

Neurónové siete pre počítačové videnie

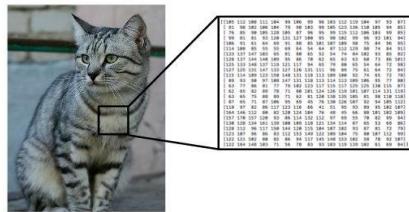
šk.r. 2021-22

Stratové funkcie
a optimalizácia

RNDr. Zuzana Černeková, PhD.
Ing. Viktor Kocur, PhD.

Opakovanie: Výzvy pri rozpoznávaní

Pohľad



Osvetlenie



Deformácie



Zakrytie



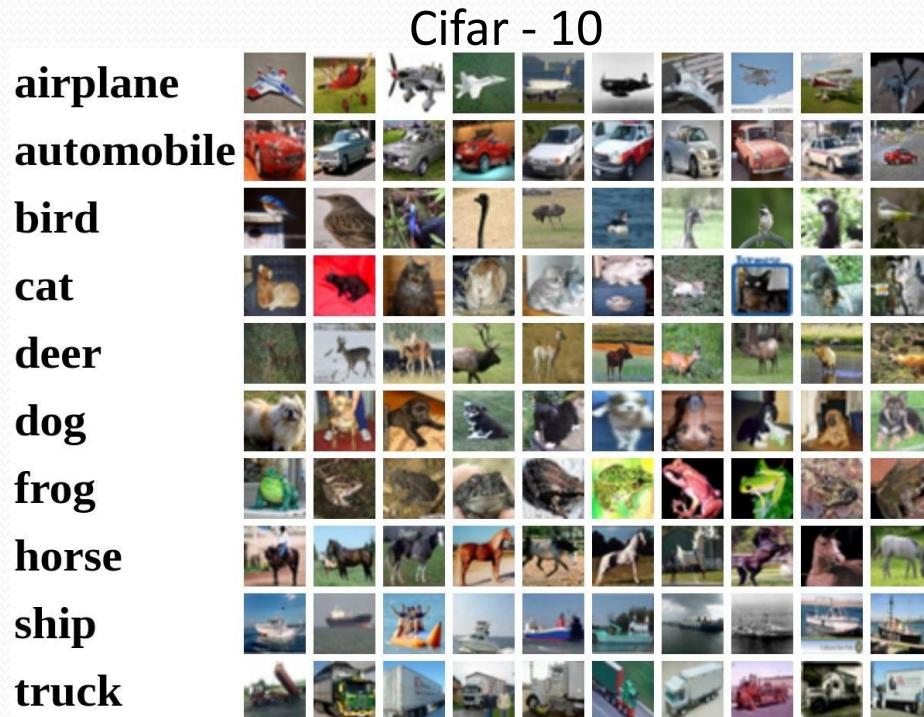
Výrazné pozadie



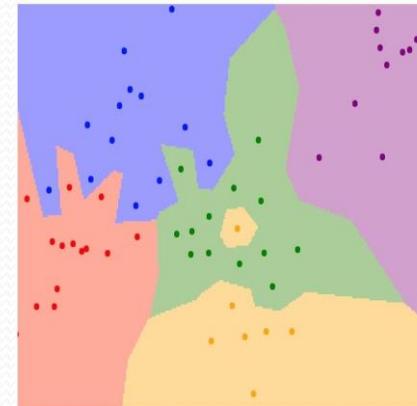
Vnútrotriedna variácia



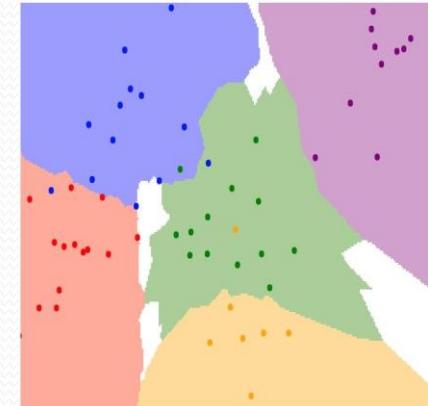
Opakovanie: dátovo riadené metódy, kNN



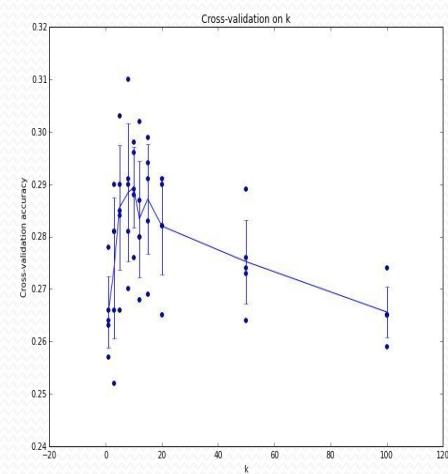
1-NN classifier



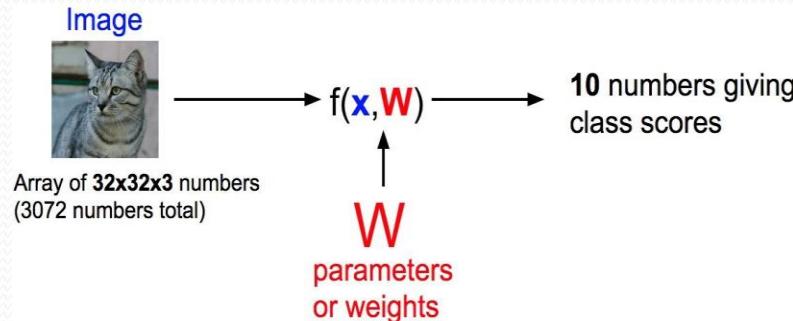
5-NN classifier



Cross validation



Opakovanie: lineárny klasifikátor



$$f(x, W) = Wx + b$$

Algebraic Viewpoint

$$f(x, W) = Wx$$

Stretch pixels into column

Input image	56	231	24	2
	0.2	-0.5	0.1	2.0
	1.5	1.3	2.1	0.0
	0	0.25	0.2	-0.3

W

Wx

$+ b$

Cat score: -96.8
Dog score: 437.9
Ship score: 61.95

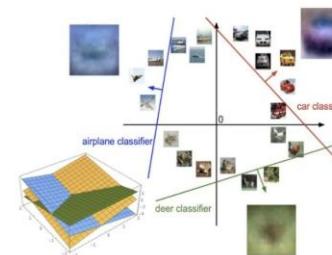
Visual Viewpoint

One template per class



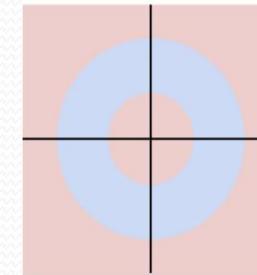
Geometric Viewpoint

Hyperplanes cutting up space



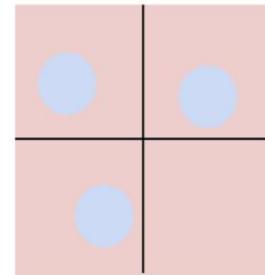
Class 1:
 $1 \leq L_2 \text{ norm} \leq 2$

Class 2:
Everything else



Class 1:
Three modes

Class 2:
Everything else



Opakovanie: lineárny klasifikátor



airplane	-3.45	-0.51	3.42
automobile	-8.87	6.04	4.64
bird	0.09	5.31	2.65
cat	2.9	-4.22	5.1
deer	4.48	-4.19	2.64
dog	8.02	3.58	5.55
frog	3.78	4.49	-4.34
horse	1.06	-4.37	-1.5
ship	-0.36	-2.09	-4.79
truck	-0.72	-2.93	6.14

Štruktúra dnešnej prednášky

- Definovať **stratovú funkciu (loss function)**, ktorá kvantifikuje našu nespokojnosť so skóre na trénovacích dátach
- Nájsť spôsob ako efektívne hľadať parametre, ktoré minimalizujú stratovú funkciu - **optimalizácia**

Predpokladajme: 3 trénovacie vzorky, 3 triedy.

Majme nejakú maticu W dostaneme skóre $f(x, W) = Wx$



mačka	3.2	1.3	2.2
auto	5.1	4.9	2.5
žaba	-1.7	2.0	-3.1

Predpokladajme: 3 trénovacie vzorky, 3 triedy.
Pre nejakú maticu W dostaneme skóre

$$f(x, W) = Wx$$



mačka	3.2	1.3	2.2
auto	5.1	4.9	2.5
žaba	-1.7	2.0	-3.1

Stratová funkcia (loss function)
hovorí ako dobrý je nás aktuálny klasifikátor

Majme databázu vzoriek

$$\{(x_i, y_i)\}_{i=1}^N$$

Kde x_i je obraz a
 y_i je (integer) label

Strata na celej databáze je suma strat na všetkých vzorkách:

$$L = \frac{1}{N} \sum_i L_i(f(x_i, W), y_i)$$

Predpokladajme: 3 trénovacie vzorky, 3 triedy.
Pre nejakú maticu W dostaneme skóre

$$f(x, W) = Wx$$



mačka	3.2	1.3	2.2
auto	5.1	4.9	2.5
žaba	-1.7	2.0	-3.1

Strata pre Multiclass SVM:

Pre danú vzorku (x_i, y_i)
kde x_i je obraz a
kde y_i je the (integer) label,

A použitím skratky s pre vektor ohodnotení (score vector):

$$s = f(x_i, W)$$

Strata pre SVM je potom v tvare:

$$\begin{aligned} L_i &= \sum_{j \neq y_i} \begin{cases} 0 & \text{if } s_{y_i} \geq s_j + 1 \\ s_j - s_{y_i} + 1 & \text{otherwise} \end{cases} \\ &= \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1) \end{aligned}$$

Strata pre Multiclass SVM:

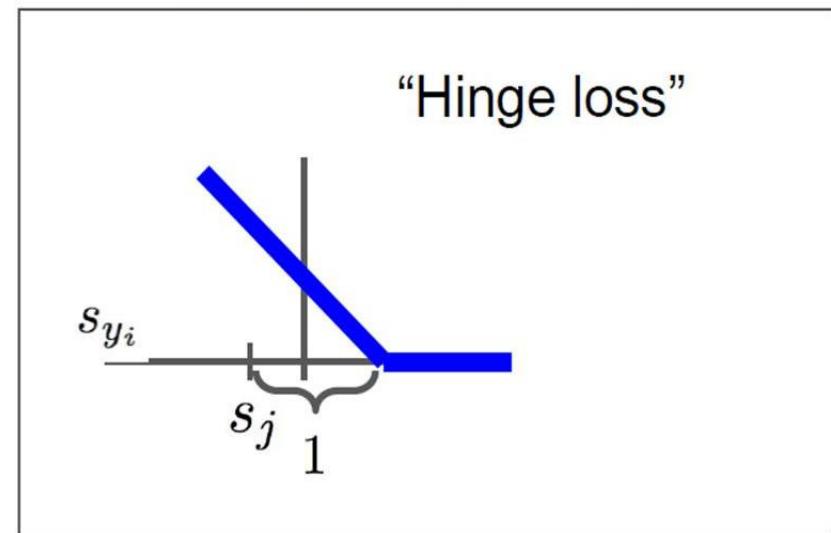
Predpokladajme: 3 trénovacie vzorky, 3 triedy.

