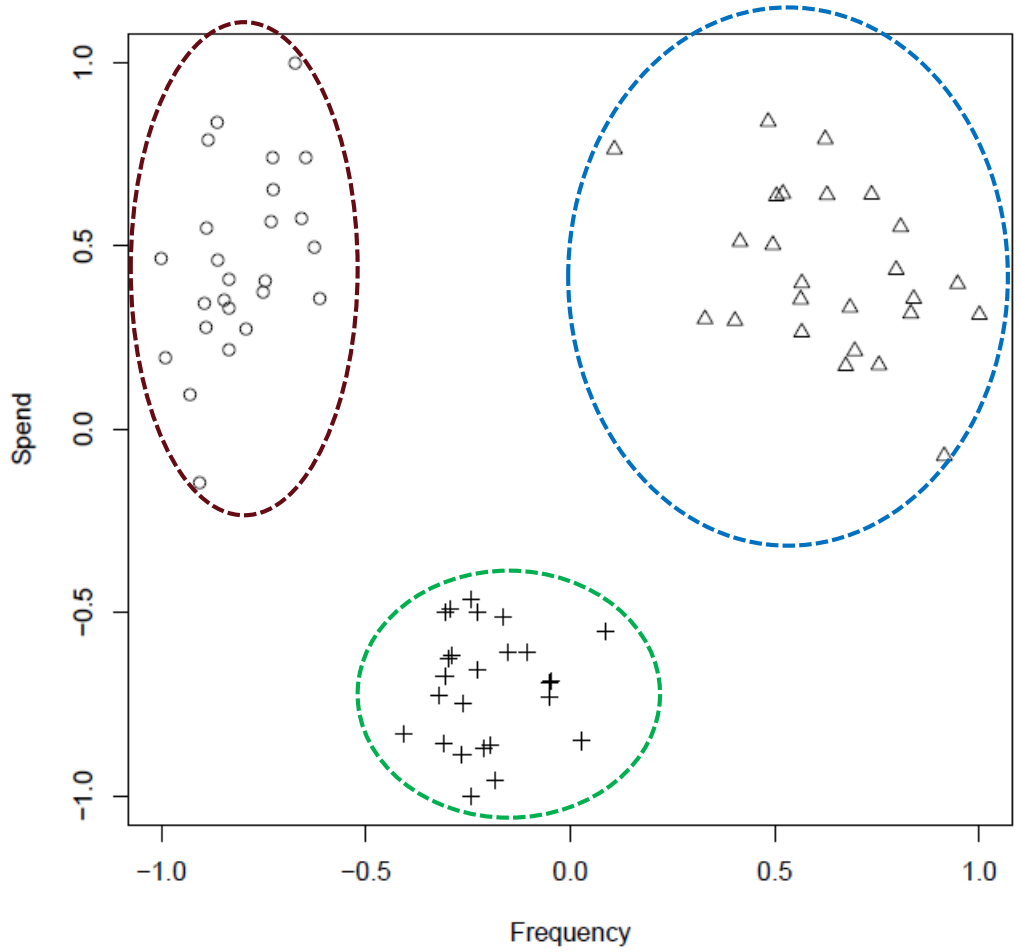
Multinomial Logistic Regression

Table: A dataset of customers of a large national retail chain.

ID	SPEND	FREQ	TYPE
1	21.6	5.4	single
2	25.7	7.1	single
3	18.9	5.6	single
4	25.7	6.8	single
		⋮	
26	107.9	5.8	business
27	92.9	5.5	business
28	122.6	6.0	business
29	107.7	5.7	business
		⋮	
47	53.2	2.6	family
48	52.4	2.0	family
49	46.1	1.4	family
50	65.3	2.2	family

