Xi'an Jiaotong-Liverpool University
西交利物浦大学

INT305 Machine Learning

Lecture 4

Support Vector Machine, SVM Loss and Softmax Loss

Sichen Liu

Department Intelligence Science

Sichen.Liu@xjtlu.edu.cn

Binary Classification with a Linear Model

- Classification: Predict a discrete-valued target
- Binary classification: Targets $t \in \{-1, +1\}$
- Linear model:

$$z = \mathbf{w}^\top \boldsymbol{x} + b$$

$$y = \operatorname{sign}(z)$$

- Question: How should we choose $\mathbf{w}$ and b?

Zero-One Loss

- We can use the 0-1 loss function, and find the weights that minimize it over data points

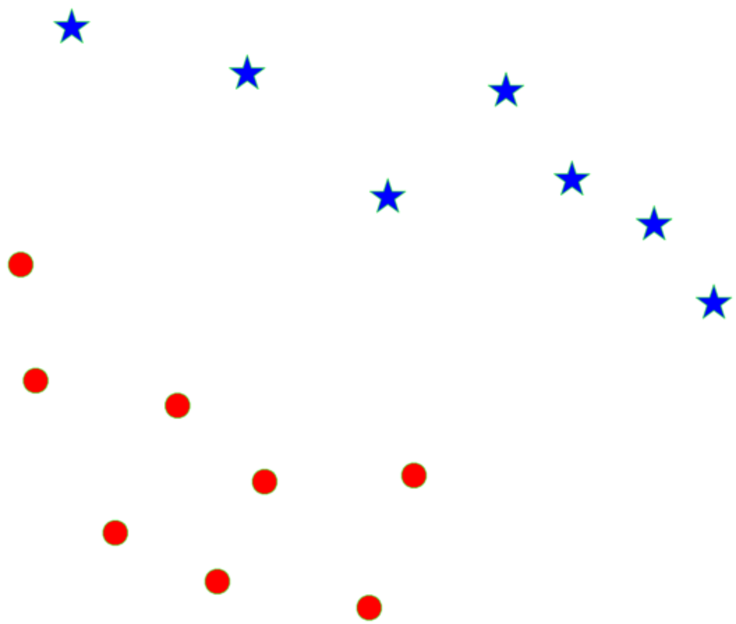
$$\boldsymbol{\mathcal{L}}_{0-1}(y,t) = \begin{cases} 0 & if\ y = t \\ 1 & if\ y \neq t \end{cases}$$

$$= \mathbb{I}\{y \neq t\}.$$

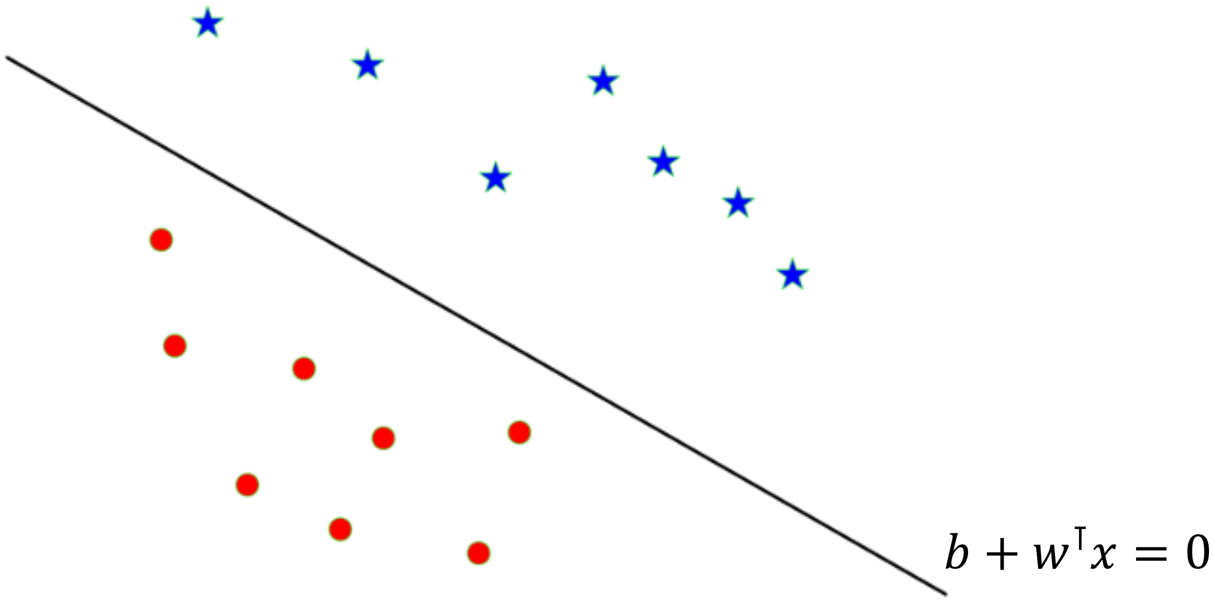
- But minimizing this loss is computationally difficult, and it can't distinguish different hypotheses that achieve the same accuracy.
- We investigated some other loss functions that are easier to minimize, e.g., logistic regression with the cross-entropy loss $\mathcal{L}_{\mathrm{CE}}$.
- Let's consider a different approach, starting from geometry of binary classifiers.

Separating Hyperplanes

Suppose we are given these data points from two different classes and want to find a linear classifier that separates them.

