Introduction to Machine Learning
Logistic Regression

Prof. Chang-Chieh Cheng

Information Technology Service Center

National Chiao Tung University

Decision surface

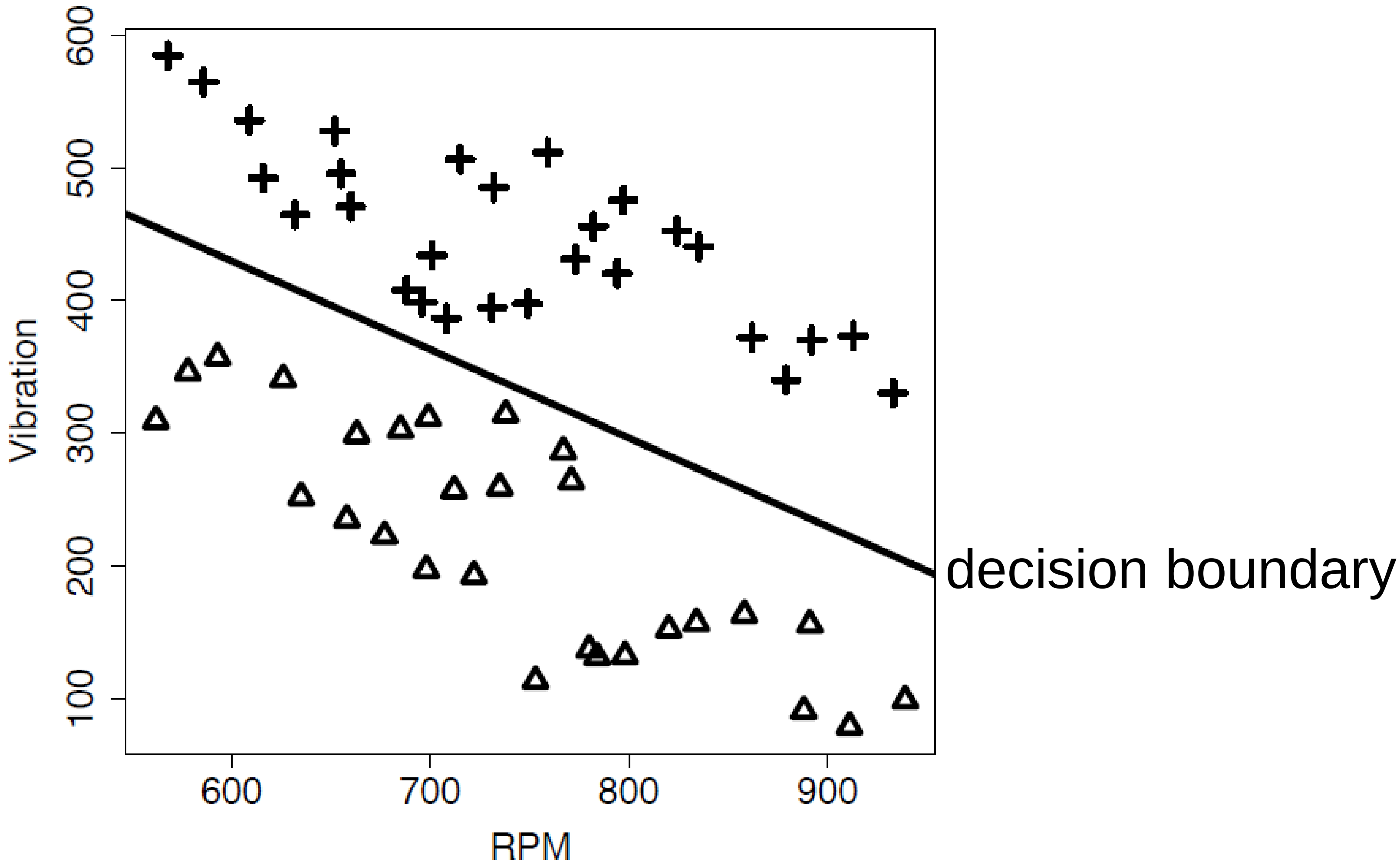
- Example: A dataset listing features for a number of generators

ID	RPM	VIBRATION	STATUS
1	568	585	good
2	586	565	good
3	609	536	good
4	616	492	good
5	632	465	good
6	652	528	good
7	655	496	good
8	660	471	good
9	688	408	good
10	696	399	good
11	708	387	good
12	701	434	good
13	715	506	good
14	732	485	good
15	731	395	good
16	749	398	good
17	759	512	good
18	773	431	good
19	782	456	good
20	797	476	good
21	794	421	good
22	824	452	good
23	835	441	good
24	862	372	good
25	879	340	good
26	892	370	good
27	913	373	good
28	933	330	good

ID	RPM	VIBRATION	STATUS
29	562	309	faulty
30	578	346	faulty
31	593	357	faulty
32	626	341	faulty
33	635	252	faulty
34	658	235	faulty
35	663	299	faulty
36	677	223	faulty
37	685	303	faulty
38	698	197	faulty
39	699	311	faulty
40	712	257	faulty
41	722	193	faulty
42	735	259	faulty
43	738	314	faulty
44	753	113	faulty
45	767	286	faulty
46	771	264	faulty
47	780	137	faulty
48	784	131	faulty
49	798	132	faulty
50	820	152	faulty
51	834	157	faulty
52	858	163	faulty
53	888	91	faulty
54	891	156	faulty
55	911	79	faulty
56	939	99	faulty

Figuring by: John D. Kelleher, et al, "Fundamentals of Machine Learning for Predictive Data Analytics - Algorithms, Worked Examples, and Case Studies," *MIT Press*, 2015.

Decision surface



- +: Good
- Δ: Faulty

Figuring by: John D. Kelleher, et al, "Fundamentals of Machine Learning for Predictive Data Analytics - Algorithms, Worked Examples, and Case Studies," *MIT Press*, 2015.

Decision surface

- Affine hyperplane
  - For a n-dimensional point $\mathbf{x} = [x_0\ x_1\ x_2\ ...\ x_n]$ in Euclidean space, where $x_0 = 1$
  - Let $\mathbf{w} = [w_0\ w_1\ ...\ w_n]$, at least one of the $w_1\ ...\ w_n$ is non-zero

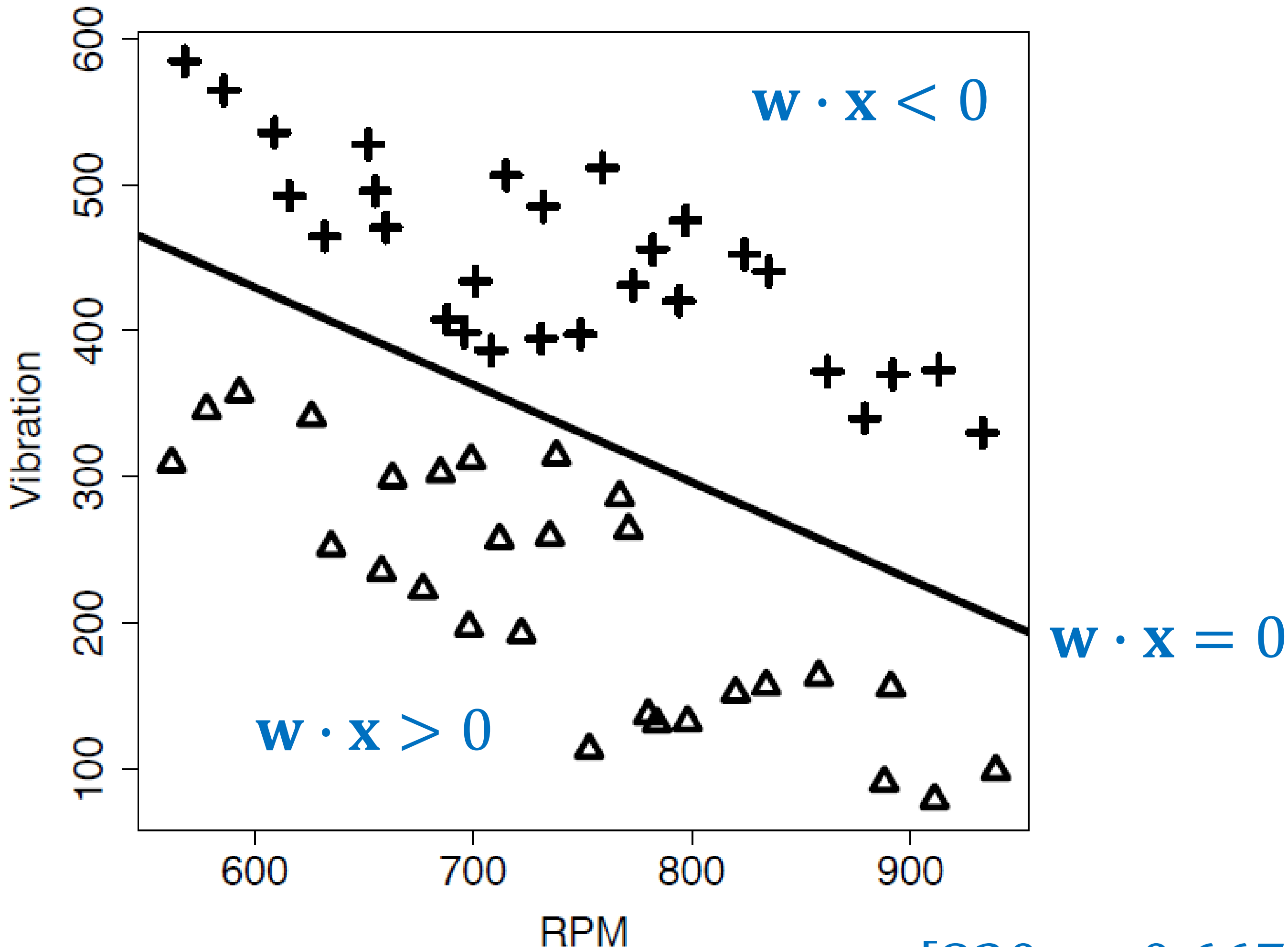
$$w_1x_1 + w_2x_2 + \cdots + w_nx_n = -w_0x_0$$

$$\sum_{j=0}^{n} w_jx_j = \mathbf{w} \cdot \mathbf{x} = 0$$

  - A hyperplane can separate the space into two half-spaces

$\mathbf{w} \cdot \mathbf{x} < 0$ and $\mathbf{w} \cdot \mathbf{x} > 0$

Decision surface



- +: Good
- Δ: Faulty

$$\mathbf{w} = [830 \;\; -0.667 \;\; -1]$$
$$\mathbf{x} = [1 \;\; RPM \;\; Vibration]$$

Figuring by: John D. Kelleher, et al, "Fundamentals of Machine Learning for Predictive Data Analytics - Algorithms, Worked Examples, and Case Studies," *MIT Press*, 2015.

