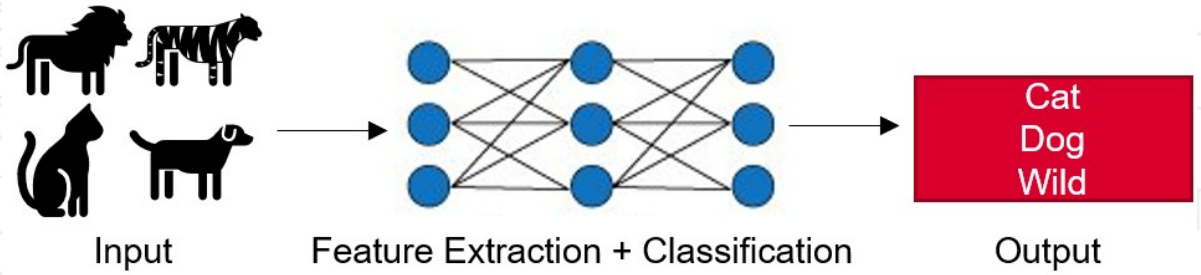
Module 2

Multi-Class Classification

Multi-Class Classification



Types of Classification Problems

- The 3 main classification problems are:

Binary Classification

- Spam
- Not spam

Multiclass Classification

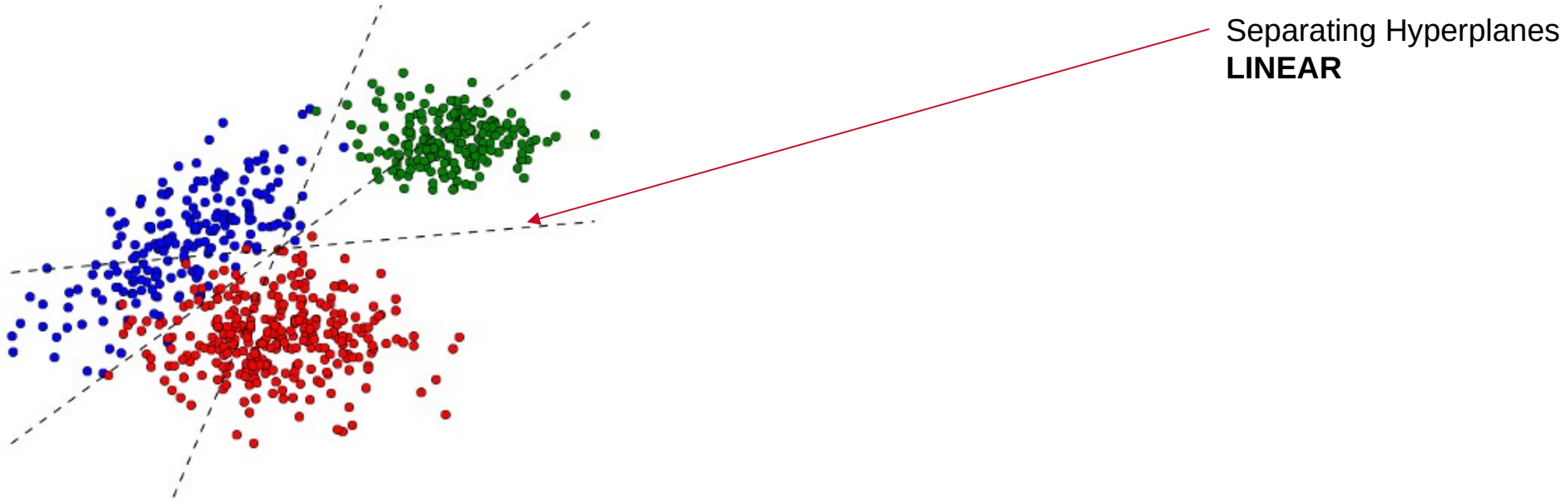
- Dog
- Cat
- Horse
- Fish
- Bird
- ...

Multi-label Classification

- Dog
- Cat
- Horse
- Fish
- Bird
- ...

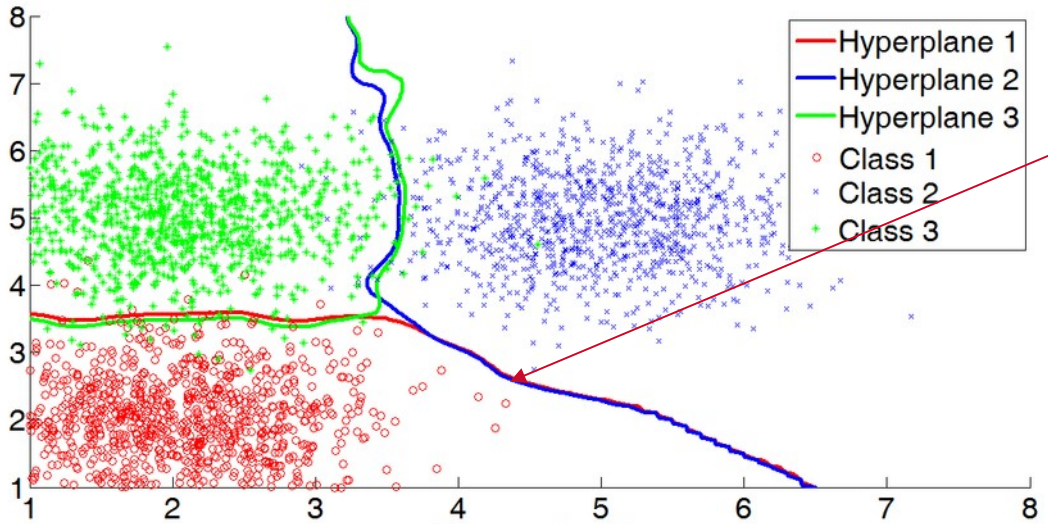
Multi-Class Classification

- is a classification task with more than two classes.



Multi-Class Classification

- is a classification task with more than two classes.



