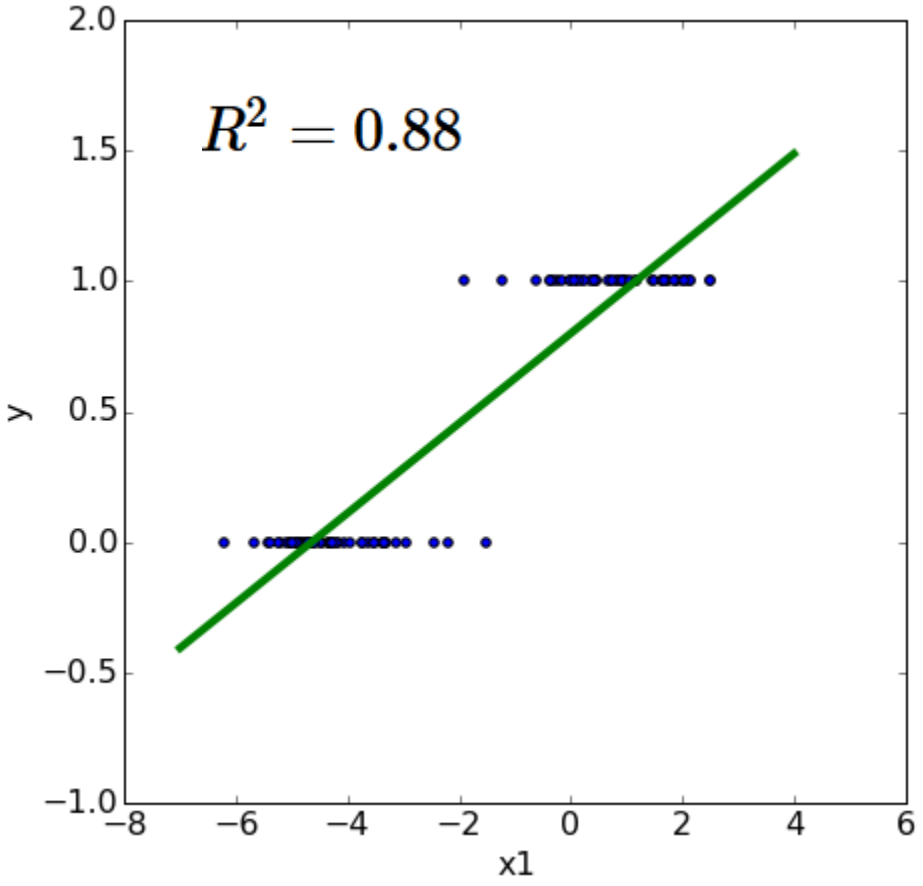
logistic regression for classification



We need to make

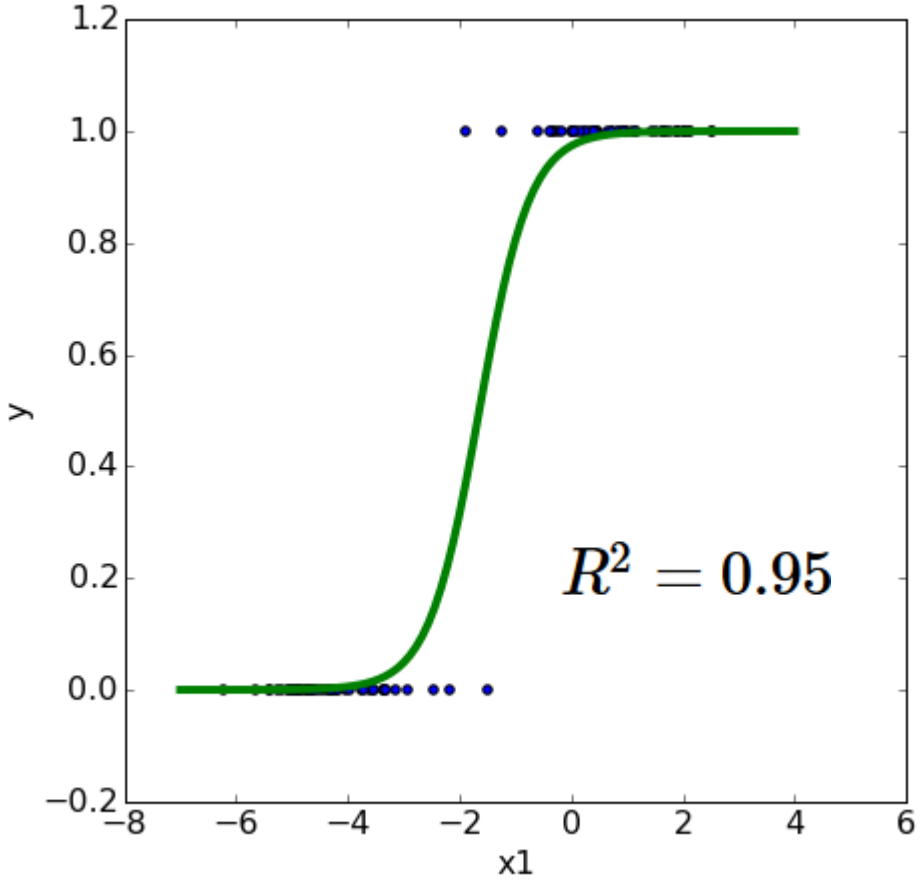
assumptions *linear relationship*

about the

model *linear model*

that generated the data.

logistic regression for classification



We need to make

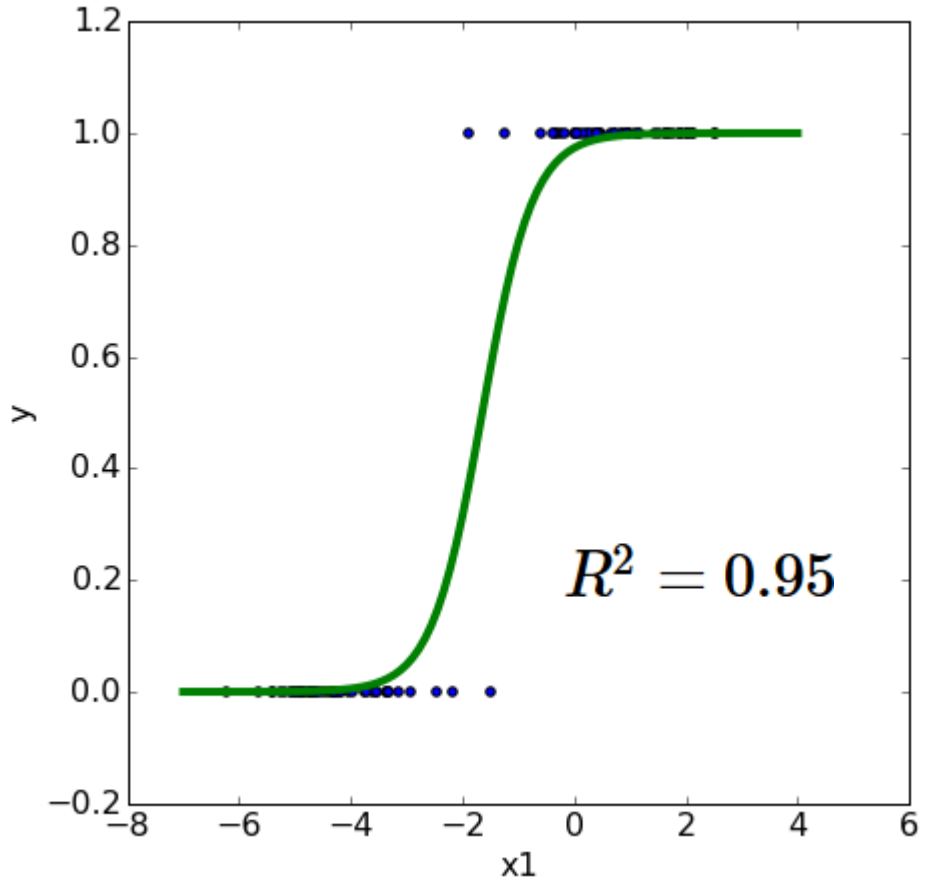
assumptions *linearly separable*

about the

model *logistic model*

that generated the data.

logistic regression: logistic model



$$f(x, \theta) = g(\theta_0 + \theta_1 x_1)$$

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

logistic regression: cost function

$$J(\theta) = -[\frac{1}{n}\sum_{i=1}^{n} y^{(i)} log(f(x^{(i)}, \theta)) + (1 - y^{(i)}) log(1 - f(x^{(i)}, \theta))]$$

We know that $y^{(i)}$ is either 0 or 1. If $y^{(i)} = 1$ then the cost function $J(\theta)$ is incremented by