Decision surface

- The learning model with decision surface

$$M_{\mathbf{w}}(\mathbf{q}) = \begin{cases} 1 & if\ \mathbf{w} \cdot \mathbf{x} \geq 0 \quad \text{(Negative)} \\ 0 & Otherwise \quad \text{(positive)} \end{cases}$$

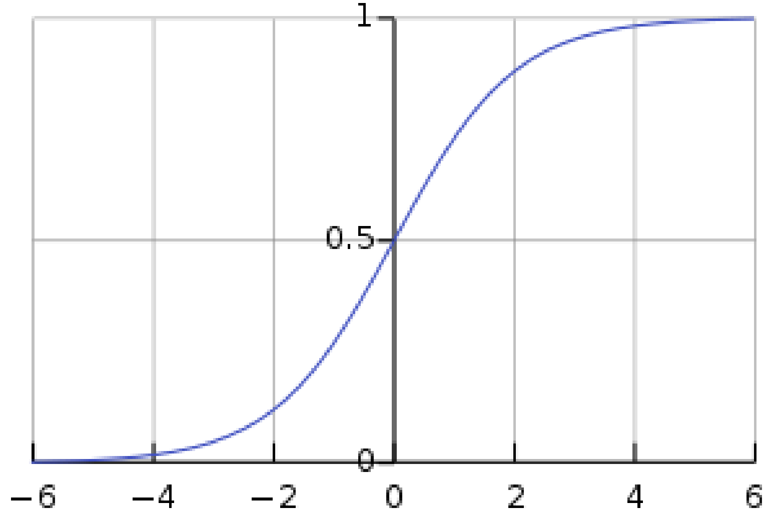
- How to find the decision surface?
  - $M_{\mathbf{w}}(\mathbf{q})$ is not a continuous function ➔ not differentiable
  - Gradient descent cannot be used!

Logistic Regression

- Logistic function or sigmoid function

$$L(x) = \frac{1}{1 + e^{-x}}$$

- The standard logistic function for *x* over a small range of real numbers such as a range contained in [−6, +6].



https://en.wikipedia.org/wiki/Logistic_function

Logistic Regression

- Hyperbolic tangent

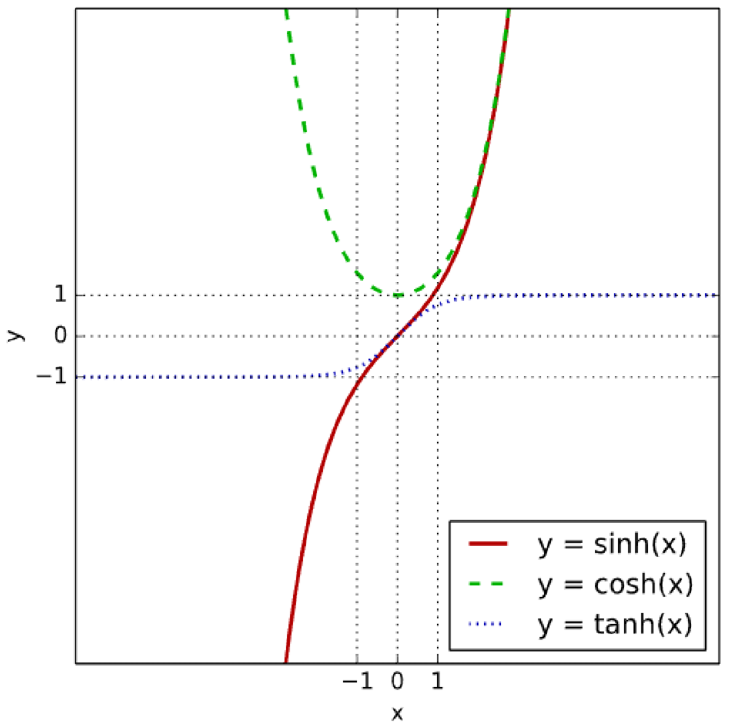
$$\tanh x = \frac{\sinh x}{\cosh x} = \frac{e^x - e^{-x}}{e^x + e^{-x}} = \frac{1 - e^{-2x}}{1 + e^{-2x}}$$

$$= \frac{e^x(1 - e^{-2x})}{e^x(1 + e^{-2x})} = \frac{1}{1 + e^{-2x}} - \frac{e^{-x}}{e^x(1 + e^{-2x})}$$

$$= L(2x) - \frac{e^{-2x}}{1 + e^{-2x}} = L(2x) - \frac{e^{-2x} + 1 - 1}{1 + e^{-2x}}$$

$$= 2L(2x) - 1$$

$$L(x) = \frac{1}{2} + \frac{1}{2} \tanh \frac{x}{2}$$



https://en.wikipedia.org/wiki/Hyperbolic_function

Logistic Regression

- The logistic function has the symmetry property

$$L(-x) = \frac{1}{1+e^{x}} = \frac{e^{-x}}{1+e^{-x}} = \frac{1+e^{-x}-1}{1+e^{-x}}$$

$$= 1 - \frac{1}{1+e^{-x}}$$

$$= 1 - L(x)$$

Logistic Regression

- Rotational symmetry

$$L(x) + L(-x) = \frac{1}{1+e^{-x}} + \frac{1}{1+e^{x}}$$

$$= \frac{(1+e^{x}) + (1+e^{-x})}{(1+e^{-x})(1+e^{x})} = 1$$

Logistic Regression

- Derivative of logistic function

$$\frac{dL(x)}{dx} = \frac{e^{-x}}{(1+e^{-x})^2} = L(x)(1 - L(x))$$

$$\frac{dL(-x)}{dx} = L(-x)(1 - L(-x))$$

  - Since $L(-x) = 1 - L(x)$

$$\frac{dL(-x)}{dx} = \frac{dL(x)}{dx}$$

Logistic Regression

- Indefinite integral of logistic function

$$\int \mathrm{L}(x)dx = x + \ln(1 + e^{-x})$$

$$= x + \ln(e^{-x}e^{x} + e^{-x})$$

$$= x + \ln(e^{-x}) + \ln(e^{x} + 1)$$

$$= \ln(e^{x} + 1)$$

Logistic Regression

- The learning model with logistic function

$$M_{\mathbf{w}}(\mathbf{x}) = L(\mathbf{w} \cdot \mathbf{x})$$

$$= \frac{1}{1 + e^{-\mathbf{w} \cdot \mathbf{x}}}$$

$\mathbf{x} = [x_0 \; x_1 \; x_2 \; \ldots \; x_n]$ and $x_0 = 1$

$$P(\text{target} = \text{negative} | \mathbf{q}) = M_{\mathbf{w}}(\mathbf{q})$$

it should be closed to 1.0

$$P(\text{target} = \text{positive} | \mathbf{q}) = 1 - M_{\mathbf{w}}(\mathbf{q})$$

it should be closed to 0.0

Logistic Regression

- Error function
  - Let *y* be the labeled target for each training data instance **x**
    - *y* = 1.0 if the target is negative
    - Otherwise *y* = 0.0
  - L2-norm error =

$$\sum_{i=1}^{m} \frac{1}{2}\left(y - M_{\mathbf{w}}(\mathbf{x}_i)\right)^2$$

Logistic Regression

$$\frac{\partial \sum_{i=1}^{m} \frac{1}{2}\left(y_i - M_{\mathbf{w}}(\mathbf{x}_i)\right)^2}{\partial w_j}$$

Chain rule

$$F(x) = f\big(g(x)\big)$$

$$F'(x) = f'\big(g(x)\big)g'(x)$$

- Let $m = 1$

$$\frac{\partial \frac{1}{2}\left(y - M_{\mathbf{w}}(\mathbf{x})\right)^2}{\partial w_j} = \left(y - M_{\mathbf{w}}(\mathbf{x})\right)\frac{\partial\left(y - M_{\mathbf{w}}(\mathbf{x})\right)}{\partial w_j} = (y - L(\mathbf{w}\cdot\mathbf{x}))\frac{\partial\left(y - L(\mathbf{w}\cdot\mathbf{x})\right)}{\partial w_j}$$

$$= (y - M_{\mathbf{w}}(\mathbf{x}))\frac{\partial\left(-L(\mathbf{w}\cdot\mathbf{x})\right)}{\partial w_j}\frac{\partial(\mathbf{w}\cdot\mathbf{x})}{\partial w_j}$$

$$= (y - M_{\mathbf{w}}(\mathbf{x}))\frac{\partial\left(-L(\mathbf{w}\cdot\mathbf{x})\right)}{\partial w_j}x_j$$

Logistic Regression

- Since $\dfrac{dL(x)}{dx} = L(x)(1 - L(x))$

$$(y - L(\mathbf{w}\cdot\mathbf{x}))\frac{\partial\big(-L(\mathbf{w}\cdot\mathbf{x})\big)}{\partial w_j}x_j = -(y - M_{\mathbf{w}}(\mathbf{x}))L(\mathbf{w}\cdot\mathbf{x})(1 - L(\mathbf{w}\cdot\mathbf{x}))x_j$$
$$= -(y - M_{\mathbf{w}}(\mathbf{x}))M_{\mathbf{w}}(\mathbf{x})(1 - M_{\mathbf{w}}(\mathbf{x}))x_j$$

- For *m* > 1

$$\frac{\partial \sum_{i=1}^{m}\frac{1}{2}\big(y_i - M_{\mathbf{w}}(\mathbf{x}_i)\big)^2}{\partial w_j}$$

$$= \sum_{i=1}^{m} -(y_i - M_{\mathbf{w}}(\mathbf{x}_i))M_{\mathbf{w}}(\mathbf{x}_i)(1 - M_{\mathbf{w}}(\mathbf{x}_i))x_{ij}$$

Logistic Regression

- To minimize error

$$w_j' = -\frac{\partial \sum_{i=1}^{m} \frac{1}{2}\left(y_i - M_{\mathbf{w}}(\mathbf{x}_i)\right)^2}{\partial w_j}$$

- ➔ $$w_j' = \sum_{i=1}^{m} (y_i - M_{\mathbf{w}}(\mathbf{x}_i))M_{\mathbf{w}}(\mathbf{x}_i)(1 - M_{\mathbf{w}}(\mathbf{x}_i))x_{ij}$$

- Therefore, increasing w_j by w'_j can let total error to approach zero

$$w_j = w_j + \alpha w_j'$$

- α: learning rate

Logistic Regression

- All feature values should be normalized.
- Normalized to [-1.0, 1.0] or [0.0. 1.0]
  - Normalization is done to have the same range of values for each of the inputs to a learning model.
  - This can guarantee stable convergence of weight and biases.

Logistic Regression

- Example
  - α = 0.02
  - Initial Weights $\mathbf{w} = [-2.9456 \quad -1.0147 \quad -2.161]$

ID	Target Level	Pred.	Error	Squared Error	w'_0	w'_1	w'_2
1	1	0.5570	0.4430	0.1963	0.1093	-0.1093	0.1093
2	1	0.5168	0.4832	0.2335	0.1207	-0.1116	0.1159
3	1	0.4469	0.5531	0.3059	0.1367	-0.1134	0.1197
4	1	0.4629	0.5371	0.2885	0.1335	-0.1033	0.1244
				...			
65	0	0.0037	-0.0037	0.0000	0.0000	0.0000	0.0000
66	0	0.0042	-0.0042	0.0000	0.0000	0.0000	0.0000
67	0	0.0028	-0.0028	0.0000	0.0000	0.0000	0.0000
68	0	0.0022	-0.0022	0.0000	0.0000	0.0000	0.0000
			Sum	24.4738	2.7031	-0.7015	1.6493
Sum of squared errors (Sum/2)				12.2369			

Figuring by: John D. Kelleher, et al, "Fundamentals of Machine Learning for Predictive Data Analytics - Algorithms, Worked Examples, and Case Studies," *MIT Press*, 2015.