Pre nejakú maticu W dostaneme skóre

$$f(x, W) = Wx$$



mačka	3.2	1.3	2.2
auto	5.1	4.9	2.5
žaba	-1.7	2.0	-3.1



$$\begin{aligned} L_i &= \sum_{j \neq y_i} \begin{cases} 0 & \text{if } s_{y_i} \geq s_j + 1 \\ s_j - s_{y_i} + 1 & \text{otherwise} \end{cases} \\ &= \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1) \end{aligned}$$

Predpokladajme: 3 trénovacie vzorky, 3 triedy.
Pre nejakú maticu W dostaneme skóre

$$f(x, W) = Wx$$



mačka	3.2	1.3	2.2
auto	5.1	4.9	2.5
žaba	-1.7	2.0	-3.1

Strata pre Multiclass SVM:

Pre danú vzorku (x_i, y_i)
kde x_i je obraz a
kde y_i je the (integer) label,

A použitím skratky s pre vektor ohodnotení (score vector):

$$s = f(x_i, W)$$

Strata pre SVM je potom v tvare:

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

Predpokladajme: 3 trénovacie vzorky, 3 triedy.
Pre nejakú maticu W dostaneme skóre

$$f(x, W) = Wx$$



mačka	3.2	1.3	2.2
auto	5.1	4.9	2.5
žaba	-1.7	2.0	-3.1
Strata	2.9		

Strata pre Multiclass SVM:

Pre danú vzorku (x_i, y_i)
kde x_i je obraz a
kde y_i je the (integer) label,

A použitím skratky s pre vektor ohodnotení (score vector):

$$s = f(x_i, W)$$

Strata pre SVM je potom v tvare:

$$\begin{aligned} L_i &= \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1) \\ &= \max(0, 5.1 - 3.2 + 1) \\ &\quad + \max(0, -1.7 - 3.2 + 1) \\ &= \max(0, 2.9) + \max(0, -3.9) \\ &= 2.9 + 0 \\ &= 2.9 \end{aligned}$$

Predpokladajme: 3 trénovacie vzorky, 3 triedy.
Pre nejakú maticu W dostaneme skóre

$$f(x, W) = Wx$$



mačka	3.2	1.3	2.2
auto	5.1	4.9	2.5
žaba	-1.7	2.0	-3.1
Strata	2.9	0	

Strata pre Multiclass SVM:

Pre danú vzorku (x_i, y_i)
kde x_i je obraz a
kde y_i je the (integer) label,

A použitím skratky s pre vektor ohodnotení (score vector):

$$s = f(x_i, W)$$

Strata pre SVM je potom v tvare:

$$\begin{aligned}
 L_i &= \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1) \\
 &= \max(0, 1.3 - 4.9 + 1) \\
 &\quad + \max(0, 2.0 - 4.9 + 1) \\
 &= \max(0, -2.6) + \max(0, -1.9) \\
 &= 0 + 0 \\
 &= 0
 \end{aligned}$$

Predpokladajme: 3 trénovacie vzorky, 3 triedy.
Pre nejakú maticu W dostaneme skóre

$$f(x, W) = Wx$$



mačka	3.2	1.3	2.2
auto	5.1	4.9	2.5
žaba	-1.7	2.0	-3.1
Strata	2.9	0	12.9

Strata pre Multiclass SVM:

Pre danú vzorku (x_i, y_i)
kde x_i je obraz a
kde y_i je the (integer) label,

A použitím skratky s pre vektor ohodnotení (score vector):

$$s = f(x_i, W)$$

Strata pre SVM je potom v tvare:

$$\begin{aligned} L_i &= \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1) \\ &= \max(0, 2.2 - (-3.1) + 1) \\ &\quad + \max(0, 2.5 - (-3.1) + 1) \\ &= \max(0, 6.3) + \max(0, 6.6) \\ &= 6.3 + 6.6 \\ &= 12.9 \end{aligned}$$

Predpokladajme: 3 trénovacie vzorky, 3 triedy.
Pre nejakú maticu W dostaneme skóre

$$f(x, W) = Wx$$



mačka	3.2	1.3	2.2
auto	5.1	4.9	2.5
žaba	-1.7	2.0	-3.1
Strata	2.9	0	12.9

Strata pre Multiclass SVM:

Pre danú vzorku (x_i, y_i)
kde x_i je obraz a
kde y_i je the (integer) label,

A použitím skratky s pre vektor ohodnotení (score vector):

$$s = f(x_i, W)$$

Strata pre SVM je potom v tvare:

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

Strata pre celú databázu je priemer:

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

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



Strata pre SVM má tvar:

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

mačka	3.2	1.3	2.2
auto	5.1	4.9	2.5
žaba	-1.7	2.0	-3.1
Strata	2.9	0	12.9

Q1: Čo sa stane s hodnotou Straty ak skóre pre auto trochu pomeníme?



Strata pre SVM má tvar:

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

mačka	3.2	1.3	2.2
auto	5.1	4.9	2.5
žaba	-1.7	2.0	-3.1
Strata	2.9	0	12.9

Q1: Čo sa stane s hodnotou Straty ak skóre pre auto trochu pomeníme?

A: Strata (Loss) sa nezmení.



Strata pre SVM má tvar:

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

mačka	3.2	1.3	2.2
auto	5.1	4.9	2.5
žaba	-1.7	2.0	-3.1
Strata	2.9	0	12.9

Q2: Aké hodnoty min/max nadobúda Strata?



Strata pre SVM má tvar:

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

mačka	3.2	1.3	2.2
auto	5.1	4.9	2.5
žaba	-1.7	2.0	-3.1
Strata	2.9	0	12.9

Q2: Aké hodnoty min/max nadobúda Strata?

A: Min = 0, Max = ∞



Strata pre SVM má tvar:

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

mačka	3.2	1.3	2.2
auto	5.1	4.9	2.5
žaba	-1.7	2.0	-3.1
Strata	2.9	0	12.9

Q3: Pri inicializácii W je malé tak, že všetky $s \approx 0$. Aká bude Strata?



Strata pre SVM má tvar:

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

mačka	3.2	1.3	2.2
auto	5.1	4.9	2.5
žaba	-1.7	2.0	-3.1
Strata	2.9	0	12.9

Q3: Pri inicializácii W je malé tak, že všetky $s \approx 0$. Aká bude Strata?

A: Počet tried minus 1. (Užitočné pri debugovaní.)



Strata pre SVM má tvar:

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

mačka	3.2	1.3	2.2
auto	5.1	4.9	2.5
žaba	-1.7	2.0	-3.1
Strata	2.9	0	12.9

Q4: Čo ak by bola Strata počítaná cez všetky triedy?
(vrátane $j=y_i$)



Strata pre SVM má tvar:

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

mačka	3.2	1.3	2.2
auto	5.1	4.9	2.5
žaba	-1.7	2.0	-3.1
Strata	2.9	0	12.9

Q4: Čo ak by bola Strata počítaná cez všetky triedy?
(vrátane $j=y_i$)

A: Hodnota Straty sa zvýši o 1.



Strata pre SVM má tvar:

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

mačka	3.2	1.3	2.2
auto	5.1	4.9	2.5
žaba	-1.7	2.0	-3.1
Strata	2.9	0	12.9

Q5: Čo ak použijeme priemer (mean) namiesto sumy?



Strata pre SVM má tvar:

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

mačka	3.2	1.3	2.2
auto	5.1	4.9	2.5
žaba	-1.7	2.0	-3.1
Strata	2.9	0	12.9

Q5: Čo ak použijeme priemer (mean) namiesto sumy?

A: Iba preškálujeme výslednú hodnotu, takže na výsledok to nemá vplyv.



Strata pre SVM má tvar:

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

mačka	3.2	1.3	2.2
auto	5.1	4.9	2.5
žaba	-1.7	2.0	-3.1
Strata	2.9	0	12.9

Q6: Čo ak použijeme $L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)^2$



Strata pre SVM má tvar:

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

mačka	3.2	1.3	2.2
auto	5.1	4.9	2.5
žaba	-1.7	2.0	-3.1
Strata	2.9	0	12.9

Q6: Čo ak použijeme $L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)^2$

A: Dostaneme iný klasifikačný algoritmus. Meníme pomer medzi dobrým a zlým skóre nelineárnym spôsobom, takže počítame inú Stratovú funkciu.

$$f(x, W) = Wx$$

$$L = \frac{1}{N} \sum_{i=1}^N \sum_{j \neq y_i} \max(0, f(x_i; W)_j - f(x_i; W)_{y_i} + 1)$$

Predpokladajme, že sme našli W také, že $L = 0$.
Je toto W jedinečné?

$$f(x, W) = Wx$$

$$L = \frac{1}{N} \sum_{i=1}^N \sum_{j \neq y_i} \max(0, f(x_i; W)_j - f(x_i; W)_{y_i} + 1)$$

Predpokladajme, že sme našli W také, že $L = 0$.
Je toto W jedinečné?

Nie! $2W$ má tiež $L = 0$.

Predpokladajme: 3 trénovacie vzorky, 3 triedy.
Pre nejakú maticu W dostaneme skóre

$$f(x, W) = Wx$$



mačka	3.2	1.3	2.2
auto	5.1	4.9	2.5
žaba	-1.7	2.0	-3.1
Strata	2.9	0	12.9

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

Before:

$$\begin{aligned}
 &= \max(0, 1.3 - 4.9 + 1) \\
 &\quad + \max(0, 2.0 - 4.9 + 1) \\
 &= \max(0, -2.6) + \max(0, -1.9) \\
 &= 0 + 0 \\
 &= 0
 \end{aligned}$$

With W twice as large:

$$\begin{aligned}
 &= \max(0, 2.6 - 9.8 + 1) \\
 &\quad + \max(0, 4.0 - 9.8 + 1) \\
 &= \max(0, -6.2) + \max(0, -4.8) \\
 &= 0 + 0 \\
 &= 0
 \end{aligned}$$

$$f(x, W) = Wx$$

$$L = \frac{1}{N} \sum_{i=1}^N \sum_{j \neq y_i} \max(0, f(x_i; W)_j - f(x_i; W)_{y_i} + 1)$$

Predpokladajme, že sme našli W také, že $L = 0$.
Je toto W jedinečné?

Nie! $2W$ má tiež $L = 0$.

Ako vybrať medzi W a $2W$?