Multinomial Logistic Regression

Three (3) Categories of customers



Multinomial Logistic Regression

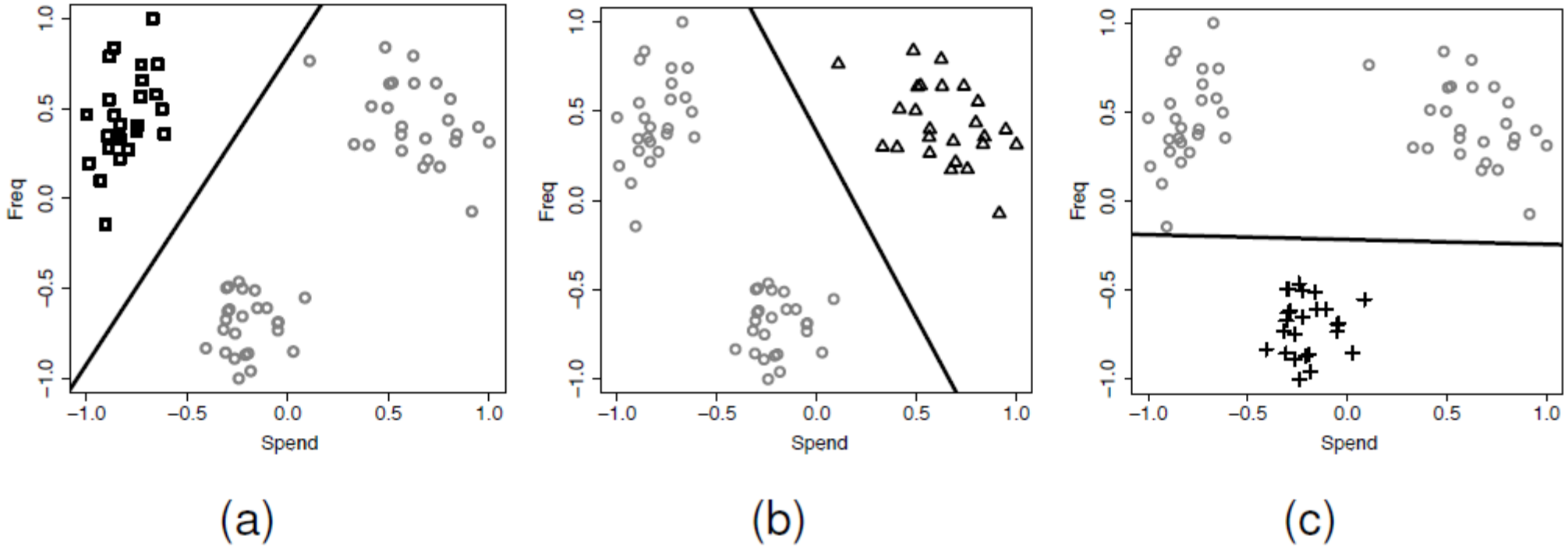


Figure: An illustration of three different **one-versus-all** prediction models for the customer type dataset that has three target levels 'single' (squares), 'business' (triangles) and 'family' (crosses).

Multinomial Logistic Regression

- For r target feature levels, we build r separate logistic regression models $\mathbb{M}_{\mathbf{w}_1}$ to $\mathbb{M}_{\mathbf{w}_r}$:

$$\begin{aligned}
\mathbb{M}_{\mathbf{w}_1}(\mathbf{d}) &= \mathit{logistic}(\mathbf{w_1} \cdot \mathbf{d}) \\
\mathbb{M}_{\mathbf{w}_2}(\mathbf{d}) &= \mathit{logistic}(\mathbf{w_2} \cdot \mathbf{d}) \\
&\vdots \\
\mathbb{M}_{\mathbf{w}_r}(\mathbf{d}) &= \mathit{logistic}(\mathbf{w_r} \cdot \mathbf{d})
\end{aligned} \tag{7}$$

where $\mathbb{M}_{\mathbf{w}_1}$ to $\mathbb{M}_{\mathbf{w}_r}$ are r different one-versus-all logistic regression models, and $\mathbf{w_1}$ to $\mathbf{w_r}$ are r different sets of weights.

Multinomial Logistic Regression

- To combine the outputs of these different models, we normalize their results using:

$$\mathbb{M}'_{\mathbf{w_k}}(\mathbf{d}) = \frac{\mathbb{M}_{\mathbf{w_k}}(\mathbf{d})}{\sum_{l \in levels(t)} \mathbb{M}_{\mathbf{w}_l}(\mathbf{d})} \quad (8)$$

where $\mathbb{M}'_{\mathbf{w_k}}(\mathbf{d})$ is a revised, normalized prediction for the one-versus-all model for the target level k.

Multinomial Logistic Regression

- The r one-versus-all logistic regression models used are trained in parallel, and the revised model outputs, $\mathbb{M}'_{\mathbf{w_k}}(\mathbf{d})$, are used when calculating the sum of squared errors for each model during the training process.
- This means that the sum of squared errors function is changed slightly to

$$L_2(\mathbb{M}_{\mathbf{w_k}}, \mathcal{D}) = \frac{1}{2}\sum_{i=1}^{n}\left(t_i - \mathbb{M}'_{\mathbf{w_k}}(\mathbf{d}_i[1])\right)^2 \qquad (9)$$

Multinomial Logistic Regression

- The revised predictions are also used when making predictions for query instances. The predicted level for a query, **q**, is the level associated with the one-versus-all model that outputs the highest result after normalization.
- We can write this as

$$\mathbb{M}(\mathbf{q}) = \underset{l \in levels(t)}{\text{argmax}}\ \mathbb{M}'_{\mathbf{w}_l}(\mathbf{q}) \tag{10}$$