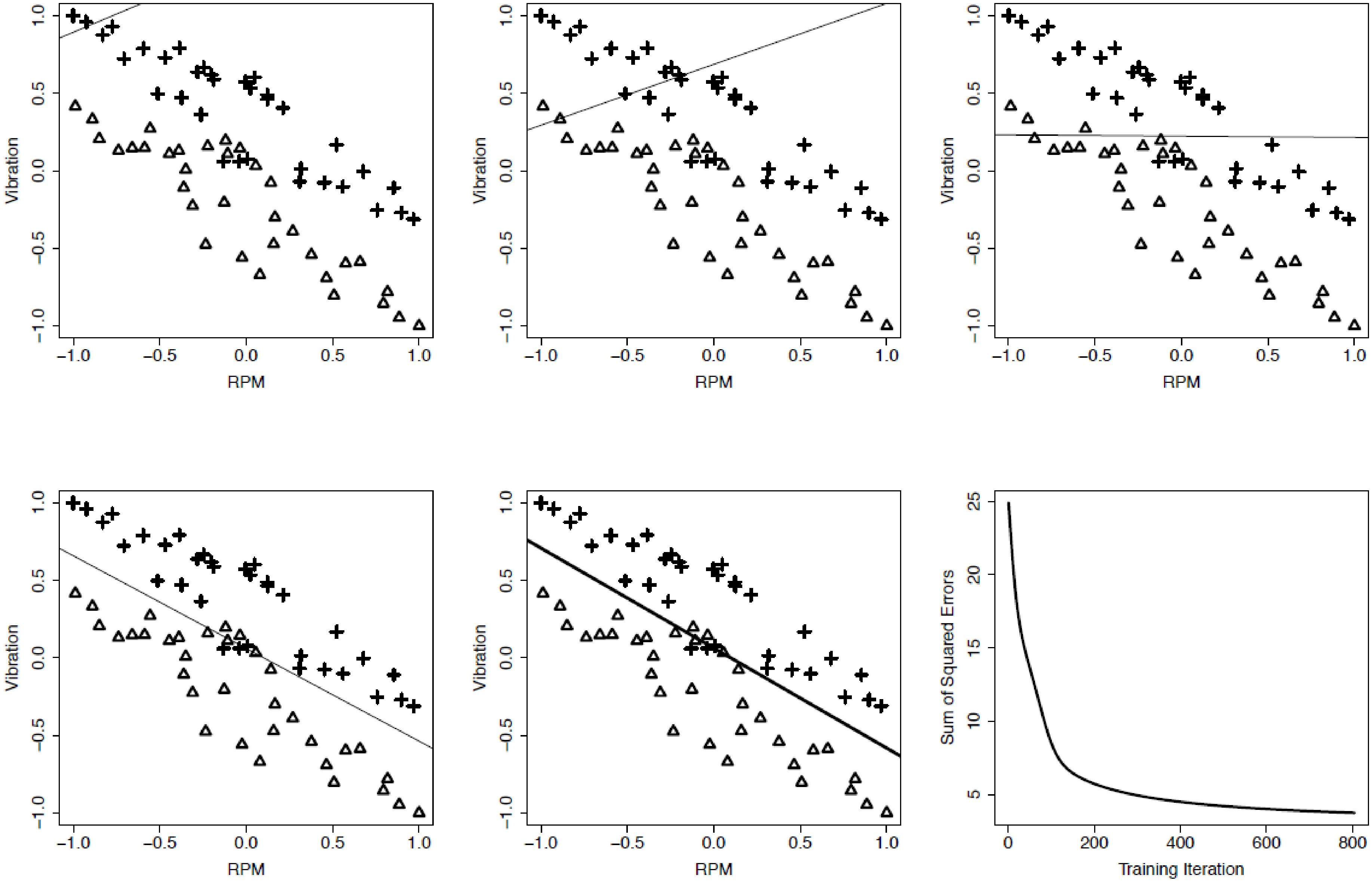
Logistic Regression

- Example
  - After the 1st iteration, all *w* are updated once,
  - $\mathbf{w} = [-2.8924 \quad -1.0287 \quad -2.194]$

ID	TARGET LEVEL	Pred.	Error	Squared Error	w_0'	w_1'	w_2'
1	1	0.5817	0.4183	0.1749	0.1018	-0.1018	0.1018
2	1	0.5414	0.4586	0.2103	0.1139	-0.1053	0.1094
3	1	0.4704	0.5296	0.2805	0.1319	-0.1094	0.1155
4	1	0.4867	0.5133	0.2635	0.1282	-0.0992	0.1194
				...			
65	0	0.0037	-0.0037	0.0000	0.0000	0.0000	0.0000
66	0	0.0043	-0.0043	0.0000	0.0000	0.0000	0.0000
67	0	0.0028	-0.0028	0.0000	0.0000	0.0000	0.0000
68	0	0.0022	-0.0022	0.0000	0.0000	0.0000	0.0000
			Sum	24.0524	2.7236	-0.6646	1.6484
			Sum of squared errors (Sum/2)	12.0262			

Figuring by: John D. Kelleher, et al, "Fundamentals of Machine Learning for Predictive Data Analytics - Algorithms, Worked Examples, and Case Studies," *MIT Press*, 2015.

Logistic Regression



Figuring by: John D. Kelleher, et al, "Fundamentals of Machine Learning for Predictive Data Analytics - Algorithms, Worked Examples, and Case Studies," *MIT Press*, 2015.

Logistic Regression

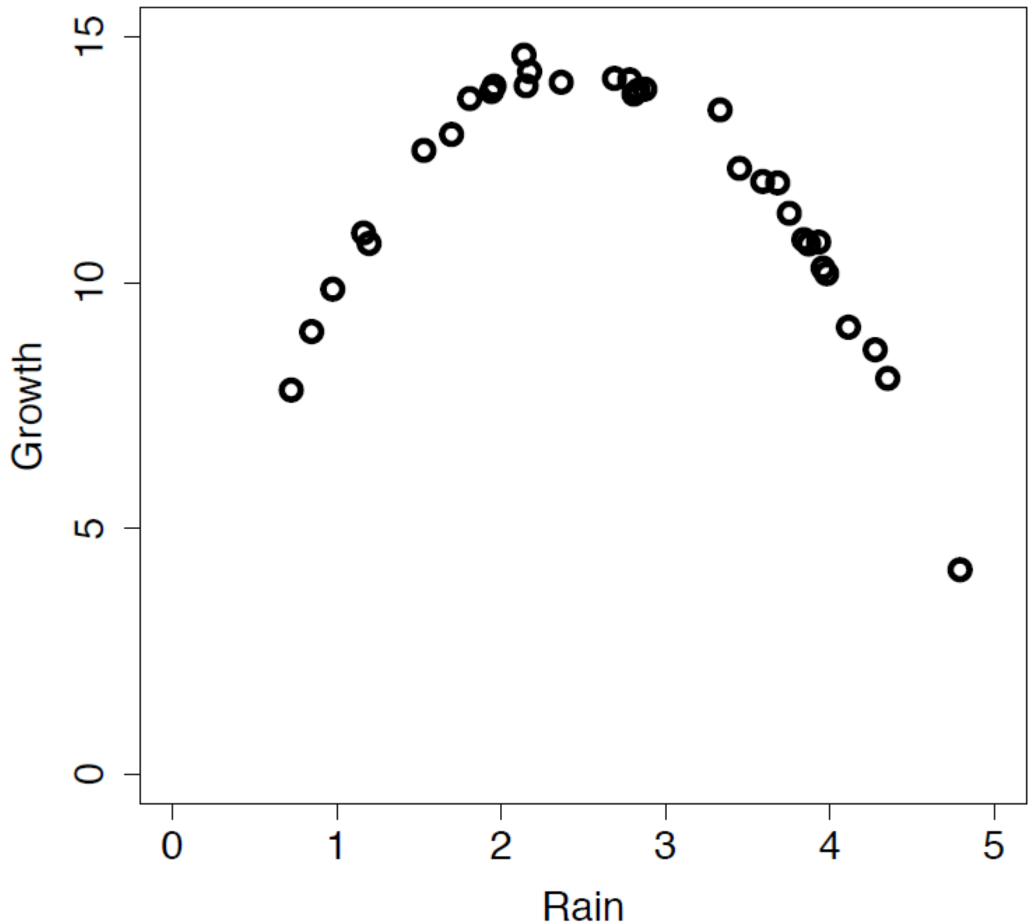
- Modeling non-linear relationships
- Example: Grass growth on Irish farms during July 2012.

ID	RAIN	GROWTH	ID	RAIN	GROWTH	ID	RAIN	GROWTH
1	2.153	14.016	12	3.754	11.420	23	3.960	10.307
2	3.933	10.834	13	2.809	13.847	24	3.592	12.069
3	1.699	13.026	14	1.809	13.757	25	3.451	12.335
4	1.164	11.019	15	4.114	9.101	26	1.197	10.806
5	4.793	4.162	16	2.834	13.923	27	0.723	7.822
6	2.690	14.167	17	3.872	10.795	28	1.958	14.010
7	3.982	10.190	18	2.174	14.307	29	2.366	14.088
8	3.333	13.525	19	4.353	8.059	30	1.530	12.701
9	1.942	13.899	20	3.684	12.041	31	0.847	9.012
10	2.876	13.949	21	2.140	14.641	32	3.843	10.885
11	4.277	8.643	22	2.783	14.138	33	0.976	9.876

Figuring by: John D. Kelleher, et al, "Fundamentals of Machine Learning for Predictive Data Analytics - Algorithms, Worked Examples, and Case Studies," *MIT Press*, 2015.

Logistic Regression

- The data scattering cannot be separated into two spaces
- Even the linear regression cannot be used



Figuring by: John D. Kelleher, et al, "Fundamentals of Machine Learning for Predictive Data Analytics - Algorithms, Worked Examples, and Case Studies," *MIT Press*, 2015.