Regularizácia

$$L(W) = \frac{1}{N} \sum_{i=1}^N L_i(f(x_i, W), y_i)$$

Strata dát: Model by mal zodpovedať trénovacím dátam

Regularizácia

$$L(W) = \underbrace{\frac{1}{N} \sum_{i=1}^N L_i(f(x_i, W), y_i)}_{\text{Strata dát}} + \lambda R(W)$$

Strata dát: Model by mal zodpovedať trénovacím dátam



Regularizácia: Zabráni aby sa model natrénoval „príliš dobre“ na trénovacích dátach

Regularizácia

λ - sila regularizácie
hyperparameter

$$L(W) = \frac{1}{N} \sum_{i=1}^N L_i(f(x_i, W), y_i) + \lambda R(W)$$

Strata dát: Model by mal zodpovedať trénovacím dátam



Regularizácia: Zabráni aby sa model natrénoval „príliš dobre“ na trénovacích dátach



Regularizácia

λ - sila regularizácie
hyperparameter

$$L(W) = \underbrace{\frac{1}{N} \sum_{i=1}^N L_i(f(x_i, W), y_i)}_{\text{Strata dát}} + \lambda R(W)$$

Strata dát: Model by mal zodpovedať trénovacím dátam



Regularizácia: Zabráni aby sa model natrénoval „príliš dobre“ na trénovacích dátach

Jednoduché príklady:

L2 regularization: $R(W) = \sum_k \sum_l W_{k,l}^2$

L1 regularization: $R(W) = \sum_k \sum_l |W_{k,l}|$

Elastic net (L1 + L2): $R(W) = \sum_k \sum_l \beta W_{k,l}^2 + |W_{k,l}|$

Regularizácia

λ - sila regularizácie
hyperparameter

$$L(W) = \frac{1}{N} \sum_{i=1}^N L_i(f(x_i, W), y_i) + \lambda R(W)$$

Strata dát: Model by mal zodpovedať trénovacím dátam



Regularizácia: Zabráni aby sa model natrénoval „príliš dobre“ na trénovacích dátach

Jednoduché príklady:

L2 regularization: $R(W) = \sum_k \sum_l W_{k,l}^2$

L1 regularization: $R(W) = \sum_k \sum_l |W_{k,l}|$

Elastic net (L1 + L2): $R(W) = \sum_k \sum_l \beta W_{k,l}^2 + |W_{k,l}|$

Zložitejšie:

Dropout

Batch normalization

Stochastic depth, fractional pooling, etc

Regularizácia

λ - sila regularizácie
hyperparameter

$$L(W) = \frac{1}{N} \sum_{i=1}^N L_i(f(x_i, W), y_i) + \lambda R(W)$$

Strata dát: Model by mal zodpovedať trénovacím dátam



Regularizácia: Zabráni aby sa model natrénoval „príliš dobre“ na trénovacích dátach



Prečo regularizovať?

- Vyjadriť preferencie pomocou váh
- Urobiť model *jednoduchý* aby fungoval na testovacích dátach
- Vylepšiť optimalizáciu

Regularizácia: Vyjadrenie preferencií

$$x = [1, 1, 1, 1]$$

$$w_1 = [1, 0, 0, 0]$$

$$w_2 = [0.25, 0.25, 0.25, 0.25]$$

L2 Regularization

$$R(W) = \sum_k \sum_l W_{k,l}^2$$

$$w_1^T x = w_2^T x = 1$$

Regularizácia: Vyjadrenie preferencií

$$x = [1, 1, 1, 1]$$

$$w_1 = [1, 0, 0, 0]$$

$$w_2 = [0.25, 0.25, 0.25, 0.25]$$

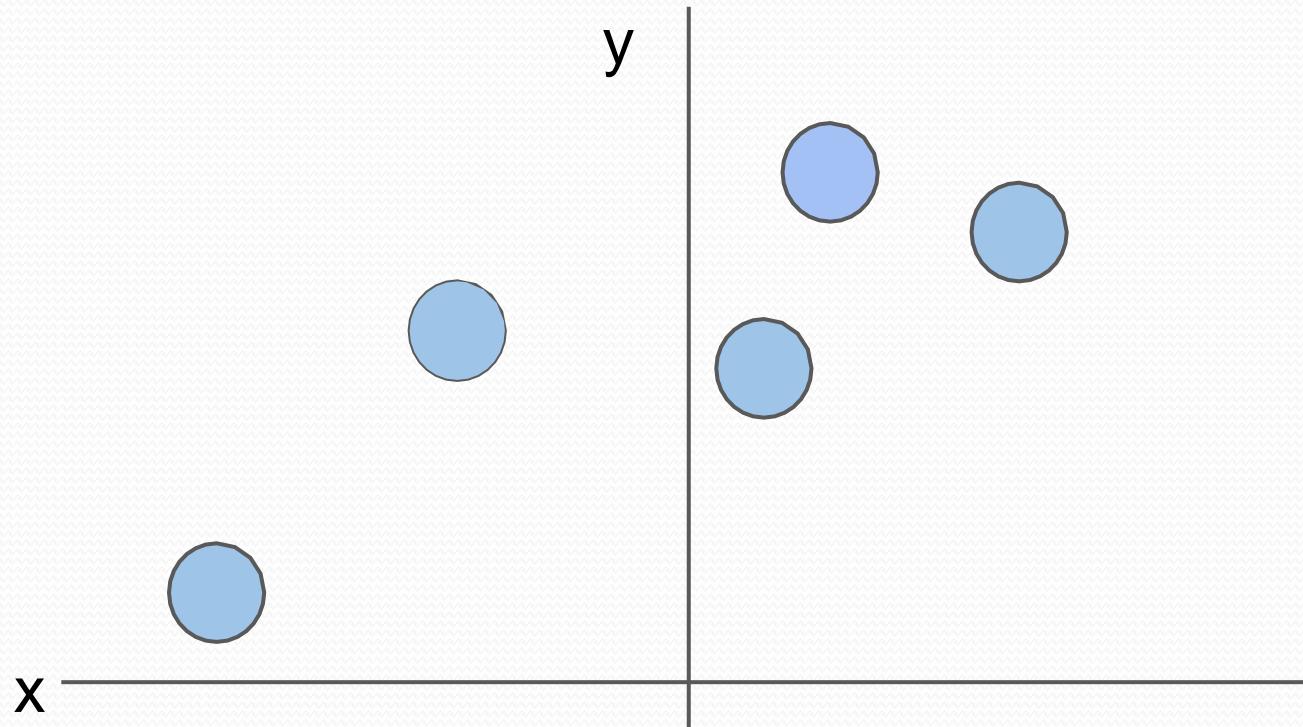
L2 Regularization

$$R(W) = \sum_k \sum_l W_{k,l}^2$$

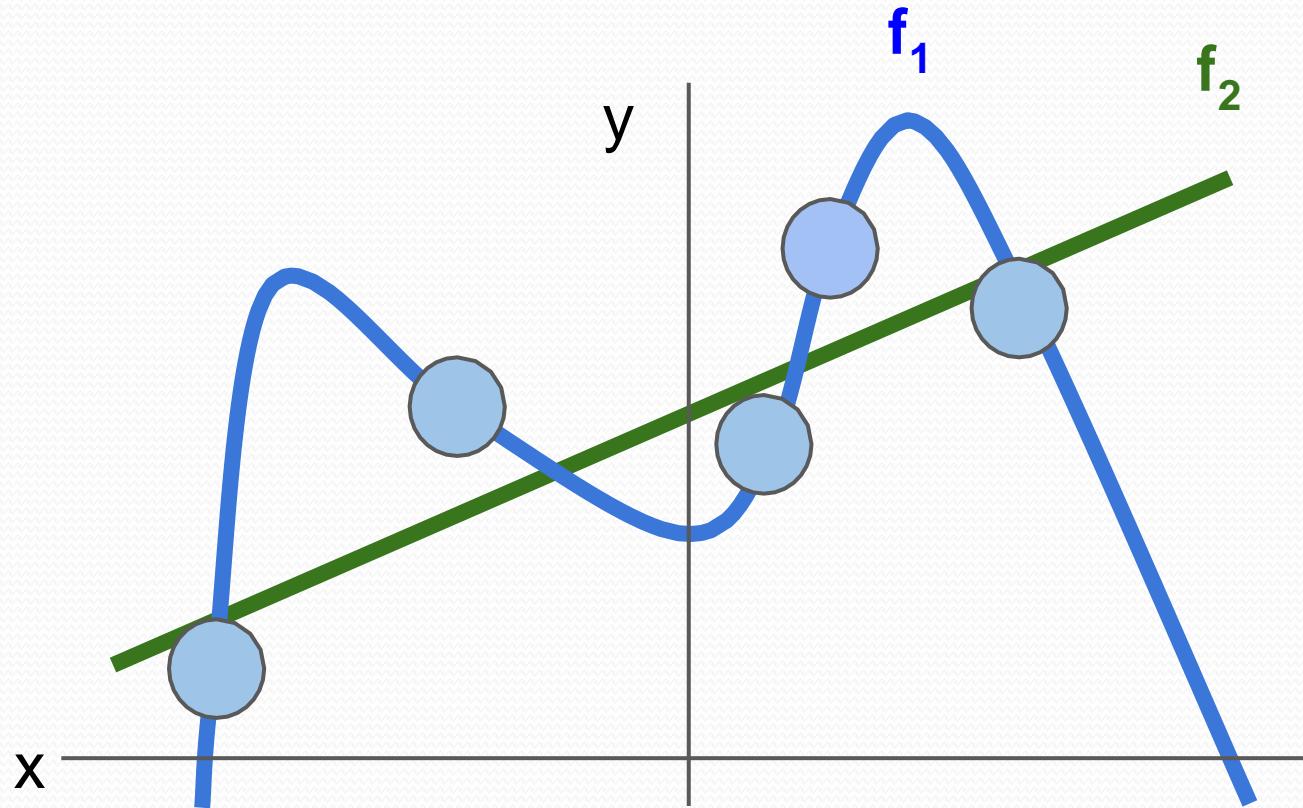
L2 regularization likes to
“spread out” the weights

$$w_1^T x = w_2^T x = 1$$

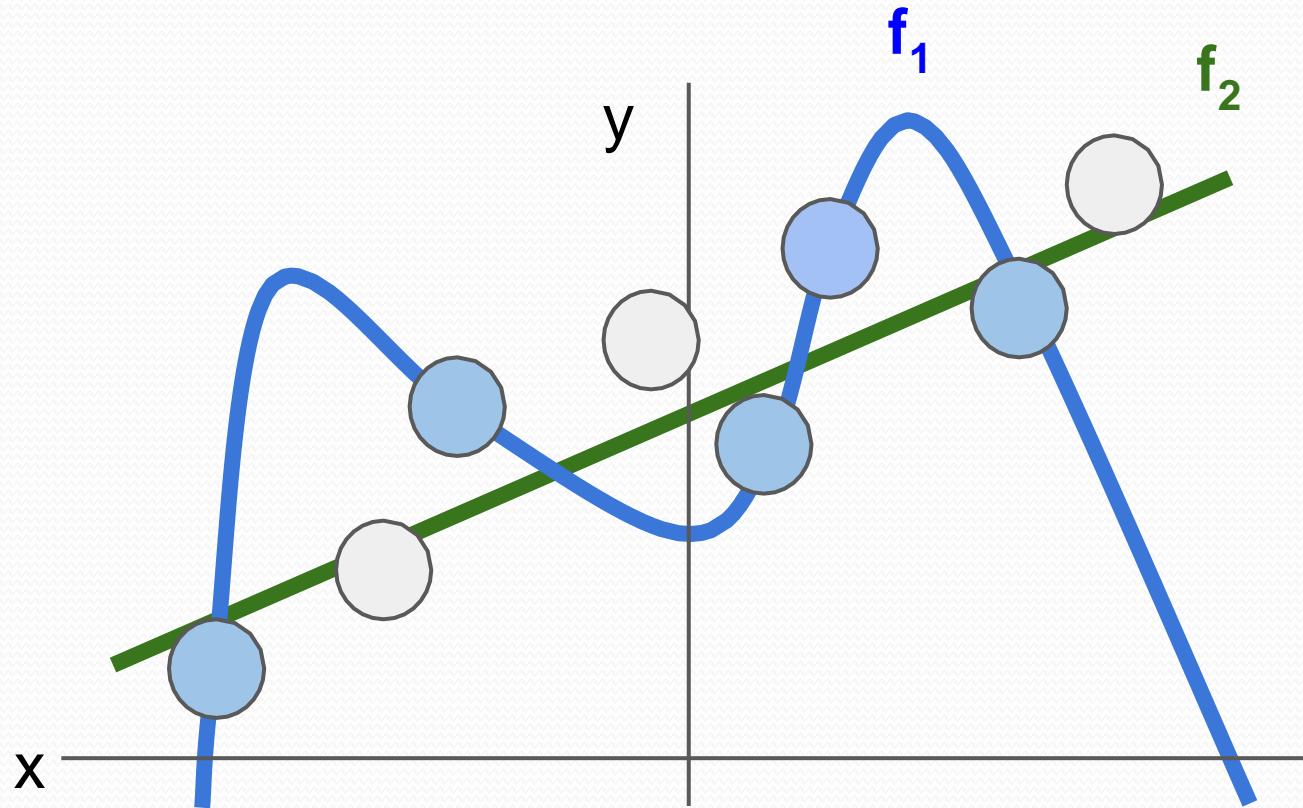
Regularizácia: Uprednostní jednoduché modely