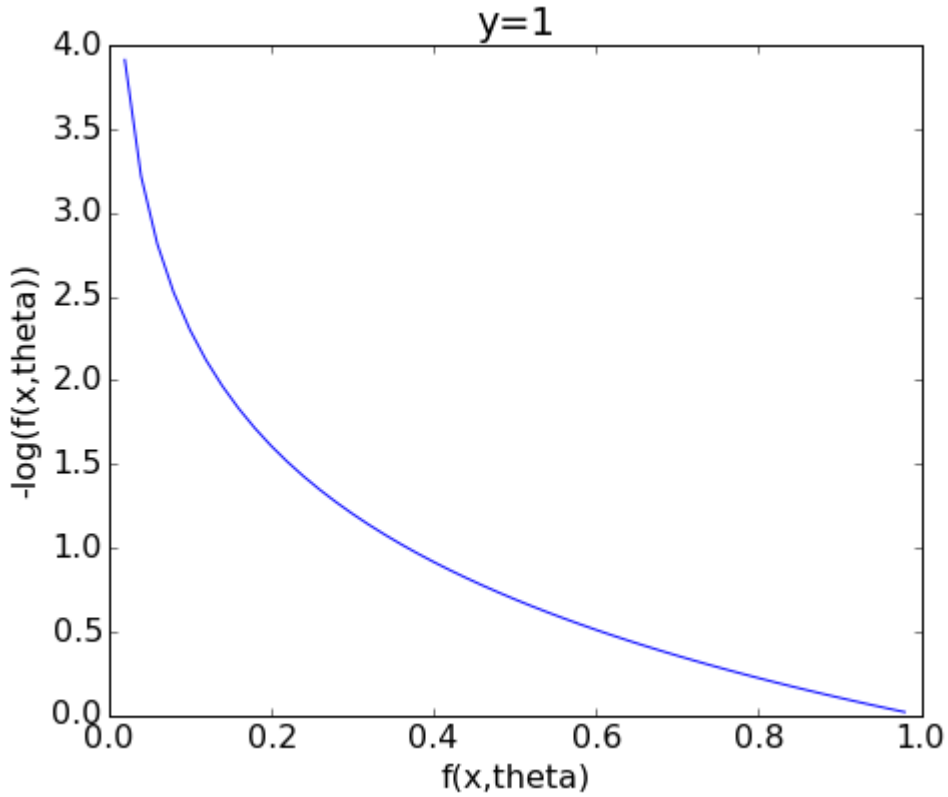
$-log(f(x^{(i)}, \theta))$.

Similarly, if $y^{(i)} = 0$ then the cost function $J(\theta)$ is incremented by

$-log(1 - f(x^{(i)}, \theta))$.

logistic regression: cost function

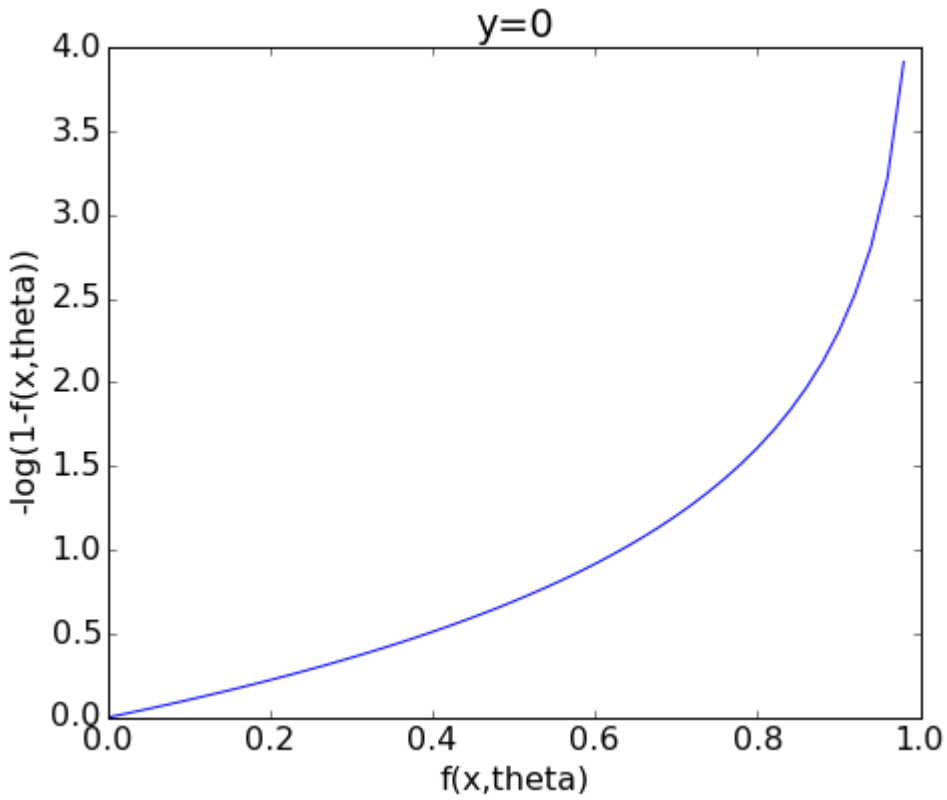
We know that $y^{(i)}$ is either 0 or 1. If $y^{(i)} = 1$ then the cost function $J(\theta)$ is incremented by $-log(f(x^{(i)}, \theta))$.



