immerse
EDUCATION

Classification and model quality

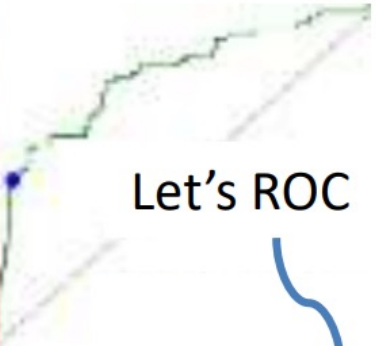
Elizabeth Savochkina | 12th June

Tuning your Machine

Shape matters

Making it easier to get fit

Cakeology

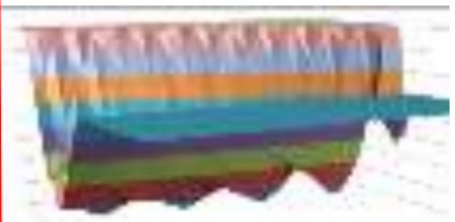


Let's ROC

Time to choose

More about classification

Some hills are too tough



Machine Learning Fundamentals

How to get fit

How do algorithms actually fit models to data?

How to get Fat

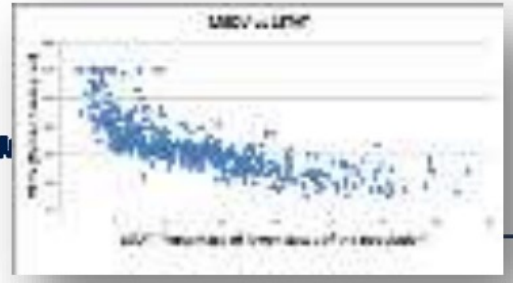
Working in higher-dimensions

Concerned about curves

Beyond straight-lines

Reducing others

Some curves are useful



Welcome to the Matrix

When Models go Bad


CNET

Your guide to a better future

Amazon facial recognition mistakenly confused 28 Congressmen with known criminals

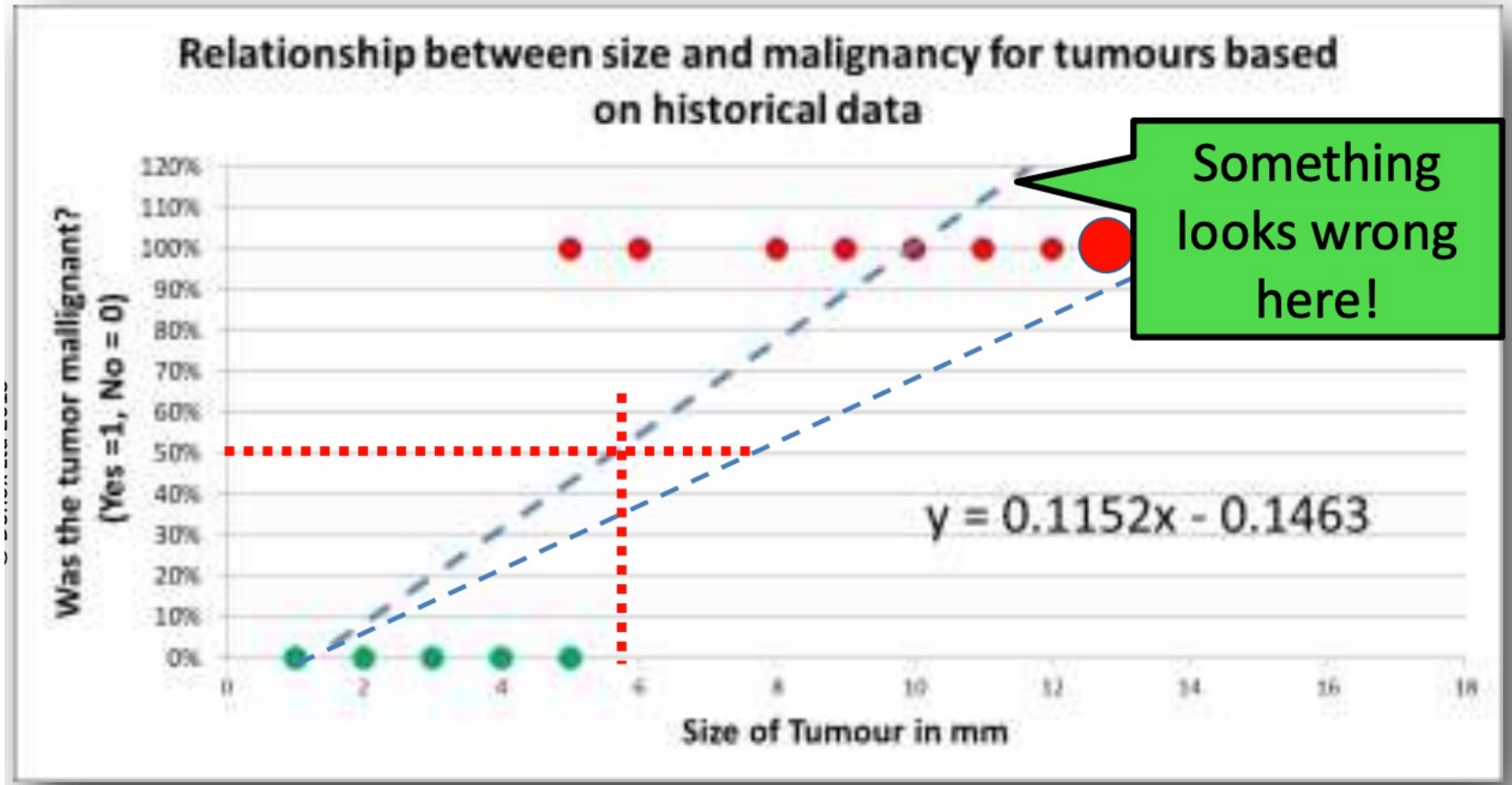
- In 2018 the American Civil Liberties Union (ACLU) used Amazon's Rekognition facial recognition technology to compare 25,000 pictures of criminals to those members of US Congress ..
- And came up with 28 matches!

Source: https://www.cnet.com/news/privacy/amazon-facial-recognition-thinks-28-congressmen-look-like-known-criminals-at-default-settings/

Predicting medical conditions

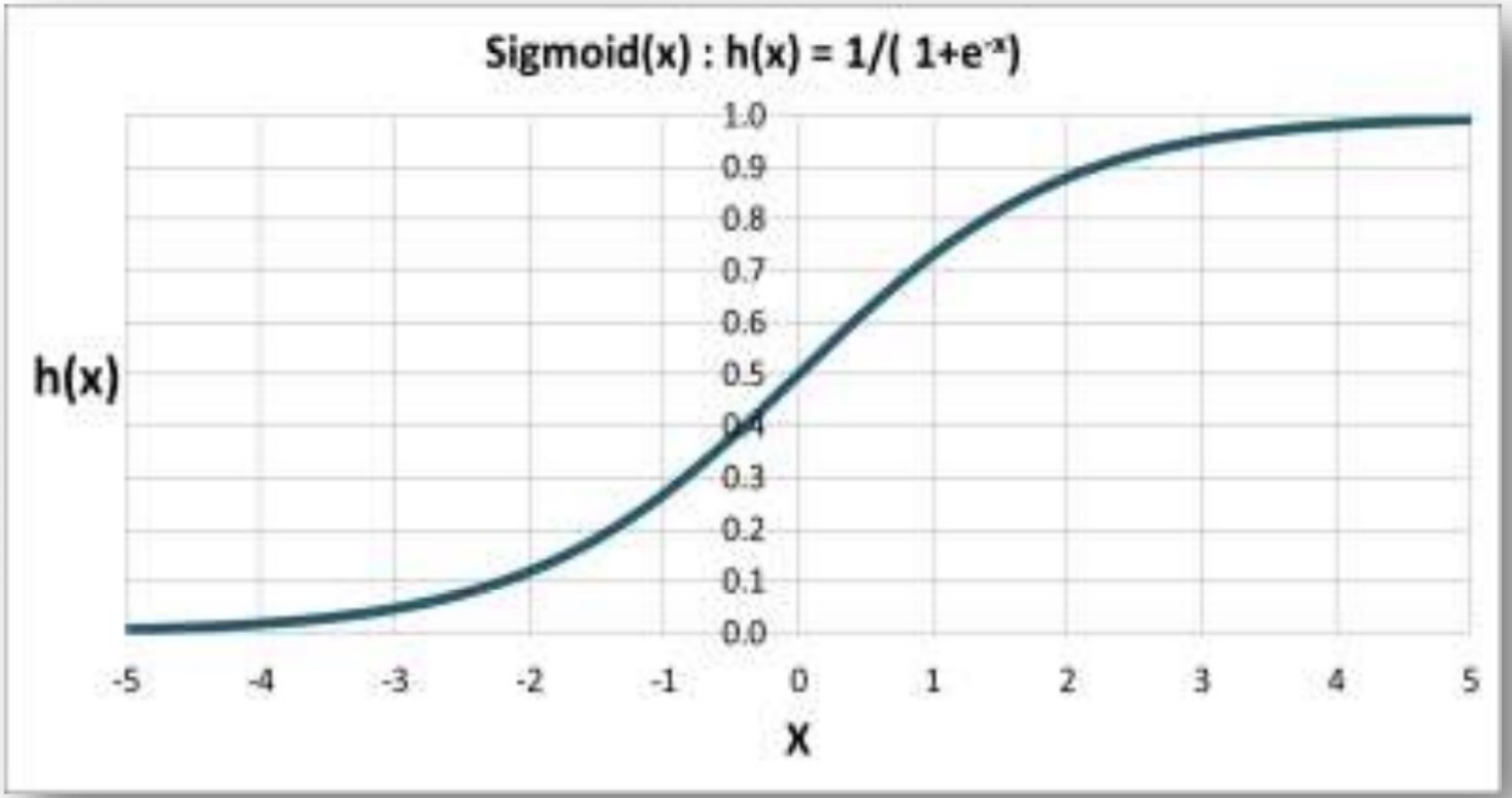
Consider the problem of predicting the probability of a tumour being malignant based on its size

Collect and review historical data



- Here we have some historical data showing the relationship between ‘Tumour size’ and malignancy
- .. So let’s just do a linear regression to make a prediction!