Separating Hyperplanes
NON-LINEAR

Multi-Class Classification

Feature Matrix (X)

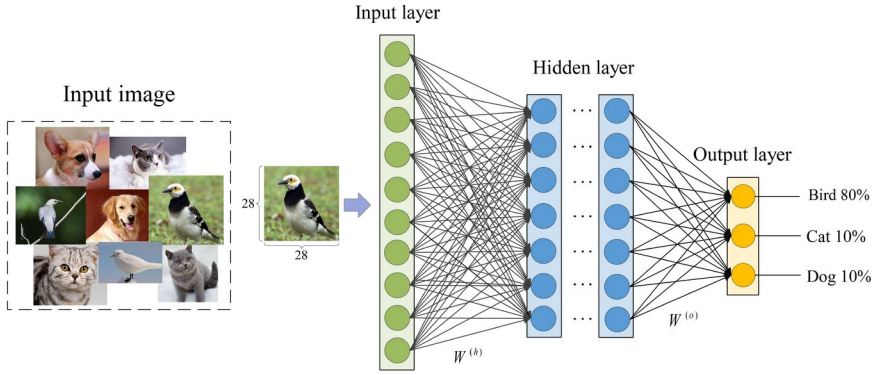
n_features ⟶

n_samples ↓

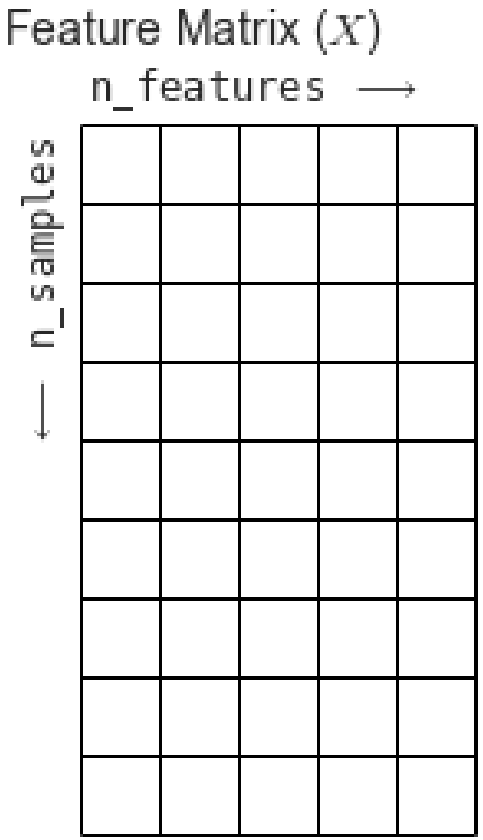
Target Vector (y)