logistic regression: cost function

Similarly, if $y^{(i)} = 0$ then the cost function $J(\theta)$ is incremented by

$-log(1 - f(x^{(i)}, \theta))$.



logistic regression

Fit a logistic model

$$f(x,\theta) = g(\theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_m x_m) = g(\theta' x)$$

to the data set such that the cost function

$$J(\theta) = -[\frac{1}{n}\sum_{i=1}^{n} y^{(i)} log(f(x^{(i)},\theta)) + (1 - y^{(i)}) log(1 - f(x^{(i)},\theta))]$$

is minimal using gradient descent

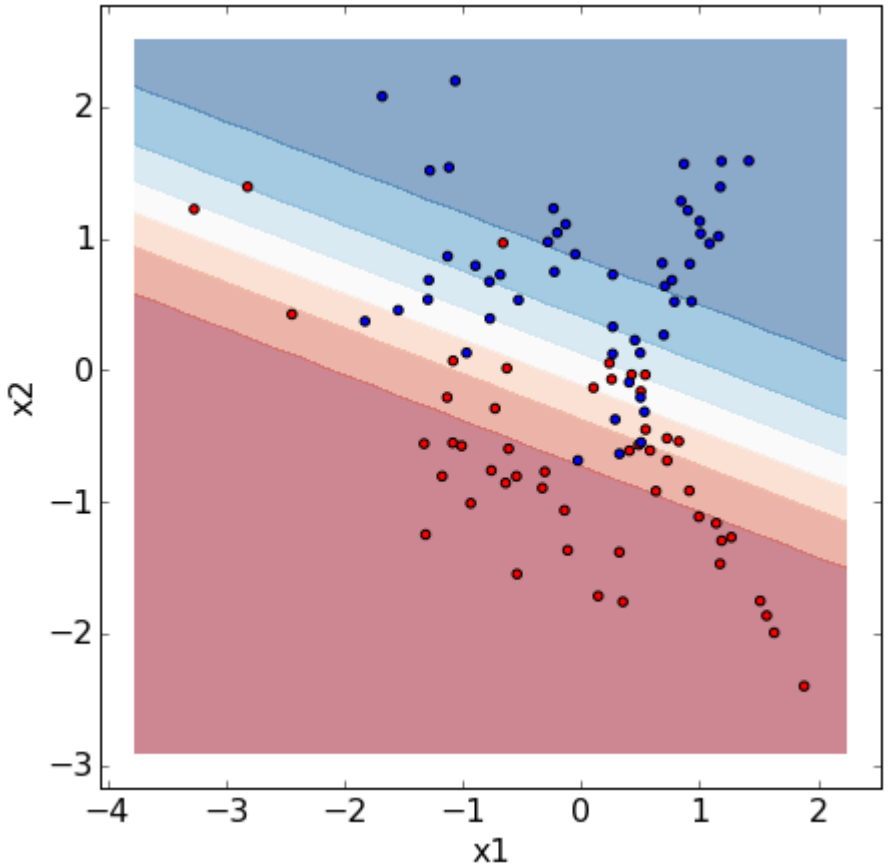
$$\theta_j := \theta_j - \alpha \frac{1}{n}\sum_{i=1}^{n} (f(x^{(i)},\theta) - y^{(i)}) x_j^{(i)}$$