Regression with basis functions

- $\mathbf{x} = [x_0\ x_1\ x_2\ \dots\ x_n]$, where $x_0 = 1$
- $\mathbf{w} = [w_0\ w_1\ \dots\ w_b]$, $k > 0$

$$y = w_0\phi_0(\mathbf{x}) + w_1\phi_1(\mathbf{x}) + \cdots + w_k\phi_b(\mathbf{x})$$

$$= \sum_{j=0}^{b} w_j\phi_j(\mathbf{x})$$

$$= \mathbf{w} \cdot \mathbf{\Phi}(\mathbf{x})$$

- where $\mathbf{\Phi}$ is a series of basis functions to transform $\mathbf{x}$

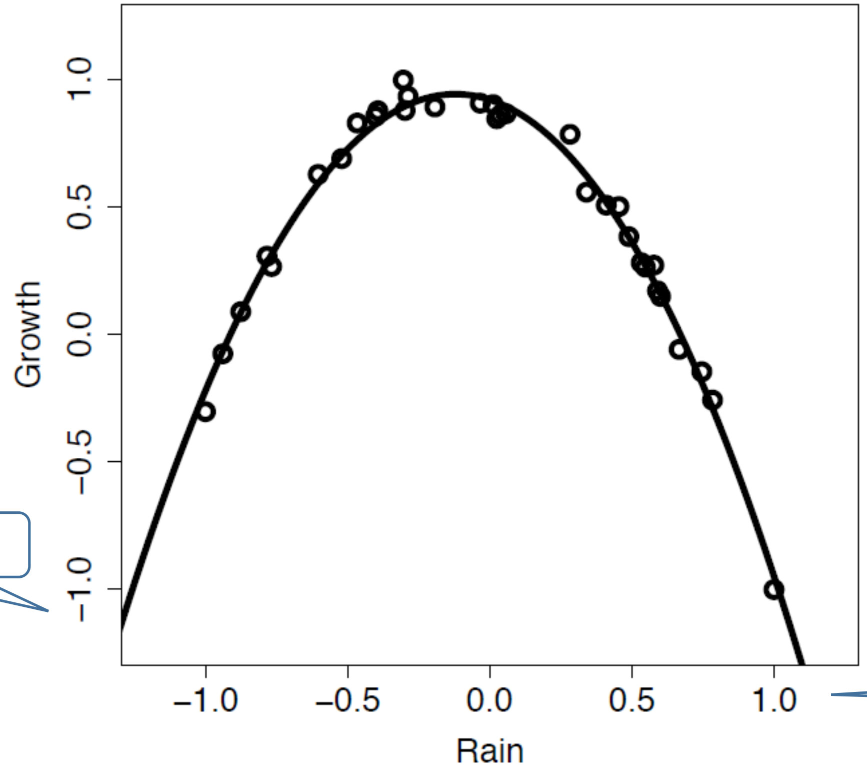
Regression with basis functions

- Example:

$$\phi_0(x) = 1$$
$$\phi_1(x) = x$$
$$\phi_2(x) = x^2$$

Using gradient descent, we have

$$\mathbf{w} = [0.3707 \quad 0.8475 \quad 1.717]$$



Normalized

Normalized

Figuring by: John D. Kelleher, et al, "Fundamentals of Machine Learning for Predictive Data Analytics - Algorithms, Worked Examples, and Case Studies," *MIT Press*, 2015.

Regression with basis functions

- Logistic regression model using basis functions

$$M_{\mathbf{w}}(\mathbf{x}) = L(\mathbf{w} \cdot \mathbf{\Phi}(\mathbf{x}))$$

$$= \frac{1}{1 + e^{-\sum_{j=0}^{k} w_j \phi_j(\mathbf{x})}}$$

Regression with basis functions

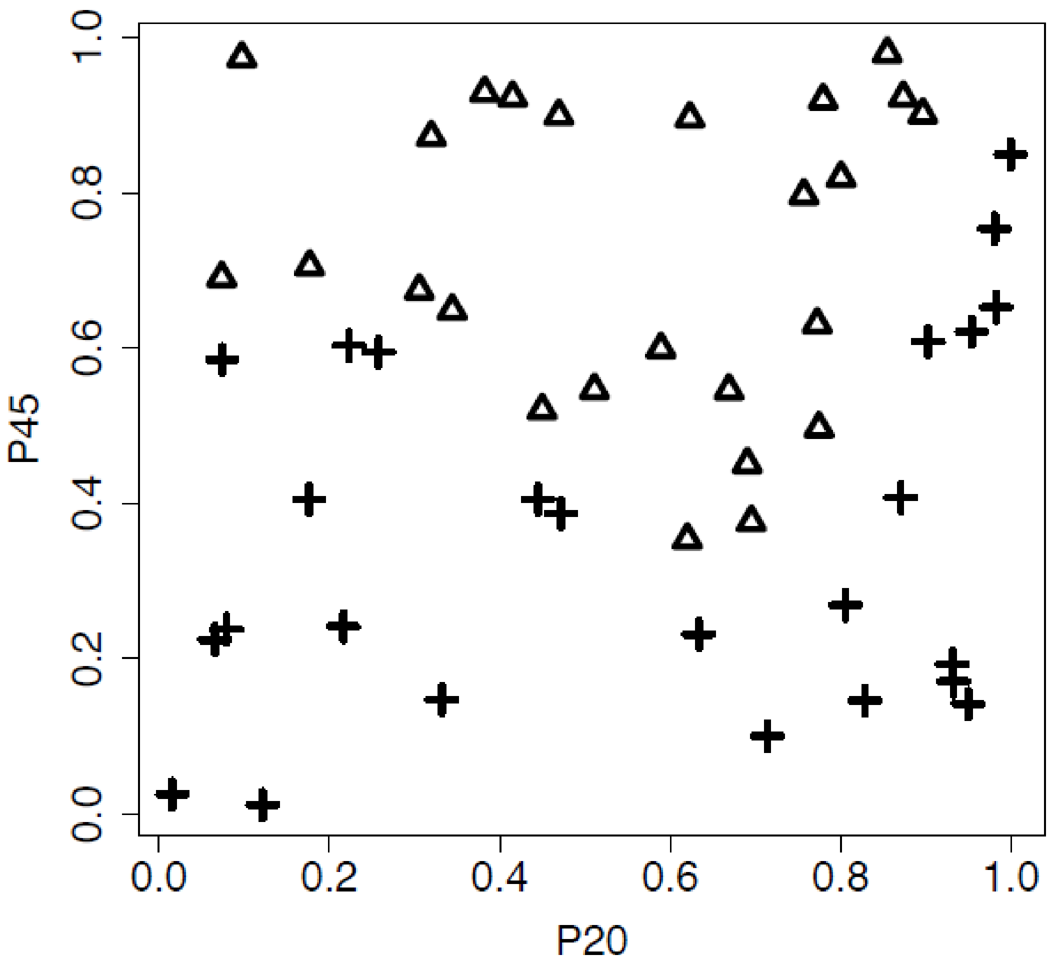
- Example: A dataset showing participants' responses to viewing 'positive' and 'negative' images measured on the EEG P20 and P45 potentials.

ID	P20	P45	TYPE	ID	P20	P45	TYPE
1	0.4497	0.4499	negative	26	0.0656	0.2244	positive
2	0.8964	0.9006	negative	27	0.6336	0.2312	positive
3	0.6952	0.3760	negative	28	0.4453	0.4052	positive
4	0.1769	0.7050	negative	29	0.9998	0.8493	positive
5	0.6904	0.4505	negative	30	0.9027	0.6080	positive
6	0.7794	0.9190	negative	31	0.3319	0.1473	positive
	⋮				⋮		

Figuring by: John D. Kelleher, et al, "Fundamentals of Machine Learning for Predictive Data Analytics - Algorithms, Worked Examples, and Case Studies," *MIT Press*, 2015.

Regression with basis functions

- Example:



+: Positive
Δ: Negative

Figuring by: John D. Kelleher, et al, "Fundamentals of Machine Learning for Predictive Data Analytics - Algorithms, Worked Examples, and Case Studies," *MIT Press*, 2015.

Regression with basis functions

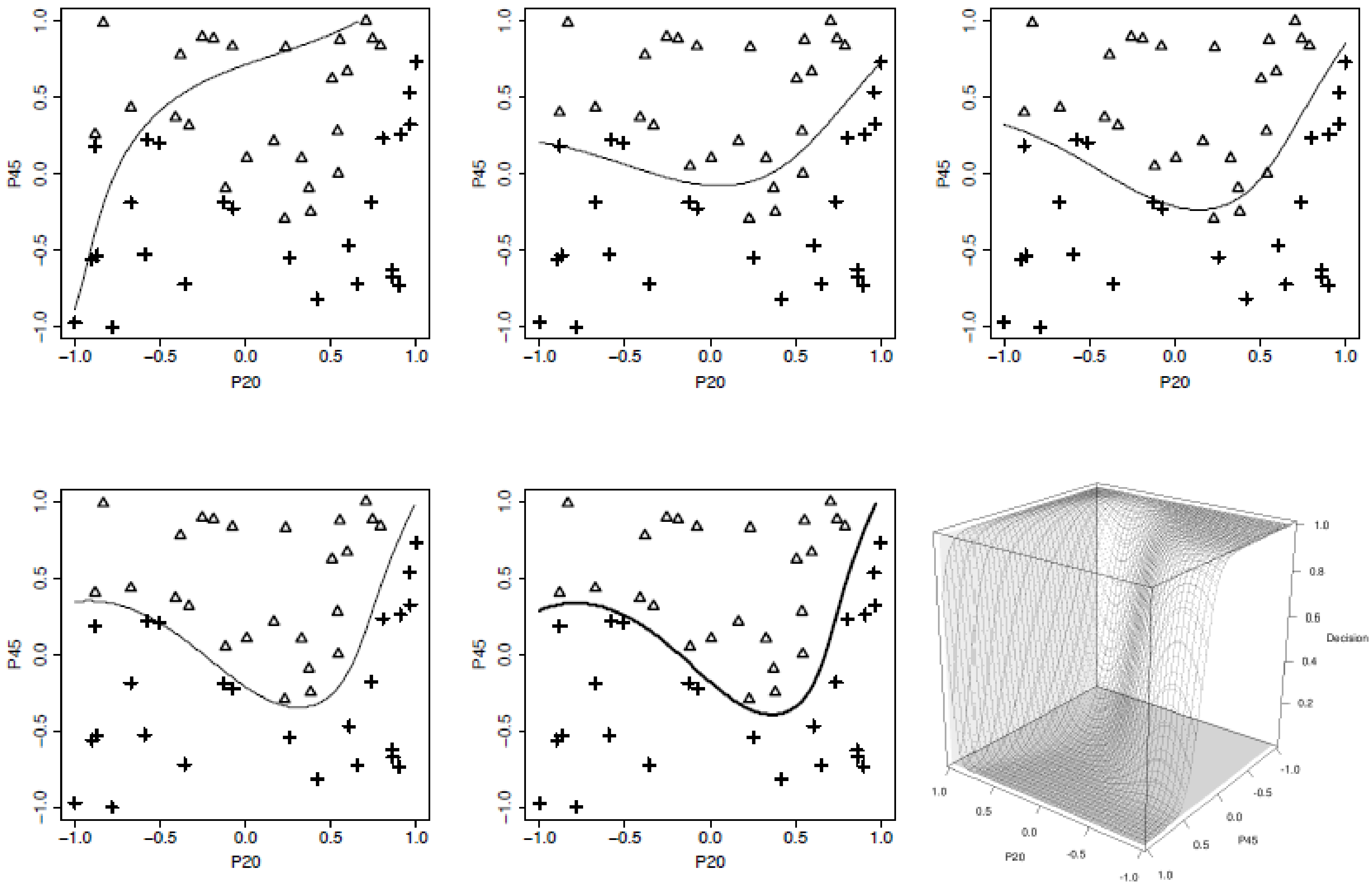
- Basis function:

$$\phi_0(x_0, x_1) = 1$$
$$\phi_1(x_0, x_1) = x_0$$
$$\phi_2(x_0, x_1) = x_1$$
$$\phi_3(x_0, x_1) = {x_0}^2$$
$$\phi_4(x_0, x_1) = {x_1}^2$$
$$\phi_5(x_0, x_1) = {x_0}^3$$
$$\phi_6(x_0, x_1) = {x_1}^3$$
$$\phi_7(x_0, x_1) = x_0 x_1$$

3rd-order polynomial surface

Regression with basis functions

- Gradient descent



Figuring by: John D. Kelleher, et al, "Fundamentals of Machine Learning for Predictive Data Analytics - Algorithms, Worked Examples, and Case Studies," *MIT Press*, 2015.