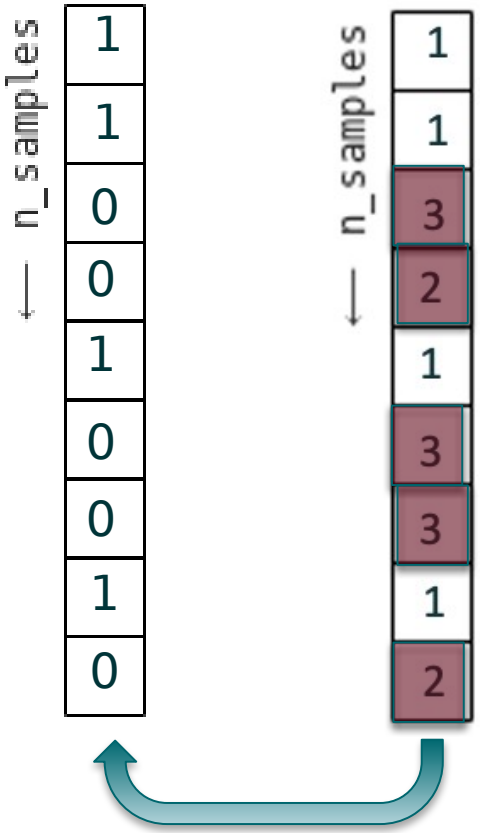
n_samples ↓

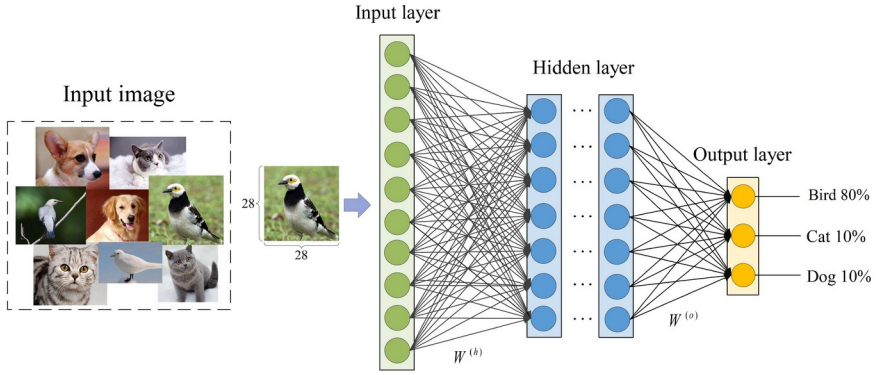
1
1
3
2
1
3
3
1
2



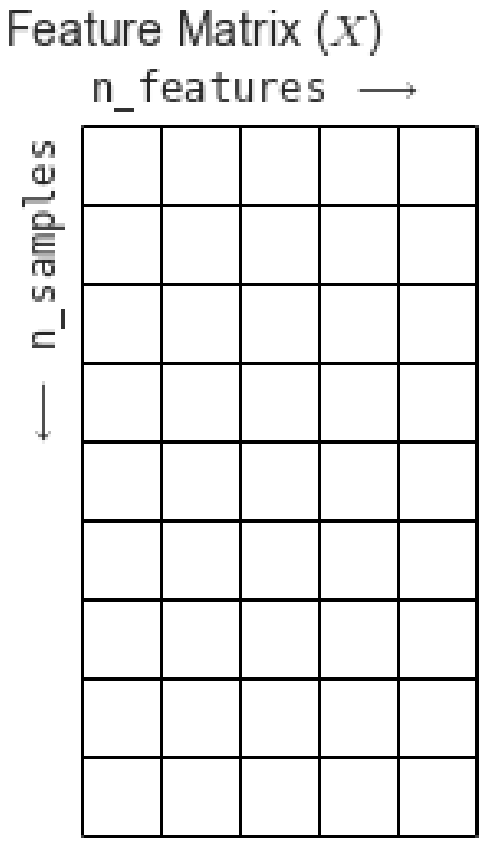
Multi-Class Classification

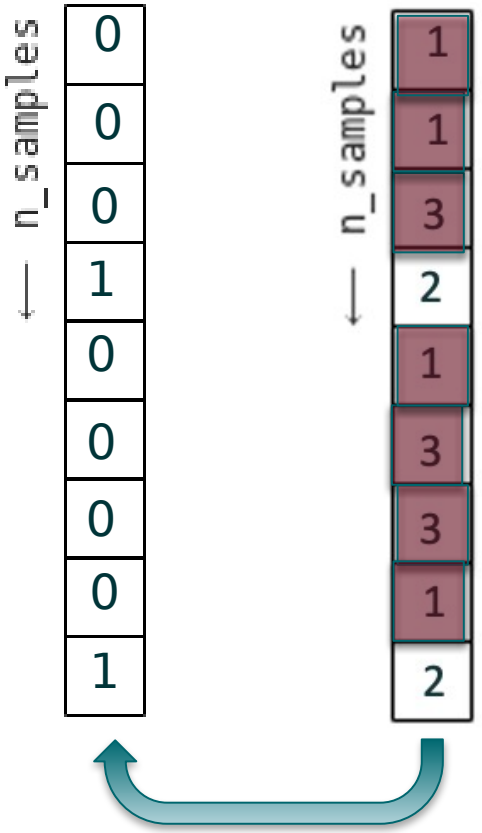


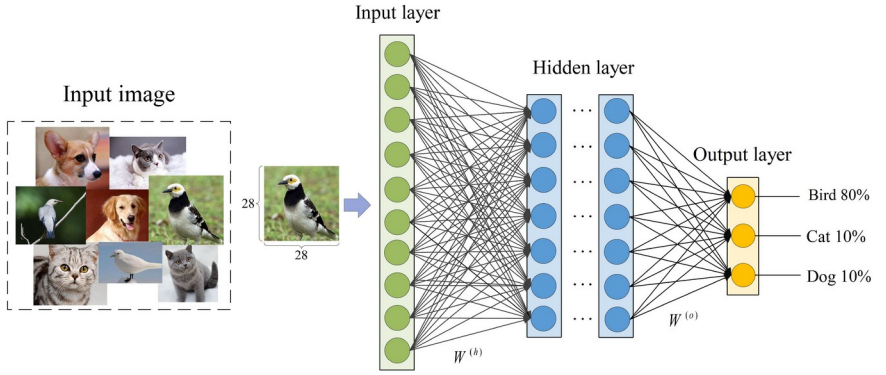




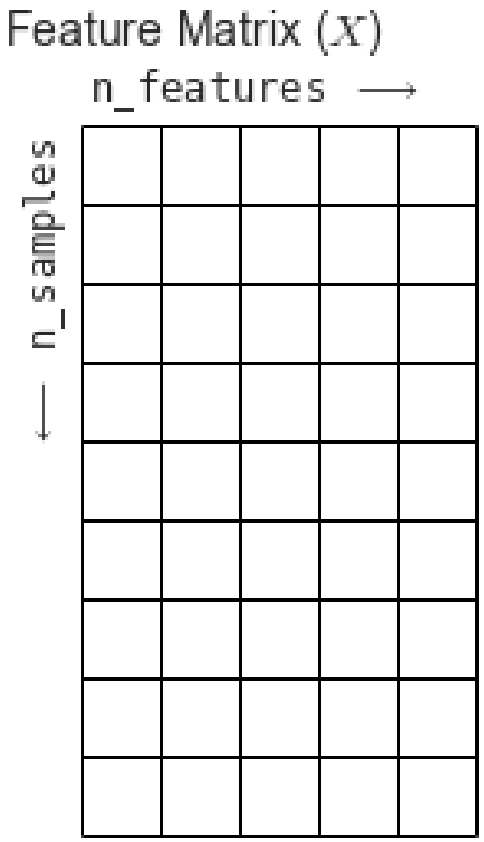
Multi-Class Classification

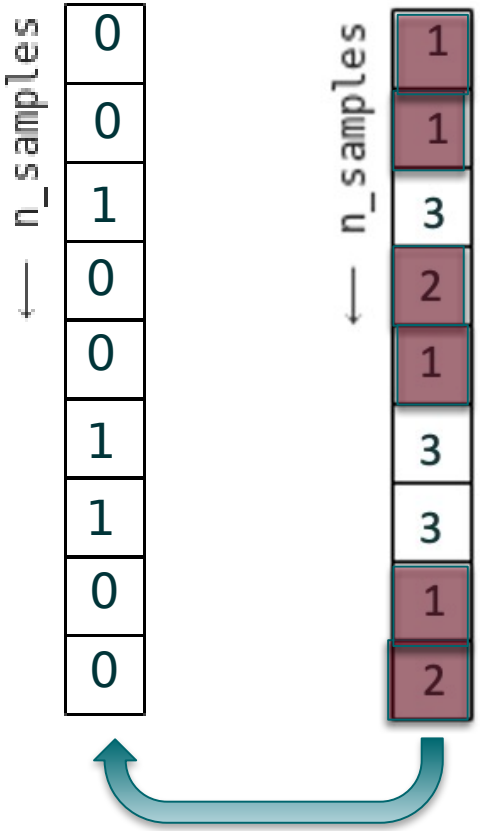


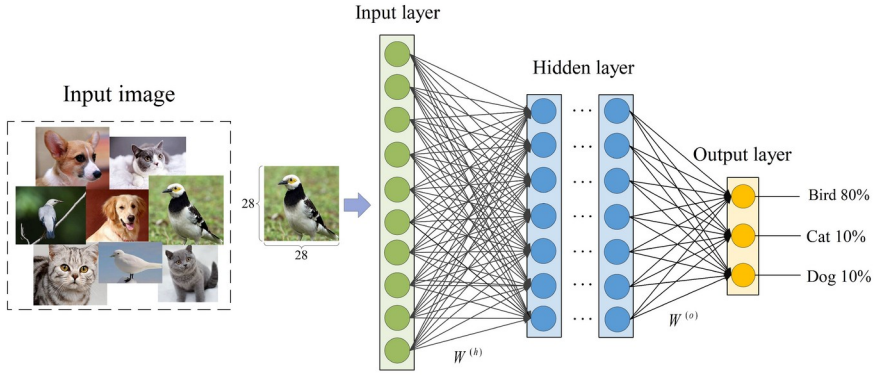




Multi-Class Classification

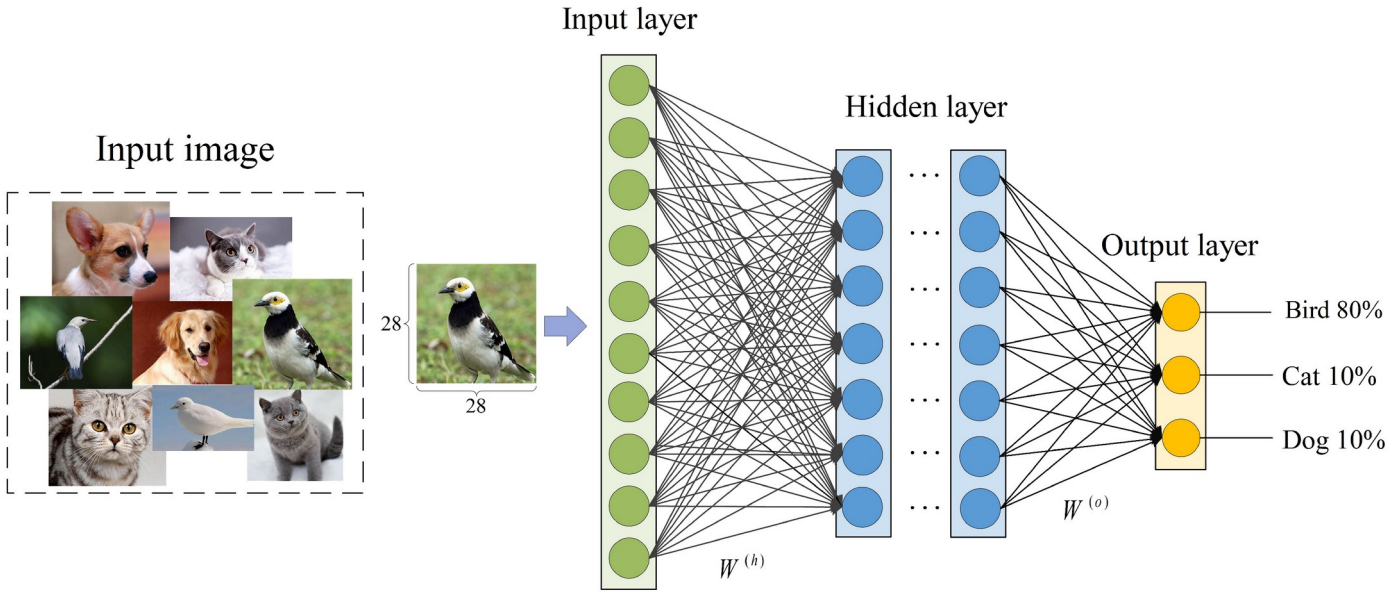






Multi-Class Classification - Propensities

- Linear and non-linear predictors output the probability of belonging to a particular class ($\hat{p}$)

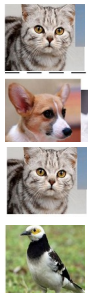


Categorical Cross-Entropy

- The most common **loss** function in multi-class classification

$$-\sum_{k} y_k \log \hat{p}_k$$

How do we determine the **distance** between this two matrices?