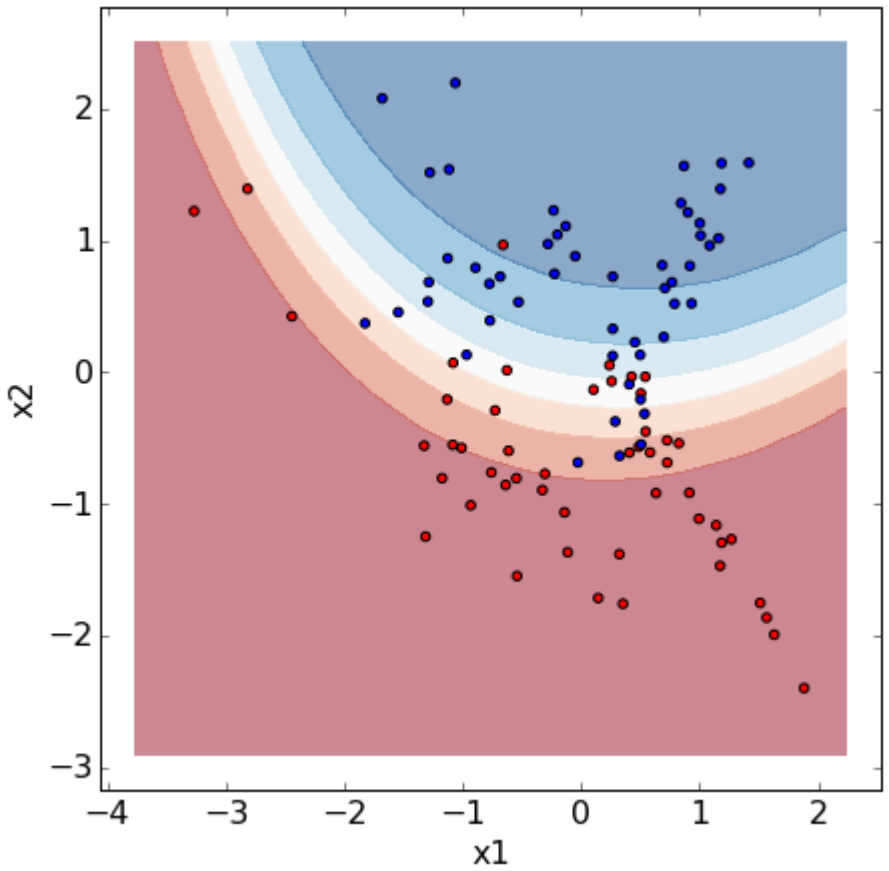
```
import sklearn.datasets as ds
from sklearn.preprocessing import StandardScaler

dataset2D = pd.read_csv("dataset2D.csv")

X = dataset2D.copy()
y = X.pop('y')

model = LogisticRegression(C=100000)
model.fit(X,y)
score = model.score(X, y)

plt.title("accuracy = %.2f" % score)
compomics_import.plot_decision_boundary(model,X,y)
plt.show()
```

```
predictions = model.predict_proba(X)
print predictions[:10]
```

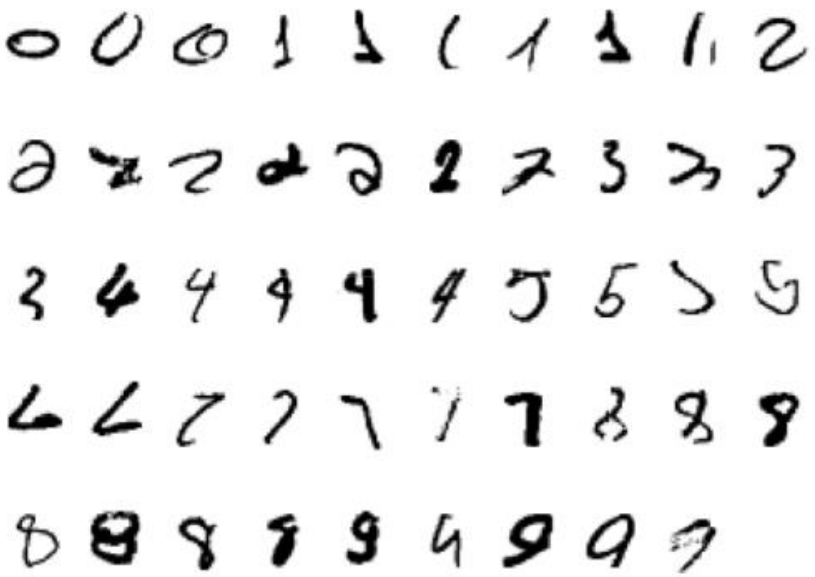
```
[[0.00553644 0.99446356]
 [0.99133532 0.00866468]
 [0.67601956 0.32398044]
 [0.03953503 0.96046497]
 [0.91253213 0.08746787]
 [0.55021616 0.44978384]
 [0.65698596 0.34301404]
 [0.0067126  0.9932874 ]
 [0.06459638 0.93540362]
 [0.02769494 0.97230506]]
```