An ‘S’ shaped (sigmoid) curve would be better



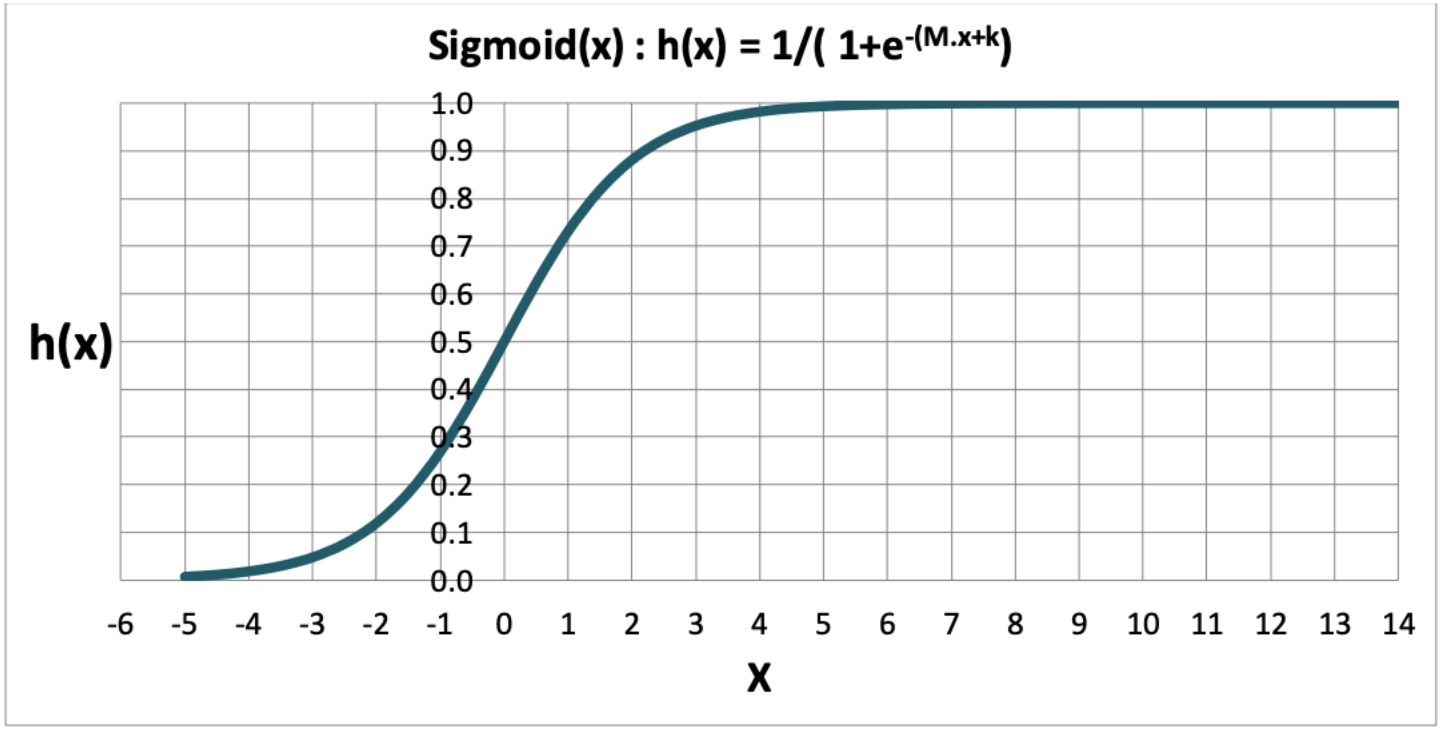
A sigmoid curve seems to be a better shape for model ..

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

‘h’ for hypothesis

We need to modify its shape to fit the data points

k	M
0	1



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

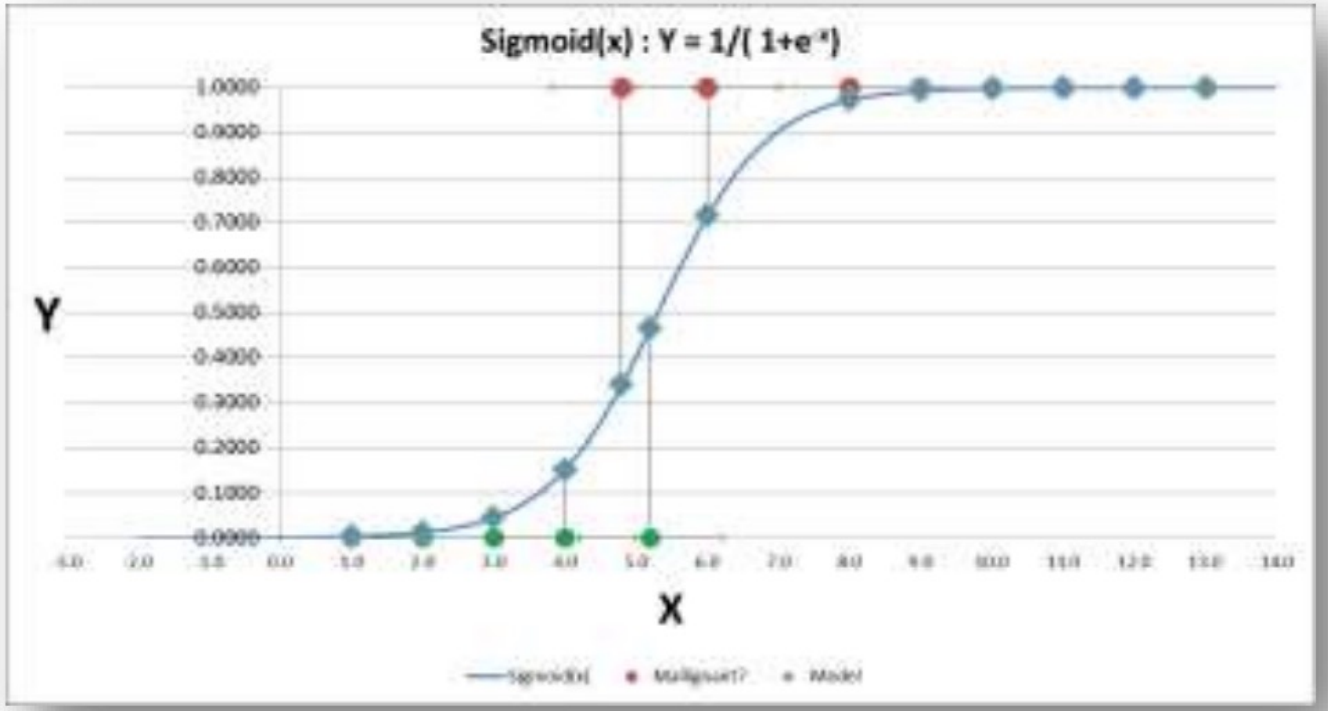
$$h(x) = \frac{1}{1 + e^{-(M.x+k)}}$$

We need a measurement to optimise a model ..

1 The Model

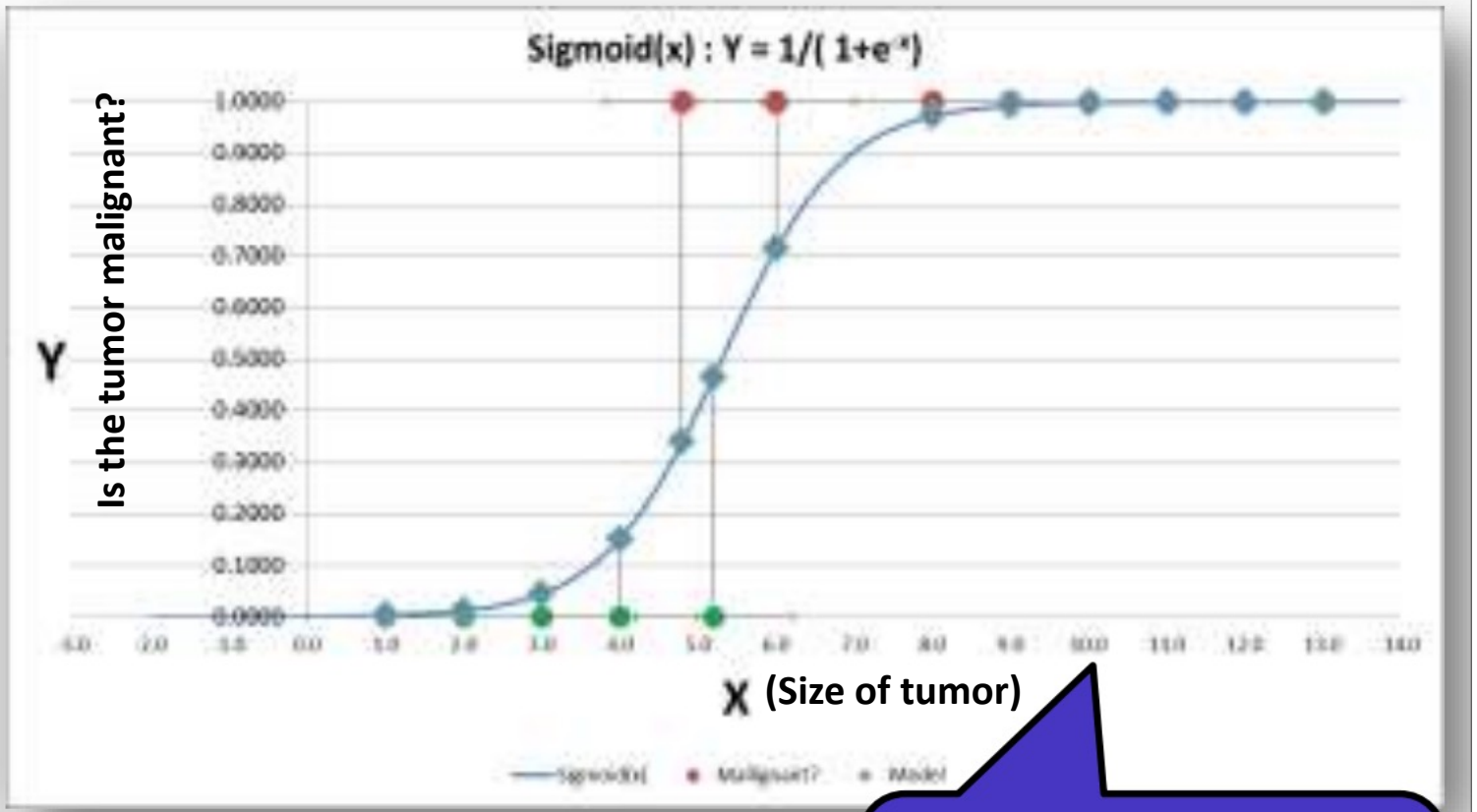
$$h(x) = \frac{1}{1 + e^{-(M.x+k)}}$$

2 How well the model matches the data



Fitting the model by minimizing the mean square error

Size of cancer cell (mm)	Mallignant?	Mallignant?		Model	Sq Error
1	No	0%		0.00	0.0000
2	No	0%		0.00	0.0000
3	No	0%		0.01	0.0001
4	No	0%		0.02	0.0004
4.8	Yes	100%		0.05	0.9081
6	Yes	100%		0.15	0.7280
5.2	No	0%		0.07	0.0049
8	Yes	100%		0.58	0.1770
9	Yes	100%		0.80	0.0417
10	Yes	100%		0.92	0.0069
11	Yes	100%		0.97	0.0010
12	Yes	100%		0.99	0.0001
13	Yes	100%		1.00	0.0000
					1.8681



Threshold (mm) that defines the person has cancer = 4.7mm tumor size

- anything above the threshold should predict – yes (100%-1), as malignant
- A new observation at 5.2mm is predicted as NOT malignant – misprediction due to a wrong loss function!

Not a good method!
See next slide

Gradient descent on mean square error ..

However, we have an issue!