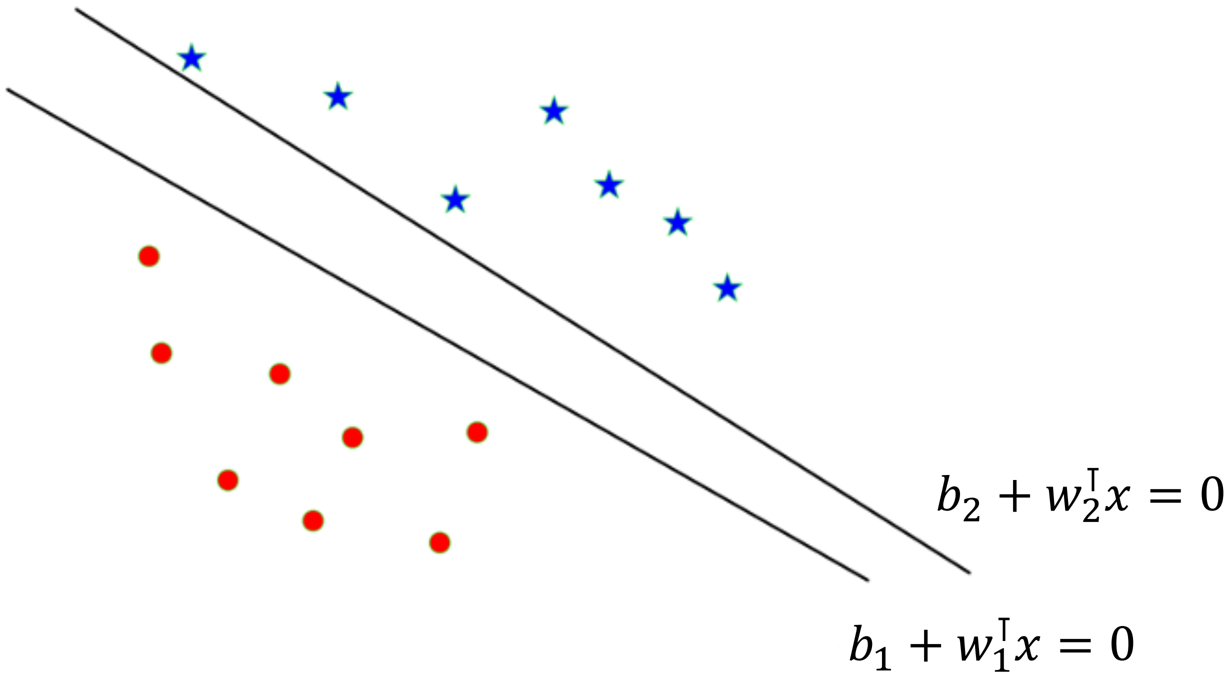


Separating Hyperplanes



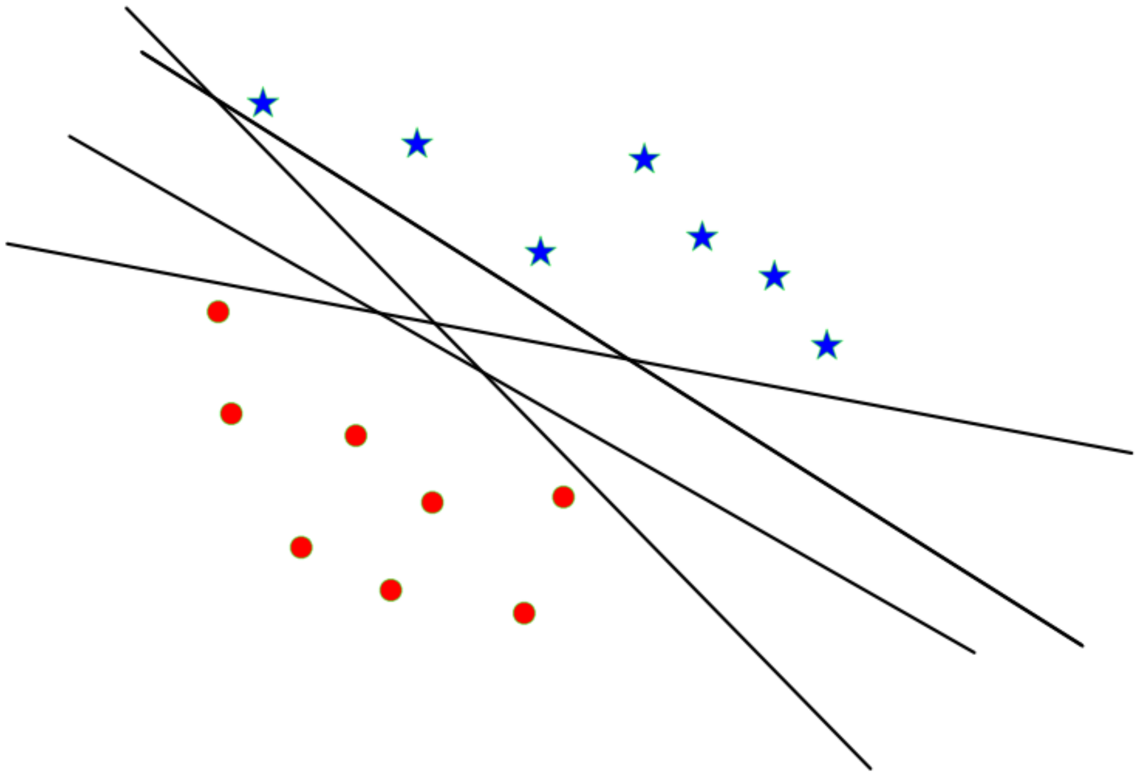
- The decision boundary looks like a line because $\mathbf{x} \in \mathbb{R}^2$, but think about it as a $D - 1$ dimensional hyperplane.
- Recall that a hyperplane is described by points $\mathbf{x} \in \mathbb{R}^D$ such that $f(\boldsymbol{x}) = \mathbf{w}^\intercal x + b = 0$.

Separating Hyperplanes



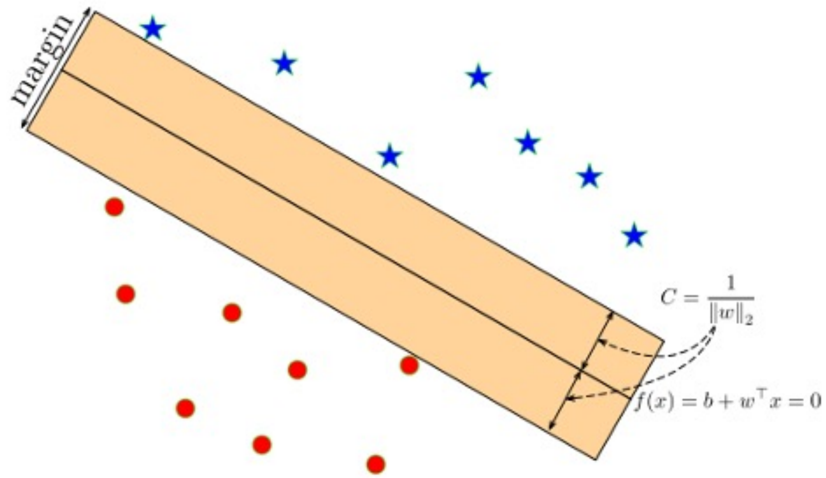
- There are multiple separating hyperplanes, described by different parameters ($\mathbf{w}$, b).