DOG	CAT	BIRD
0	1	0
1	0	0
0	1	0
0	0	1

p_dog	p_cat	p_bird
0.05	0.90	0.05
0.85	0.10	0.05
0.80	0.10	0.10
0.10	0.15	0.75

Categorical Cross-Entropy

- The most common **loss** function in multi-class classification

$$-\sum_{k} y_k \log \hat{p}_k$$

DOG	CAT	BIRD
0	1	0
1	0	0
0	1	0
0	0	1

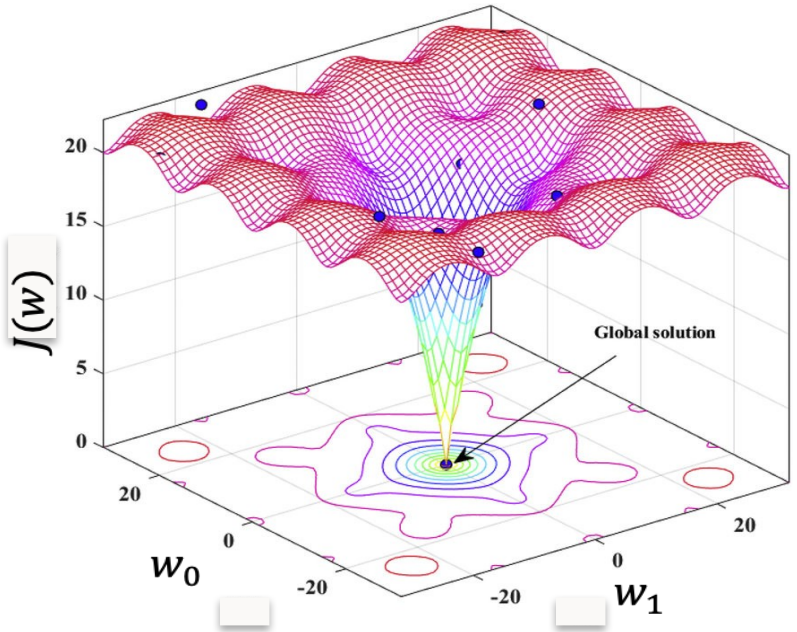
p_dog	p_cat	p_bird
0.05	0.90	0.05
0.85	0.10	0.05
0.80	0.10	0.10
0.10	0.15	0.75

$$-(0 \log 0.05 + 1 \log 0.90 + 0 \log 0.05) = 0.1054$$

Categorical Cross-Entropy

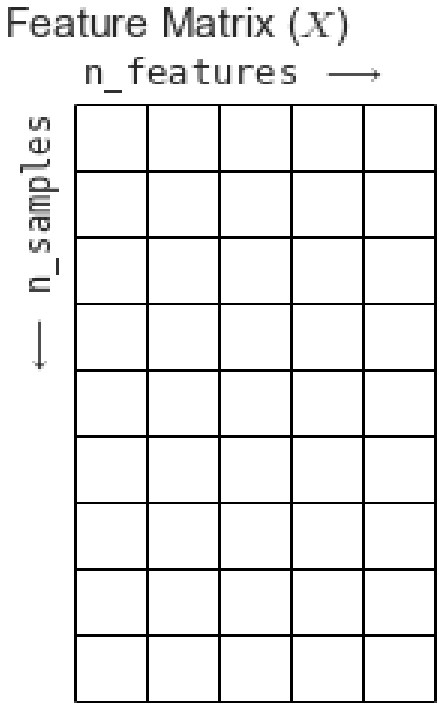
- The cost function is averaged over the observations

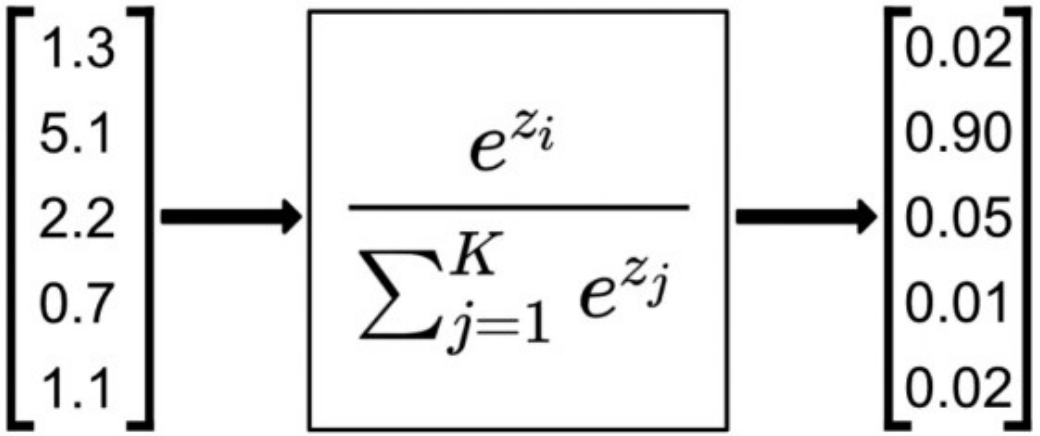
$$J(w) = -\frac{1}{n}\sum_{i}\sum_{k} y_k \log \hat{p}_k$$



Combining Predictions

- We use the softmax function to combine predictions



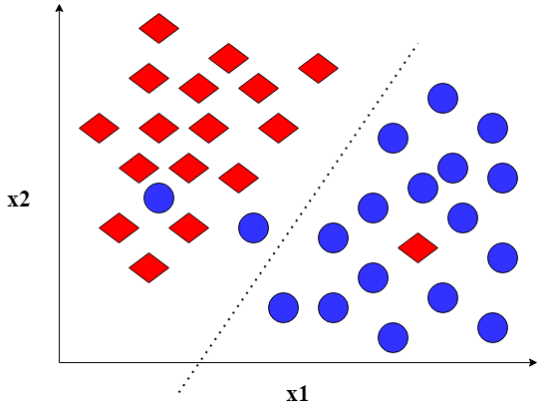


Binary Classification Algorithms?