We can't tune the model by minimising the mean square error

Because the parameter space is not bowl-shaped

(Non-convex surface)

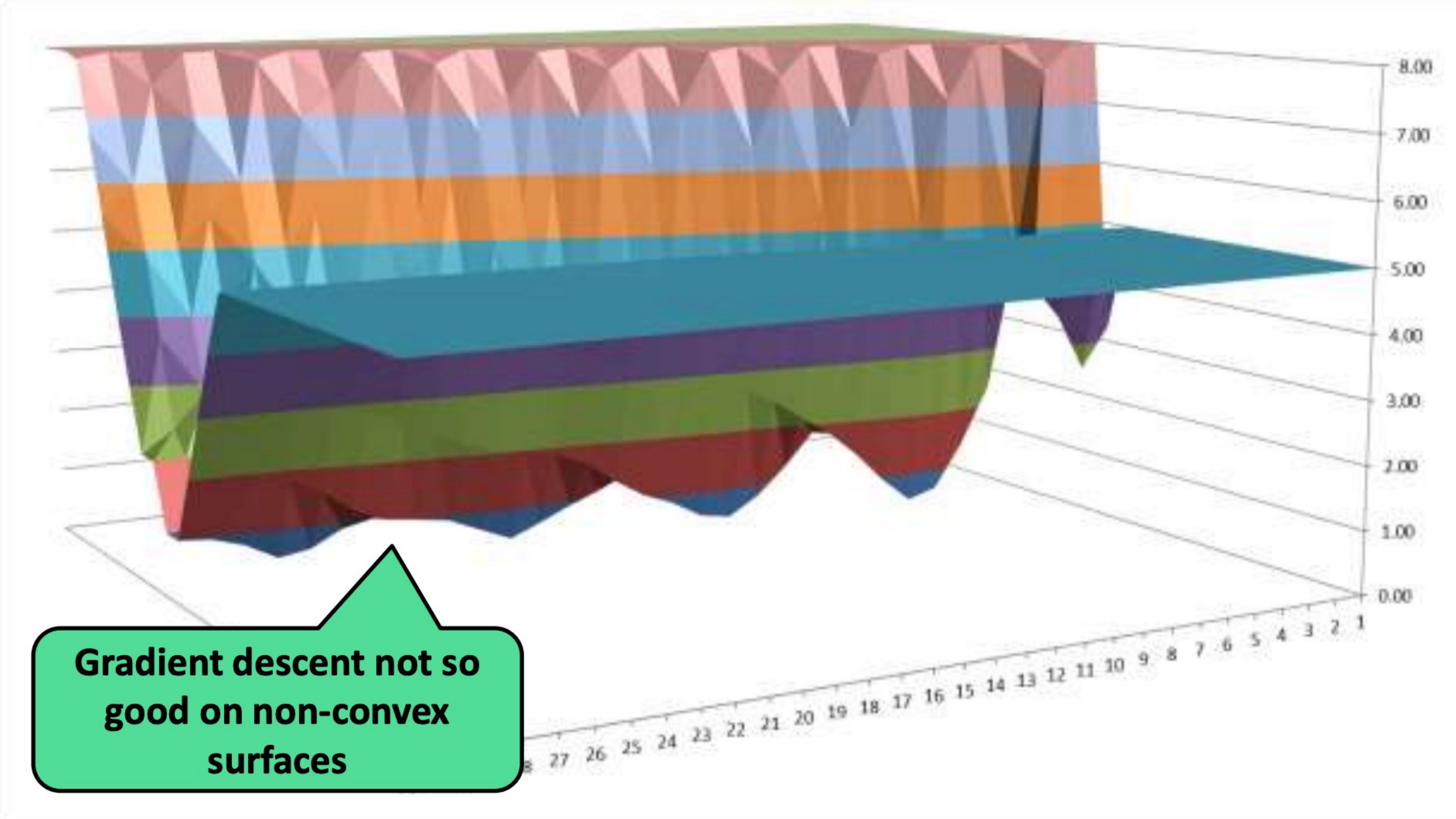
M \ k -->	-150	-140	-130	-120	-110	-100	-90	-80	-70	-60	-50	-40	-30	-20	-10	0	10	20	30	40	50	60	70	80	90	100	110	120
0.0	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	3.25	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00
0.3	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	7.98	2.58	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00
0.6	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	7.56	3.49	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00
0.9	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	4.55	4.05	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00
1.2	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	7.97	2.32	4.36	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00
1.5	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	7.09	1.41	4.55	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00
1.8	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	5.06	0.71	4.67	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00
2.1	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	7.86	3.56	0.56	4.76	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00
2.4	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	6.59	2.50	0.95	4.82	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00
2.7	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	7.98	5.24	1.98	1.44	4.87	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00
3.0	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	7.50	4.15	1.76	1.84	4.90	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00
3.3	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	6.31	3.28	1.25	2.15	4.93	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00
3.6	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	7.92	5.32	2.60	0.82	2.45	4.95	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00
3.9	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	7.03	4.52	2.05	0.53	2.72	4.96	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00
4.2	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	7.99	6.12	3.82	1.98	0.61	2.89	4.97	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00
4.5	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	7.66	5.37	3.11	1.91	0.87	3.01	4.98	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00
4.8	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	6.85	4.78	2.69	1.59	1.06	3.14	4.98	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00
5.1	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	7.95	6.01	4.04	2.10	1.10	1.35	3.29	4.99	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00
5.4	8.00	8.00	8.00	8.00	8.00	8.00	8.00	8.00	7.19	5.41	3.64	2.00	0.92	1.69	3.47	4.99	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00
5.7	8.00	8.00	8.00	8.00	8.00	8.00	8.00	7.99	6.69	4.91	3.02	1.99	0.70	1.89	3.64	4.99	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00
6.0	8.00	8.00	8.00	8.00	8.00	8.00	8.00	7.78	5.98	4.25	2.78	1.96	0.50	1.98	3.78	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00
6.3	8.00	8.00	8.00	8.00	8.00	8.00	8.00	6.99	5.44	3.93	2.16	1.81	0.70	2.05	3.87	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00
6.6	8.00	8.00	8.00	8.00	8.00	8.00	7.97	6.47	4.97	3.42	2.00	1.36	0.91	2.20	3.92	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00
6.9	8.00	8.00	8.00	8.00	8.00	8.00	7.33	5.97	4.53	3.00	2.00	1.03	0.99	2.45	3.96	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00
7.2	8.00	8.00	8.00	8.00	8.00	8.00	6.95	5.48	4.00	2.84	2.00	0.97	1.05	2.69	3.98	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00
7.5	8.00	8.00	8.00	8.00	8.00	7.85	6.25	4.99	3.85	2.25	1.99	0.86	1.25	2.85	3.99	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00
7.8	8.00	8.00	8.00	8.00	8.00	7.04	5.97	4.78	3.20	2.01	1.92	0.61	1.59	2.94	4.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00
8.1	8.00	8.00	8.00	8.00	7.98	6.89	5.51	4.07	2.99	2.00	1.64	0.53	1.84	2.97	4.01	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00
8.4	8.00	8.00	8.00	8.00	7.48	6.10	5.00	3.98	2.89	2.00	1.16	0.79	1.95	2.99	4.03	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00
8.7	8.00	8.00	8.00	8.00	6.99	5.97	4.91	3.71	2.36	2.00	1.01	0.94	1.98	3.00	4.04	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00
9.0	8.00	8.00	8.00	7.91	6.78	5.53	4.25	3.07	2.01	2.00	0.99	0.99	2.00	3.01	4.07	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00

Mean Squared Error Loss

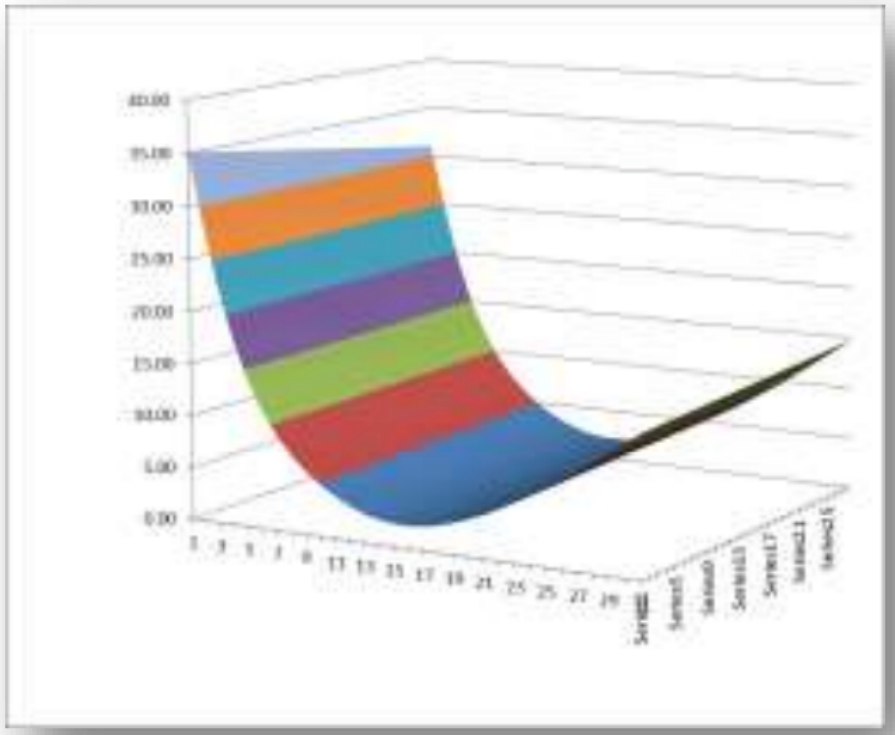
$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

If we say:
Actual value = 1, predicted = 0
MSE = (1 - 0)² = 1

So, the real class is = 1 but the prediction is 0.
MSE does not strongly penalize misclassifications *even for the perfect mismatch!*


8.00
7.00
6.00
5.00
4.00
3.00
2.00
1.00
0.00
27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1
Gradient descent not so good on non-convex surfaces

We need an alternative to mean square error ..



$$Cost = \begin{cases} -\log\big(h(x)\big), if\ y = 1, \\ -\log\big(1 - h(x)\big), if\ y = 0) \end{cases}$$

Definition of natural logarithm

When

$$e^y = x$$

Then base e logarithm of x is

$$\ln(x) = \log_e(x) = y$$

The e constant or Euler's number is:

$$e \approx 2.71828183$$

Ln of 0

The natural logarithm of zero is undefined:

$$\ln(0) \text{ is undefined}$$

The limit near 0 of the natural logarithm of x, when x approaches zero, is minus infinity:

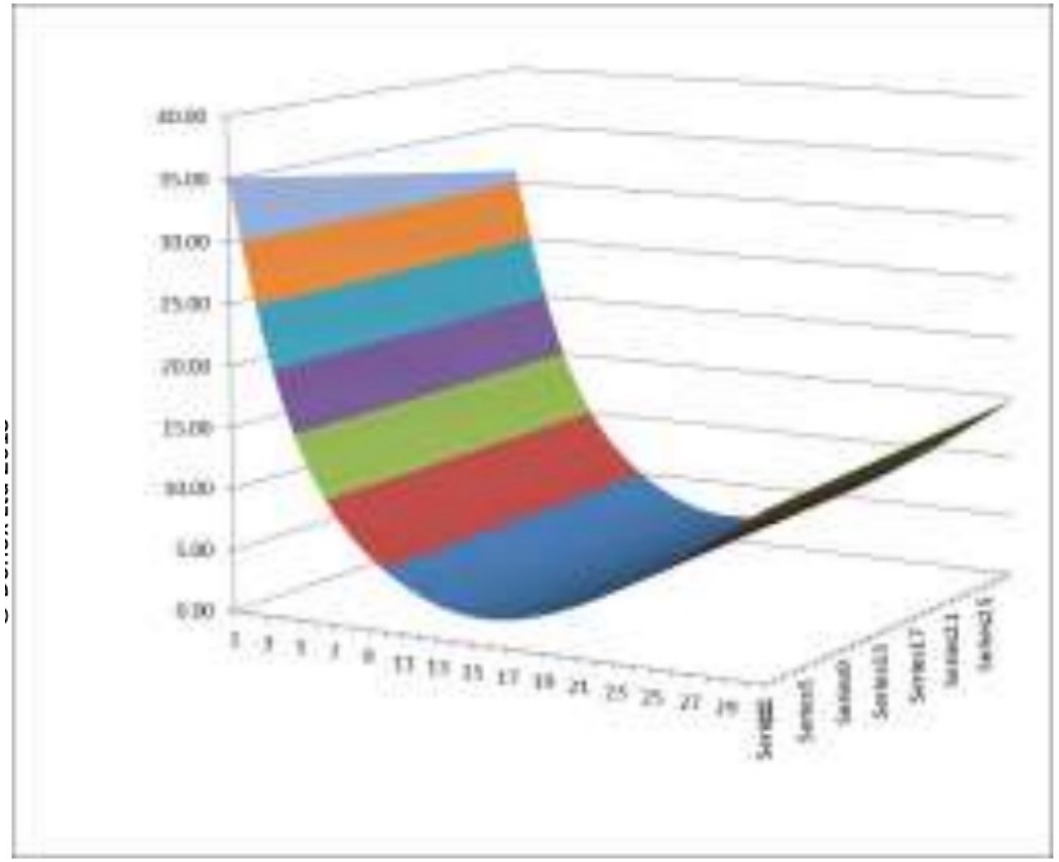
$$\lim_{x \to 0^+} \ln(x) = -\infty$$

$$LogLoss = \frac{-1}{N} \sum_{i=1}^{N} \sum_{j=1}^{M} x_{ij} * \ log(p_{ij})$$

If we say:
Actual value = 1, predicted = 0
LogLoss = -(1 * log(0) + 0 * log(1)) = tends to infinity !!

LogLoss <u>strongly</u> penalizes misclassifications where loss tends to infinity!

We need an alternative to mean square error ..

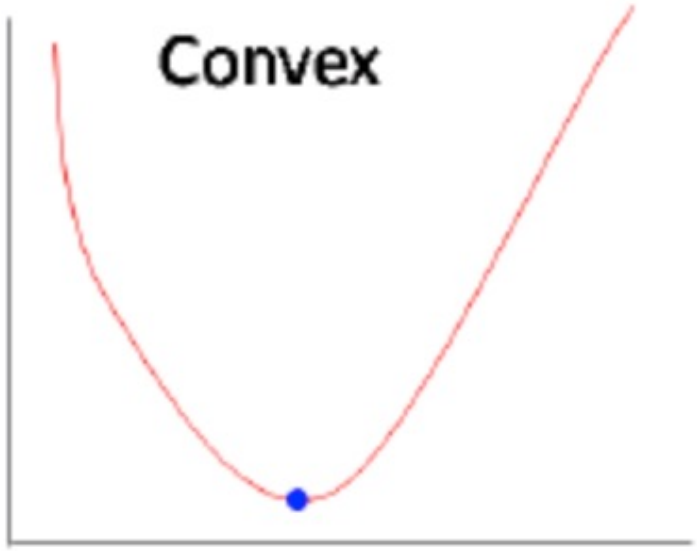


k -->

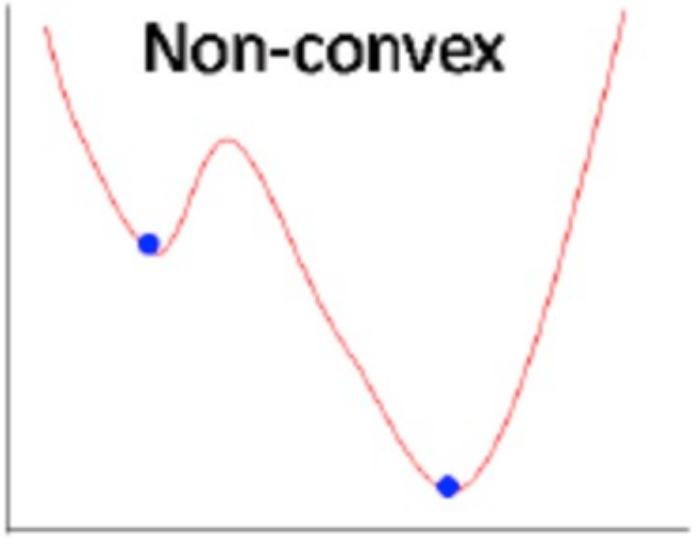
M	-10.8	-10.8	-10.8	-10.7	-10.7	-10.7	-10.7	-10.7	-10.6	-10.6	-10.6	-10.6	-10.6	-10.5	-10.5	-10.5	-10.5	-10.5	-10.4	-10.4	-10.4	-10.4	-10.4	-10.3	-10.3	-10.3	-10.3	-10.3
0.70	35.08	34.93	34.77	34.62	34.47	34.31	34.16	34.01	33.86	33.70	33.55	33.40	33.25	33.10	32.94	32.79	32.64	32.49	32.34	32.19	32.04	31.88	31.73	31.58	31.43	31.28	31.13	30.98
0.80	28.37	28.22	28.08	27.94	27.80	27.66	27.51	27.37	27.23	27.09	26.95	26.81	26.67	26.53	26.39	26.25	26.11	25.98	25.84	25.70	25.56	25.42	25.29	25.15	25.01	24.88	24.74	24.61
0.90	22.51	22.38	22.26	22.14	22.01	21.89	21.77	21.64	21.52	21.40	21.28	21.15	21.03	20.91	20.79	20.67	20.55	20.43	20.31	20.20	20.08	19.96	19.84	19.73	19.61	19.49	19.38	19.26
1.00	17.77	17.66	17.56	17.45	17.35	17.24	17.14	17.04	16.94	16.83	16.73	16.63	16.53	16.43	16.33	16.23	16.13	16.03	15.93	15.84	15.74	15.64	15.54	15.45	15.35	15.25	15.16	15.06
1.10	14.07	13.98	13.90	13.81	13.73	13.64	13.56	13.47	13.39	13.30	13.22	13.14	13.05	12.97	12.89	12.81	12.73	12.64	12.56	12.48	12.40	12.32	12.24	12.16	12.09	12.01	11.93	11.85
1.20	11.23	11.16	11.09	11.02	10.95	10.88	10.81	10.74	10.67	10.60	10.53	10.46	10.39	10.33	10.26	10.19	10.13	10.06	9.99	9.93	9.86	9.80	9.73	9.67	9.60	9.54	9.47	9.41
1.30	9.04	8.98	8.92	8.87	8.81	8.75	8.69	8.63	8.58	8.52	8.46	8.41	8.35	8.30	8.24	8.19	8.13	8.08	8.02	7.97	7.91	7.86	7.81	7.75	7.70	7.65	7.59	7.54
1.40	7.34	7.29	7.24	7.19	7.14	7.10	7.05	7.00	6.95	6.91	6.86	6.81	6.76	6.72	6.67	6.63	6.58	6.53	6.49	6.44	6.40	6.35	6.31	6.26	6.22	6.18	6.13	6.09
1.50	5.99	5.94	5.90	5.86	5.82	5.78	5.74	5.70	5.66	5.62	5.58	5.54	5.50	5.46	5.42	5.38	5.34	5.30	5.26	5.23	5.19	5.15	5.11	5.08	5.04	5.00	4.96	4.93
1.60	4.88	4.85	4.81	4.78	4.74	4.71	4.67	4.64	4.60	4.57	4.53	4.50	4.47	4.43	4.40	4.37	4.33	4.30	4.27	4.24	4.20	4.17	4.14	4.11	4.08	4.05	4.02	3.99
1.70	3.97	3.94	3.91	3.88	3.85	3.83	3.80	3.77	3.74	3.71	3.68	3.65	3.63	3.60	3.57	3.54	3.52	3.49	3.46	3.44	3.41	3.39	3.36	3.34	3.31	3.29	3.26	3.24
1.80	3.24	3.22	3.19	3.17	3.15	3.12	3.10	3.08	3.05	3.03	3.01	2.99	2.97	2.95	2.92	2.90	2.88	2.86	2.84	2.82	2.80	2.78	2.77	2.75	2.73	2.71	2.69	2.67
1.90	2.69	2.67	2.65	2.63	2.62	2.60	2.58	2.57	2.55	2.54	2.52	2.51	2.49	2.48	2.46	2.45	2.44	2.42	2.41	2.40	2.38	2.37	2.36	2.35	2.34	2.33	2.31	2.30
2.00	2.31	2.30	2.29	2.28	2.27	2.26	2.25	2.25	2.24	2.23	2.22	2.21	2.20	2.20	2.19	2.18	2.18	2.17	2.16	2.16	2.15	2.15	2.14	2.14	2.13	2.13	2.13	2.12
2.10	2.13	2.12	2.12	2.12	2.11	2.11	2.11	2.11	2.10	2.10	2.10	2.10	2.10	2.10	2.10	2.10	2.10	2.10	2.10	2.10	2.11	2.11	2.11	2.11	2.11	2.12	2.12	2.12
2.20	2.12	2.12	2.12	2.13	2.13	2.13	2.14	2.14	2.15	2.15	2.16	2.16	2.17	2.18	2.18	[illegible]	2.20	2.20	2.21	2.22	2.23	2.24	2.24	2.25	2.26	2.27	2.28	2.29
2.30	2.27	2.28	2.29	2.30	2.31	2.32	2.33	2.34	2.35	2.36	2.37	2.39	2.40	2.41	2.42	[illegible]	2.45	2.46	2.48	2.49	2.50	2.52	2.53	2.55	2.56	2.58	2.59	2.61
2.40	2.57	2.58	2.60	2.61	2.63	2.65	2.66	2.68	2.69	2.71	2.73	2.75	2.76	2.78	2.80	[illegible]	2.84	2.86	2.87	2.89	2.91	2.93	2.95	2.97	2.99	3.01	3.03	3.06
2.50	2.99	3.01	3.03	3.06	3.08	3.10	3.12	3.14	3.16	3.18	3.21	3.23	3.25	3.27	3.30	3.32	[illegible]	3.37	3.39	3.41	3.44	3.46	3.49	3.51	3.54	3.56	3.59	3.61
2.60	3.53	3.55	3.58	3.60	3.63	3.65	3.68	3.70	3.73	3.76	3.78	3.81	3.84	3.86	3.89	3.92	[illegible]	3.97	4.00	4.03	4.06	4.09	4.11	4.14	4.17	4.20	4.23	4.26
2.70	4.16	4.18	4.21	4.24	4.27	4.30	4.33	4.36	4.39	4.42	4.45	4.48	4.51	4.54	4.57	4.60	[illegible]	[illegible]	4.70	4.73	4.76	4.79	4.82	4.86	4.89	4.92	4.95	4.99
2.80	4.86	4.89	4.92	4.96	4.99	5.02	5.05	5.09	5.12	5.15	5.19	5.22	5.25	5.29	5.32	5.36	[illegible]	[illegible]	5.46	5.50	5.53	5.57	5.60	5.64	5.67	5.71	5.74	5.78
2.90	5.63	5.67	5.70	5.74	5.77	5.81	5.84	5.88	5.92	5.95	5.99	6.03	6.06	6.10	6.14	6.17	6.21	[illegible]	[illegible]	6.32	6.36	6.40	6.44	6.48	6.51	6.55	6.59	6.63
3.00	6.46	6.50	6.54	6.57	6.61	6.65	6.69	6.73	6.77	6.81	6.84	6.88	6.92	6.96	7.00	7.04	7.08	[illegible]	[illegible]	7.20	7.24	7.28	7.32	7.36	7.40	7.44	7.49	7.53
3.10	7.34	7.38	7.42	7.46	7.50	7.54	7.58	7.62	7.66	7.70	7.74	7.79	7.83	7.87	7.91	7.95	8.00	[illegible]	[illegible]	8.12	8.16	8.21	8.25	8.29	8.34	8.38	8.42	8.47
3.20	8.26	8.30	8.34	8.38	8.43	8.47	8.51	8.56	8.60	8.64	8.68	8.73	8.77	8.82	8.86	8.90	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	9.31	9.35	9.40	9.44
3.30	9.21	9.26	9.30	9.35	9.39	9.43	9.48	9.52	9.57	9.62	9.66	9.71	9.75	9.80	9.84	9.89	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	10.31	10.36	10.40	10.45
3.40	10.20	10.25	10.29	10.34	10.39	10.43	10.48	10.53	10.57	10.62	10.67	10.72	10.76	10.81	10.86	10.91	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	11.35	11.39	11.44	11.49
3.50	11.22	11.27	11.32	11.37	11.41	11.46	11.51	11.56	11.61	11.66	11.71	11.76	11.81	11.86	11.91	11.96	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	12.41	12.46	12.51	12.56
3.60	12.27	12.32	12.37	12.42	12.47	12.52	12.57	12.62	12.67	12.72	12.78	12.83	12.88	12.93	12.98	13.03	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	13.50	13.56	13.61	13.66
3.70	13.35	13.40	13.45	13.50	13.56	13.61	13.66	13.71	13.77	13.82	13.87	13.92	13.98	14.03	14.08	14.14	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	14.62	14.68	14.73	14.79