Multinomial Logistic Regression

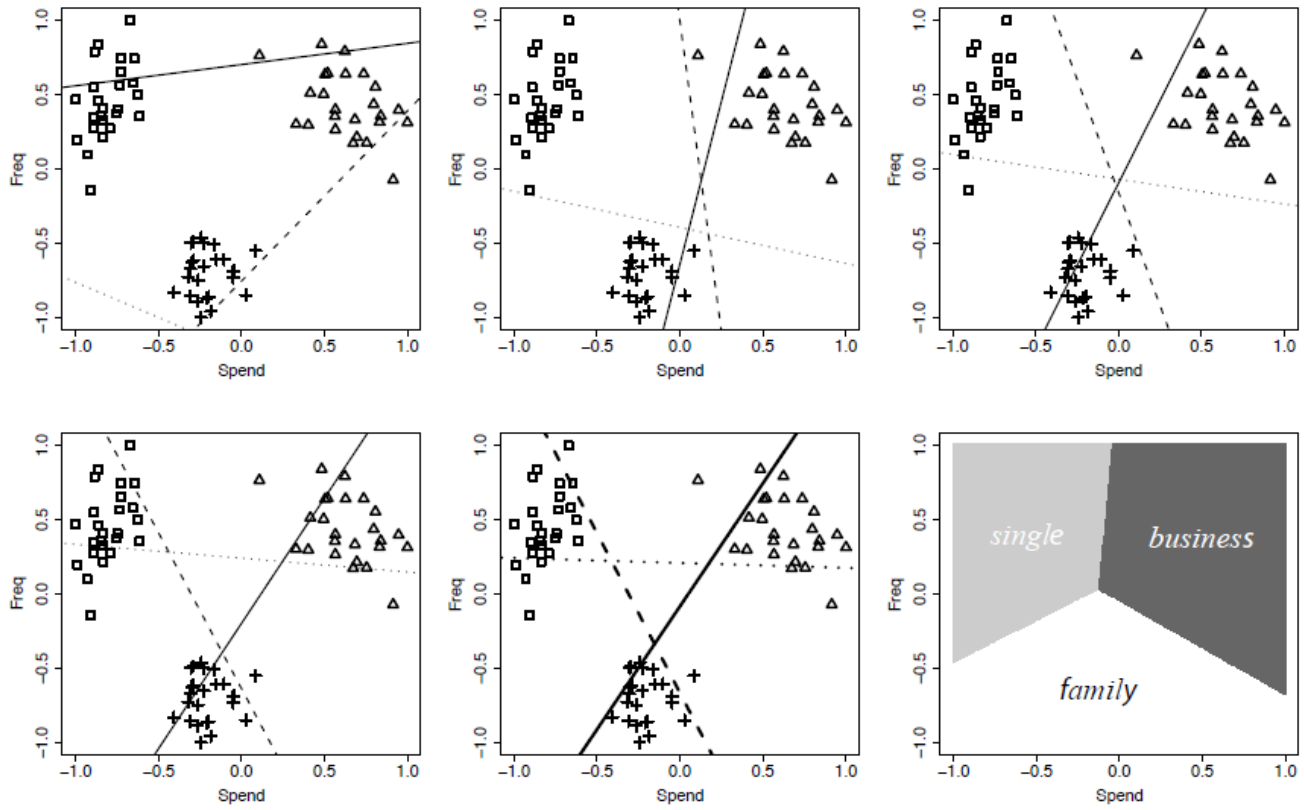
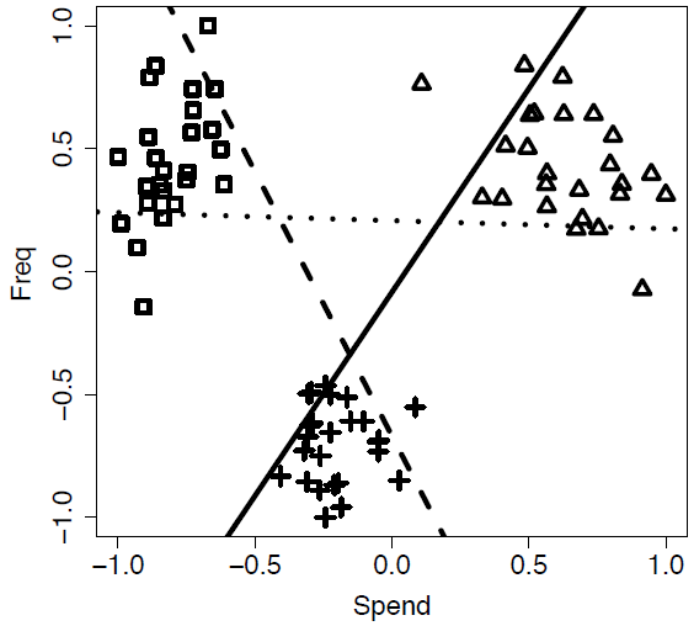


Figure: A selection of the models developed during the gradient descent process for the customer group dataset. Squares represent instances with the 'single' target level, triangles the 'business' level and crosses the 'family' level. (f) illustrates the overall decision boundaries that are learned between the three target levels.