Separating Hyperplanes



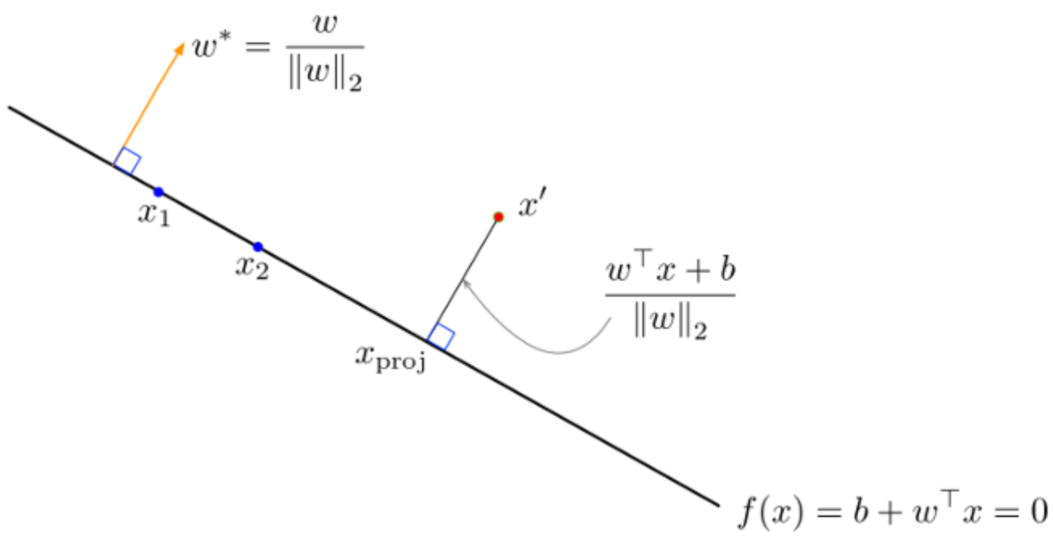
Optimal Separating Hyperplane

Optimal Separating Hyperplane: A hyperplane that separates two classes and maximizes the distance to the closest point from either class, i.e., maximize the **margin** of the classifier.



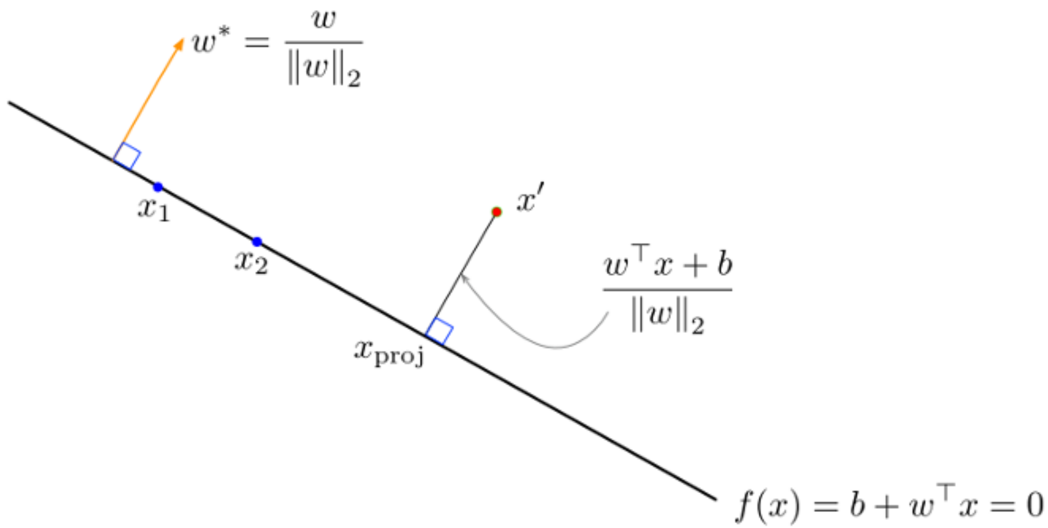
Intuitively, ensuring that a classifier is not too close to any data points leads to better generalization on the test data.

Geometry of Points and Planes



- Recall that the decision hyperplane is orthogonal (perpendicular) to $\mathbf{w}$.
- The vector $\mathbf{w}^* = \frac{\mathbf{w}}{\|\mathbf{w}\|_2}$ is a unit vector pointing in the same direction as $\mathbf{w}$.
- The same hyperplane could equivalently be defined in terms of $\mathbf{w}^*$.

Geometry of Points and Planes



The (signed) distance of a point $\mathbf{x}'$ to the hyperplane is

$$\frac{\mathbf{w}^\top \mathbf{x}' + b}{\|\mathbf{w}\|_2}$$

Maximizing Margin as an Optimization Problem

- Recall: the classification for the i-th data point is correct when

$$\operatorname{sign}\left(\mathbf{w}^\top \mathbf{x}^{(i)} + b\right) = t^{(i)}$$

- This can be rewritten as

$$t^{(i)}\left(\mathbf{w}^\top \mathbf{x}^{(i)} + b\right) > 0$$

- Enforcing a margin of C:

$$t^{(i)} \cdot \underbrace{\frac{(\mathbf{w}^\top \mathbf{x}^{(i)} + b)}{\|\mathbf{w}\|_2}}_{\text{signed distance}} \geq C$$

Maximizing Margin as an Optimization Problem

Max-margin objective:

$$\max_{\mathbf{w},\, b} C$$

$$\text{s.t.} \frac{t^{(i)}(\mathbf{w}^\top \mathbf{x}^{(i)} + b)}{\|\mathbf{w}\|_2} \geq C \qquad i = 1, \cdots, N$$

Plug in $C = 1/\|\mathbf{w}\|_2$ and simplify:

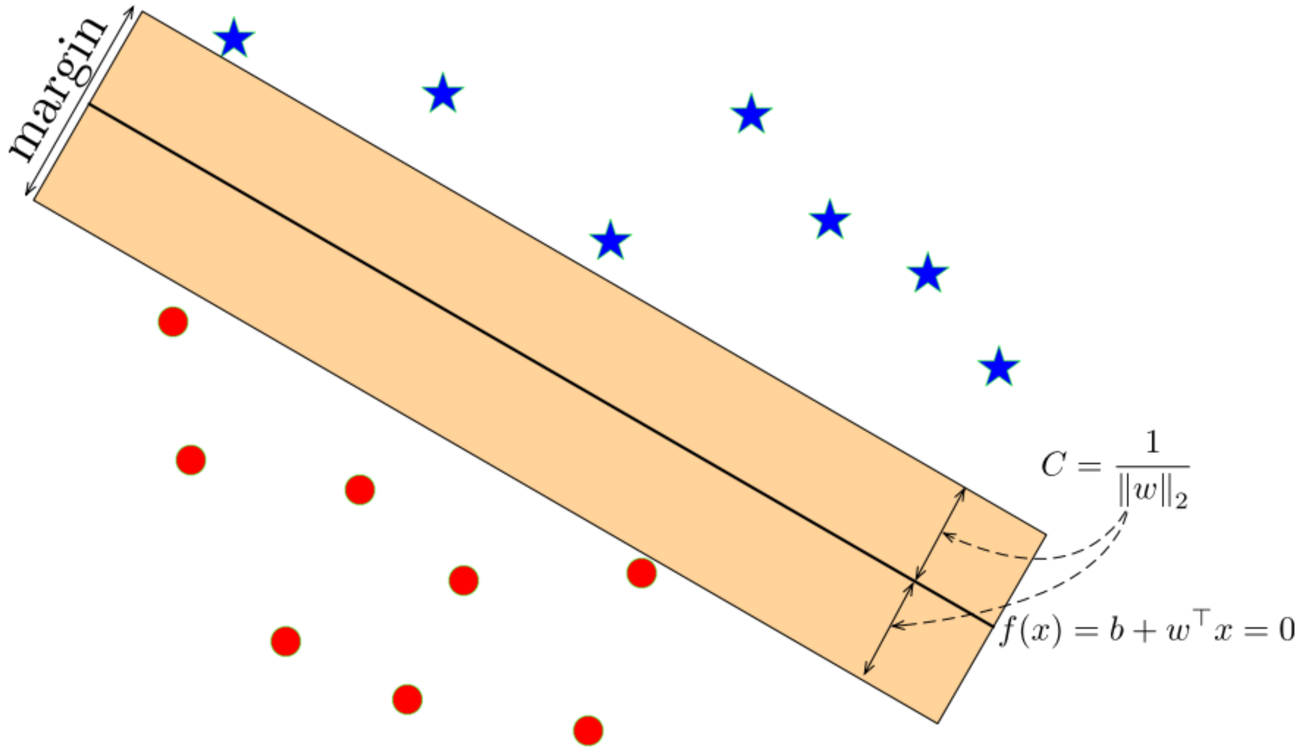
$$\underbrace{\frac{t^{(i)}(\mathbf{w}^\top \mathbf{x}^{(i)} + b)}{\|\mathbf{w}\|_2} \geq \frac{1}{\|\mathbf{w}\|_2}}_{\text{geometric margin constraint}} \iff \underbrace{t^{(i)}(\mathbf{w}^\top \mathbf{x}^{(i)} + b) \geq 1}_{\text{algebraic margin constraint}}$$

Equivalent optimization objective:

$$\min \|\mathbf{w}\|_2^2$$

$$\text{s.t. } t^{(i)}(\mathbf{w}^\top \mathbf{x}^{(i)} + b) \geq 1 \qquad i = 1, \cdots, N$$

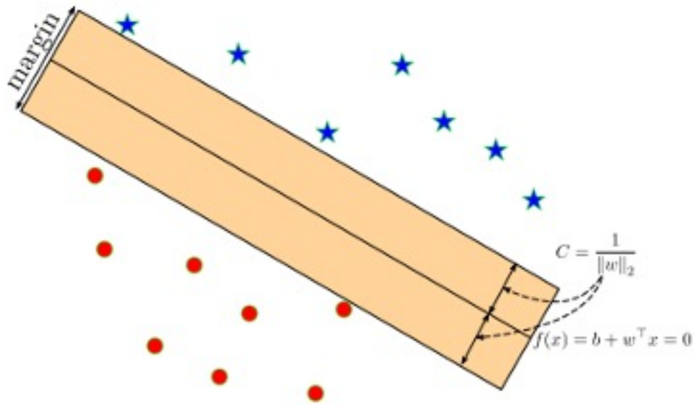
Maximizing Margin as an Optimization Problem



Maximizing Margin as an Optimization Problem

Algebraic max-margin objective:

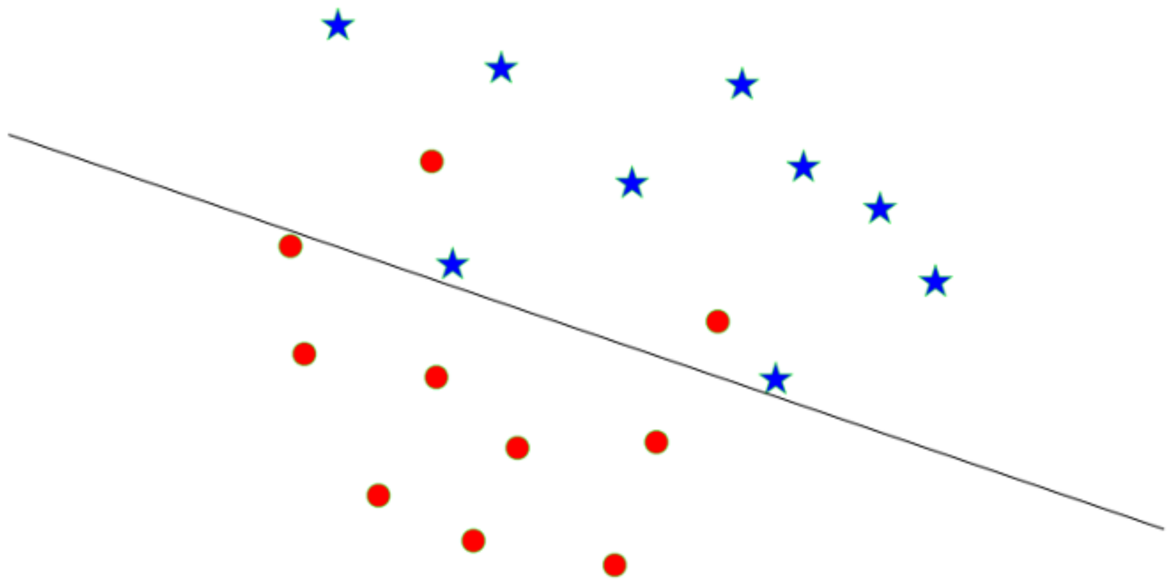
$$\min \|\mathbf{w}\|_2^2$$
$$\text{s.t. } t^{(i)}(\mathbf{w}^\top \mathbf{x}^{(i)} + b) \geq 1 \qquad i = 1, \cdots, N$$