- We can still use binary classification algorithms in multi-class classification
- Strategies: One-Vs-All (One-Vs-Rest) and One-Vs-One

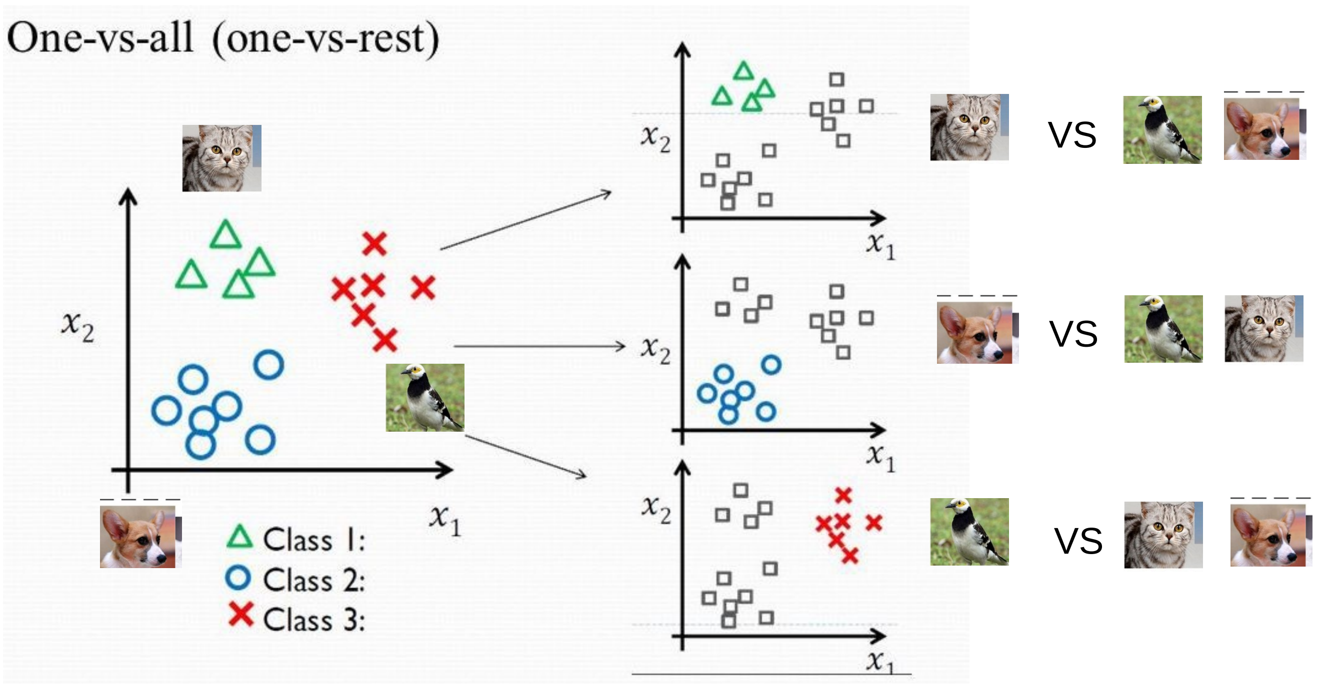
Example: **LOGISTIC REGRESSION**

$$\hat{p} = \frac{1}{1 + e^{-(w_0 + w_1 x_1 + \ldots.)}} = \frac{1}{1 + e^{Xw}}$$



One-Vs-All (One-Vs-Rest)

- This approach implies fitting the model multiple times
- In each fit the binary classifier compares one class against the rest



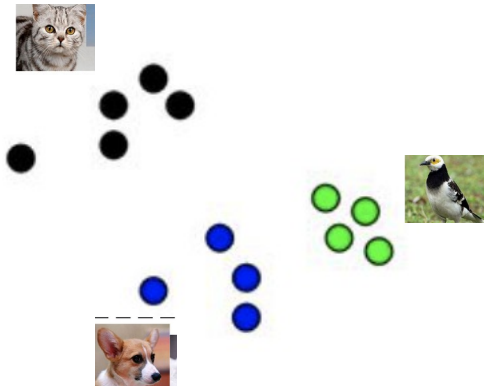
$$\hat{p} = \frac{1}{1+\ e^{-(w_0+w_1x_1+\ ....)}} = \frac{1}{1+\ e^{Xw}}$$

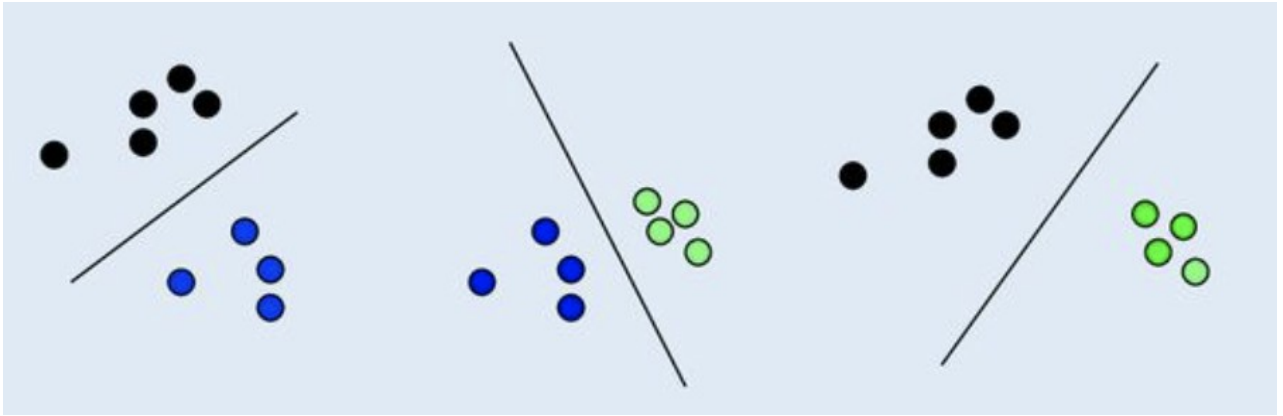
$$\hat{p} = \frac{1}{1+\ e^{-(w_0+w_1x_1+\ ....)}} = \frac{1}{1+\ e^{Xw}}$$

$$\hat{p} = \frac{1}{1+\ e^{-(w_0+w_1x_1+\ ....)}} = \frac{1}{1+\ e^{Xw}}$$

One-Vs-One

- This approach implies fitting the model multiple times
- In each fit the binary classifier compares one class against each other class