Multinomial Logistic Regression

- Example: A dataset of customers of a large national retail chain

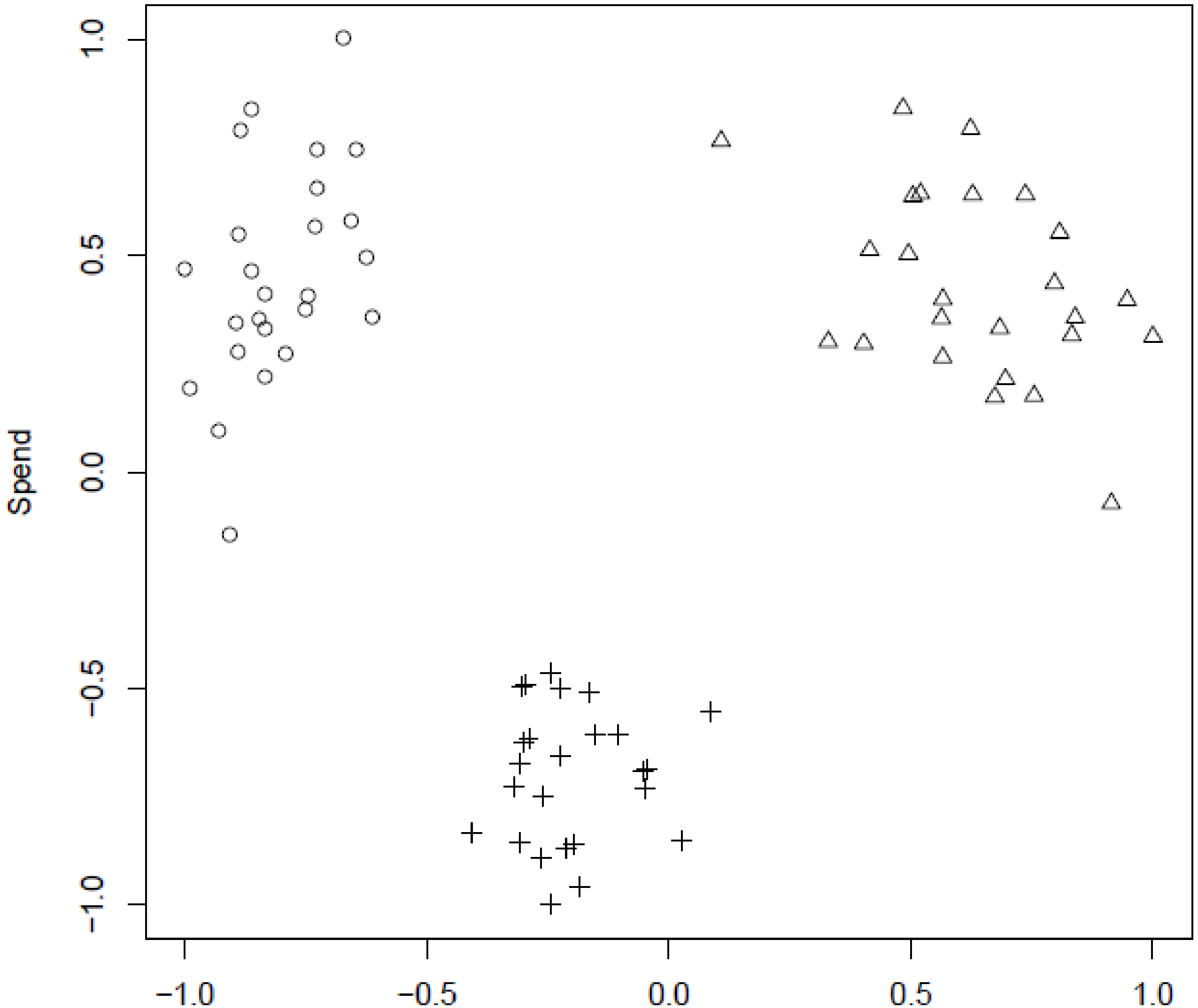
ID	SPEND	FREQ	TYPE
1	21.6	5.4	single
2	25.7	7.1	single
3	18.9	5.6	single
4	25.7	6.8	single
		⋮	
26	107.9	5.8	business
27	92.9	5.5	business

ID	SPEND	FREQ	TYPE
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

Figuring by: John D. Kelleher, et al, "Fundamentals of Machine Learning for Predictive Data Analytics - Algorithms, Worked Examples, and Case Studies," *MIT Press*, 2015.

Multinomial Logistic Regression

- Example: A dataset of customers of a large national retail chain



O: Single
+: Family
Δ: Business

Figuring by: John D. Kelleher, et al, "Fundamentals of Machine Learning for Predictive Data Analytics - Algorithms, Worked Examples, and Case Studies," *MIT Press*, 2015.

Multinomial Logistic Regression

- Given the *r* target levels $T = \{t_1 \quad t_2 \quad \ldots \quad t_r\}$
- *r* logistic regression models:
  - ➔Each of them is a one-versus-all separation

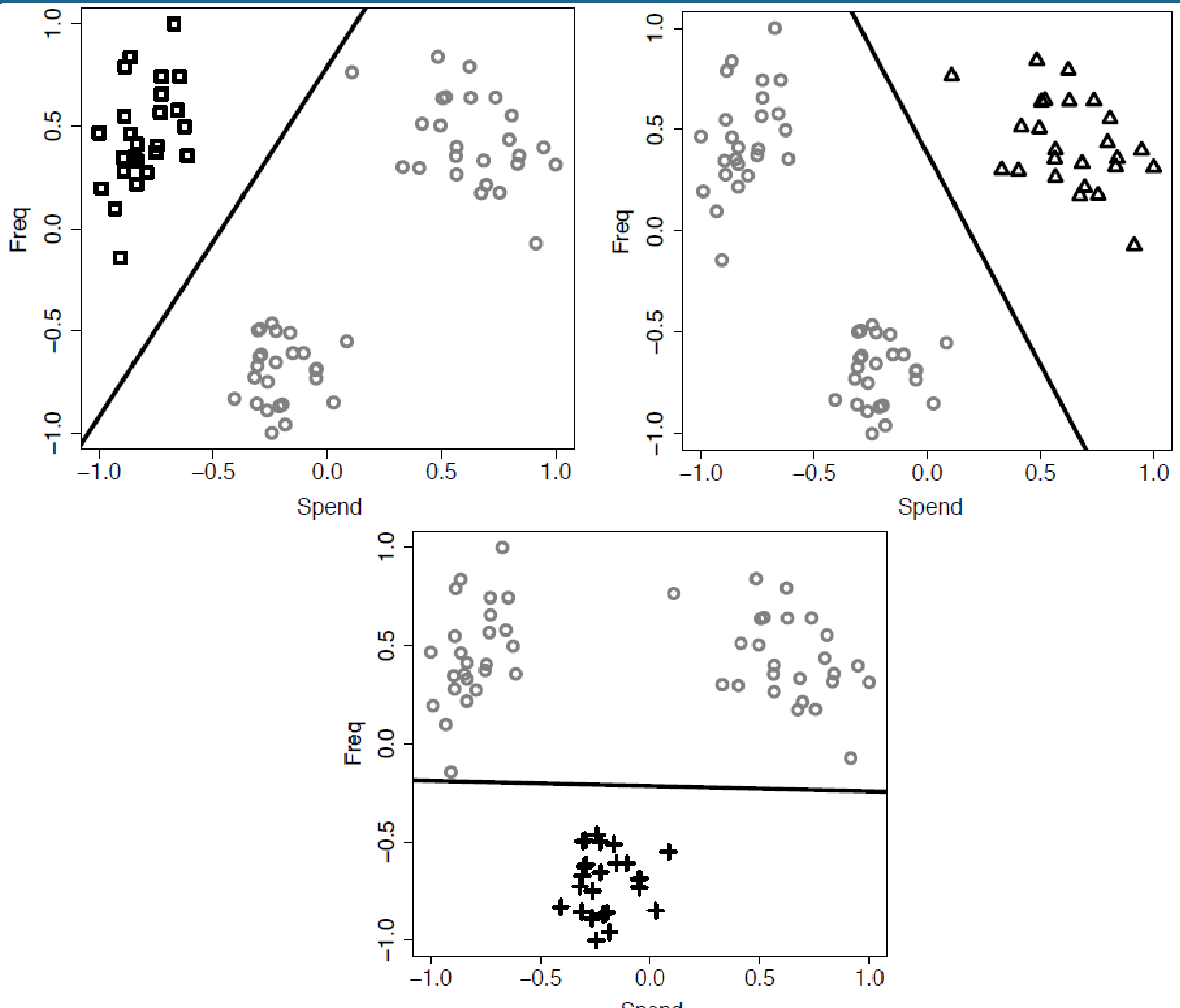
$$M_{\mathbf{w}_1}(\mathbf{x}) = L(\mathbf{w}_1 \cdot \mathbf{x})$$

$$M_{\mathbf{w}_2}(\mathbf{x}) = L(\mathbf{w}_2 \cdot \mathbf{x})$$

...

$$M_{\mathbf{w}_r}(\mathbf{x}) = L(\mathbf{w}_r \cdot \mathbf{x})$$

Multinomial Logistic Regression



Figuring by: John D. Kelleher, et al, "Fundamentals of Machine Learning for Predictive Data Analytics - Algorithms, Worked Examples, and Case Studies," *MIT Press*, 2015.

Multinomial Logistic Regression

- To combine the outputs of these different models, we normalize the result of each logistic model:

$$M'_{\mathbf{w}_k}(\mathbf{x}) = \frac{M_{\mathbf{w}_k}(\mathbf{x})}{\sum_{c=1}^{r} M_{\mathbf{w}_c}(\mathbf{x})}$$

- To verify each step of gradient descent, we can compute the L2 error of all training data instances

$$E(M_{\mathbf{w}_k}, \mathbf{X}) = \frac{1}{2}\sum_{i=1}^{m}\left(y_i - M'_{\mathbf{w}_k}(\mathbf{x}_i)\right)^2$$

  - where y_i is 0 if the target of $\mathbf{x}_i$ is t_k; otherwise y_i is 1.