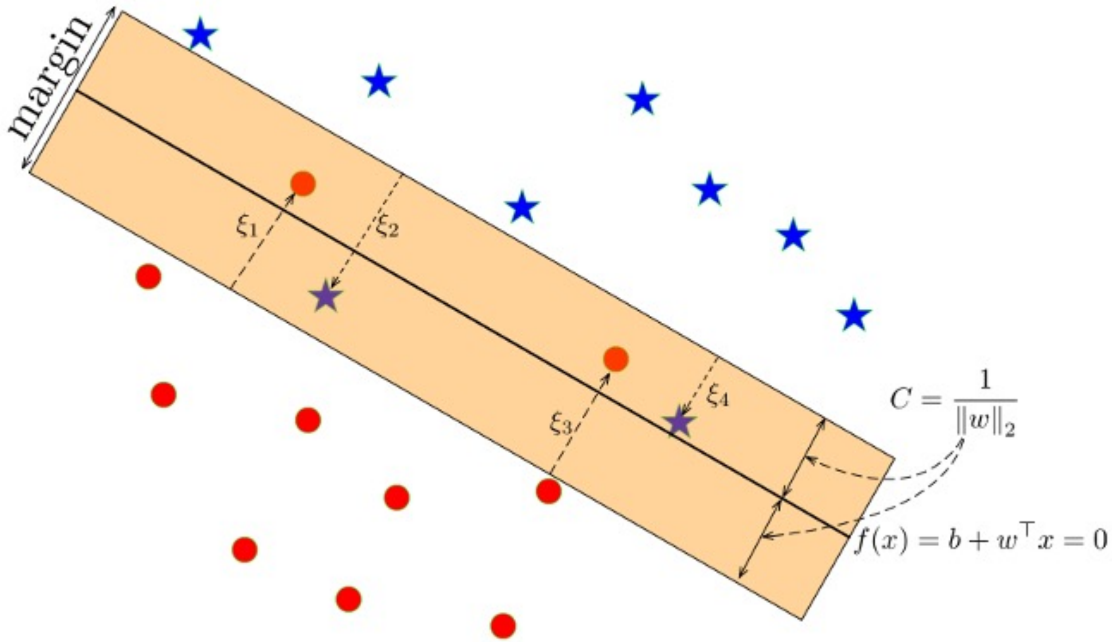
- Observe: if the margin constraint is not tight for $\mathbf{x}^{(i)}$, we could remove it from the training set and the optimal $\mathbf{w}$ would be the same.
- The important training examples are the ones with algebraic margin 1, and are called **support vectors**.
- Hence, this algorithm is called the (hard) **Support Vector Machine (SVM)** (or Support Vector Classifier).
- SVM-like algorithms are often called **max-margin** or **large-margin**.

Non-Separable Data Points

How can we apply the max-margin principle if the data are **not** linearly separable?



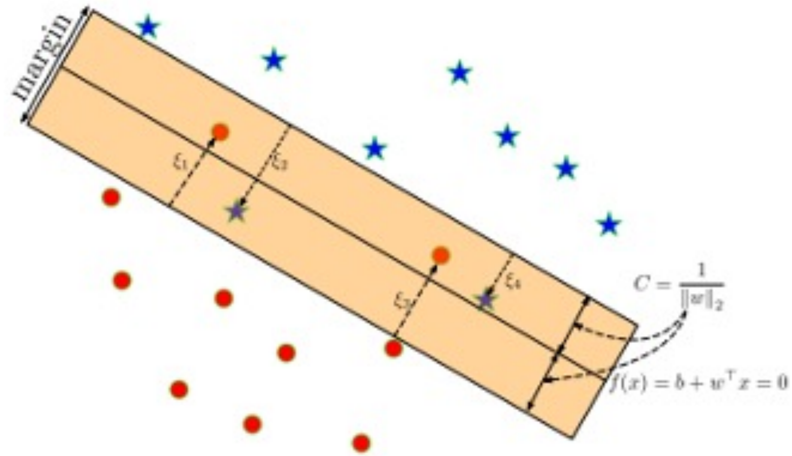
Maximizing Margin for Non-Separable Data Points



Main idea:

- Allow some points to be within the margin or even be misclassified; we represent this with **slack variables** ξ_i.
- But constrain or penalize the total amount of slack.

Maximizing Margin for Non-Separable Data Points



- **Soft margin constraint**:

$$\frac{t^{(i)}(\mathbf{w}^\top \mathbf{x}^{(i)} + b)}{\|\mathbf{w}\|_2} \geq C(1 - \xi_i),$$

for $\xi_i \geq 0$.

- Penalize $\sum_i \xi_i$.

Maximizing Margin for Non-Separable Data Points

Soft-margin SVM objective:

$$\min_{\mathbf{w},\, b,\, \xi} \frac{1}{2} \|\mathbf{w}\|_2^2 + \gamma \sum_{i=1}^{N} \xi_i$$

$$\text{s.t.}\; t^{(i)}\left(\mathbf{w}^\top \mathbf{x}^{(i)} + b\right) \geq 1 - \xi_i \qquad i = 1, \cdots, N$$

$$\xi_i \geq 0 \qquad i = 1, \cdots, N$$

- γ is a hyperparameter that trades off the margin with the amount of slack.
  - For $\gamma = 0$, we'll get $\mathbf{w} = 0$. (why?)
  - As $\gamma \to \infty$, we get the hard-margin objective.
- Note: it is also possible to constrain $\sum_i \xi_i$ instead of penalizing it.