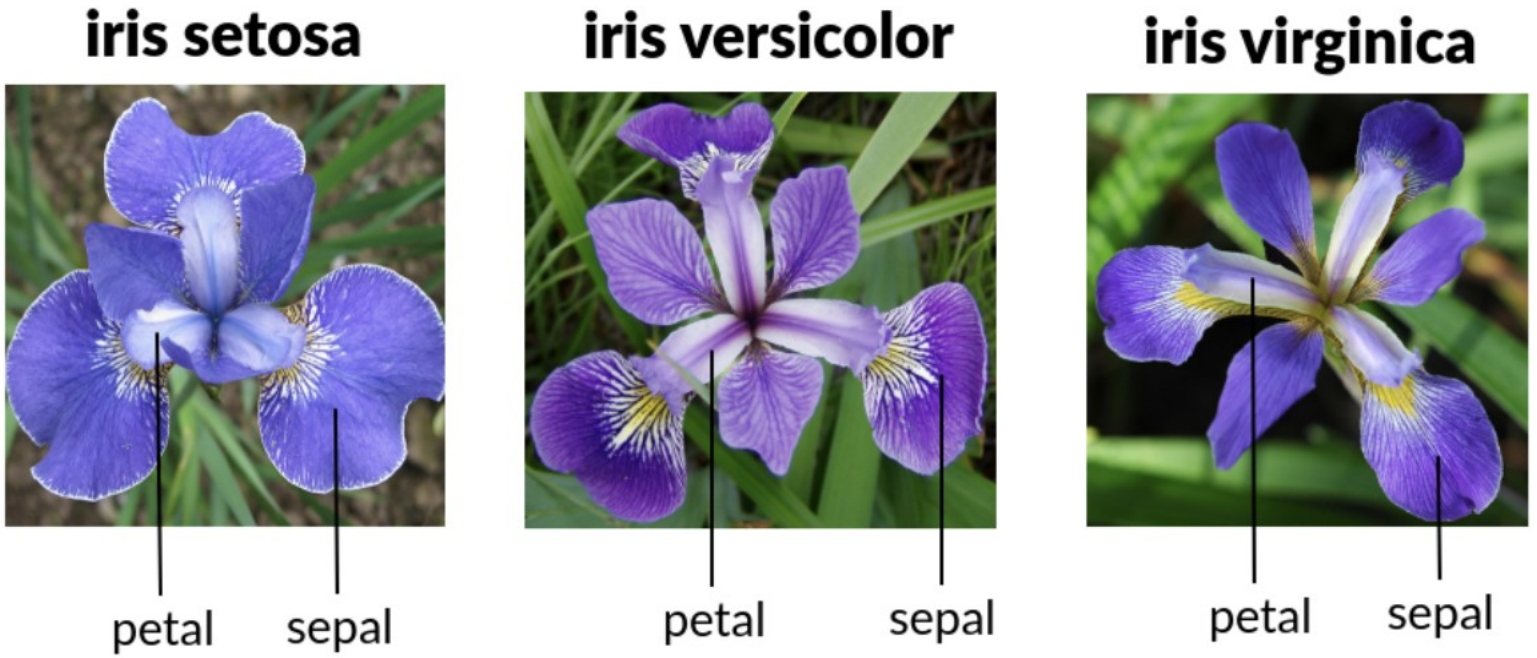
$$\hat{p} = \frac{1}{1 + e^{Xw}}$$

$$\hat{p} = \frac{1}{1 + e^{Xw}}$$

$$\hat{p} = \frac{1}{1 + e^{Xw}}$$

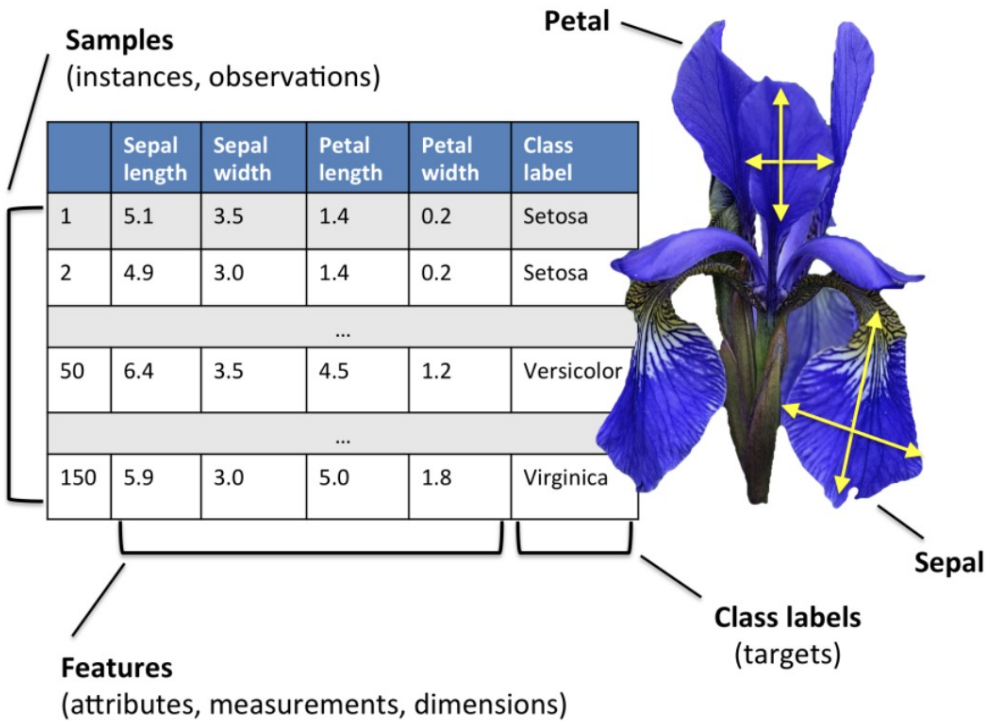
IRIS Dataset

- Classical ML dataset



IRIS Dataset

- Classical ML dataset



	Sepal length	Sepal width	Petal length	Petal width	Class label
1	5.1	3.5	1.4	0.2	Setosa
2	4.9	3.0	1.4	0.2	Setosa
...					
50	6.4	3.5	4.5	1.2	Versicolor
...					
150	5.9	3.0	5.0	1.8	Virginica