Multinomial Logistic Regression

- The multinomial logistic model then is

$$M(\mathbf{x}) = \underset{t_k \in T}{\arg\max}\, M'_{\mathbf{w}_k}(\mathbf{x})$$

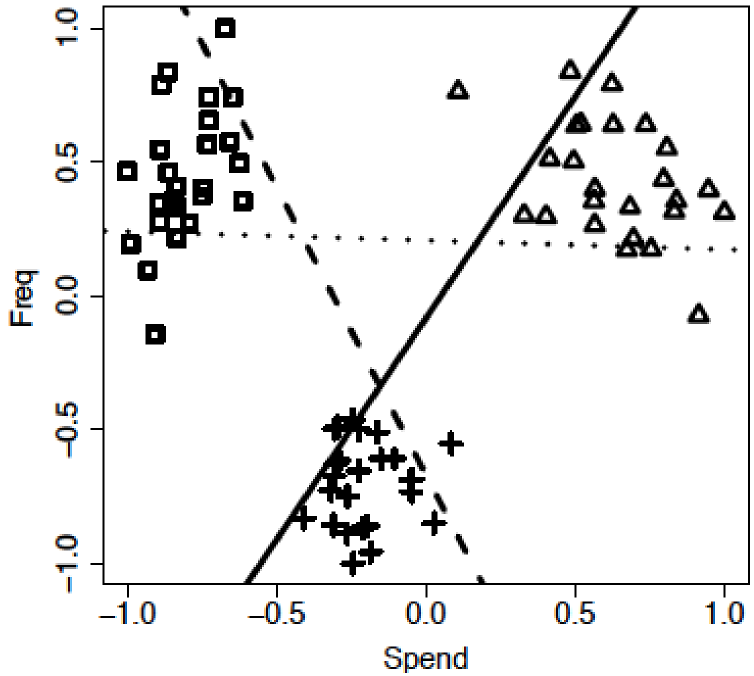
Multinomial Logistic Regression

- Example

$$Single(O):\ M_{\mathbf{w}_1}(\mathbf{x}) = L([0.7993 \quad -15.9 \quad 9.5974] \cdot \mathbf{x})$$

$$Family(+):\ M_{\mathbf{w}_2}(\mathbf{x}) = L([3.6526 \quad -0.58 \quad 17.5886] \cdot \mathbf{x})$$

$$Business(\Delta):\ M_{\mathbf{w}_3}(\mathbf{x}) = L([4.6419 \quad 14.94 \quad 6.9457] \cdot \mathbf{x})$$



Figuring by: John D. Kelleher, et al, "Fundamentals of Machine Learning for Predictive Data Analytics - Algorithms, Worked Examples, and Case Studies," *MIT Press*, 2015.

Multinomial Logistic Regression

- Example
  - **q** = [1 25.67 6.12] ➔ normalize ➔ **q** = [1 -0.7279 0.4789]

$$Single(O)\text{: } M_{\mathbf{w}_1}(\mathbf{q}) = 0.9999$$

$$Family(+)\text{: } M_{\mathbf{w}_2}(\mathbf{q}) = 0.01278$$

$$Business(\Delta)\text{: } M_{\mathbf{w}_3}(\mathbf{q}) = 0.0518$$

$$Single(O)\text{: } M'_{\mathbf{w}_1}(\mathbf{q}) = 0.9393$$

$$Family(+)\text{: } M_{\mathbf{w}_2}(\mathbf{q}) = 0.0120$$

$$Business(\Delta)\text{: } M_{\mathbf{w}_3}(\mathbf{q}) = 0.0487$$

Multinomial Logistic Regression

- Softmax

$$M_{\mathbf{w}_k}(\mathbf{x}) = \frac{e^{\mathbf{w}_k \cdot \mathbf{x}}}{\sum_{c=1}^{r} e^{\mathbf{w}_c \cdot \mathbf{x}}}$$

$$M(\mathbf{x}) = \arg\max_{t_k \in T} M_{\mathbf{w}_k}(\mathbf{x})$$

- But, what is the error function for minimizing with gradient descent?
  - Cross Entropy

Multinomial Logistic Regression

- Multinomial targets

$$precision(l) = \frac{TP(l)}{TP(l) + FP(l)}$$

$$recall(l) = \frac{TP(l)}{TP(l) + FN(l)}$$

  - where l is a target level

Multinomial Logistic Regression

- Average class accuracy

$$\frac{1}{|levels(t)|} \sum_{l \in levels(t)} \text{recall}_l$$

- Harmonic average class accuracy

$$\frac{1}{\frac{1}{|levels(t)|} \sum_{l \in levels(t)} \frac{1}{\text{recall}_l}}$$

Harmonic average of *n* numbers:

$$H = \frac{n}{\frac{1}{x_1} + \frac{1}{x_2} + \cdots + \frac{1}{x_n}}$$

Multinomial Logistic Regression

- Multinomial targets
  - Example: bacterial species identification

ID	Target	Prediction
1	durionis	fructosus
2	ficulneus	fructosus
3	fructosus	fructosus
4	ficulneus	ficulneus
5	durionis	durionis
6	pseudo.	pseudo.
7	durionis	fructosus
8	ficulneus	ficulneus
9	pseudo.	pseudo.
10	pseudo.	fructosus
11	fructosus	fructosus
12	ficulneus	ficulneus
13	durionis	durionis
14	fructosus	fructosus
15	fructosus	ficulneus

ID	Target	Prediction
16	ficulneus	ficulneus
17	ficulneus	ficulneus
18	fructosus	fructosus
19	durionis	durionis
20	fructosus	fructosus
21	fructosus	fructosus
22	durionis	durionis
23	fructosus	fructosus
24	pseudo.	fructosus
25	durionis	durionis
26	pseudo.	pseudo.
27	fructosus	fructosus
28	ficulneus	ficulneus
29	fructosus	fructosus
30	fructosus	fructosus

Figuring by: John D. Kelleher, et al, "Fundamentals of Machine Learning for Predictive Data Analytics - Algorithms, Worked Examples, and Case Studies," *MIT Press*, 2015.

Multinomial Logistic Regression

- Multinomial targets
  - Example: bacterial species identification

		Prediction				Recall
		'durionis'	*'ficulneus'*	*'fructosus'*	*'pseudo.'*	
Target	*'durionis'*	5	0	2	0	0.714
	'ficulneus'	0	6	1	0	0.857
	'fructosus'	0	1	10	0	0.909
	'pseudo.'	0	0	2	3	0.600
	Precision	1.000	0.857	0.667	1.000	

  - Harmonic average class accuracy

$$\frac{1}{\frac{1}{4}\left(\frac{1}{0.714}+\frac{1}{0.857}+\frac{1}{0.909}+\frac{1}{0.600}\right)}=\frac{1}{1.333}=75.000\%$$

Figuring by: John D. Kelleher, et al, "Fundamentals of Machine Learning for Predictive Data Analytics - Algorithms, Worked Examples, and Case Studies," *MIT Press*, 2015.

Logistic Regression in sklearn

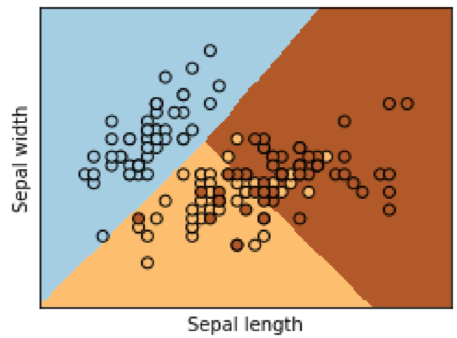
- Example of iris
  - https://github.com/jameschengcs/ml/blob/master/logistic.py

```
import numpy as np
from sklearn import linear_model, datasets

iris = datasets.load_iris()
X = iris.data[:, :2]  # we only take the first two features.
Y = iris.target

logreg = linear_model.LogisticRegression(C=1e5)
logreg.fit(X, Y)

# Create test data
x_min, x_max = X[:, 0].min() - .5, X[:, 0].max() + .5
y_min, y_max = X[:, 1].min() - .5, X[:, 1].max() + .5
xx, yy = np.meshgrid(np.arange(x_min, x_max, h),
                     np.arange(y_min, y_max, h))
Z = logreg.predict(np.c_[xx.ravel(), yy.ravel()])
```



Prediction scores

- all classification prediction models return a score
- score ≧ threshold ➔Positive
- otherwise ➔Negative

ID	Target	Prediction	Score	Outcome
7	ham	ham	0.001	TN
11	ham	ham	0.003	TN
15	ham	ham	0.059	TN
13	ham	ham	0.064	TN
19	ham	ham	0.094	TN
12	spam	ham	0.160	FN
2	spam	ham	0.184	FN
3	ham	ham	0.226	TN
16	ham	ham	0.246	TN
1	spam	ham	0.293	FN

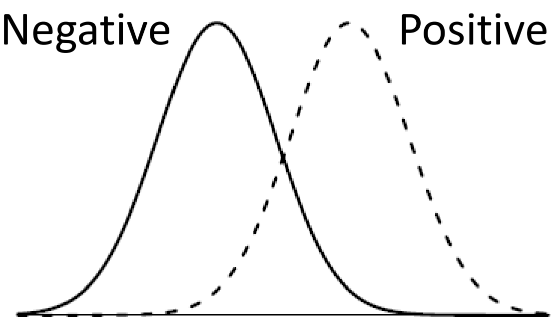
ID	Target	Prediction	Score	Outcome
5	ham	ham	0.302	TN
14	ham	ham	0.348	TN
17	ham	spam	0.657	FP
8	spam	spam	0.676	TP
6	spam	spam	0.719	TP
10	spam	spam	0.781	TP
18	spam	spam	0.833	TP
20	ham	spam	0.877	FP
9	spam	spam	0.960	TP
4	spam	spam	0.963	TP

Figuring by: John D. Kelleher, et al, "Fundamentals of Machine Learning for Predictive Data Analytics - Algorithms, Worked Examples, and Case Studies," *MIT Press*, 2015.

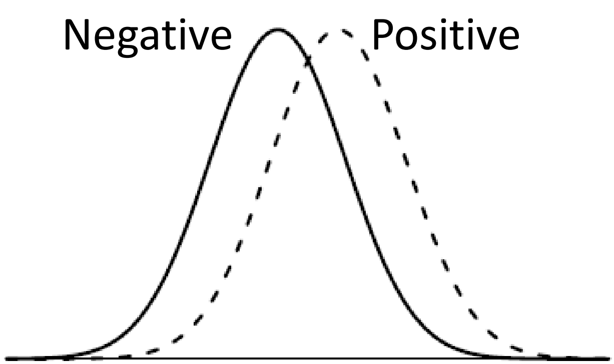
spam is positive and **ham is negative** in this case
Target is spam and prediction is also spam➔TP
Target is spam and but prediction is ham➔FN
Target is ham and prediction is also ham➔TN
Target is ham and but prediction is spam ➔FP

Prediction scores

- How well the distributions of scores produced by the model for different target levels are separated?
- Which model is better?
  - Model 1



  - Model 2



Figuring by: John D. Kelleher, et al, "Fundamentals of Machine Learning for Predictive Data Analytics - Algorithms, Worked Examples, and Case Studies," *MIT Press*, 2015.