From Margin Violation to Hinge Loss

Let's simplify the soft margin constraint by eliminating ξ_i. Recall:

$$t^{(i)}\left(\mathbf{w}^\top \mathbf{x}^{(i)} + b\right) \geq 1 - \xi_i \qquad i = 1, \cdots, N$$

$$\xi_i \geq 0 \qquad i = 1, \cdots, N$$

- Rewrite as $\xi_i \geq 1 - t^{(i)}\left(\mathbf{w}^\top \mathbf{x}^{(i)} + b\right)$.
- **Case 1**: $1 - t^{(i)}\left(\mathbf{w}^\top \mathbf{x}^{(i)} + b\right) \leq 0$
  - ➢ The smallest non-negative ξ_i that satisfies the constraint is $\xi_i = 0$.
- **Case 2**: $1 - t^{(i)}\left(\mathbf{w}^\top \mathbf{x}^{(i)} + b\right) > 0$
  - ➢ The smallest ξ_i that satisfies the constraint is $\xi_i = 1 - t^{(i)}\left(\mathbf{w}^\top \mathbf{x}^{(i)} + b\right)$.
- Hence, $\xi_i = \max\{0, 1 - t^{(i)}\left(\mathbf{w}^\top \mathbf{x}^{(i)} + b\right)\}$.
- Therefore, the slack penalty can be written as

$$\sum_{i=1}^{N} \xi_i = \sum_{i=1}^{N} \max\{0, 1 - t^{(i)}\left(\mathbf{w}^\top \mathbf{x}^{(i)} + b\right)\}$$

From Margin Violation to Hinge Loss

If we write $y^{(i)}(\mathbf{w}, b) = \mathbf{w}^\top \mathbf{x} + b$, then the optimization problem can be written as

$$\min_{\mathbf{w},\, b,\, \xi} \sum_{i=1}^{N} \max\{0, 1 - t^{(i)} y^{(i)}(\mathbf{w}, b)\} + \frac{1}{2\gamma} \|\mathbf{w}\|_2^2$$

- The loss function $\mathcal{L}_{\mathrm{H}}(y, t) = \max\{0, 1 - ty\}$ is called the **hinge** loss.
- The second term is the L_2-norm of the weights.
- Hence, the soft-margin SVM can be seen as a linear classifier with hinge loss and an L_2 regularizer.

Multiclass SVM Loss

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:

cat	**3.2**	1.3	2.2
car	5.1	**4.9**	2.5
frog	-1.7	2.0	**-3.1**

Multiclass SVM loss:

Given an example $(\boldsymbol{x}_i, \boldsymbol{y}_i)$, where $\boldsymbol{x}_i$ is the image and $\boldsymbol{y}_i$ is the (integer) label.

and using the shorthand for the scores vector: $s = f(\boldsymbol{x}_i, W)$

The SVM loss has the form:

$$\boxed{L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)}$$

Multiclass SVM Loss

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:

cat	**3.2**	1.3	2.2
car	5.1	**4.9**	2.5
frog	-1.7	2.0	**-3.1**
Losses:	2.9		

Multiclass SVM loss:

Given an example $(\boldsymbol{x}_i, \boldsymbol{y}_i)$,
where $\boldsymbol{x}_i$ is the image and $\boldsymbol{y}_i$ is the (integer) label.

and using the shorthand for the scores vector: $s = f(\boldsymbol{x}_i, W)$

The SVM loss has the form:

$$\boxed{L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)}$$

$$= \max(0, 5.1 - 3.2 + 1) + \max(0, -1.7 - 3.2 + 1)$$
$$= \max(0, 2.9) + \max(0, -3.9)$$
$$= 2.9 + 0$$
$$= 2.9$$

Multiclass SVM Loss

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:

cat	**3.2**	1.3	2.2
car	5.1	**4.9**	2.5
frog	-1.7	2.0	**-3.1**
Losses:	2.9	0	

Multiclass SVM loss:

Given an example $(\boldsymbol{x}_i, \boldsymbol{y}_i)$, where $\boldsymbol{x}_i$ is the image and $\boldsymbol{y}_i$ is the (integer) label.

and using the shorthand for the scores vector: $s = f(\boldsymbol{x}_i, W)$

The SVM loss has the form:

$$\boxed{L_i = \sum_{j \neq \boldsymbol{y}_i} \max(0, s_j - s_{\boldsymbol{y}_i} + 1)}$$

$$= \max(0, 1.3 - 4.9 + 1)$$
$$+ \max(0, 2.0 - 4.9 + 1)$$
$$= \max(0, -2.6) + \max(0, -1.9)$$
$$= 0 + 0$$
$$= 0$$

Multiclass SVM Loss

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:

cat	**3.2**	1.3	2.2
car	5.1	**4.9**	2.5
frog	-1.7	2.0	**-3.1**
Losses:	2.9	0	12.9

Multiclass SVM loss:

Given an example $(\boldsymbol{x}_i, \boldsymbol{y}_i)$, where $\boldsymbol{x}_i$ is the image and $\boldsymbol{y}_i$ is the (integer) label.

and using the shorthand for the scores vector: $s = f(\boldsymbol{x}_i, W)$

The SVM loss has the form:

$$\boxed{L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)}$$

$$\begin{aligned}
&= \max(0, 2.2 - (-3.1) + 1) \\
&\quad + \max(0, 2.5 - (-3.1) + 1) \\
&= \max(0, 6.3) + \max(0, 6.6) \\
&= 6.3 + 6.6 \\
&= 12.9
\end{aligned}$$

Multiclass SVM Loss

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:

cat	**3.2**	1.3	2.2
car	5.1	**4.9**	2.5
frog	-1.7	2.0	**-3.1**
Losses:	2.9	0	12.9

Multiclass SVM loss:

Given an example $(\boldsymbol{x}_i, \boldsymbol{y}_i)$, where $\boldsymbol{x}_i$ is the image and $\boldsymbol{y}_i$ is the (integer) label.

and using the shorthand for the scores vector: $s = f(\boldsymbol{x}_i, W)$

The SVM loss has the form:

$$\boxed{L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)}$$

and the full training loss is the mean over all examples in the training data:

$$L = \frac{1}{N} \sum_{i=1}^{N} L_i$$

$$L = (2.9 + 0 + 12.9)/3$$
$$= \mathbf{5.27}$$

Multiclass SVM Loss

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:

cat	**3.2**	1.3	2.2
car	5.1	**4.9**	2.5
frog	-1.7	2.0	**-3.1**
Losses:	2.9	0	12.9

Multiclass SVM loss:

Given an example $(\boldsymbol{x}_i, \boldsymbol{y}_i)$,
where $\boldsymbol{x}_i$ is the image and $\boldsymbol{y}_i$ is the (integer) label.

and using the shorthand for the scores vector: $s = f(\boldsymbol{x}_i, W)$

The SVM loss has the form:

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

Q: what if the sum was instead over all classes? (including j=y_i)

Multiclass SVM Loss

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:

cat	**3.2**	1.3	2.2
car	5.1	**4.9**	2.5
frog	-1.7	2.0	**-3.1**
Losses:	2.9	0	12.9

Multiclass SVM loss:

Given an example $(\boldsymbol{x}_i, \boldsymbol{y}_i)$,
where $\boldsymbol{x}_i$ is the image and $\boldsymbol{y}_i$ is the (integer) label.

and using the shorthand for the scores vector: $s = f(\boldsymbol{x}_i, W)$

The SVM loss has the form:

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

Q2: what if we used a mean instead of a sum here

Multiclass SVM Loss

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:

cat	**3.2**	1.3	2.2
car	5.1	**4.9**	2.5
frog	-1.7	2.0	**-3.1**
Losses:	2.9	0	12.9

Multiclass SVM loss:

Given an example $(\boldsymbol{x}_i, \boldsymbol{y}_i)$,
where $\boldsymbol{x}_i$ is the image and $\boldsymbol{y}_i$ is the (integer) label.

and using the shorthand for the scores vector: $s = f(\boldsymbol{x}_i, W)$

The SVM loss has the form:

$$L_i = \sum_{j \neq \boldsymbol{y}_i} \max(0, s_j - s_{\boldsymbol{y}_i} + 1)$$

Q3: what if we used

$$L_i = \sum_{j \neq \boldsymbol{y}_i} \max(0, s_j - s_{\boldsymbol{y}_i} + 1)^2$$

Multiclass SVM Loss

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:

cat	**3.2**	1.3	2.2
car	5.1	**4.9**	2.5
frog	-1.7	2.0	**-3.1**
Losses:	2.9	0	12.9

Multiclass SVM loss:

Given an example $(\boldsymbol{x}_i, \boldsymbol{y}_i)$, where $\boldsymbol{x}_i$ is the image and $\boldsymbol{y}_i$ is the (integer) label.

and using the shorthand for the scores vector: $s = f(\boldsymbol{x}_i, W)$

The SVM loss has the form:

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

Q4: what is the min/max possible loss

Multiclass SVM Loss

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:



cat	**3.2**	1.3	2.2
car	5.1	**4.9**	2.5
frog	-1.7	2.0	**-3.1**
Losses:	2.9	0	12.9

Multiclass SVM loss:

Given an example $(\boldsymbol{x}_i, \boldsymbol{y}_i)$,
where $\boldsymbol{x}_i$ is the image and $\boldsymbol{y}_i$ is the (integer) label.

and using the shorthand for the scores vector: $s = f(\boldsymbol{x}_i, W)$

The SVM loss has the form:

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

Q5: usually at initialization W are small numbers so all s ~=0. What is the loss?

Softmax

Softmax Classifier (Multinomial Logistic Regression)

cat	**3.2**
car	5.1
frog	-1.7

Softmax

Softmax Classifier (Multinomial Logistic Regression)

scores = unnormalized log probabilities of the classes.

$$\boxed{s = f(\boldsymbol{x}_i; W)}$$

cat	**3.2**
car	5.1
frog	-1.7

Softmax

Softmax Classifier (Multinomial Logistic Regression)

scores = unnormalized log probabilities of the classes.

$$\boxed{P(Y = k|X = \boldsymbol{x}_i) = \frac{e^{s_k}}{\sum_j e^{s_j}}} \text{ where } \boxed{s = f(\boldsymbol{x}_i; W)}$$

cat	**3.2**
car	5.1
frog	-1.7

Softmax

Softmax Classifier (Multinomial Logistic Regression)

scores = unnormalized log probabilities of the classes.

$$\boxed{P(Y=k|X=\boldsymbol{x}_i)=\boxed{\frac{e^{s_k}}{\sum_j e^{s_j}}}}\ \text{where}\ \boxed{s=f(\boldsymbol{x}_i;W)}$$

Softmax function

cat	**3.2**
car	5.1
frog	-1.7

Softmax

Softmax Classifier (Multinomial Logistic Regression)

scores = unnormalized log probabilities of the classes.

$$\boxed{P(Y=k|X=\boldsymbol{x}_i)=\frac{e^{s_k}}{\sum_j e^{s_j}}} \text{ where } \boxed{s=f(\boldsymbol{x}_i;W)}$$

cat	**3.2**
car	5.1
frog	-1.7

Want to maximize the log likelihood, or (for a loss function) to minimize the negative log likelihood of the correct class:

$$\boxed{L_i=-\log P(Y=\boldsymbol{y}_i|X=\boldsymbol{x}_i)}$$

Softmax

Softmax Classifier (Multinomial Logistic Regression)

cat	**3.2**
car	5.1
frog	-1.7

unnormalized log probabilities

$$L_i = -\log\left(\frac{e^{s_{y_i}}}{\sum_j e^{s_j}}\right)$$

Softmax

Softmax Classifier (Multinomial Logistic Regression)

$$L_i = -\log\left(\frac{e^{s_{y_i}}}{\sum_j e^{s_j}}\right)$$

unnormalized probabilities

	unnormalized log probabilities	exp	unnormalized probabilities
cat	**3.2**	→	**24.5**
car	5.1	→	164.0
frog	-1.7	→	0.18

Softmax

Softmax Classifier (Multinomial Logistic Regression)

$$L_i = -\log\left(\frac{e^{s_{y_i}}}{\sum_j e^{s_j}}\right)$$

	unnormalized log probabilities	$\xrightarrow{\text{exp}}$	unnormalized probabilities	$\xrightarrow{\text{normalize}}$	probabilities
cat	**3.2**		**24.5**		**0.13**
car	5.1		164.0		0.87
frog	-1.7		0.18		0.00

Softmax

Softmax Classifier (Multinomial Logistic Regression)

$$L_i = -\log\left(\frac{e^{s_{y_i}}}{\sum_j e^{s_j}}\right)$$

unnormalized probabilities

	unnormalized log probabilities	exp → unnormalized probabilities	normalize → probabilities
cat	**3.2**	**24.5**	**0.13**
car	5.1	164.0	0.87
frog	-1.7	0.18	0.00