Regularizácia: Uprednostní jednoduché modely



Regularizácia: Uprednostní jednoduché modely



Regularizácia zabraňuje aby sa model fitoval na dátá *príliš dobre* a teda aby sme nefitovali aj šum v dátach

Softmax Clasifikátor (Multinomial Logistic Regression)

Chceme interpretovať skóre klasifikátora ako
pravdepodobnosti



mačka	3.2
auto	5.1
žaba	-1.7

Iné Stratové funkcie: Softmax Clasifikátor (Multinomial Logistic Regression)

Chceme interpretovať skóre klasifikátora ako
pravdepodobnosti



$$s = f(x_i; W)$$

$$P(Y = k | X = x_i) = \frac{e^{s_k}}{\sum_j e^{s_j}}$$

Softmax
Funkcia

mačka	3.2
auto	5.1
žaba	-1.7

Softmax Clasifikátor (Multinomial Logistic Regression)

Chceme interpretovať skóre klasifikátora ako
pravdepodobnosti



$$s = f(x_i; W)$$

$$P(Y = k | X = x_i) = \frac{e^{s_k}}{\sum_j e^{s_j}}$$

Softmax
Funkcia

Pravdepodobnosti
musia byť ≥ 0

mačka	3.2	
auto	5.1	$\xrightarrow{\text{exp}}$
žaba	-1.7	

$$\begin{matrix} 24.5 \\ 164.0 \\ 0.18 \end{matrix}$$

nenormalizované
pravdepodobnosti

Softmax Clasifikátor (Multinomial Logistic Regression)

Chceme interpretovať skóre klasifikátora ako
pravdepodobnosti



$$s = f(x_i; W)$$

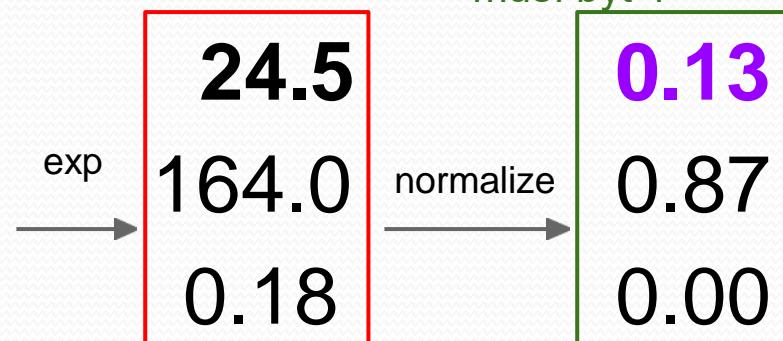
$$P(Y = k | X = x_i) = \frac{e^{s_k}}{\sum_j e^{s_j}}$$

Softmax
Funkcia

Pravdepodobnosti
musia byť ≥ 0

Suma
pravdepodobností
musí byť 1

mačka	3.2
auto	5.1
žaba	-1.7



nenormalizované
pravdepodobnosti

pravdepodobnosti

Softmax Clasifikátor (Multinomial Logistic Regression)

Chceme interpretovať skóre klasifikátora ako
pravdepodobnosti



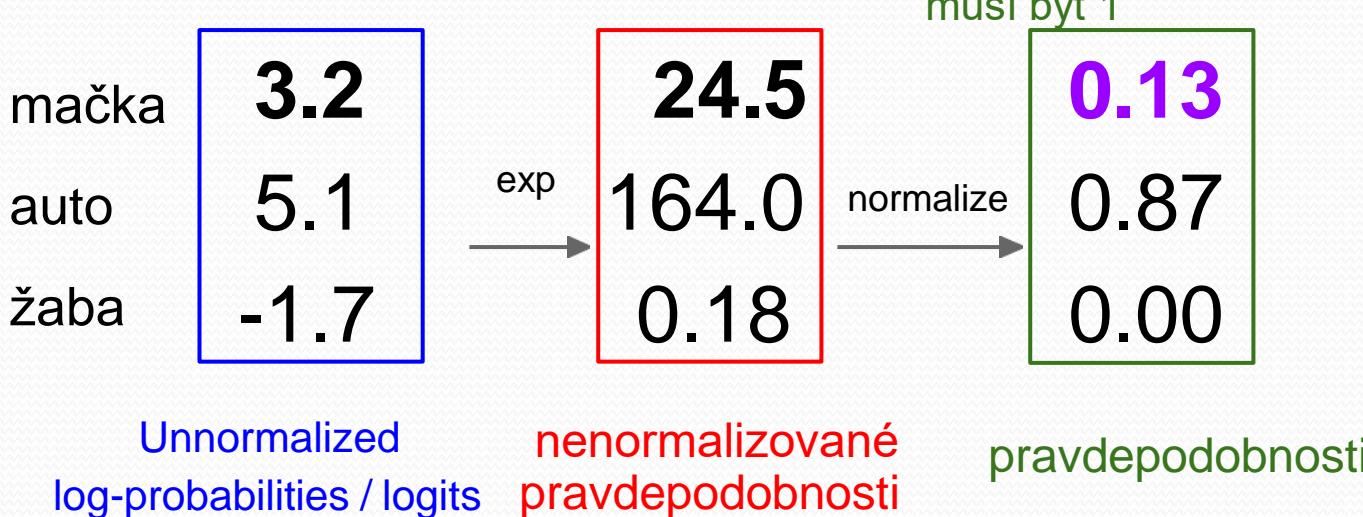
$$s = f(x_i; W)$$

$$P(Y = k | X = x_i) = \frac{e^{s_k}}{\sum_j e^{s_j}}$$

Softmax
Funkcia

Pravdepodobnosti
musia byť ≥ 0

Suma
pravdepodobností
musí byť 1



Softmax Clasifikátor (Multinomial Logistic Regression)

Chceme interpretovať skóre klasifikátora ako
pravdepodobnosti



$$s = f(x_i; W)$$

$$P(Y = k | X = x_i) = \frac{e^{s_k}}{\sum_j e^{s_j}}$$

Softmax
Funkcia

Pravdepodobnosti
musia byť ≥ 0

Suma
pravdepodobností
musí byť 1

mačka	3.2
auto	5.1
žaba	-1.7

exp

24.5
164.0
0.18

normalize

0.13
0.87
0.00

$$L_i = -\log(0.13) = 0.89$$

Unnormalized
log-probabilities / logits

nenormalizované
pravdepodobnosti

pravdepodobnosti

Softmax Clasifikátor (Multinomial Logistic Regression)

Chceme interpretovať skóre klasifikátora ako
pravdepodobnosti



$$s = f(x_i; W)$$

Pravdepodobnosti
musia byť ≥ 0

mačka	3.2
auto	5.1
žaba	-1.7

exp

24.5
164.0
0.18

normalize

$$P(Y = k | X = x_i) = \frac{e^{s_k}}{\sum_j e^{s_j}}$$

Softmax
Funkcia

Suma
pravdepodobností
musí byť 1

0.13
0.87
0.00

$$L_i = -\ln(0.13) \\ = 2.04$$

Metóda maximálnej vierošodnosti
(Maximum Likelihood Estimation)
Vyberáme pravdepodobnosti tak,
aby sme maximalizovali
vierošodnosť pozorovaných dát

Unnormalized
log-probabilities / logits

nenormalizované
pravdepodobnosti

pravdepodobnosti

Softmax Clasifikátor (Multinomial Logistic Regression)

Chceme interpretovať skóre klasifikátora ako
pravdepodobnosti



$$s = f(x_i; W)$$

$$P(Y = k | X = x_i) = \frac{e^{s_k}}{\sum_j e^{s_j}}$$

Softmax
Funkcia

Pravdepodobnosti
musia byť ≥ 0

Suma
pravdepodobností
musí byť 1

$$L_i = -\log P(Y = y_i | X = x_i)$$

mačka	3.2
auto	5.1
žaba	-1.7

exp

24.5
164.0
0.18

normalize

0.13
0.87
0.00

porovnaj

1.00
0.00
0.00

Unnormalized
log-probabilities / logits

nenormalizované
pravdepodobnosti

pravdepodobnosti

Správne
pravd.

Softmax Clasifikátor (Multinomial Logistic Regression)

Chceme interpretovať skóre klasifikátora ako
pravdepodobnosti



$$s = f(x_i; W)$$

$$P(Y = k | X = x_i) = \frac{e^{s_k}}{\sum_j e^{s_j}}$$

Softmax
Funkcia

Pravdepodobnosti
musia byť ≥ 0

Suma
pravdepodobností
musí byť 1

mačka	3.2
auto	5.1
žaba	-1.7

\exp

24.5
164.0
0.18

normalize

0.13
0.87
0.00

Unnormalized
log-probabilities / logits

nenormalizované
pravdepodobnosti

pravdepodobnosťí

$$L_i = -\log P(Y = y_i | X = x_i)$$

Kullback–Leibler
divergence

$$D_{KL}(P \| Q) = \sum_y P(y) \log \frac{P(y)}{Q(y)}$$

Správne
pravd.