M ~ 2
K ~ -10

Logarithmic loss



MSE loss



- MSE function is non-convex for binary classification
- it is *not guaranteed* to minimize the Cost function.

In Python .. Easier than Excel!

In [1]:
```
1 from sklearn.linear_model import LogisticRegression
```

In [2]:
```
1 x = [ [ 1. ], [ 2. ], [ 3. ], [ 4. ], [ 4.8], [ 6. ],
2                [ 5.2], [ 8. ], [ 9. ], [10. ], [11. ], [12. ], [13. ]]
3 Y = [0,0,0,0,1,1,0,1,1,1,1,1,1]
```

In [3]:
```
1 logisticRegr = LogisticRegression(solver='liblinear', C=1e9)
```

In [ ]:
```
1 logisticRegr.fit(x,Y)
```

In [5]:
```
1 print(logisticRegr.coef_)
2 print(logisticRegr.intercept_)
```

```
[[2.16244588]]
[-10.8314298]
```

Student task .. Replicate this in RapidMiner

The Language of Model Quality

Recall Week 1 : Logistical Regression for Credit Analysis

		Prediction	
		No Default	Default
Reality	No Default	Computer said "They will not default". We gave them a loan. They did not default. Good business!	Computer said "They will default". We didn't give them a loan. Lost business. Not good
	Default	Computer said "They will not default". We gave them a loan. They did default. Bad!	Computer said "They will default". We did not give them a loan. They would have defaulted in reality Good decision

		Prediction	
		No Default	Default
Reality	No Default	Computer said "They will not default". We gave them a loan. They did not default. Good business!	Computer said "They will default". We didn't give them a loan. Lost business. Not good
	Default	Computer said "They will not default". We gave them a loan. They did default. Bad!	Computer said "They will default". We did not give them a loan. They would have defaulted in reality Good decision

TP - A true positive is an outcome where the model correctly predicts the positive class.
TN - A true negative is an outcome where the model correctly predicts the negative class.
FP - A false positive is an outcome where the model incorrectly predicts the positive class.
FN - A false positive is an outcome where the model incorrectly predicts the negative class.

F_1-score

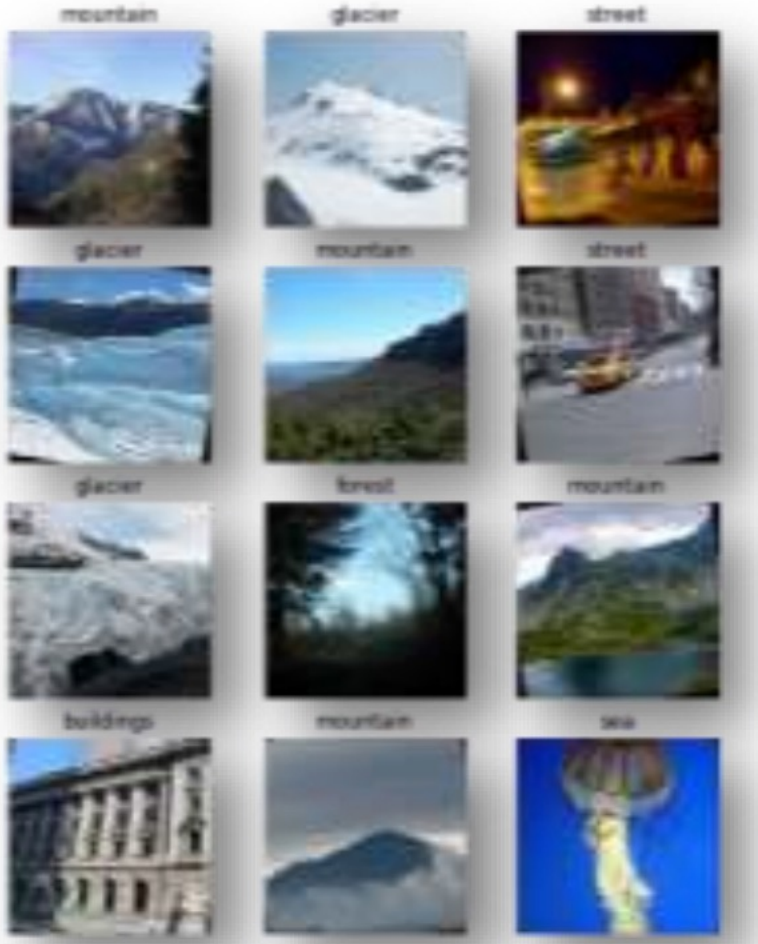
- The F_1-score is a way of combining the precision and recall of a model
- Defined as the harmonic mean of the model's precision and recall.

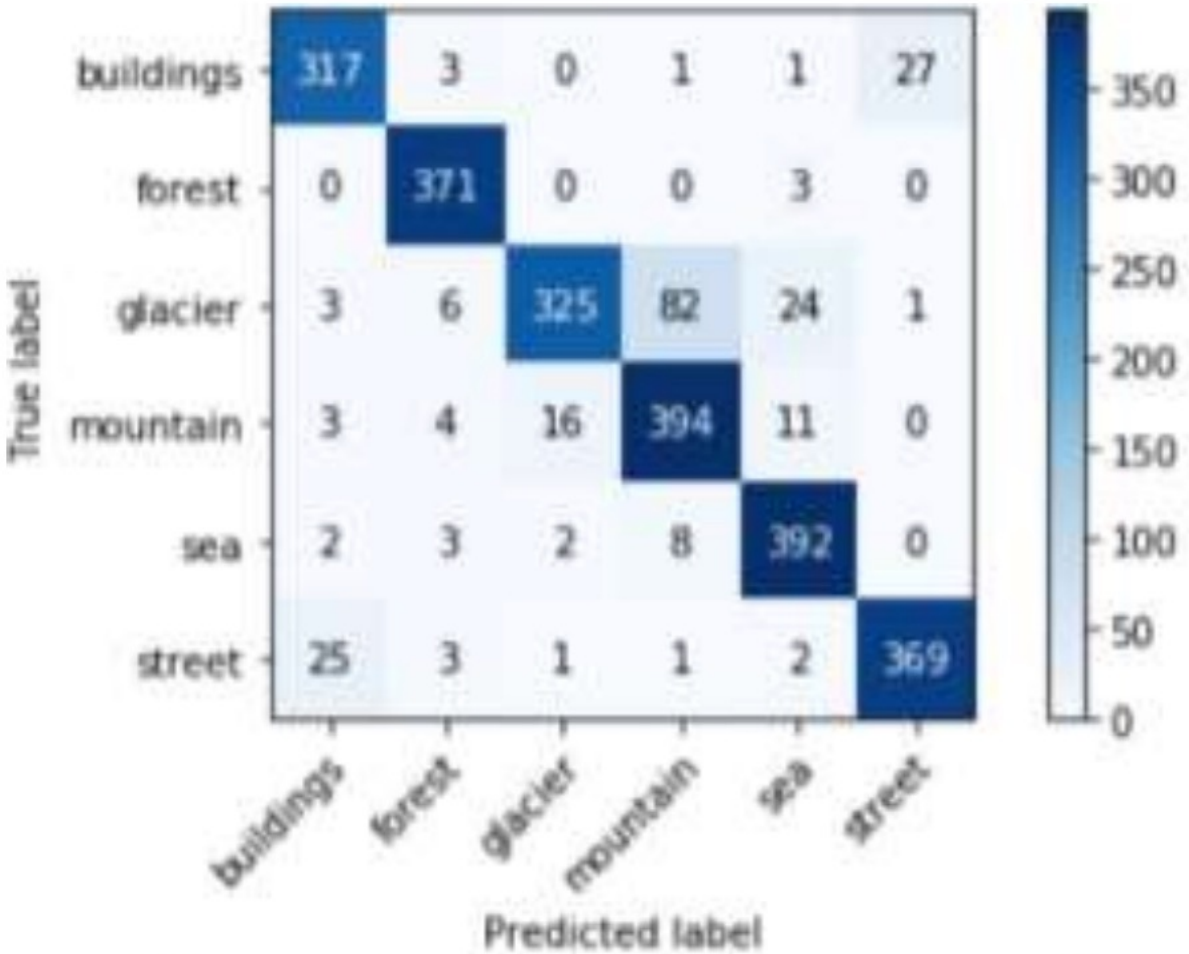
$$F_1 = \frac{2}{\frac{1}{\text{recall}} \times \frac{1}{\text{precision}}} = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$$
$$= \frac{\text{tp}}{\text{tp} + \frac{1}{2}(\text{fp} + \text{fn})}$$

What is a good F_1-score?