Multinomial Logistic Regression



$$\mathbb{M}_{\mathbf{w}\,'single'}(\mathbf{q}) = \mathit{Logistic}(0.7993 - 15.9030 \times \textsc{Spend} + 9.5974 \times \textsc{Freq})$$

$$\mathbb{M}_{\mathbf{w}\,'family'}(\mathbf{q}) = \mathit{Logistic}(3.6526 + -0.5809 \times \textsc{Spend} - 17.5886 \times \textsc{Freq})$$

$$\mathbb{M}_{\mathbf{w}\,'business'}(\mathbf{q}) = \mathit{Logistic}(4.6419 + 14.9401 \times \textsc{Spend} - 6.9457 \times \textsc{Freq})$$

Multinomial Logistic Regression

- For a query instance with SPEND $= 25.67$ and FREQ $= 6.12$, which are normalized to SPEND $= -0.7279$ and FREQ $= 0.4789$, the predictions of the individual models would be

$$\mathbb{M}_{\mathbf{w}\,'single'}(\mathbf{q}) = \mathit{Logistic}(0.7993 - 15.9030 \times (-0.7279) + 9.5974 \times 0.4789)$$
$$= 0.9999$$
$$\mathbb{M}_{\mathbf{w}\,'family'}(\mathbf{q}) = \mathit{Logistic}(3.6526 + -0.5809 \times (-0.7279) - 17.5886 \times 0.4789)$$
$$= 0.01278$$
$$\mathbb{M}_{\mathbf{w}\,'business'}(\mathbf{q}) = ?$$

Multinomial Logistic Regression

- For a query instance with SPEND $= 25.67$ and FREQ $= 6.12$, which are normalized to SPEND $= -0.7279$ and FREQ $= 0.4789$, the predictions of the individual models would be

$$\begin{aligned}
\mathbb{M}_{\mathbf{w}\,'single'}(\mathbf{q}) &= \mathit{Logistic}(0.7993 - 15.9030 \times (-0.7279) + 9.5974 \times 0.4789) \\
&= 0.9999 \\
\mathbb{M}_{\mathbf{w}\,'family'}(\mathbf{q}) &= \mathit{Logistic}(3.6526 + -0.5809 \times (-0.7279) - 17.5886 \times 0.4789) \\
&= 0.01278 \\
\mathbb{M}_{\mathbf{w}\,'business'}(\mathbf{q}) &= \mathit{Logistic}(4.6419 + 14.9401 \times (-0.7279) - 6.9457 \times 0.4789) \\
&= 0.0518
\end{aligned}$$

Multinomial Logistic Regression

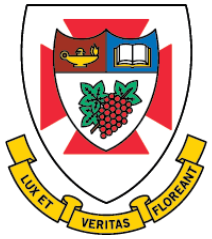
- These predictions would be normalized as follows:

$$\mathbb{M}'_{\mathbf{w}\,'single'}(\mathbf{q}) = \frac{0.9999}{0.9999 + 0.01278 + 0.0518} = 0.9393$$

$$\mathbb{M}'_{\mathbf{w}\,'family'}(\mathbf{q}) = \frac{0.01278}{0.9999 + 0.01278 + 0.0518} = 0.0120$$

$$\mathbb{M}'_{\mathbf{w}\,'business'}(\mathbf{q}) = \frac{0.0518}{0.9999 + 0.01278 + 0.0518} = 0.0487$$

- This means the overall prediction for the query instance is *'single'*, as this gets the highest normalized score.



THE UNIVERSITY OF WINNIPEG | Professional, Applied and Continuing Education

Support Vector Machines
SVM

SVM

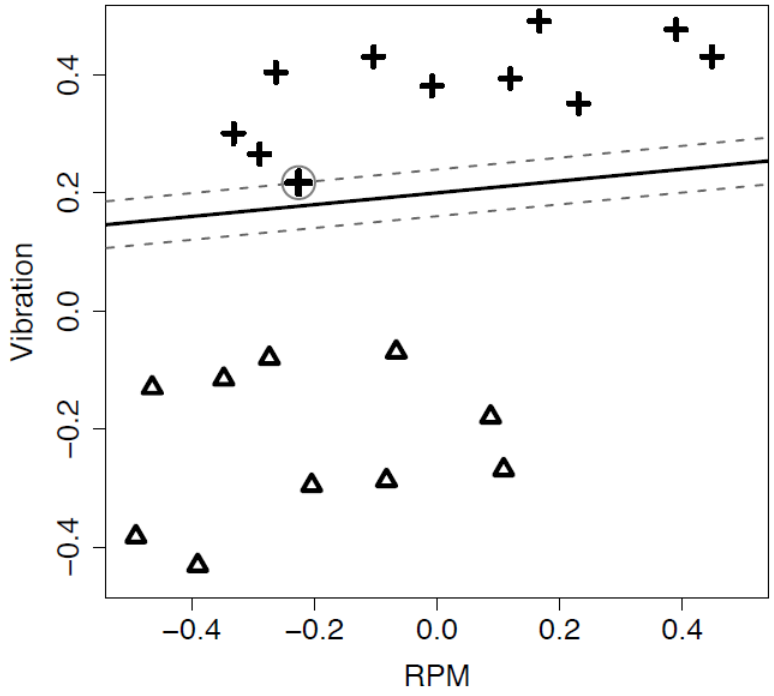


Figure: A small sample of the generators dataset with two features, RPM and VIBRATION, and two target levels, *'good'* (shown as crosses) and *'bad'* (shown as triangles). A decision boundary with a very small margin.