1.00
0.00
0.00

Softmax Clasifikátor (Multinomial Logistic Regression)

Chceme interpretovať skóre klasifikátora ako
pravdepodobnosti



$$s = f(x_i; W)$$

$$P(Y = k | X = x_i) = \frac{e^{s_k}}{\sum_j e^{s_j}}$$

Softmax
Funkcia

Pravdepodobnosti
musia byť ≥ 0

mačka	3.2
auto	5.1
žaba	-1.7

\exp

24.5
164.0
0.18

normalize

0.13
0.87
0.00

Suma
pravdepodobností
musí byť 1

$$L_i = -\log P(Y = y_i | X = x_i)$$

Unnormalized
log-probabilities / logits

nenormalizované
pravdepodobnosti

pravdepodobnosti

Správne
pravd.

porovnaj	1.00
Cross Entropy	0.00
$H(P, Q) = H(p) + D_{KL}(P \ Q)$	0.00

Softmax Clasifikátor (Multinomial Logistic Regression)

Chceme interpretovať skóre klasifikátora ako
pravdepodobnosti



$$s = f(x_i; W)$$

$$P(Y = k | X = x_i) = \frac{e^{s_k}}{\sum_j e^{s_j}}$$

Softmax
Funkcia

mačka	3.2
auto	5.1
žaba	-1.7

Maximalizovať pravdepodobnosti správnej triedy

$$L_i = -\log P(Y = y_i | X = x_i) = -\log\left(\frac{e^{s_{y_i}}}{\sum_j e^{s_j}}\right)$$

Softmax Clasifikátor (Multinomial Logistic Regression)

Chceme interpretovať skóre klasifikátora ako
pravdepodobnosti



$$s = f(x_i; W)$$

$$P(Y = k | X = x_i) = \frac{e^{s_k}}{\sum_j e^{s_j}}$$

Softmax
Funkcia

mačka	3.2
auto	5.1
žaba	-1.7

Maximalizovať pravdepodobnosti správnej triedy

$$L_i = -\log P(Y = y_i | X = x_i) = -\log\left(\frac{e^{s_{y_i}}}{\sum_j e^{s_j}}\right)$$

Q1: Akú min/max hodnotu môže nadobudnúť L_i ?

Softmax Clasifikátor (Multinomial Logistic Regression)

Chceme interpretovať skóre klasifikátora ako
pravdepodobnosti



$$s = f(x_i; W)$$

$$P(Y = k | X = x_i) = \frac{e^{s_k}}{\sum_j e^{s_j}}$$

Softmax
Funkcia

mačka	3.2
auto	5.1
žaba	-1.7

Maximalizovať pravdepodobnosti správnej triedy

$$L_i = -\log P(Y = y_i | X = x_i) = -\log\left(\frac{e^{s_{y_i}}}{\sum_j e^{s_j}}\right)$$

Q1: Akú min/max hodnotu môže nadobudnúť L_i ?

A: min 0, max infinity

Softmax Clasifikátor (Multinomial Logistic Regression)

Chceme interpretovať skóre klasifikátora ako
pravdepodobnosti



$$s = f(x_i; W)$$

$$P(Y = k | X = x_i) = \frac{e^{s_k}}{\sum_j e^{s_j}}$$

Softmax
Funkcia

mačka	3.2
auto	5.1
žaba	-1.7

Maximalizovať pravdepodobnosti správnej triedy

$$L_i = -\log P(Y = y_i | X = x_i) = -\log\left(\frac{e^{s_{y_i}}}{\sum_j e^{s_j}}\right)$$

Q2: Ak inicializujeme všetky s na približne rovnakú hodnotu a počet tried je C, akú hodnotu bude mať L_i ?

Softmax Clasifikátor (Multinomial Logistic Regression)

Chceme interpretovať skóre klasifikátora ako
pravdepodobnosti



$$s = f(x_i; W)$$

$$P(Y = k | X = x_i) = \frac{e^{s_k}}{\sum_j e^{s_j}}$$

Softmax
Funkcia

mačka	3.2
auto	5.1
žaba	-1.7

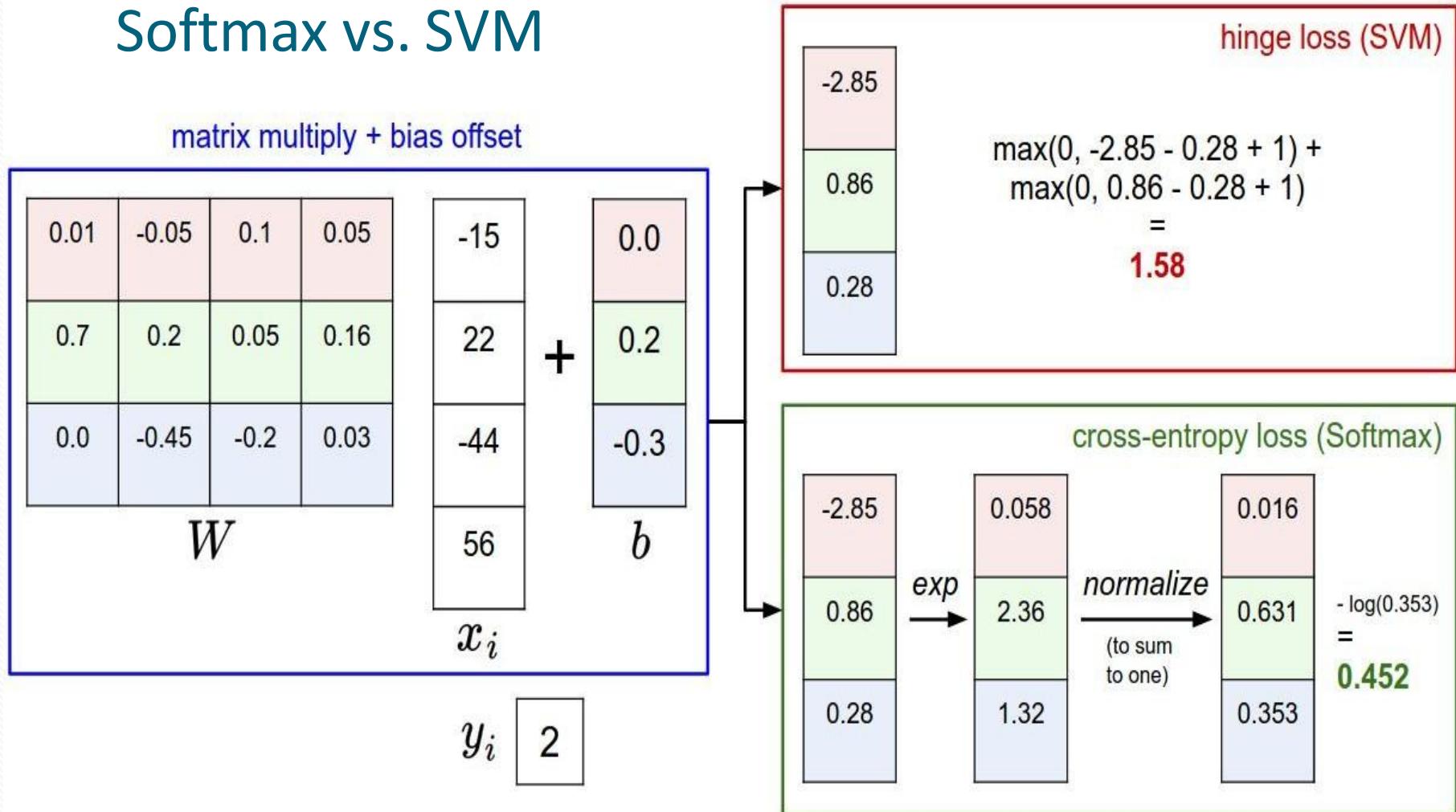
Maximalizovať pravdepodobnosti správnej triedy

$$L_i = -\log P(Y = y_i | X = x_i) = -\log\left(\frac{e^{s_{y_i}}}{\sum_j e^{s_j}}\right)$$

Q2: Ak inicializujeme všetky s na približne rovnakú hodnotu a počet tried je C, akú hodnotu bude mať L_i ?

A: $-\log(1/C) = \log(C)$, eg $\log(10) \approx 2.3$

Softmax vs. SVM



Softmax vs. SVM

$$L_i = -\log\left(\frac{e^{s_{y_i}}}{\sum_j e^{s_j}}\right) \quad L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

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 $y_i = 0$

Q: Čo sa stane s hodnotou Straty v obidvoch prípadoch, ak skóre pre niektorú vzorku trochu pomeníme?

Opakovanie

- Máme databázu vzoriek (x, y)
- Máme **funciu ohodnotení (score function):** $s = f(x; W) = Wx$
- Máme **stratovú funkciu:**

Softmax

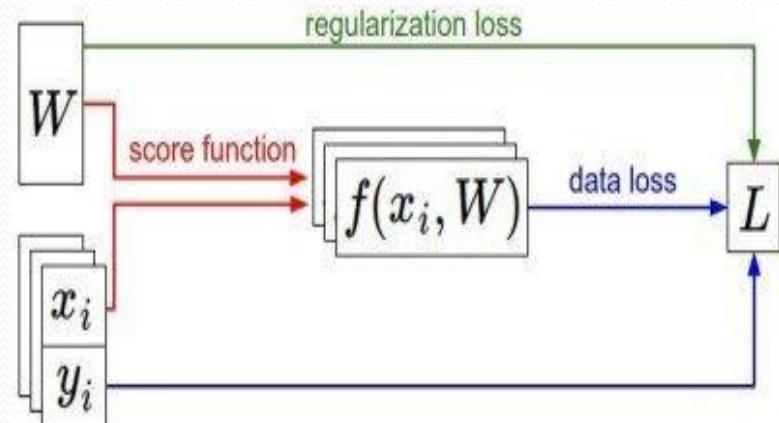
$$L_i = -\log\left(\frac{e^{sy_i}}{\sum_j e^{sj}}\right)$$

SVM

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

Full loss

$$L = \frac{1}{N} \sum_{i=1}^N L_i + R(W)$$



Opakovanie

Ako nájdem najlepšie W ?

- Máme databázu vzoriek (x, y)
- Máme **funciu ohodnotení (score function):** $s = f(x; W) = Wx$ e.g.
- Máme **stratovú funkciu:**

Softmax

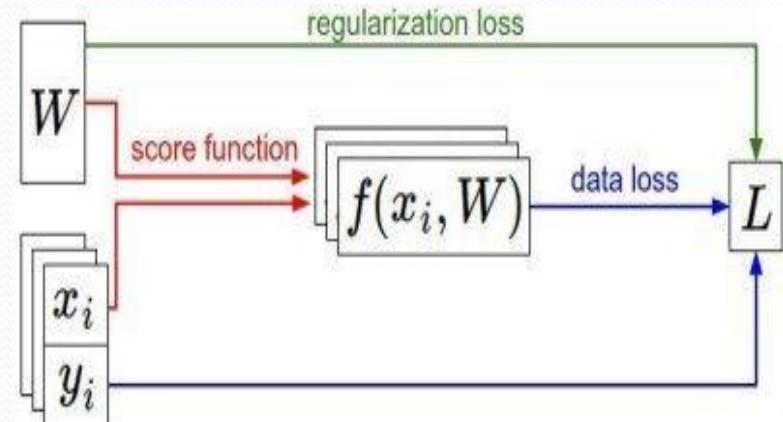
$$L_i = -\log\left(\frac{e^{sy_i}}{\sum_j e^{sj}}\right)$$

SVM

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

Full loss

$$L = \frac{1}{N} \sum_{i=1}^N L_i + R(W)$$



Optimalizácia

- Každý bod na tejto krajine zodpovedá nejakému nastaveniu parametrov W
- Výška v každom bode zodpoveda hodnote Straty
- Úlohou je nájsť dno údolia, najnižší bod



Stratégia #1: Random search

- Prvý nápad – veľmi zlý
- Postupne testujem rôzne W v náhodnom poradí a vypočítame Stratu
- Pri náhodnom rozdelení do 10 tried je uspešnosť 10%
- Ak dostaneme 15,5% sme lepší
- **ALE state of the art** je 95%