Python

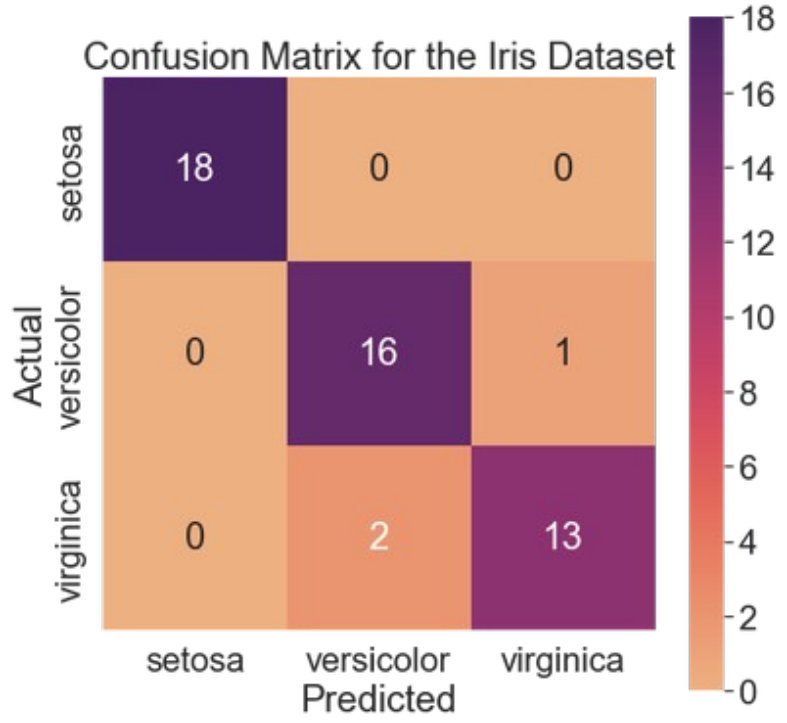
Metrics

- Used to determine if model is performing well

Scoring	Function	Comment
Classification		
'accuracy'	`metrics.accuracy_score`	
'balanced_accuracy'	`metrics.balanced_accuracy_score`	
'top_k_accuracy'	`metrics.top_k_accuracy_score`	
'average_precision'	`metrics.average_precision_score`	
'neg_brier_score'	`metrics.brier_score_loss`	
'f1'	`metrics.f1_score`	for binary targets
'f1_micro'	`metrics.f1_score`	micro-averaged
'f1_macro'	`metrics.f1_score`	macro-averaged
'f1_weighted'	`metrics.f1_score`	weighted average
'f1_samples'	`metrics.f1_score`	by multilabel sample
'neg_log_loss'	`metrics.log_loss`	requires `predict_proba` support
'precision' etc.	`metrics.precision_score`	suffixes apply as with 'f1'
'recall' etc.	`metrics.recall_score`	suffixes apply as with 'f1'
'jaccard' etc.	`metrics.jaccard_score`	suffixes apply as with 'f1'
'roc_auc'	`metrics.roc_auc_score`	
'roc_auc_ovr'	`metrics.roc_auc_score`	
'roc_auc_ovo'	`metrics.roc_auc_score`	
'roc_auc_ovr_weighted'	`metrics.roc_auc_score`	
'roc_auc_ovo_weighted'	`metrics.roc_auc_score`	

Metrics Review: Confusion Matrix

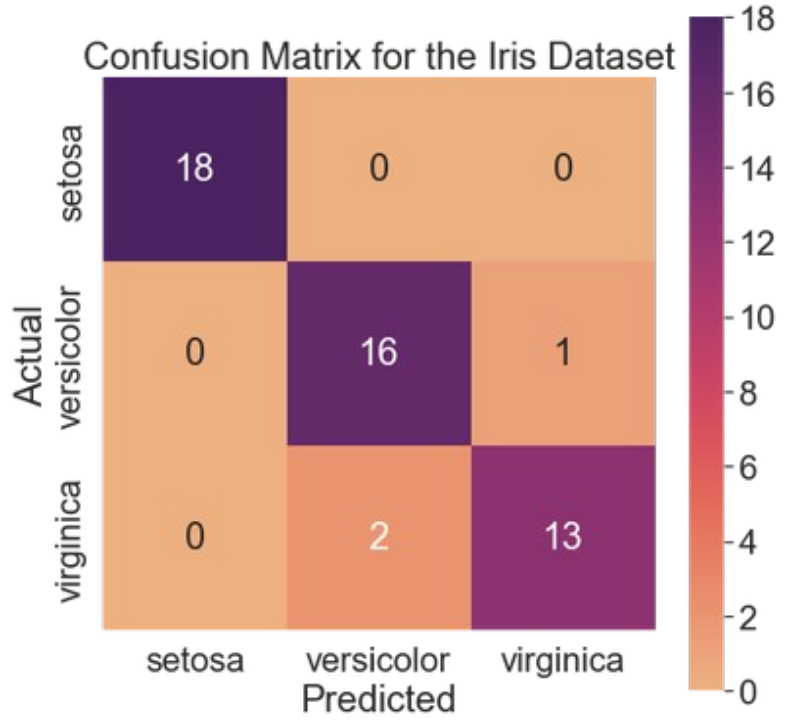
- The confusion matrix gives a summary of overall performance
- Accuracy can be derived as the fraction of correct predictions



$$\text{accuracy} = \frac{18 + 16 + 13}{50}$$

Metrics Review: Recall (Sensitivity)

- Detection rate for all classes
- We can use macro average (average by the number of classes).



$$\text{recall setosa} = \frac{18}{18} = 1$$

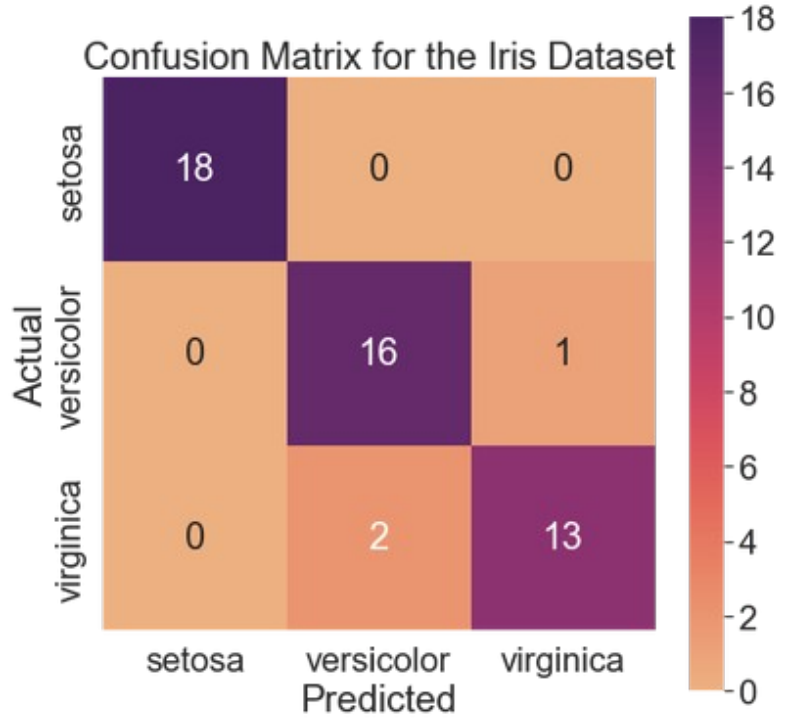
$$\text{recall versicolor} = \frac{16}{17} = 0.9412$$

$$\text{recall virgin}ica = \frac{13}{15} = 0.8667$$

$$\text{recall macro} = \frac{1 + 0.9412 + 0.8667}{3} = 0.9360$$

Metrics Review: Recall (Sensitivity)