SVM

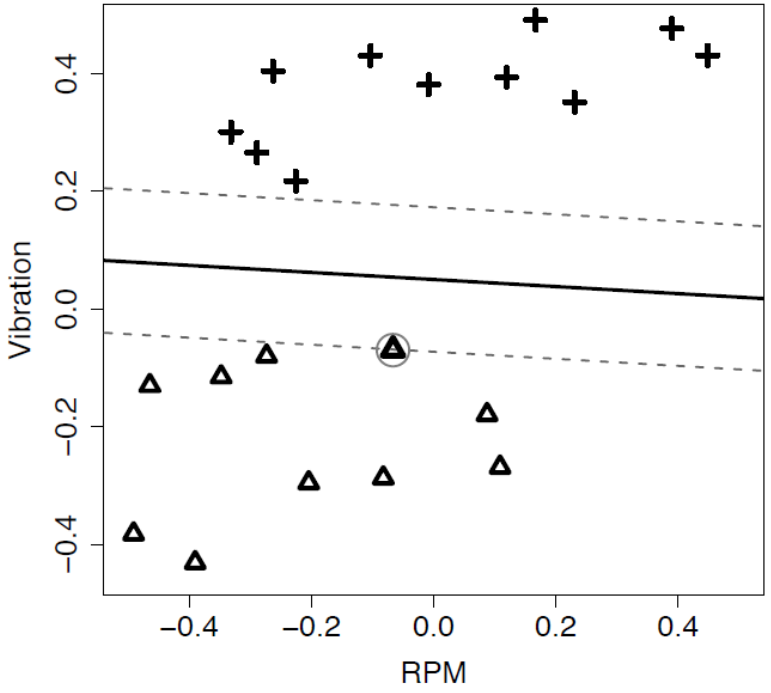


Figure: A small sample of the generators dataset with two features, RPM and VIBRATION, and two target levels, *'good'* (shown as crosses) and *'bad'* (shown as triangles). A decision boundary with a large margin.

SVM

- Training a support vector machine involves searching for the decision boundary, or **separating hyperplane**, that leads to the maximum margin as this will best separate the levels of the target feature.
- The instances in a training dataset that fall along the margin extents, and so define the margins, are known as the **support vectors** and define the decision boundary.

SVM

- We define the separating hyperplane as follows:

$$w_0 + \mathbf{w} \cdot \mathbf{d} = 0 \tag{11}$$

- For instances above a separating hyperplane

$$w_0 + \mathbf{w} \cdot \mathbf{d} > 0$$

and for instances below a separating hyperplane

$$w_0 + \mathbf{w} \cdot \mathbf{d} < 0$$

SVM

- We build a support vector machine prediction model so that instances with the negative target level result in the model outputting ≤ -1 and instances with the positive target level result in the model outputting $\geq +1$.
- The space between the outputs of -1 and $+1$ allows for the margin.

SVM

- A support vector machine model is defined as

$$\mathbb{M}_{\boldsymbol{\alpha},w_0}(\mathbf{q}) = \sum_{i=1}^{s} (t_i \times \boldsymbol{\alpha}[i] \times (\mathbf{d}_i \cdot \mathbf{q}) + w_0) \qquad (12)$$

where

- $\mathbf{q}$ is the set of descriptive features for a query instance;
- $(\mathbf{d}_1, t_1), \ldots, (\mathbf{d}_s, t_s)$ are s support vectors (instances composed of descriptive features and a target feature);
- w_0 is the first weight of the decision boundary;
- and $\boldsymbol{\alpha}$ is a set of parameters determined during the training process (there is a parameter for each support vector $\boldsymbol{\alpha}[1], \ldots, \boldsymbol{\alpha}[s]$).[1]

SVM

- Training a support vector machine is framed as a **constrained quadratic optimization problem**
- This type of problem is defined in terms of:
  - a set of constraints, and
  - an optimization criterion.

SVM

- The required **constraints** required by the training process are

$$w_0 + \mathbf{w} \cdot \mathbf{d} \leq -1 \text{ for } t_i = -1 \tag{13}$$

and:

$$w_0 + \mathbf{w} \cdot \mathbf{d} \geq +1 \text{ for } t_i = +1 \tag{14}$$

- We can combine these two constraints into a single constraint (remember t_i is always equal to either -1 or $+1$):