Stratégia #2: Sledovať svah

- Na základe lokálnej geometrie
- Pomaly sa posúvať smerom dole svahom
- V 1-rozmernom priestore, svah (sklon) sa určí ako derivácia funkcie:

$$\frac{df(x)}{dx} = \lim_{h \rightarrow 0} \frac{f(x + h) - f(x)}{h}$$

- Vo viacrozmerných dimenziách **gradient** je vektor (parciálnych derivácií) pozdĺž každej dimenzie
- Svah v akomkoľvek smere je **skalárny súčin** daného smeru a gradientu
- Smer najstrmšieho zostupu je v smere **negatívneho gradientu**

aktuálne W:

[0.34,
-1.11,
0.78,
0.12,
0.55,
2.81,
-3.1,
-1.5,
0.33,...]

loss 1.25347

gradient dW:

[-2.5,
?,
?,
?,
?,
?,
?,
?,
?,
?,
?,
?,
?,
?,
?,
?,...]

aktuálne W:

[0.34,
-1.11,
0.78,
0.12,
0.55,
2.81,
-3.1,
-1.5,
0.33,...]
loss 1.25347

W + h (1. dim):

[0.34 + **0.0001**,
-1.11,
0.78,
0.12,
0.55,
2.81,
-3.1,
-1.5,
0.33,...]
loss 1.25322

gradient dW:

[-2.5,
?,
?,
?,
?,
?,
?,
?,
?,
?
?,...]

aktuálne W:

[0.34,
-1.11,
0.78,
0.12,
0.55,
2.81,
-3.1,
-1.5,
0.33,...]

loss 1.25347

W + h (1. dim):

[0.34 + **0.0001**,
-1.11,
0.78,
0.12,
0.55,
2.81,
-3.1,
-1.5,
0.33,...]

loss 1.25322

gradient dW:

[-2.5,
?,
?.

$$\frac{(1.25322 - 1.25347)}{0.0001} = -2.5$$

$$\frac{df(x)}{dx} = \lim_{h \rightarrow 0} \frac{f(x + h) - f(x)}{h}$$

?,
?,...]

aktuálne W :

[0.34,
-1.11,
0.78,
0.12,
0.55,
2.81,
-3.1,
-1.5,
0.33,...]

loss 1.25347

$W + h$ (2. dim):

[0.34,
-1.11 + **0.0001**,
0.78,
0.12,
0.55,
2.81,
-3.1,
-1.5,
0.33,...]

loss 1.25353

gradient dW :

[-2.5,
?,
?,
?,
?,
?,
?,
?,
?,
?
?,...]

aktuálne W:

[0.34,
-1.11,
0.78,
0.12,
0.55,
2.81,
-3.1,
-1.5,
0.33,...]

loss 1.25347

W + h (2. dim):

[0.34,
-1.11 + 0.0001,
0.78,
0.12,
0.55,
2.81,
-3.1,
-1.5,
0.33,...]

loss 1.25353

gradient dW:

[-2.5,
0.6,
?,
?,
?,
?,
?,
?,
?]

$$\frac{(1.25353 - 1.25347)}{0.0001} = 0.6$$

$$\frac{df(x)}{dx} = \lim_{h \rightarrow 0} \frac{f(x+h) - f(x)}{h}$$

?,...]

aktuálne W :

[0.34,
-1.11,
0.78,
0.12,
0.55,
2.81,
-3.1,
-1.5,
0.33,...]

loss 1.25347

$W + h$ (3. dim):

[0.34,
-1.11,
0.78 + 0.0001,
0.12,
0.55,
2.81,
-3.1,
-1.5,
0.33,...]

loss 1.25347

gradient dW :

[-2.5,
0.6,
?,
?,
?,
?,
?,
?,
?,
?,
?,...]

aktuálne W:

[0.34,
-1.11,
0.78,
0.12,
0.55,
2.81,
-3.1,
-1.5,
0.33,...]

loss 1.25347

W + h (3. dim):

[0.34,
-1.11,
0.78 + **0.0001**,
0.12,
0.55,
2.81,
-3.1,
-1.5,
0.33,...]

loss 1.25347

gradient dW:

[-2.5,
0.6,
0,
?,
?]

$$\frac{(1.25347 - 1.25347)}{0.0001} = 0$$

$$\frac{df(x)}{dx} = \lim_{h \rightarrow 0} \frac{f(x+h) - f(x)}{h}$$

?,...]

aktuálne W:

[0.34,
-1.11,
0.78,
0.12,
0.55,
2.81,
-3.1,
-1.5,
0.33,...]

loss 1.25347

W + h (3. dim):

[0.34,
-1.11,
0.78 + **0.0001**,
0.12,
0.55,
2.81,
-3.1,
-1.5,
0.33,...]

loss 1.25347

gradient dW:

[-2.5,
0.6,
0,
?,
?]

Numerický Gradient

- Pomalý! Musíme prejsť cez všetky dimenzie
- Treba aproximovať

?,...]

Strata je nejaká funkcia W :

$$L = \frac{1}{N} \sum_{i=1}^N L_i + \sum_k W_k^2$$

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

$$s = f(x; W) = Wx$$

Chceme $\nabla_W L$

Budeme používať **analytický gradient**

- je presnejší a výrazne rýchlejší

aktuálne W:

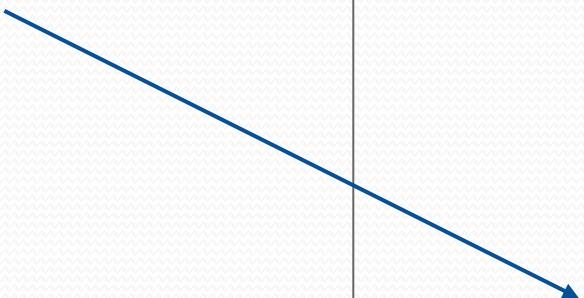
[0.34,
-1.11,
0.78,
0.12,
0.55,
2.81,
-3.1,
-1.5,
0.33,...]

loss 1.25347

$dW = \dots$
(some function
data and W)

gradient dW:

[-2.5,
0.6,
0,
0.2,
0.7,
-0.5,
1.1,
1.3,
-2.1,...]



In summary:

- Numerický gradient: aproximácia, pomalé, ľahké napísať
- Analytický gradient: presný, rýchly, náchylný na chyby

=>

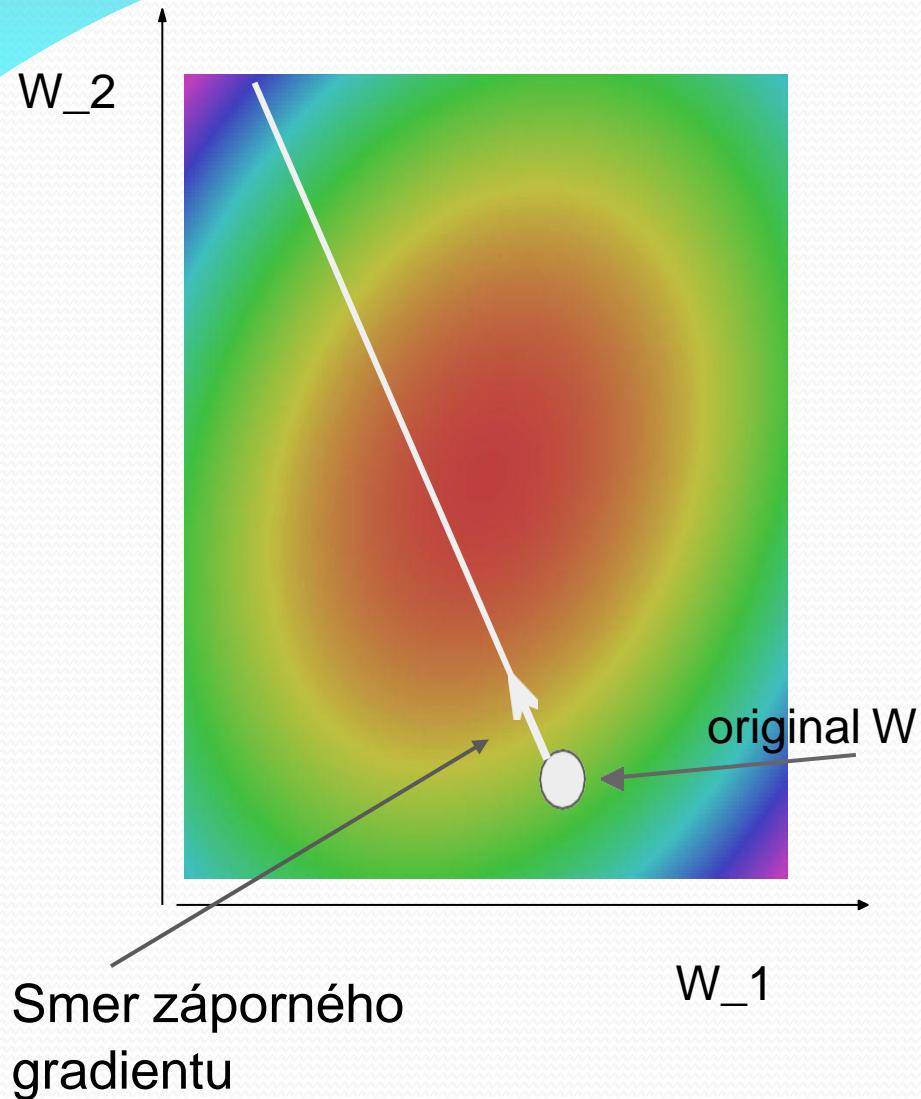
V praxi: Vždy používajte analytický gradient, ale kontrolujte implementáciu pomocou numerického gradientu.
Toto sa nazýva **gradient check**.

Gradient Descent

- Základ trénovania všetkých NN

```
# Vanilla Gradient Descent  
  
while True:  
    weights_grad = evaluate_gradient(loss_fun, data, weights)  
    weights += - step_size * weights_grad # perform parameter update
```

- Step size (learning rate) – veľmi dôležitý **hyperparameter**



- Pohybujem sa vždy v smere negatívneho gradientu
- V praxi iné algoritmy na hľadanie update rules
 - Gradient descent with momentum
 - ADAM
- Neskôr v prednáškach

Stochastic Gradient Descent (SGD)

$$L(W) = \frac{1}{N} \sum_{i=1}^N L_i(x_i, y_i, W) + \lambda R(W)$$

$$\nabla_W L(W) = \frac{1}{N} \sum_{i=1}^N \nabla_W L_i(x_i, y_i, W) + \lambda \nabla_W R(W)$$

Celú sumu je problém vypočítať ked N je veľké!