- Detection rate for all classes
- We can use micro average (sum true detections over all obs).



$$\text{recall micro} = \frac{18 + 16 + 13}{18 + 17 + 15} = 0.9400$$

Python

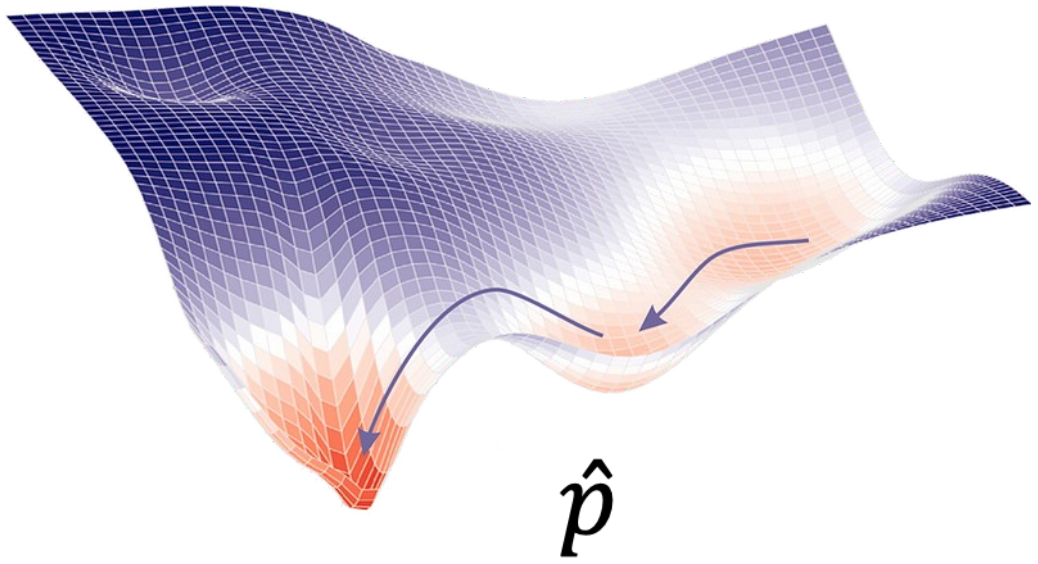
Common Cost Functions Neural Nets

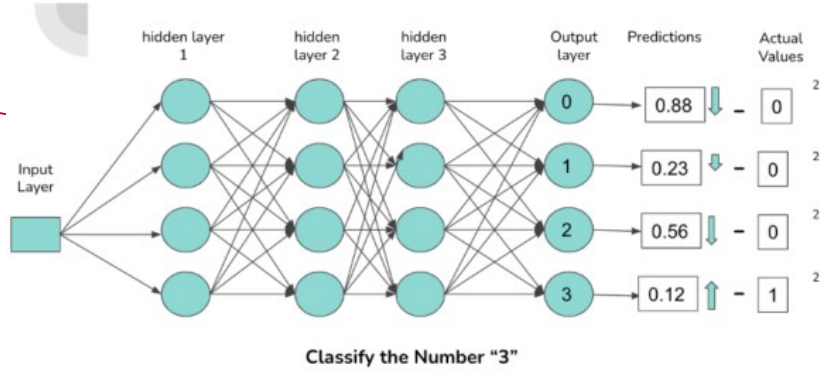
- Categorical Cross-Entropy
  - Loss Function

$$J(w) = -\sum_{k} y_k \log \hat{p}_k$$

  - Gradient

$$\nabla J(w) = \hat{p} - y$$

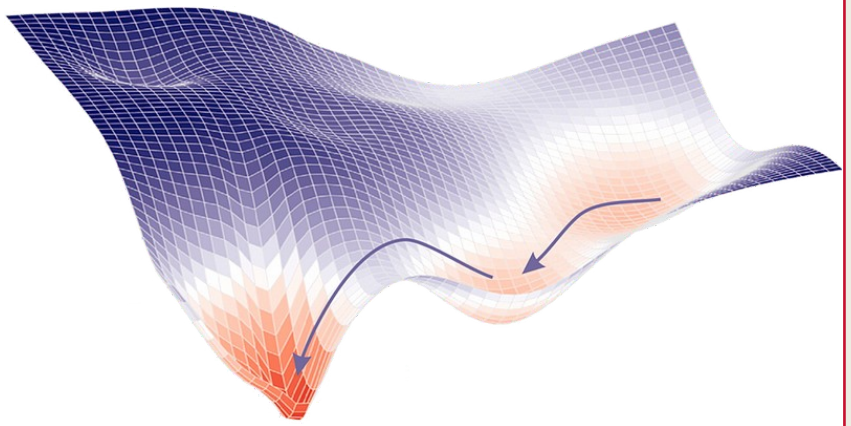


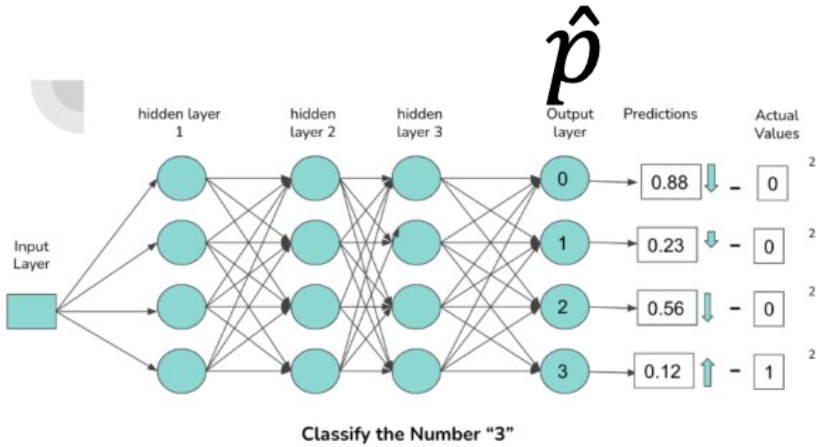


Classify the Number "3"

Activation Functions Neural Nets

- Common Activations on Layers
  - ReLu
  - Tanh
  - Sigmoid
- Output Layer Activation
  - Softmax





Classify the Number "3"

Python