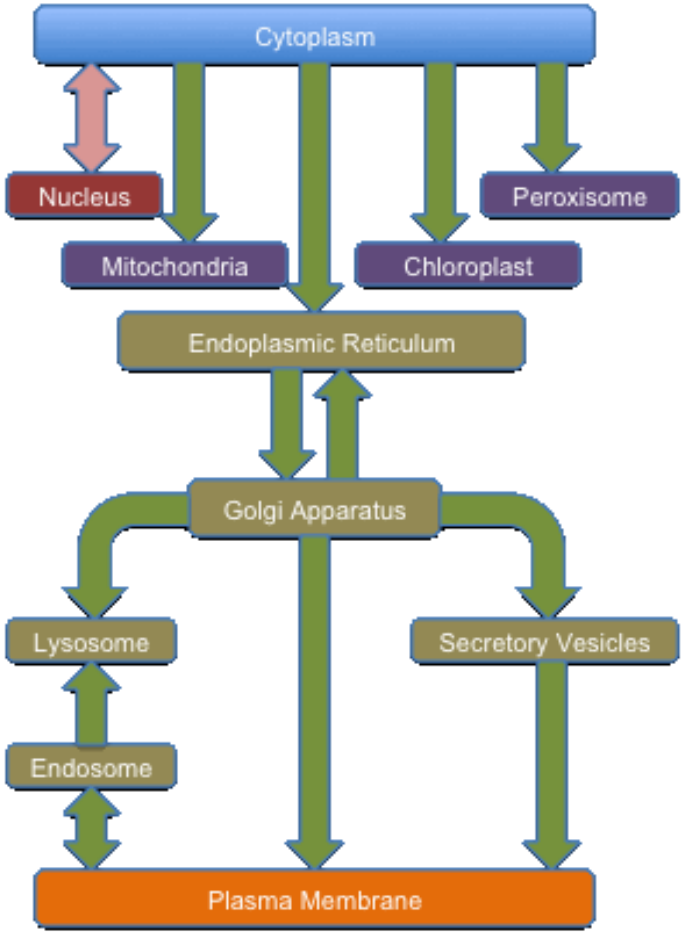
non-linear logistic regression





multiclass classification



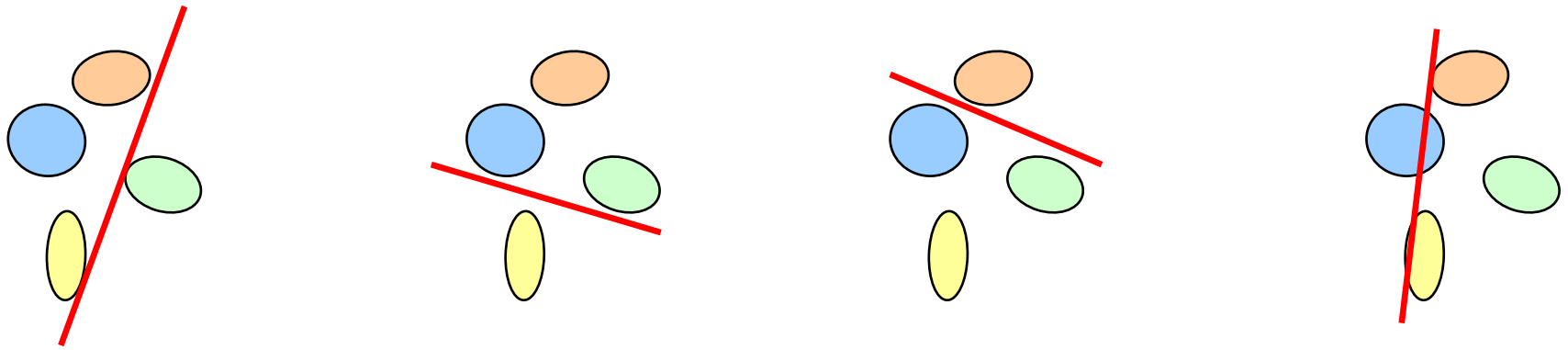


Source: https://www.rostlab.org/services/locDB/

one against all

One-against-all → k by k

+1	−1	−1	−1
−1	+1	−1	−1
−1	−1	+1	−1
−1	−1	−1	+1



one against one

All-pairs → k by $\binom{k}{2}$

+1	+1	+1	0	0	0
−1	0	0	+1	+1	0
0	−1	0	−1	0	+1
0	0	−1	0	−1	−1