Performance

- Prediction scores
  - Threshold increases TPR decreases
  - TP rate (TPR) = TP / (TP + FN)
    - Threshold = 0.0 ➔ Every thing is positive ➔ FN = 0
  - Threshold increases TNR increases
  - TN rate (TNR) = TN / (TN + FP)
    - Threshold = 0.0 ➔ No negative ➔ TN = 0

Performance

- Different thresholds generate different performances
- Example:

(a) Threshold: 0.75

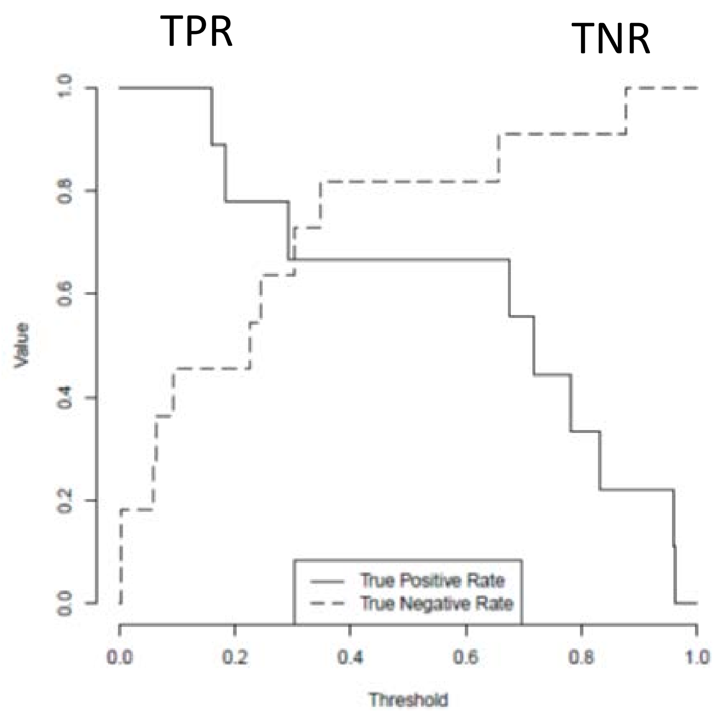
		Prediction 'spam'	**Prediction** 'ham'
Target	*'spam'*	4	4
	'ham'	2	10

(b) Threshold: 0.25

		Prediction 'spam'	**Prediction** 'ham'
Target	*'spam'*	7	2
	'ham'	4	7

Performance

- Changing values of TPR and TNR

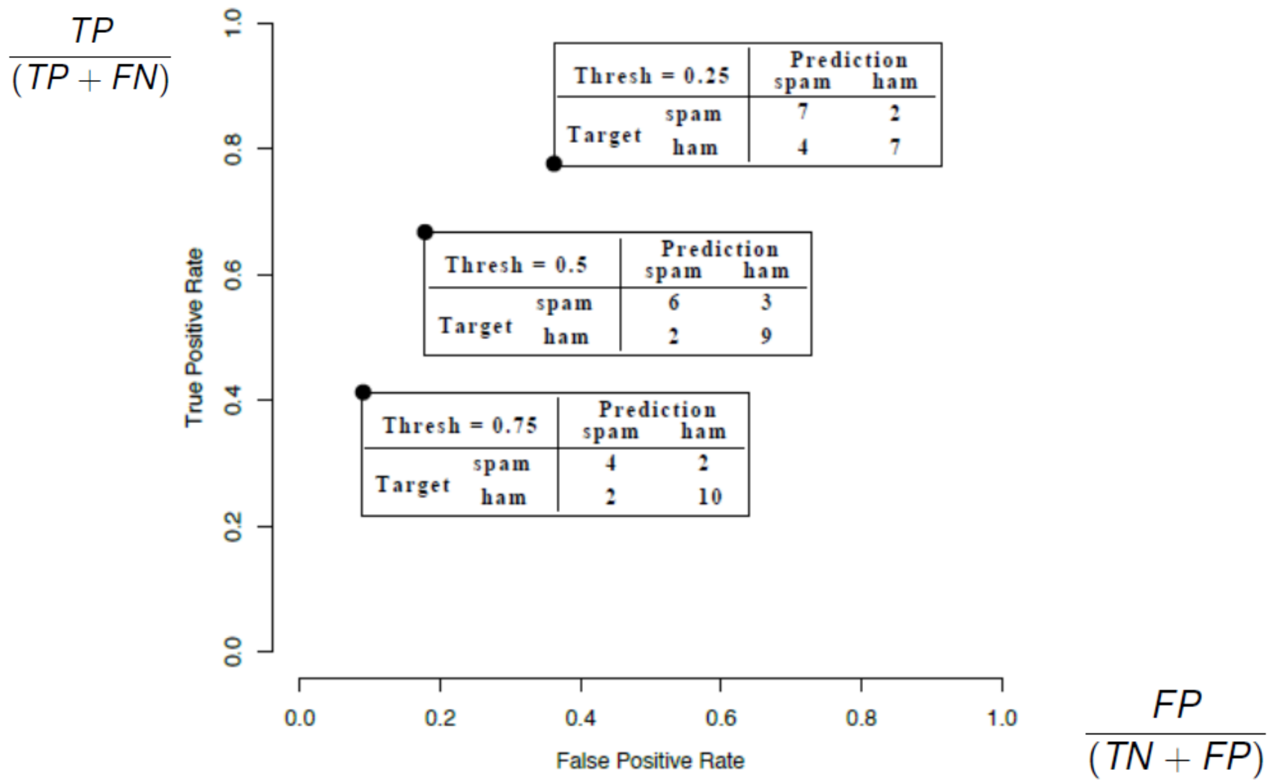


Figuring by: John D. Kelleher, et al, "Fundamentals of Machine Learning for Predictive Data Analytics - Algorithms, Worked Examples, and Case Studies," *MIT Press*, 2015.

TP rate (TPR) = TP / (TP + FN)
TN rate (*TNR*) = TN / (TN + FP)

ROC Curve

- Receiver operating characteristic curve (**ROC curve**)



Figuring by: John D. Kelleher, et al, "Fundamentals of Machine Learning for Predictive Data Analytics - Algorithms, Worked Examples, and Case Studies," *MIT Press*, 2015.

ROC Curve

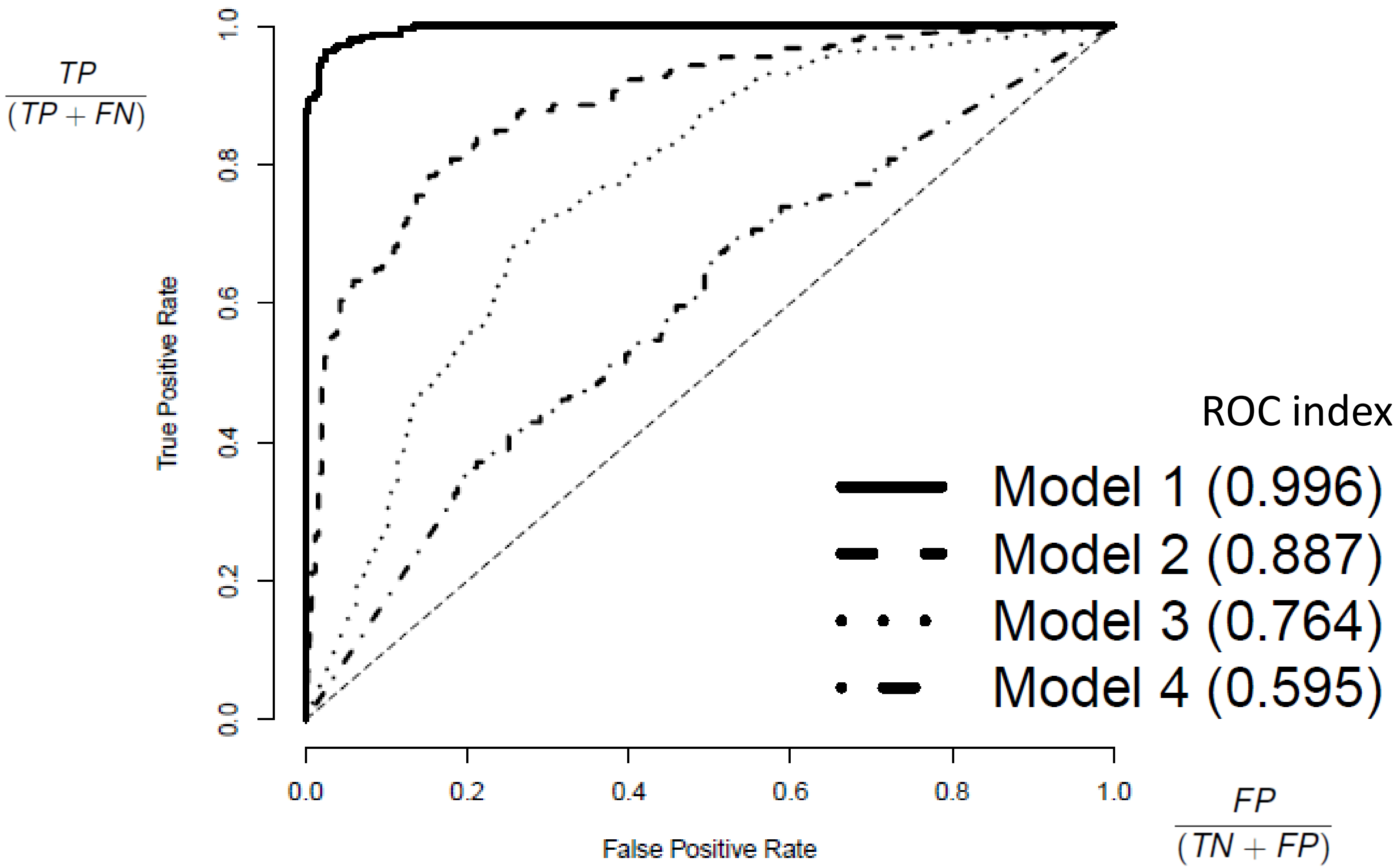
- ROC index
  - Given a set of thresholds $T = \{t_1, t_2, \ldots, t_m\}$

$$R = \sum_{i=2}^{m} \frac{(FPR(t_i) - FPR(t_{i-1}))(TPR(t_i) + TPR(t_{i-1}))}{2}$$

  - R is above 0.7 that indicates a strong model; otherwise, weak model

ROC Curve

- ROC curve



Figuring by: John D. Kelleher, et al, "Fundamentals of Machine Learning for Predictive Data Analytics - Algorithms, Worked Examples, and Case Studies," *MIT Press*, 2015.