- F1 score ranges from 0 to 1
- 1 is a perfect score indicating that the model predicts each observation correctly.
- In general:
  - > 0.9 Very good
  - 0.8 - 0.9 Good
  - 0.5 - 0.8 OK
  - < 0.5 Not good

Multi-Class Classification

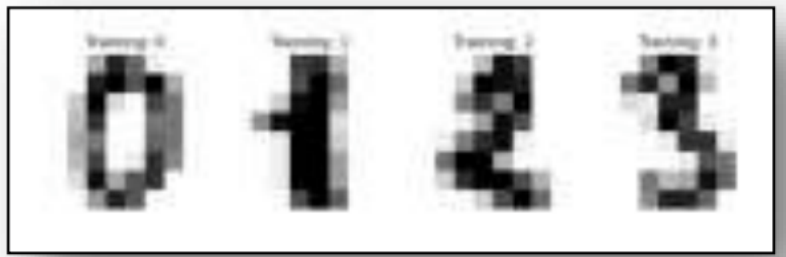


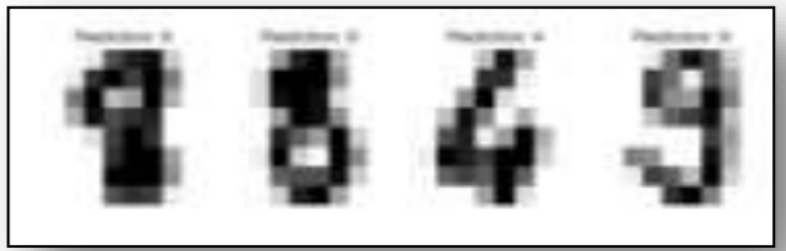


Training set ~2200 images of each classification

Source: https://medium.com/swlh/image-classification-tutorials-in-pytorch-transfer-learning-19ebc329e200

sklearn.metrics.classification_report





Classification report for classifier SVC(gamma=0.001):

	precision	recall	f1-score	support
0	1.00	0.99	0.99	88
1	0.99	0.97	0.98	91
2	0.99	0.99	0.99	86
3	0.98	0.87	0.92	91
4	0.99	0.96	0.97	92
5	0.95	0.97	0.96	91
6	0.99	0.99	0.99	91
7	0.96	0.99	0.97	89
8	0.94	1.00	0.97	88
9	0.93	0.98	0.95	92
accuracy			0.97	899

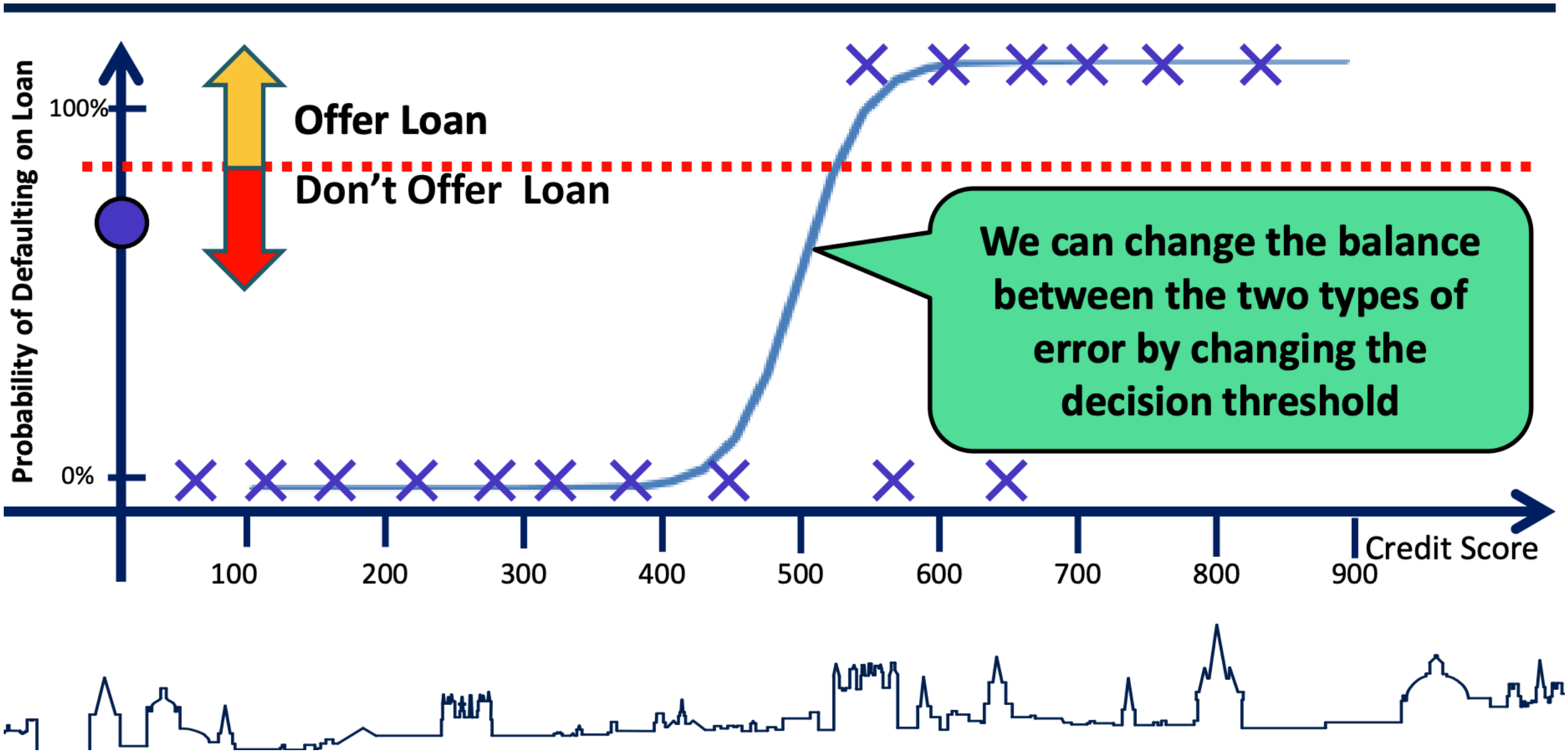
Confusion matrix:

```
[
[87 0 0 0 1 0 0 0 0 0]
[ 0 88 1 0 0 0 0 0 1 1]
[ 0 0 85 1 0 0 0 0 0 0]
[ 0 0 0 79 0 3 0 4 5 0]
[ 0 0 0 0 88 0 0 0 0 4]
[ 0 0 0 0 0 88 1 0 0 2]
[ 0 1 0 0 0 0 90 0 0 0]
[ 0 0 0 0 0 1 0 88 0 0]
[ 0 0 0 0 0 0 0 0 88 0]
[ 0 0 0 1 0 1 0 0 0 90]]
```

Sources: https://scikit-learn.org/stable/auto_examples/classification/plot_digits_classification.html#sphx-glr-auto-examples-classification-plot-digits-classification-py
https://scikit-learn.org/stable/modules/generated/sklearn.metrics.classification_report.html