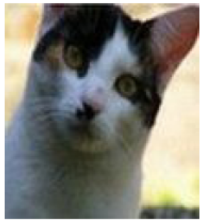
unnormalized log probabilities

probabilities

→ L_i = -log(0.13)
= **2.04**

Softmax

Softmax Classifier (Multinomial Logistic Regression)



$$L_i = -\log\left(\frac{e^{s_{y_i}}}{\sum_j e^{s_j}}\right)$$

Q: What is the min/max possible loss L_i?

unnormalized probabilities

	unnormalized log probabilities		unnormalized probabilities		probabilities
cat	**3.2**	exp →	**24.5**	normalize →	**0.13**
car	5.1		164.0		0.87
frog	-1.7		0.18		0.00

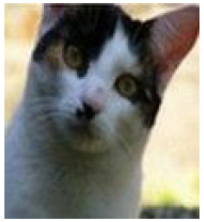
→ L_i = -log(0.13)
= **2.04**

unnormalized log probabilities

probabilities

Softmax

Softmax Classifier (Multinomial Logistic Regression)



$$L_i = -\log\left(\frac{e^{s_{y_i}}}{\sum_j e^{s_j}}\right)$$

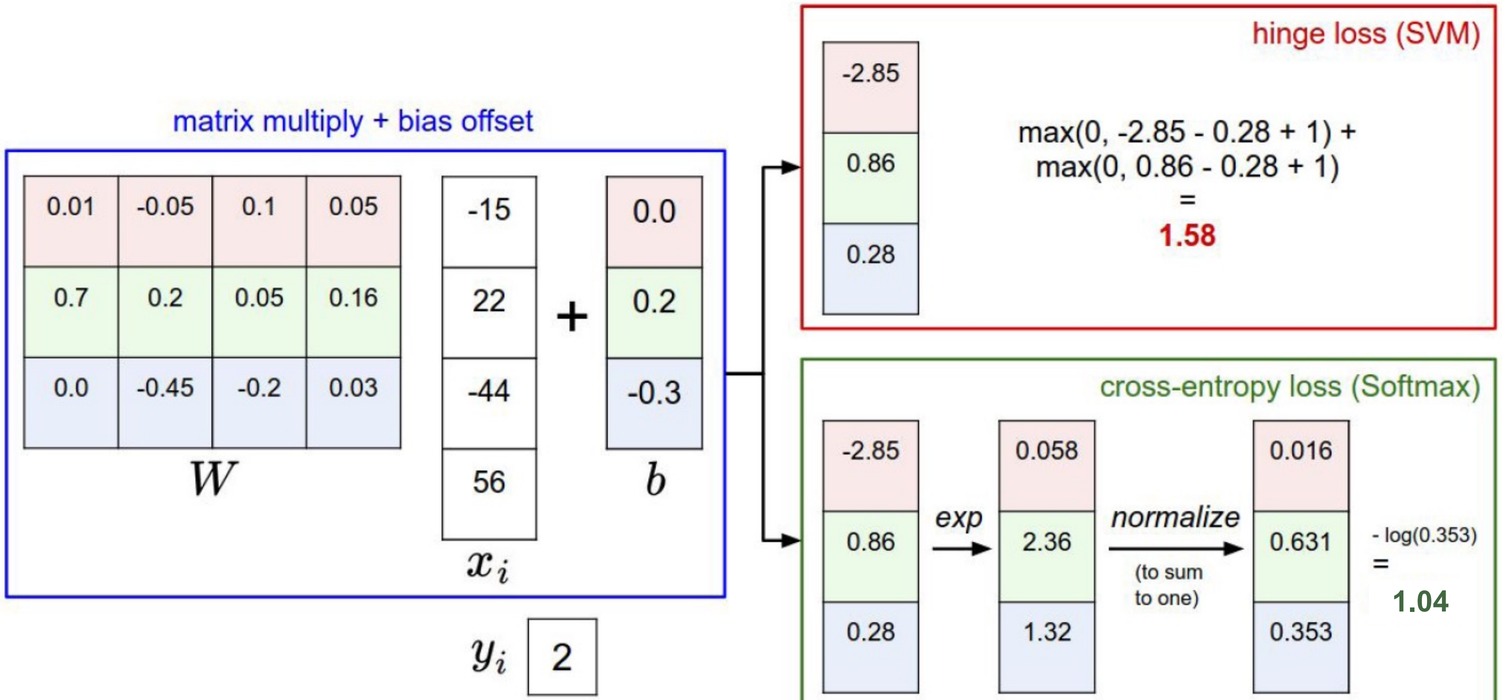
Q5: usually at initialization W are small numbers, so all s ~= 0. What is the loss?

	unnormalized log probabilities	unnormalized probabilities	probabilities
cat	**3.2**	**24.5**	**0.13**
car	5.1	164.0	0.87
frog	-1.7	0.18	0.00

exp → normalize →

L_i = -log(0.13)
= **2.04**

SVM & Softmax



matrix multiply + bias offset
0.01 -0.05 0.1 0.05
0.7 0.2 0.05 0.16
0.0 -0.45 -0.2 0.03
W
-15
22
-44
56
x_i
+
0.0
0.2
-0.3
b
y_i 2
hinge loss (SVM)
-2.85
0.86
0.28
max(0, -2.85 - 0.28 + 1) +
max(0, 0.86 - 0.28 + 1)
=
1.58
cross-entropy loss (Softmax)
-2.85
0.86
0.28
exp
0.058
2.36
1.32
normalize
(to sum
to one)
0.016
0.631
0.353
- log(0.353)
=
1.04

SVM & Softmax

Softmax vs. SVM

$$L_i = -\log\left(\frac{e^{s_{y_i}}}{\sum_j e^{s_j}}\right)$$

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

SVM & Softmax

Softmax vs. SVM

$$L_i = -\log\left(\frac{e^{s_{y_i}}}{\sum_j e^{s_j}}\right)$$

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

assume scores:
[10, -2, 3]
[10, 9, 9]
[10, -100, -100]
and $\boxed{y_i = 0}$

Q: suppose I take a datapoint and I jiggle a bit (changing its score slightly). What happens to the loss in both cases?