$$t_i \times (w_0 + \mathbf{w} \cdot \mathbf{d}) \geq 1 \tag{15}$$

SVM

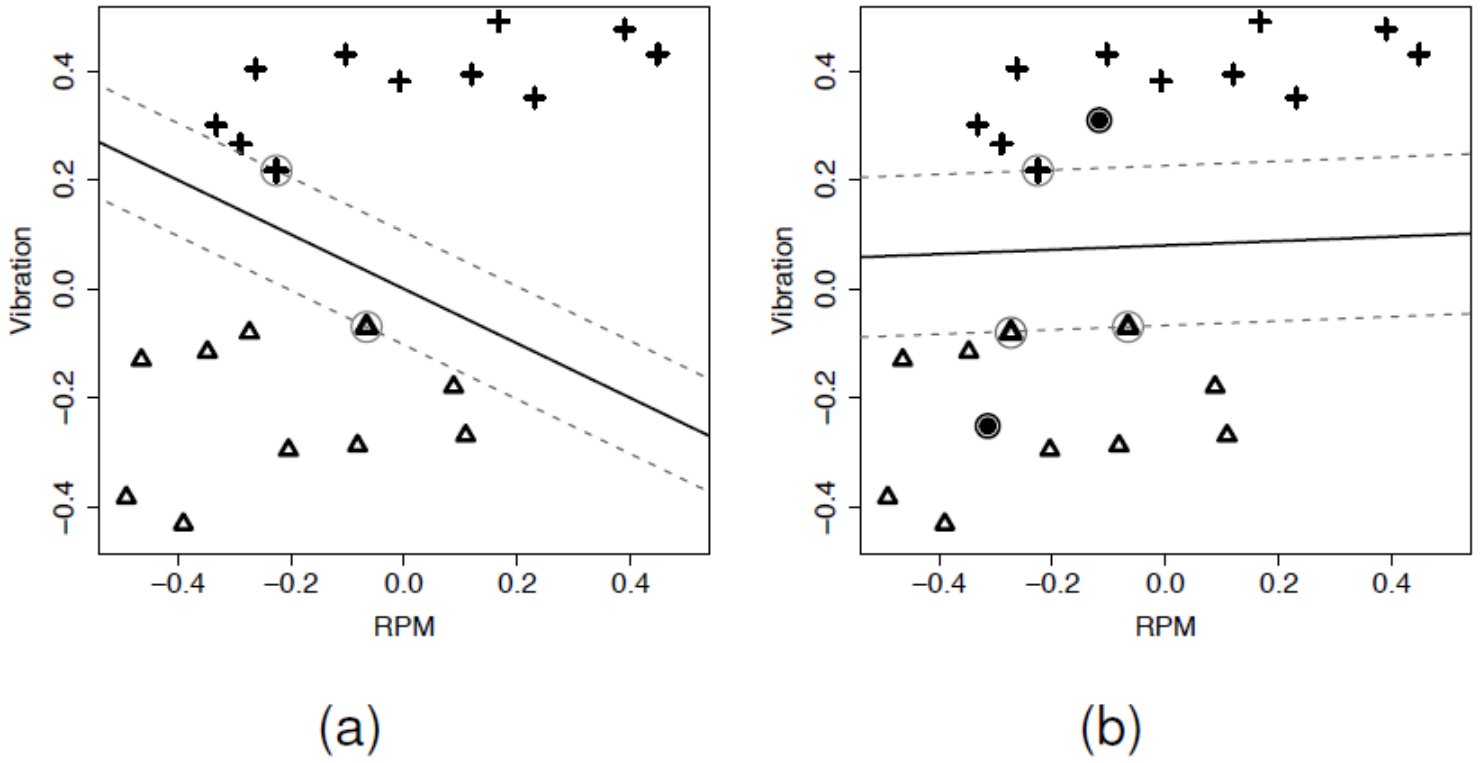


Figure: Different margins that satisfy the constraint in Equation (15). The instances that define the margin are highlighted in each case. (b) shows the maximum margin and also shows two query instances represented as black dots.

SVM

- The **optimization** criterion used is defined in terms of the perpendicular distance from any instance to the decision boundary and is given by

abs(w0 + w . d)

$$dist(\mathbf{d}) = \frac{w_0 + abs(\mathbf{w} \cdot \mathbf{d})}{||\mathbf{w}||}$$

where $||\mathbf{w}||$ is known as the **Euclidean norm** of $\mathbf{w}$ and is calculated as

$$||\mathbf{w}|| = \sqrt{\mathbf{w}[1]^2 + \mathbf{w}[2]^2 + \ldots + \mathbf{w}[m]^2}$$

SVM

- For instances along the **margin extents**, $abs(\mathbf{w} \cdot \mathbf{d} + w_0) = 1$.
- So, the distance from any instance along the margin extents to the decision boundary is $\frac{1}{||\mathbf{w}||}$, and because the margin is symmetrical to either side of the decision boundary, the size of the margin is $\frac{2}{||\mathbf{w}||}$.