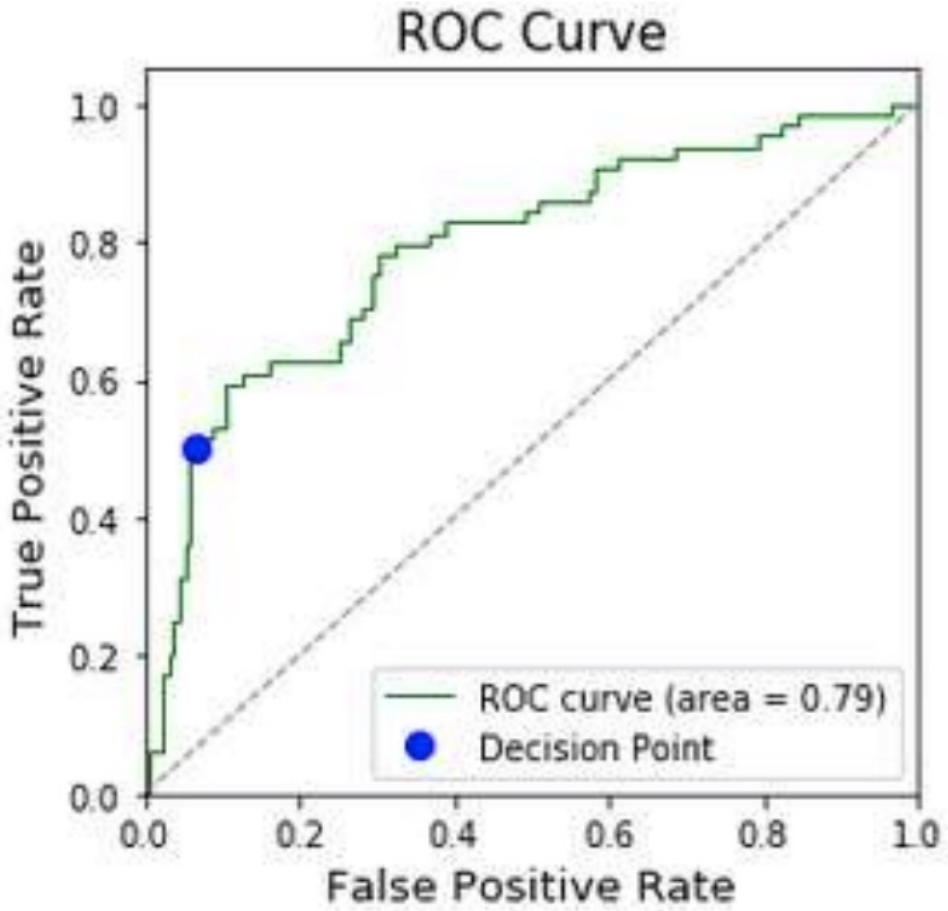
Moving the decision point

Setting the Decision Threshold

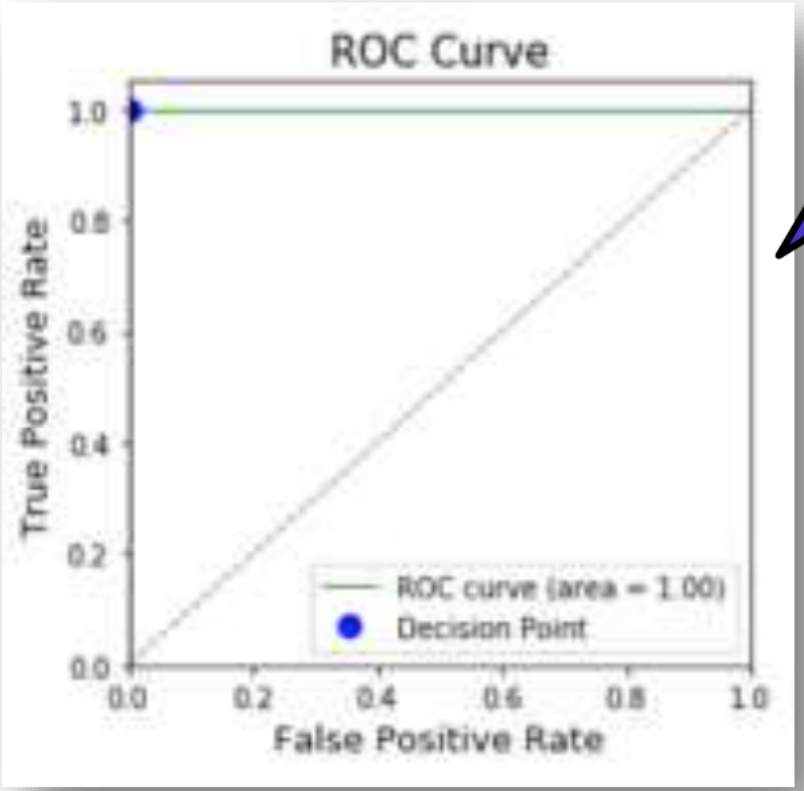


Let’s experiment with different decision thresholds

- The ‘Receiver Operating Characteristic’ (ROC) curve does just that …
  - Systematically sweep through levels of decision threshold
  - Plot the ‘True Positive Rate’ vs. ‘False positive Rate’
- You decide your decision point based on what matters most

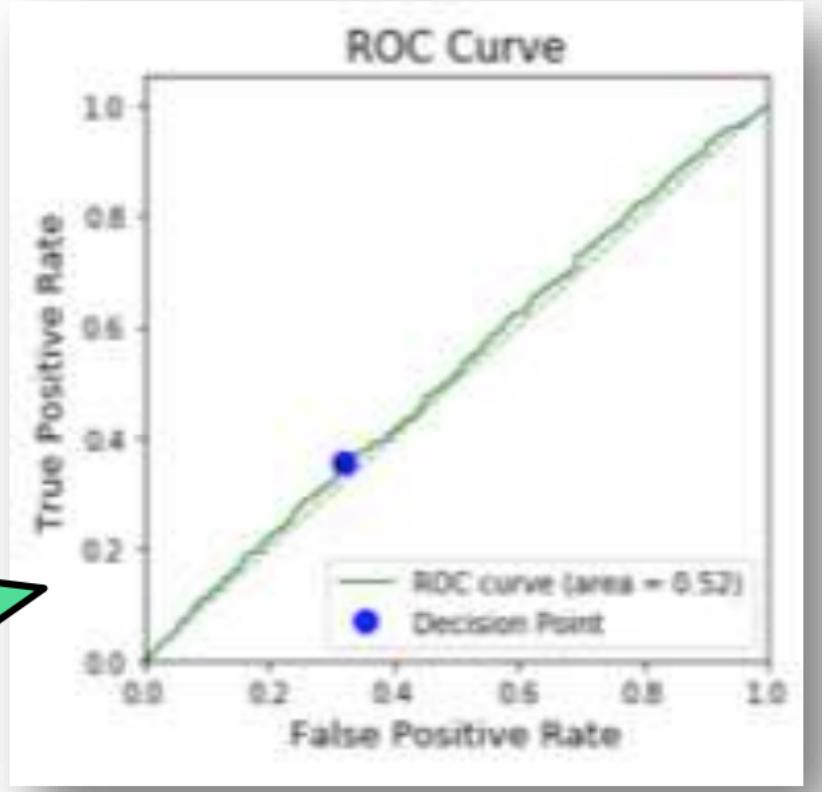


What you should look for in your ROC?

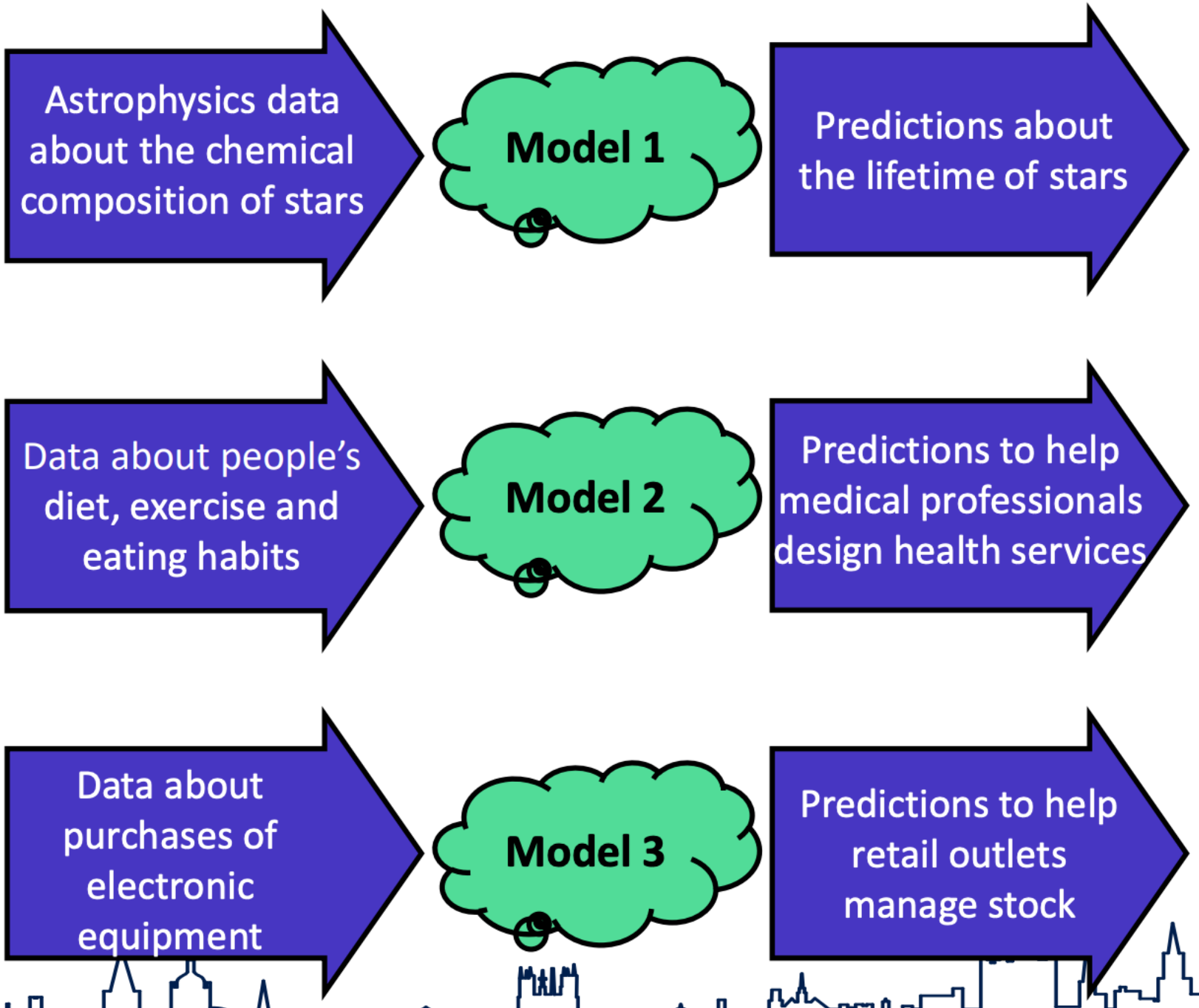


Perfect classifier
100% true positives,
0% false positives

Purely Random Classifier
Just as many false positives
as true positives



A Final Thought on Model Quality ..

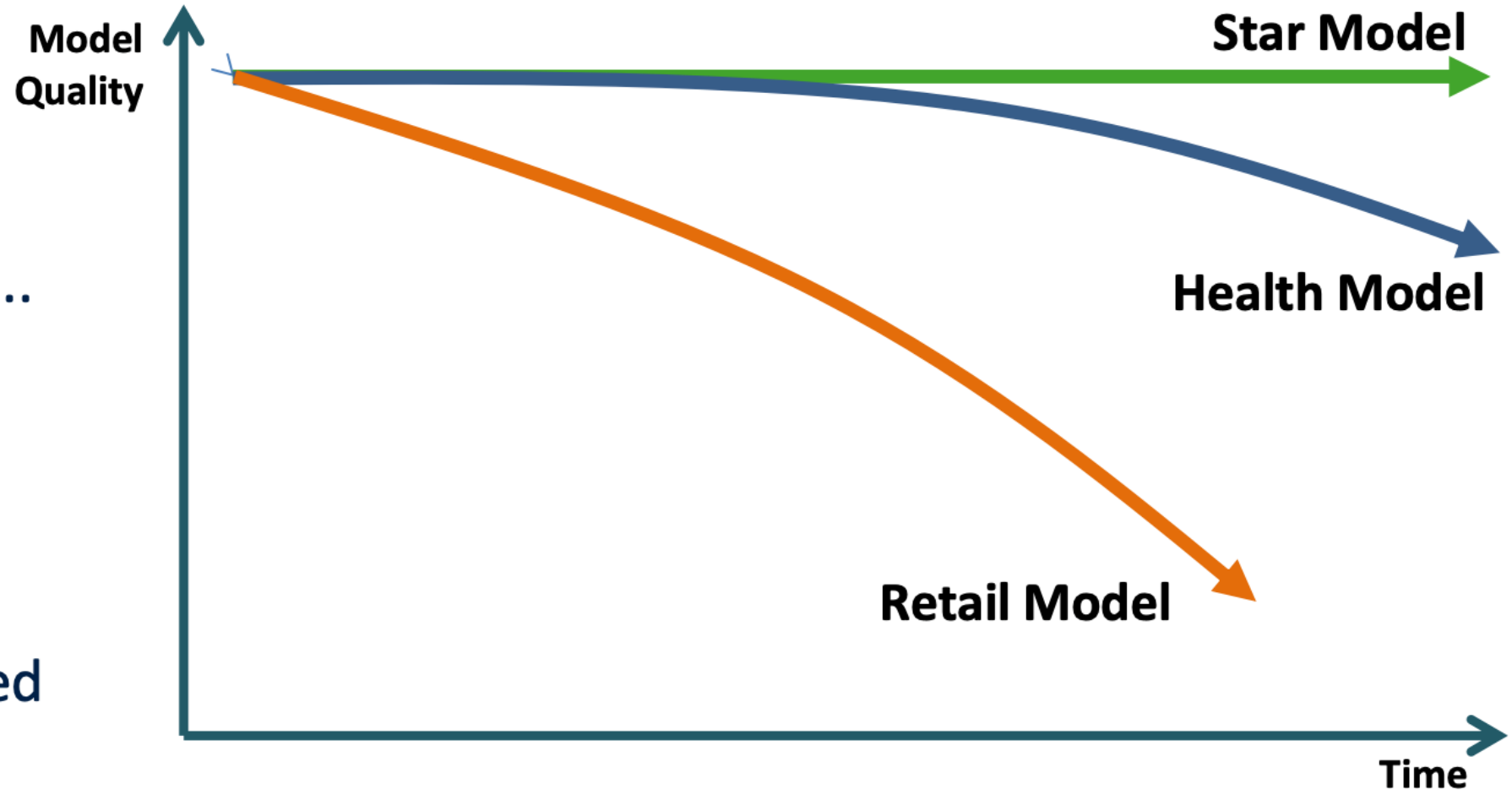


Thinking about measures of model quality ..