Cross entropy

- Definition

$$H(p,q) = -\sum_{x} p(x)\log q(x)$$

  - where *p* and *q* are two probabilities over the same underlying set of events measures
    - for example:
      - $p \in \{p(\text{Positive}), p(\text{Negative})\}$
      - and also $q \in \{q(\text{Positive}), q(\text{Negative})\}$

Cross entropy

- Cross entropy can be an error function for logistic regression

$$H(p,q) = -\sum_{x} p(x)\log q(x)$$

  - *p:* target of training data
    - $p \in \{p(\text{Positive}) = y, p(\text{Negative}) = 1 - y\}$
  - *q*: The prediction of model
    - $q \in \{q(\text{Positive}) = M_{\mathbf{w}}(\mathbf{x}), q(\text{Negative}) = 1 - M_{\mathbf{w}}(\mathbf{x})\}$

$$H(p,q) = -y\log M_{\mathbf{w}}(\mathbf{x}) - (1-y)\log(1 - M_{\mathbf{w}}(\mathbf{x}))$$

Cross entropy

- 100 % accuracy
  - y = 1 and $M_{\mathbf{w}}(\mathbf{x})$ also predicts 1

$$H(p,q) = -1\log 1 - (1-1)\log(1-1) = \mathbf{0}$$

  - y = 0 and $M_{\mathbf{w}}(\mathbf{x})$ also predicts 0

$$H(p,q) = -0\log 0 - (1-0)\log(1-0) = \mathbf{0}$$

Cross entropy

- 50 % accuracy
  - y = 1 and $M_{\mathbf{w}}(\mathbf{x})$ predicts 0.5

$$H(p,q) = -1\log 0.5 - (1-1)\log(1-0.5) = \mathbf{-\log 0.5 > 0}$$

  - y = 0 and $M_{\mathbf{w}}(\mathbf{x})$ predicts 0.5

$$H(p,q) = -0\log 0.5 - (1-0)\log(1-0.5) = \mathbf{-\log 0.5 > 0}$$

- Accuracy and cross entropy are in inverse proportion

Cross entropy

- For *m* training data instances

$$G(\mathbf{w}) = \frac{1}{m}\sum_{i=1}^{m} -y_i \log M_{\mathbf{w}}(\mathbf{x}_i) - (1-y_i)\log(1-M_{\mathbf{w}}(\mathbf{x}_i))$$

  - where $M_{\mathbf{w}}(\mathbf{x}) = L(\mathbf{w}\cdot\mathbf{x}) = \frac{1}{1+e^{-\mathbf{w}\cdot\mathbf{x}}}$
  - Finding a **w** such that *G*(**w**) is minimum
  - *G*(**w**) is convex
    - A function $f(x)$ is convex if $\frac{d^2 f(x)}{dx} \geq 0$
    - $\frac{d^2 G(\mathbf{w})}{d\mathbf{w}} \geq 0$
    - The minimum of *G*(**w**) is existed and unique.

Cross entropy

- Minimizing cross entropy by gradient descent
  - Let ln be the logarithm and $L = M_{\mathbf{w}}(\mathbf{x}_i)$

$$g_i(L) = -y_i \ln L - (1 - y_i)\ln(1 - L)$$

  - Then, $$G(L) = \frac{1}{m}\sum_{i=1}^{m} g_i(L)$$

$$\frac{\partial g_i(L)}{\partial L} = \frac{\partial(-y_i \ln L)}{\partial L} + \frac{\partial(-(1 - y_i)\ln(1 - L))}{\partial L}$$

$$\frac{d\ln x}{dx} = \frac{1}{x}$$

$$= \frac{-y_i}{L} + \frac{1 - y_i}{1 - L}$$

$$= \frac{L - y_i}{L(1 - L)}$$

Cross entropy

- Minimizing cross entropy by gradient descent

let $\mathbf{u} = \mathbf{w} \cdot \mathbf{x}$

$$\frac{\partial g_i(L)}{\partial \mathbf{u}} = \frac{\partial g_i(L)}{\partial L} \frac{L}{\partial \mathbf{u}}$$

$$\frac{dL(x)}{dx} = L(x)(1 - L(x))$$

$$\frac{\partial g_i(L)}{\partial \mathbf{u}} = \frac{L - y_i}{L(1 - L)} L(1 - L)$$

$$= L - y_i$$

$$= M_{\mathbf{w}}(\mathbf{x}_i) - y_i$$

Then, we have

$$\frac{\partial g_i(L)}{\partial w_j} = (L - y_i) x_{ij}$$

$$\frac{\partial g}{\partial w_j} = \frac{dg}{d\mathbf{u}} \frac{d\mathbf{u}}{dw_j}$$

and $$\frac{\partial G(L)}{\partial w_j} = \frac{1}{m} \sum_{i=1}^{m} (L - y_i) x_{ij}$$

Cross entropy

- Convex

$$\frac{\partial^2 g_i(L)}{\partial \mathbf{u}} = \frac{\partial (L - y_i)}{\partial \mathbf{u}}$$

$$= L(1 - L)$$

$$\frac{\partial^2 g_i(L)}{\partial w_j} = x_{ij} \frac{\partial (L - y_i)}{\partial w_j}$$

$$= x_{ij}^2 L(1 - L)$$

$$\frac{dL(\mathbf{u})}{dw_j} = \frac{dL(\mathbf{u})}{d\mathbf{u}} \frac{d\mathbf{u}}{dw_j}$$

- Because $0.25 \geq L(1 - L) \geq 0$, the cross entropy error function is convex

Cross entropy

- Derivative of Softmax

$$\frac{d\frac{f(x)}{g(x)}}{dx} = \frac{\frac{df(x)}{dx}g(x) - f(x)\frac{dg(x)}{dx}}{g(x)g(x)}$$

$$\frac{de^x}{dx} = e^x$$

$$\frac{d(e^{x_1} + e^{x_2} + \cdots + e^{x_k} + \cdots + e^{x_n})}{dx_k} = e^{x_k}$$

$$S_{\mathbf{w}_k}(\mathbf{x}) = \frac{e^{\mathbf{w}_k \cdot \mathbf{x}}}{\sum_{c=1}^{r} e^{\mathbf{w}_c \cdot \mathbf{x}}}$$

$$\frac{\partial S_{\mathbf{w}_k}(\mathbf{x})}{\partial \mathbf{w}_k} = \frac{\partial \frac{e^{\mathbf{w}_k \cdot \mathbf{x}}}{\sum_{c=1}^{r} e^{\mathbf{w}_c \cdot \mathbf{x}}}}{\partial \mathbf{w}_k}$$

$$= \frac{e^{\mathbf{w}_k \cdot \mathbf{x}} \sum_{c=1}^{r} e^{\mathbf{w}_c \cdot \mathbf{x}} - e^{\mathbf{w}_k \cdot \mathbf{x}} e^{\mathbf{w}_k \cdot \mathbf{x}}}{\sum_{c=1}^{r} e^{\mathbf{w}_c \cdot \mathbf{x}} \sum_{c=1}^{r} e^{\mathbf{w}_c \cdot \mathbf{x}}}$$

$$= \frac{e^{\mathbf{w}_k \cdot \mathbf{x}}}{\sum_{c=1}^{r} e^{\mathbf{w}_c \cdot \mathbf{x}}}\left(1 - \frac{e^{\mathbf{w}_k \cdot \mathbf{x}}}{\sum_{c=1}^{r} e^{\mathbf{w}_c \cdot \mathbf{x}}}\right)$$

$$= S_{\mathbf{w}_k}(\mathbf{x})(1 - S_{\mathbf{w}_k}(\mathbf{x}))$$

similar to the logistic function

Cross entropy

- Derivative of Softmax

$$\frac{\partial S_{\mathbf{w}_k}(\mathbf{x})}{\partial \mathbf{w}_l} = \frac{\partial \frac{e^{\mathbf{w}_k \cdot \mathbf{x}}}{\sum_{c=1}^{r} e^{\mathbf{w}_c \cdot \mathbf{x}}}}{\partial \mathbf{w}_l}$$

$$= \frac{0 - e^{\mathbf{w}_k \cdot \mathbf{x}} e^{\mathbf{w}_l \cdot \mathbf{x}}}{\sum_{c=1}^{r} e^{\mathbf{w}_c \cdot \mathbf{x}} \sum_{c=1}^{r} e^{\mathbf{w}_c \cdot \mathbf{x}}}$$

$$= -S_{\mathbf{w}_k}(\mathbf{x}) S_{\mathbf{w}_l}(\mathbf{x})$$

Cross entropy

- Cross entropy of Softmax

$$g_i(S) = -\sum_{c=1}^{r} y_{ic} \ln s_{\mathbf{w}_c}(\mathbf{x}_i)$$

$$G(S) = \sum_{i=1}^{m} \sum_{c=1}^{r} (-y_{ic} \ln s_{\mathbf{w}_c}(\mathbf{x}_i))$$

Cross entropy

- Derivative of *g*

$$\frac{\partial g_i(S)}{\partial \mathbf{w}_k} = -\sum_{c=1}^{r} \frac{\partial y_{ic} \ln S_{\mathbf{w}_c}(\mathbf{x}_i)}{\partial \mathbf{w}_k} = -\sum_{c=1}^{r} y_{ic} \frac{\partial \ln S_{\mathbf{w}_c}(\mathbf{x}_i)}{\partial \mathbf{w}_k} = -\sum_{c=1}^{r} \frac{y_{ic}}{S_{\mathbf{w}_c}(\mathbf{x}_i)} \frac{\partial S_{\mathbf{w}_c}(\mathbf{x}_i)}{\partial \mathbf{w}_k}$$

$$= -\frac{y_{ik}}{S_{\mathbf{w}_k}(\mathbf{x}_i)} \frac{\partial S_{\mathbf{w}_k}(\mathbf{x}_i)}{\partial \mathbf{w}_k} - \sum_{\boldsymbol{c \neq k}}^{r} \frac{y_{ic}}{S_{\mathbf{w}_c}(\mathbf{x}_i)} \frac{\partial S_{\mathbf{w}_c}(\mathbf{x}_i)}{\partial \mathbf{w}_k}$$

$$\frac{d \ln x}{dx} = \frac{1}{x}$$

$$= -\frac{y_{ik}}{S_{\mathbf{w}_k}(\mathbf{x}_i)} S_{\mathbf{w}_k}(\mathbf{x}_i)\left(1 - S_{\mathbf{w}_k}(\mathbf{x}_i)\right) - \sum_{c \neq k}^{r} \frac{y_{ic}}{S_{\mathbf{w}_c}(\mathbf{x}_i)} \left(-S_{\mathbf{w}_k}(\mathbf{x}_i) S_{\mathbf{w}_c}(\mathbf{x}_i)\right)$$

$$= -y_{ik} + y_{ik} S_{\mathbf{w}_k}(\mathbf{x}_i) + \sum_{c \neq k}^{r} y_{ic} S_{\mathbf{w}_k}(\mathbf{x}_i)$$

$$F(x) = f\left(g(x)\right)$$
$$F'(x) = f'\left(g(x)\right) g'(x)$$

$$= -y_{ik} + S_{\mathbf{w}_k}(\mathbf{x}_i) \sum_{c=1}^{r} y_{ic}$$

$$= S_{\mathbf{w}_k}(\mathbf{x}_i) - y_{ik}$$

Cross entropy

- Derivative of *G*

$$\frac{\partial G(S)}{\partial \mathbf{w}_k} = \sum_{i=1}^{m} \left(S_{\mathbf{w}_k}(\mathbf{x}_i) - y_{ik} \right)$$

similar to the logistic function

Logistic Regression in tensorflow

- Example of MNIST
  - https://github.com/jameschengcs/ml/blob/master/logistic_tf.py