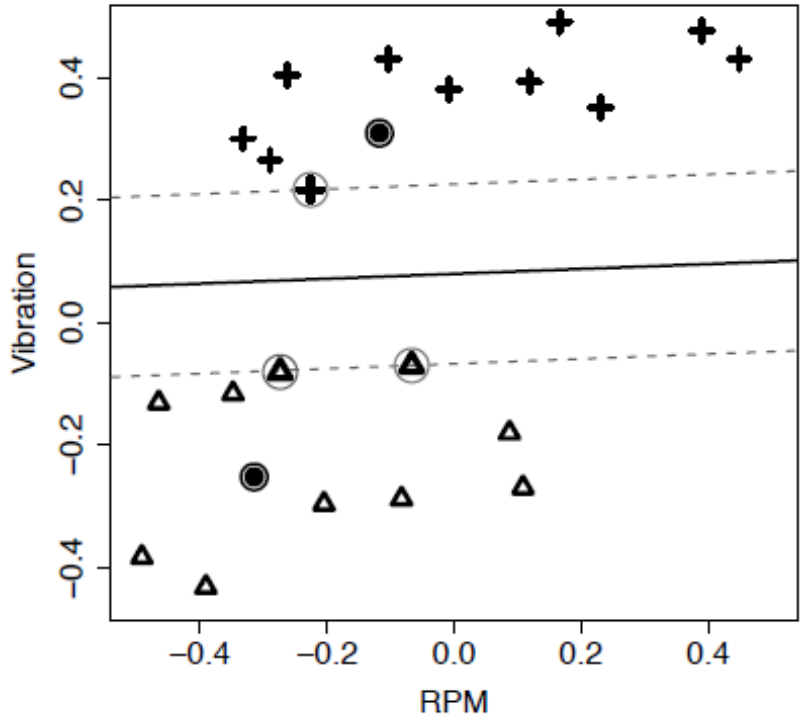
SVM

- The goal when training a support vector machine is
  1. maximize $\frac{2}{||\mathbf{w}||}$
  2. subject to the constraint

$$t_i \times (w_0 + \mathbf{w} \cdot \mathbf{d}) \geq 1$$

SVM

- The optimal decision boundary and associated support vectors for the example we have been following
- In this case 'good' is the positive level and set to +1, and 'faulty' is the negative level and set to -1.



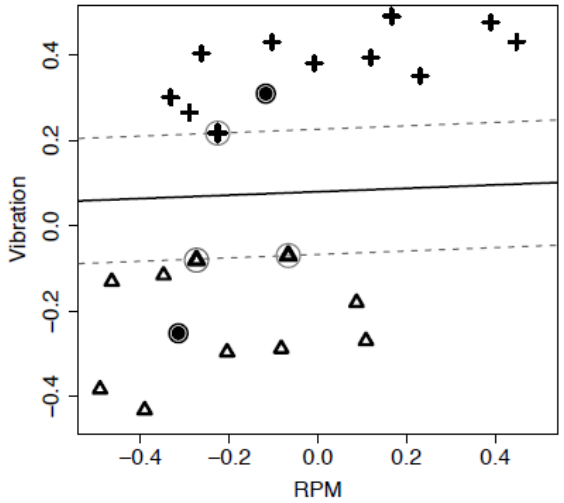
SVM

- The descriptive feature values and target feature values for the support vectors in these cases are
  - $(\langle -0.225, 0.217 \rangle, +1)$,
  - $(\langle -0.066, -0.069 \rangle, -1)$,
  - $(\langle -0.273, -0.080 \rangle, -1)$.
- The value of w_0 is -0.1838,
- The values of the α parameters are

$$\langle 22.056, 6.998, 16.058 \rangle.$$

23.056

SVM



- The plot shows the position of two new query instances for this problem.
- The descriptive feature values for these querys are
  1. $\mathbf{q_1} = \langle -0.314, -0.251 \rangle$
  2. $\mathbf{q_2} = \langle -0.117, 0.31 \rangle$.

SVM

- For the first query instance, $\mathbf{q_1} = \langle -0.314, -0.251 \rangle$, the output of the support vector machine model is:

$$
\begin{aligned}
&\mathbb{M}_{\boldsymbol{\alpha}, w_0}(\mathbf{q_1}) \\
&\quad = (1 \times 23.056 \times ((-0.225 \times -0.314) + (0.217 \times -0.251)) - 0.1838) \\
&\qquad + (-1 \times 6.998 \times ((-0.066 \times -0.314) + (-0.069 \times -0.251)) - 0.1838) \\
&\qquad + (-1 \times 16.058 \times ((-0.273 \times -0.314) + (-0.080 \times -0.251)) - 0.1838 \\
&\quad = -2.145
\end{aligned}
$$

- The model output is less than -1, so this query is predicted to be a *'faulty'* generator.
- For the second query instance, the model output is ?

SVM

- **Basis functions** can be used with support vector machines to handle data that is not **linearly separable**
- To use basis functions we update Equation (15) to

$$t_i \times (w_0 + \mathbf{w} \cdot \phi(\mathbf{d})) \geq 1 \text{ for all } i \qquad (16)$$

where ϕ is a set of basis functions applied to the descriptive features $\mathbf{d}$, and $\mathbf{w}$ is a set of weights containing one weight for each member of ϕ.

SVM

- Typically, the number of basis functions is larger than the number of descriptive features, so the application of the basis functions moves the data into a higher-dimensional space.
- The expectation is that a linear separating hyperplane will exist in this higher-dimensional space even though it does not in the original feature space.

SVM

- The prediction model in this case becomes

$$\mathbb{M}_{\boldsymbol{\alpha},\phi,w_0}(\mathbf{q}) = \sum_{i=1}^{s} (t_i \times \boldsymbol{\alpha}[i] \times (\phi(\mathbf{d}_i) \cdot \phi(\mathbf{q})) + w_0) \qquad (17)$$

- This equation requires a dot product calculation between the result of applying the basis functions to the query instance and to each of the support vectors which is repeated multiple times during the training process.