Aproximujeme sumu pomocou **minibatch**

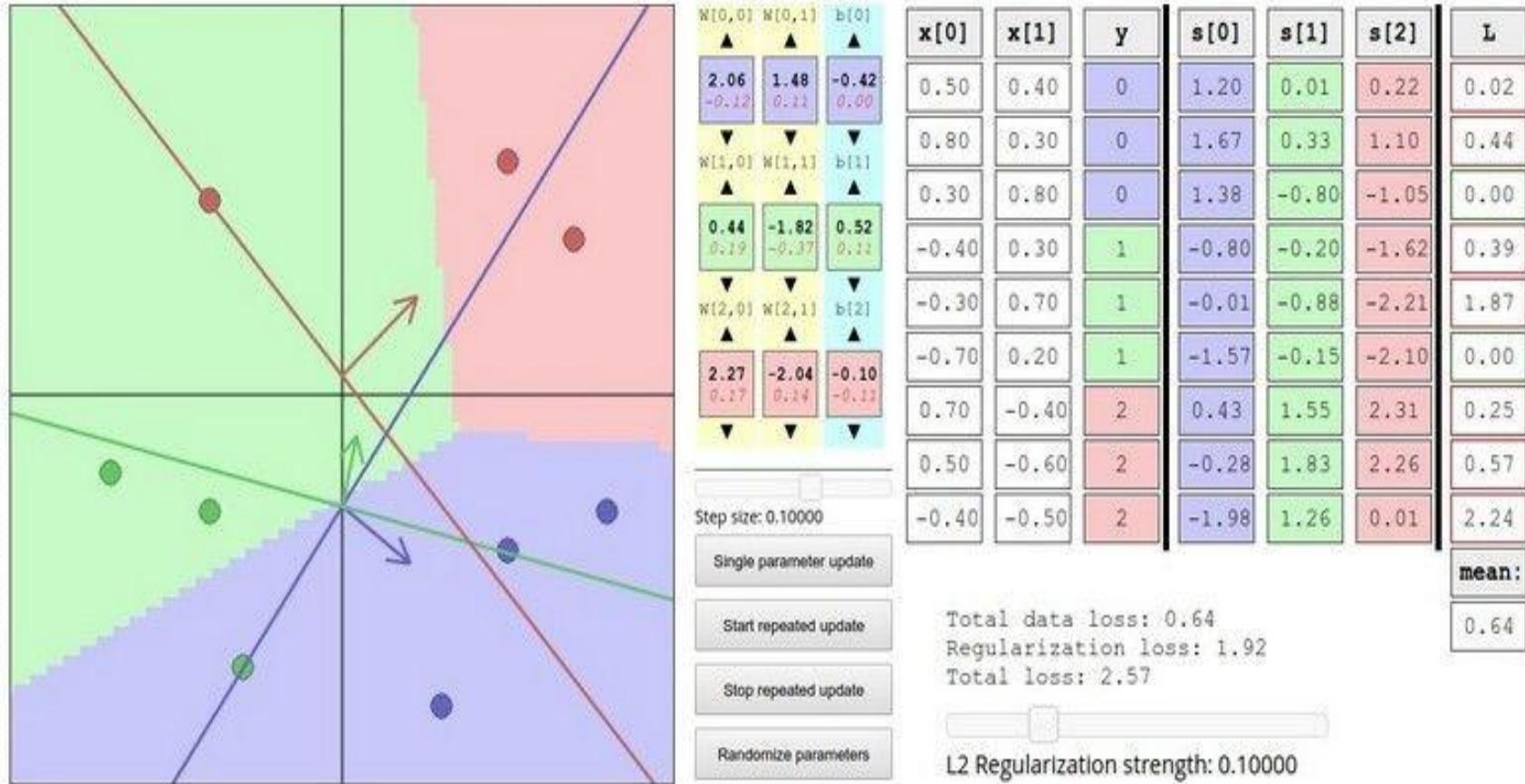
Najčastejšie
32 / 64 / 128 vzoriek

```
# Vanilla Minibatch Gradient Descent
```

```
while True:
```

```
    data_batch = sample_training_data(data, 256) # sample 256 examples
    weights_grad = evaluate_gradient(loss_fun, data_batch, weights)
    weights += - step_size * weights_grad # perform parameter update
```

Interactive Web Demo



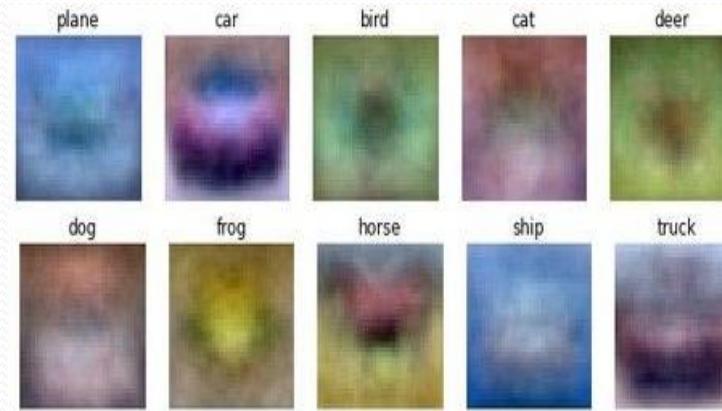
<http://vision.stanford.edu/teaching/cs231n-demos/linear-classify/>

Bokom: Príznaky v obraze (Image Features)

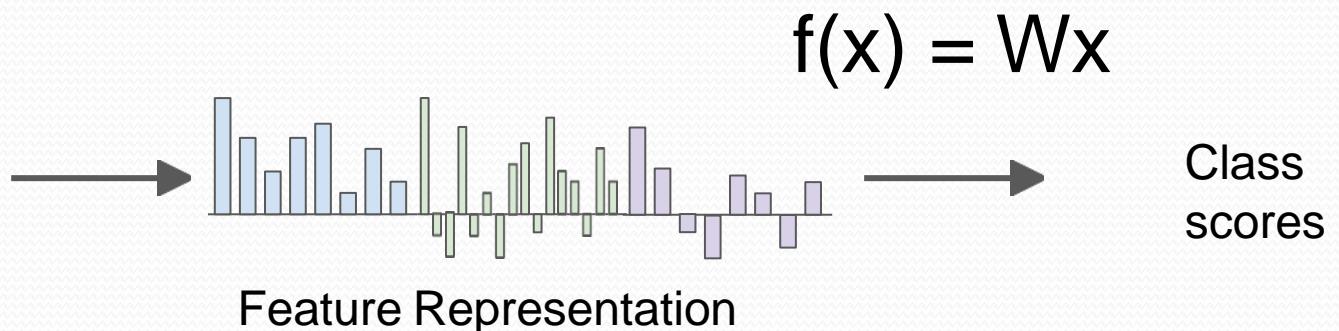


Class scores

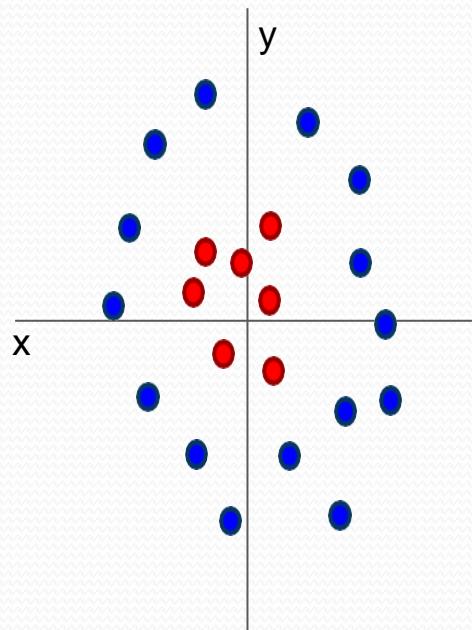
$$f(x) = Wx$$



Bokom: Príznaky v obraze (Image Features)