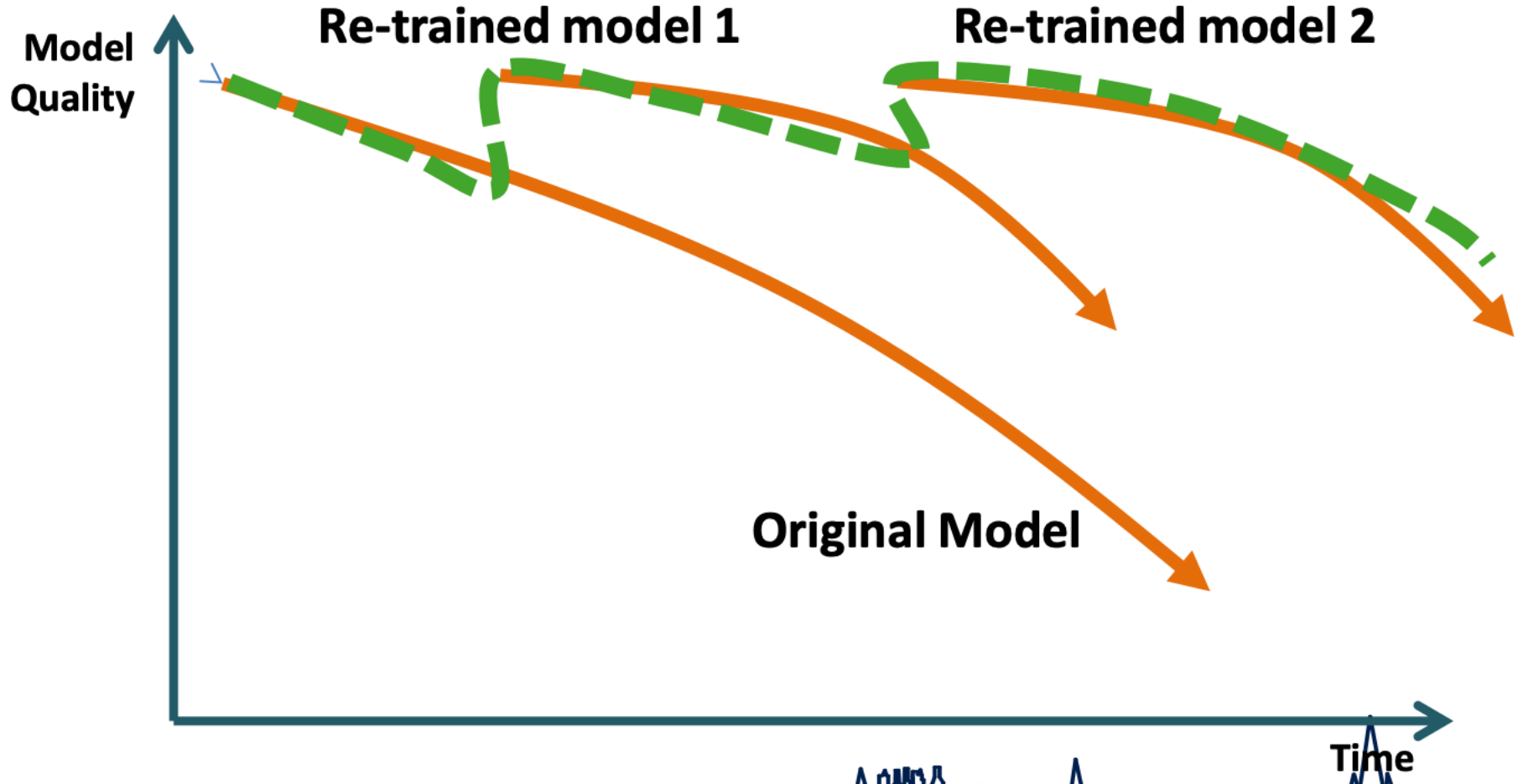
Why might we have to think differently about measuring quality of the three models on the left?

How does model quality change over time?

- The real world changes over time ..
- Some things that were true yesterday .. May be less true today
- Real world data may change ..
- And models may need to be updated



Models may need to be monitored and re-trained

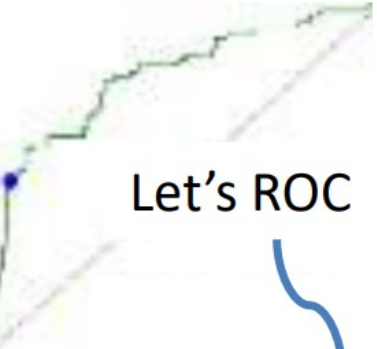


Conclusion

Tuning your Machine

Shape matters

Making it easier to get fit



Let's ROC

How to get fit

How do algorithms actually fit models to data?

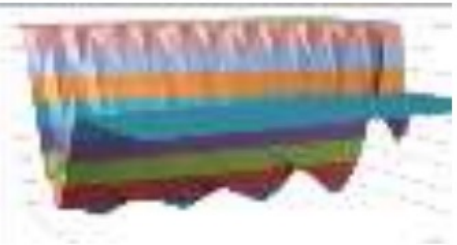
Cakeology

Machine Learning Fundamentals

Time to choose

More about classification

Some hills are too tough



How to get Fat

Working in higher-dimensions

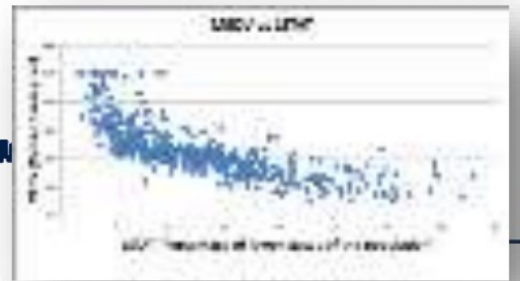
Welcome to the Matrix

Concerned about curves

Beyond straight-lines

Reducing others

Some curves are useful



Out-takes
(gag reel)

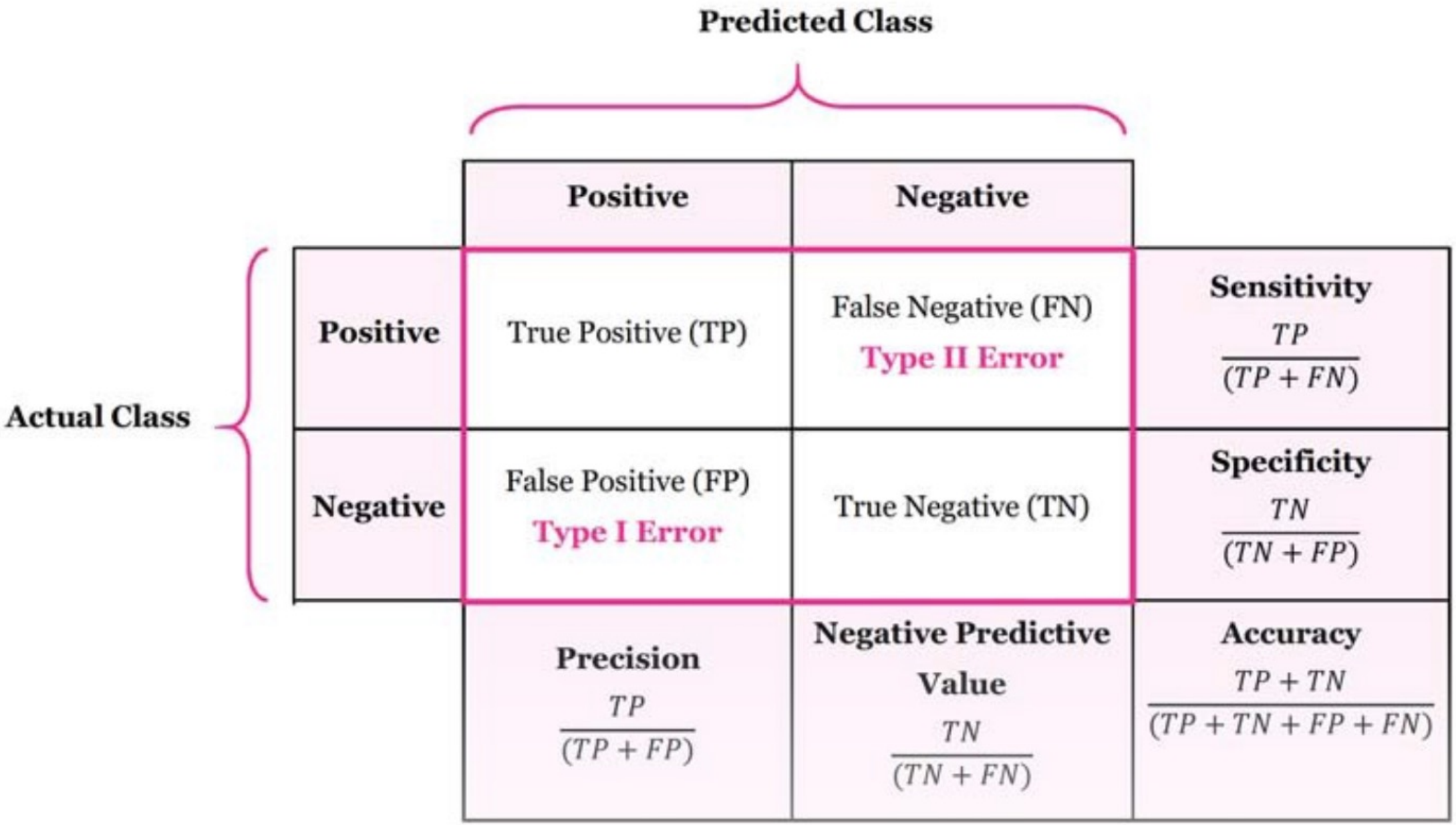
Lessons Learnt ..

Always read the manual!

Trust your own knowledge, skill and insight

Don’t assume that the model must be right

There may be more than one ‘right’ answer


Predicted Class
Positive
Negative
Actual Class
Positive
True Positive (TP)
False Negative (FN)
Type II Error
Sensitivity
$\frac{TP}{(TP + FN)}$
Negative
False Positive (FP)
Type I Error
True Negative (TN)
Specificity
$\frac{TN}{(TN + FP)}$
Precision
$\frac{TP}{(TP + FP)}$
Negative Predictive Value
$\frac{TN}{(TN + FN)}$
Accuracy
$\frac{TP + TN}{(TP + TN + FP + FN)}$