SVM

- A dot product is a computationally expensive operation,
- We can use a clever trick to avoid it:
  - the same result obtained by calculating the dot product of the descriptive features of a support vector and a query instance after having applied the basis functions can be obtained by applying a much less costly kernel function, *kernel*, to the original descriptive feature values of the support vector and the query.

SVM

- The prediction equation becomes

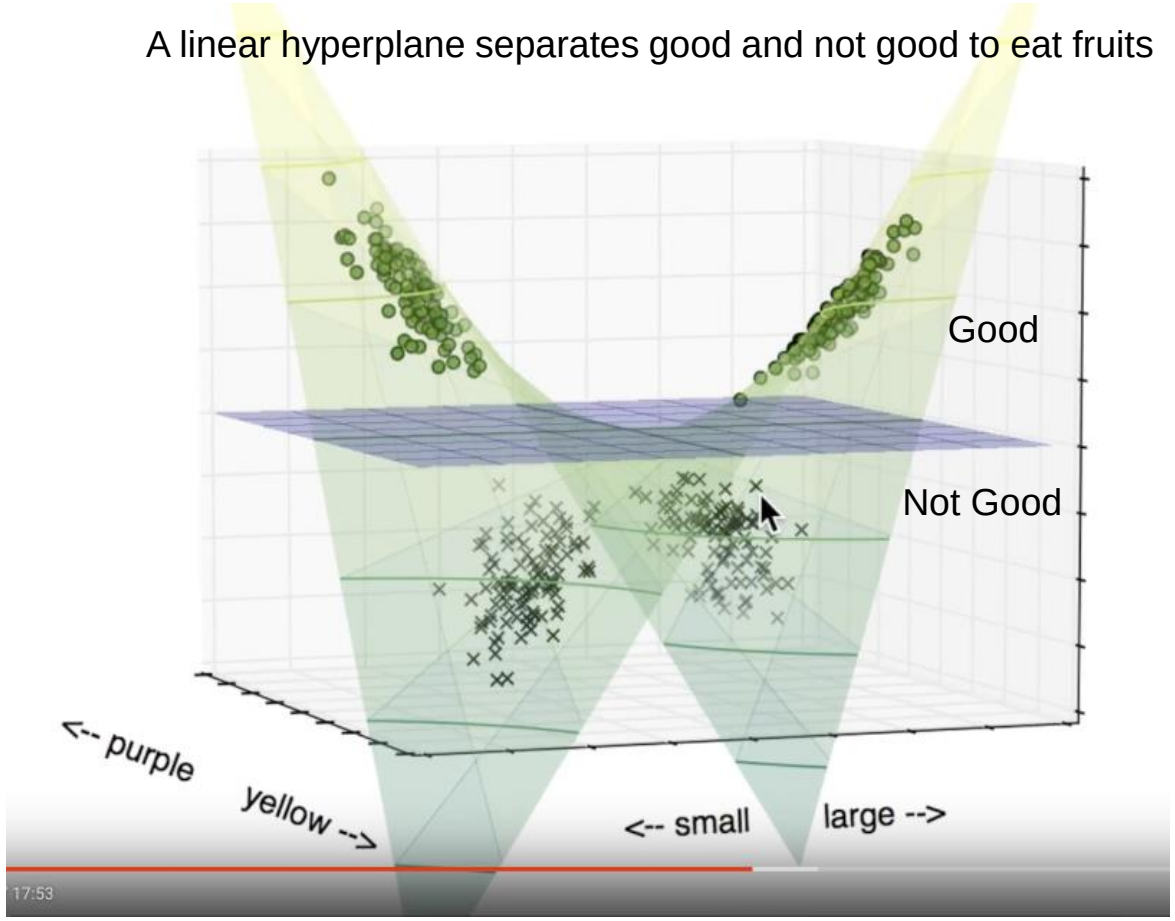
$$\mathbb{M}_{\boldsymbol{\alpha},kernel,w_0}(\mathbf{q}) = \sum_{i=1}^{s} (t_i \times \boldsymbol{\alpha}[i] \times kernel\,(\mathbf{d}_i, \mathbf{q}) + w_0) \qquad (18)$$

Kernel Trick

Kernel Trick

Good to eat and not good to eat fruits are linearly inseparable

A linear hyperplane separates good and not good to eat fruits



SVM

- A wide range of standard kernel functions can be used with support vector machines including:

Linear kernel

$$kernel(\mathbf{d}, \mathbf{q}) = \mathbf{d} \cdot \mathbf{q} + c$$

where c is an optional constant

Polynomial kernel

$$kernel(\mathbf{d}, \mathbf{q}) = (\mathbf{d} \cdot \mathbf{q} + 1)^p$$

where p is the degree of a polynomial function

Gaussian radial basis kernel

$$kernel(\mathbf{d}, \mathbf{q}) = \exp(-\gamma ||\mathbf{d} - \mathbf{q}||^2)$$

where γ is a manually chosen tuning parameter