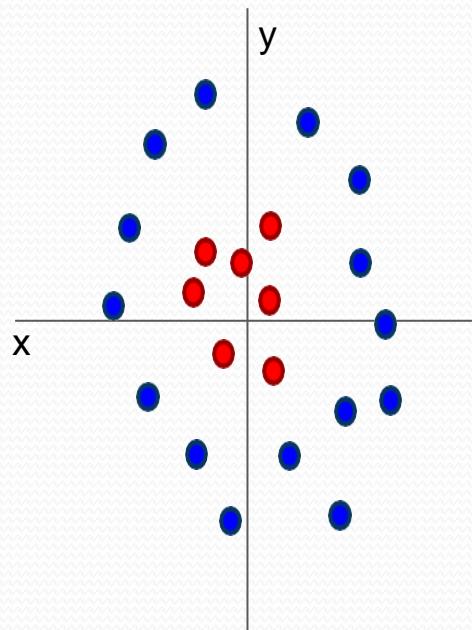


Príznaky: motivácia



Nevieme oddeliť červené
a modré pomocou
lineárneho klasifikátora

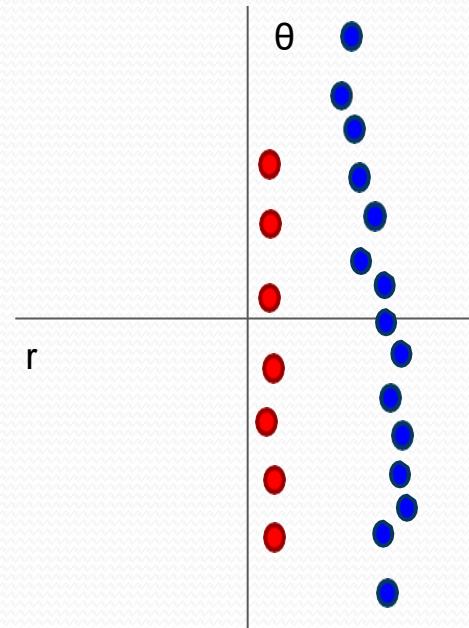
Príznaky: motivácia



$$f(x, y) = (r(x, y), \theta(x, y))$$

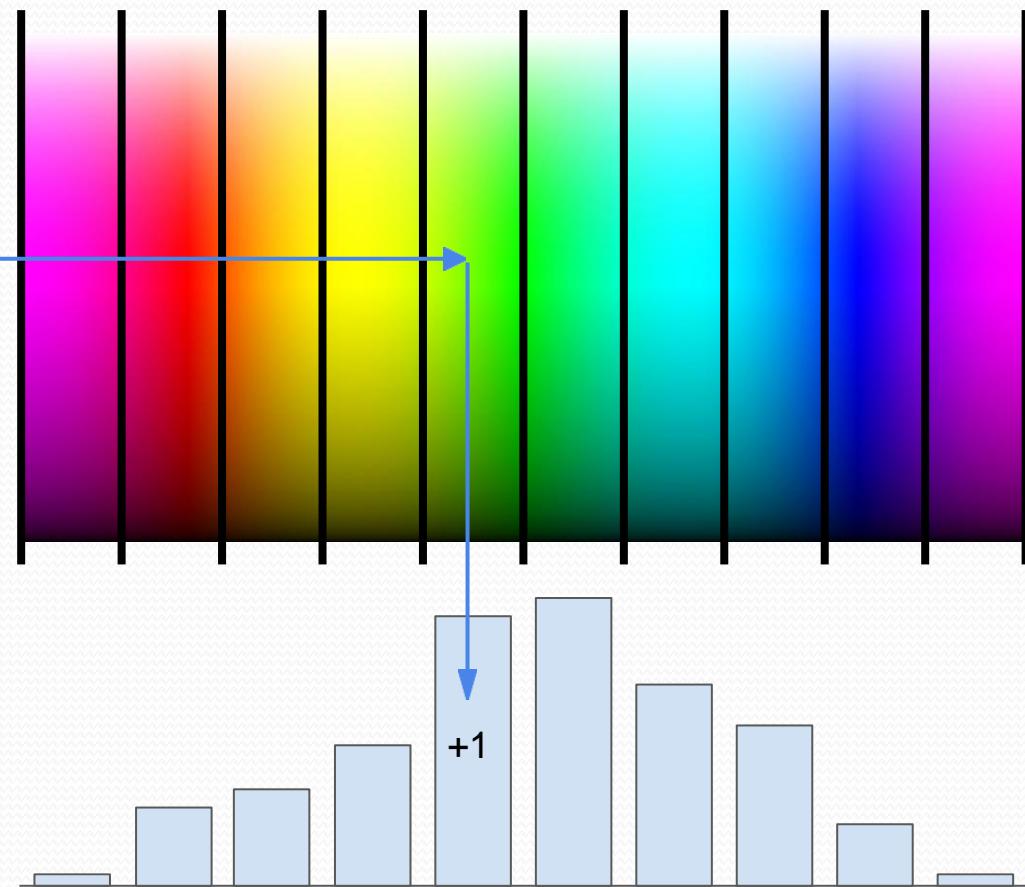
Polárne súradnice

Nevieme oddeliť červené
a modré pomocou
lineárneho klasifikátora



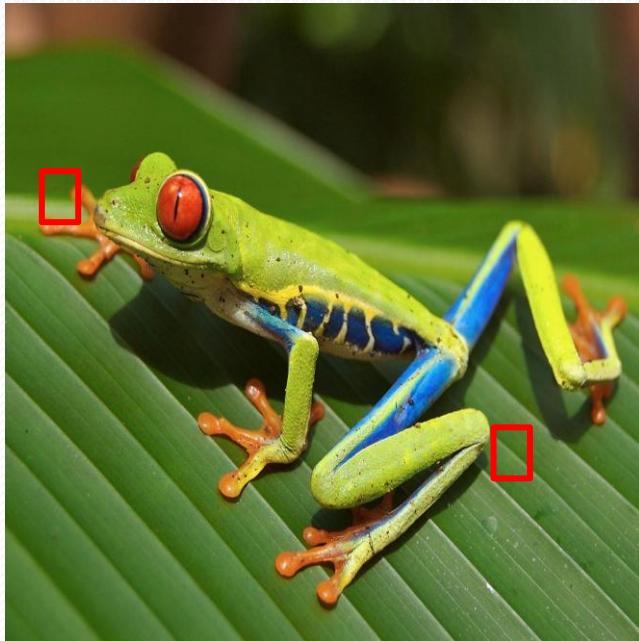
Po aplikovaní transformácie
príznakov (feature transform),
body môžeme separovať

Príklad: Farebný histogram

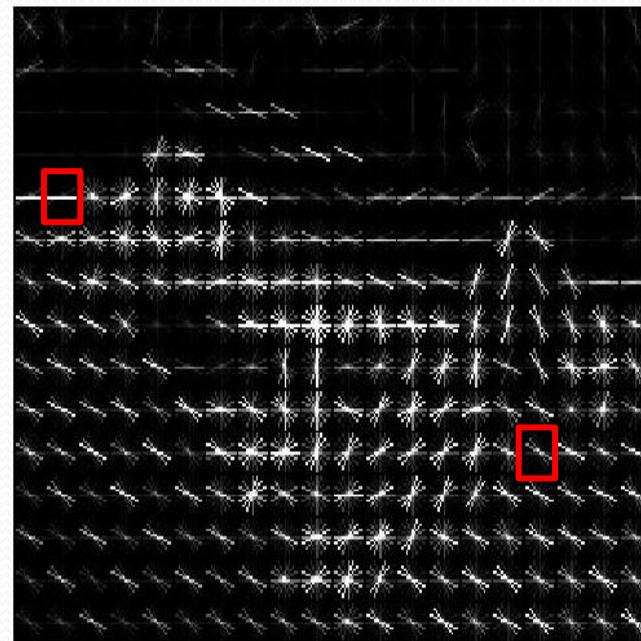


Príklad: Histogram orientovaných gradientov

Histogram of Oriented Gradients (HoG)



Divide image into 8x8 pixel regions
Within each region quantize edge direction into 9 bins



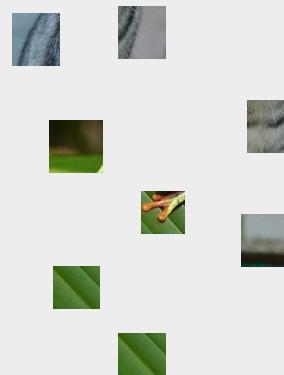
Example: 320x240 image gets divided into 40x30 bins; in each bin there are 9 numbers so feature vector has $30 \times 40 \times 9 = 10,800$ numbers

Príklad: Bag of Words

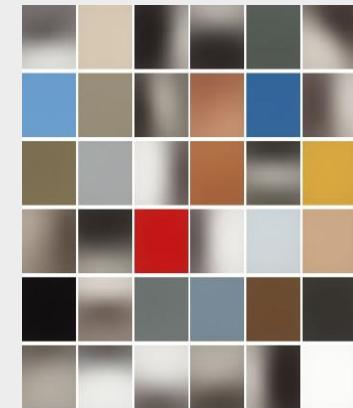
Step 1: Build codebook



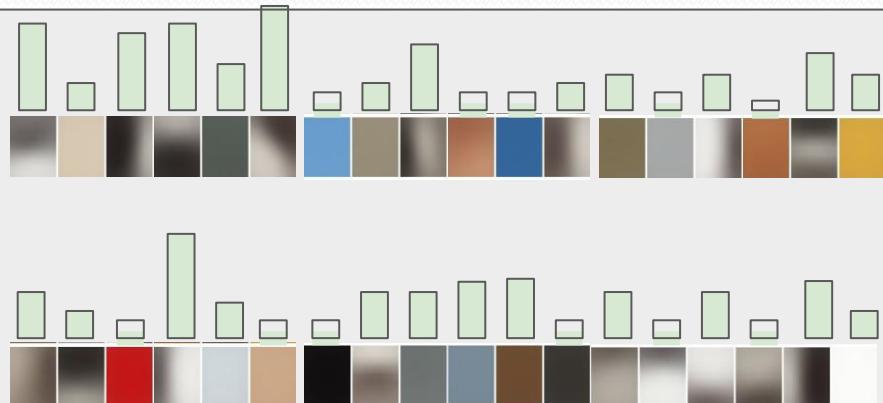
Extract
random
patches



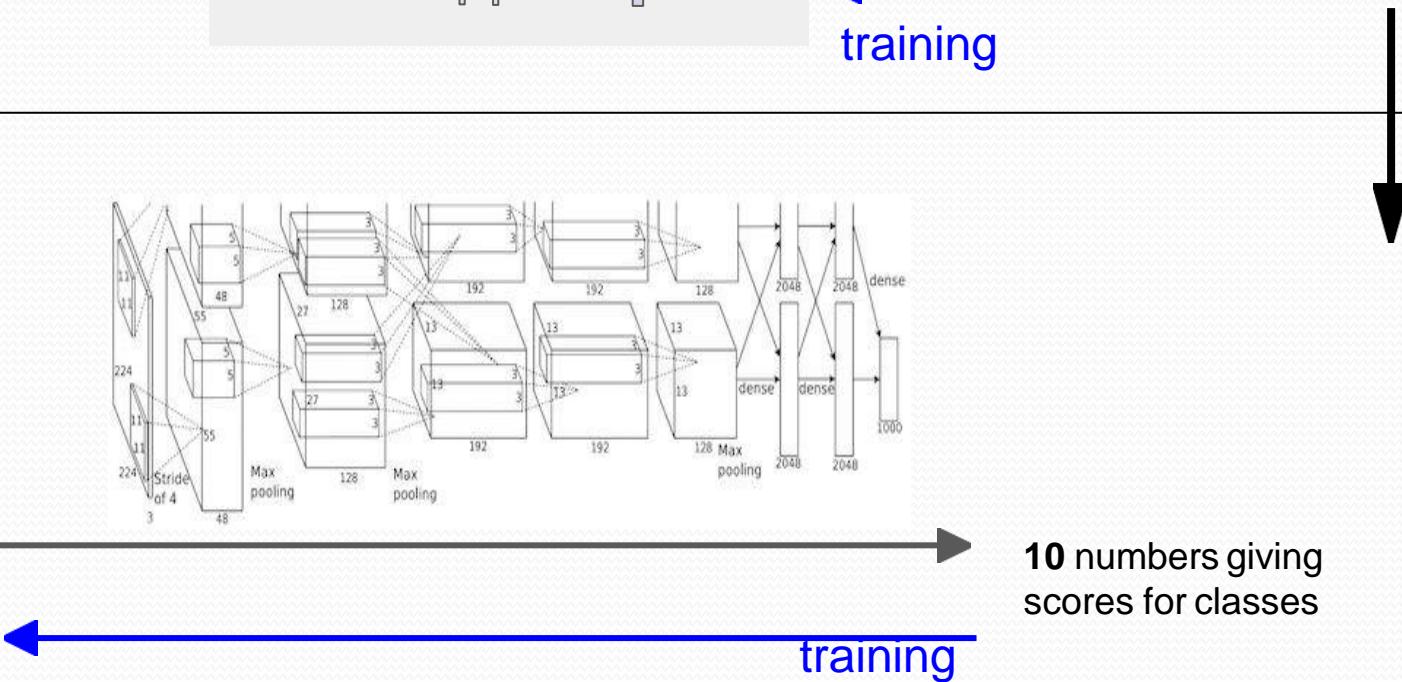
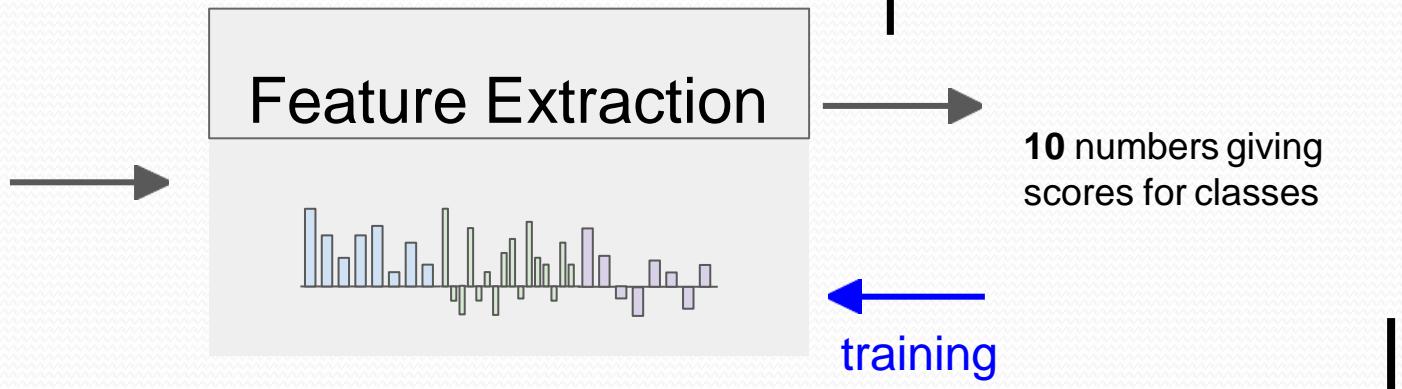
Cluster patches
to form
“codebook” of
“visual words”



Step 2: Encode images



Príznaky vs ConvNets



Nasledujúca prednáška:

Úvod do NN

Spätná propagácia (Backpropagation)