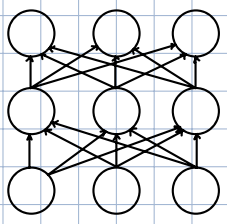
DEEP LEARNING BOOK - IAN GOODFELLOW (SOME POINTS FROM CHAPTER 1)

- QUINTESSENTIAL OF REPRESENTATION LEARNING ALGO → AUTOENCODER
- DEEP LEARNING → SOLVES CENTRAL PROBLEM OF REPRESENTATION LEARNING. → BUILDING COMPLEX CONCEPTS OUT OF SIMPLE CONCEPTS.
- ESSENTIAL OF DEEP LEARNING → MULTILAYER PERCEPTRON (MLP) or FEEDFORWARD DEEP NETWORK
  - E.G. BREAKS DOWN COMPLICATED MAPPING INTO SERIES OF NESTED MAPPINGS.



- DEEP LEARNING HAD A LOT OF DIFFERENT NAMES. → CYBERNETICS (40s-60s) → CONNECTIONISM (80-95) → DEEP LEARNING (06)
- MORE USEFUL WITH MORE DATA AVAILABLE
- GROWN IN SIZE → COMPUTATIONAL INFRASTRUCTURE
- SOLVING INCREASINGLY COMPLEX PROBLEMS
- ONE OF DEEP LEARNING NAMES ⇒ ARTIFICIAL NEURAL NETWORK (ANN) → ENGINEERED SYSTEMS INSPIRED BY BRAIN.

- NEUROSCIENCE HAS GIVEN REASON TO HOPE THAT A SINGLE DL ALGO CAN SOLVE MANY DIFFERENT TASKS → IMPORTANT SOURCE OF INSPIRATION.
  ↳ NEOCOGNITRON → BASIS OF MODERN CONVOLUTION NETWORK. (POWERFUL MODEL ARCHITECTURE FOR PROCESSING IMAGES)
- MOST NEURAL NETWORKS TODAY → BASED ON RECTIFIED LINEAR UNIT
- DON'T VIEW DEEP LEARNING AS AN ATTEMPT TO SIMULATE BRAIN.

CHAPTER 3

CHOOSING CLASSIFICATION ALGORITHM

- TAKES PRACTICE AND TIME
- "NO FREE LUNCH THEOREM" - DAVID H. WOLPERT - THERE ISN'T ONE CLASSIFIER THAT WORKS BEST FOR ALL SCENARIOS.
  ↳ CHOOSE A HANDFUL OF MODELS, THEN SELECT ONE THAT WORKS. → MODEL TRAINING VERY DEPENDANT ON DATA OVER TRAINING MODEL.
- 5 STEPS FOR SUPERVISED TRAINING:
  1. SELECT FEATURES + COLLECT LABELLED TRAING DATA.
  2. CHOOSE PERFORMANCE METRIC
  3. CHOOSE LEARNING ALGORITHM + TRAINING MODEL
  4. EVALUATE PERFORMANCE OF MODEL.
  5. FINETUNE MODEL

- DISCLAIMER → FEATURE SELECTION + PREPROCESSING + ETC.. → COVERED LATER IN BOOK

SCIKIT-LEARN - PERCEPTRON

- USER-FRIENDLY + OPTIMISED IMPLEMENTATION OF SEVERAL CLASSIFICATION ALGORITHMS.
- BUILDING PERCEPTRON WITH SCIKIT-LEARN USING IRIS DATASET.
- 'train_test_split' → SPLIT DATA INTO TEST + TRAINING DATA.
- MANY MACHINE LEARNING + OPTIMISATION ALGORITHMS → REQUIRE FEATURE SCALING FOR OPTIMAL PERFORMANCE → E.G. GRADIENT DESCENT.

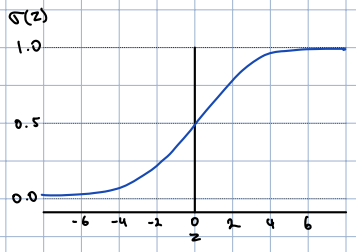
LOGISTIC REGRESSION + CONDITIONAL PROBABILITIES

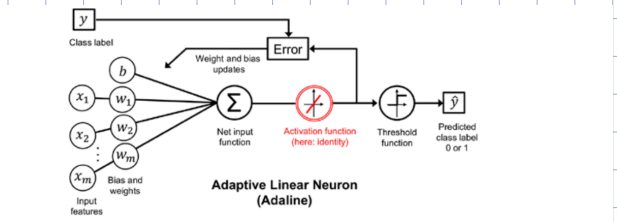
- LOGISTIC REGRESSION → CLASSIFICATION MODEL → PERFORMS WELL ON LINEARLY SEPARABLE CLASSES.
- ONE OF THE MOST WIDELY USED ALGORITHMS FOR CLASSIFICATION
- LINEAR MODEL FOR BINARY CLASSIFICATION.
- LOGISTIC REGRESSION FOR MULTIPLE CLASSES → MULTINOMIAL LOGISTIC REGRESSION OR SOFTMAX REGRESSION (LOOK INTO IT)
- MAIN MECHANICS:
  - ODDS ⇒ ODDS IN FAVOUR OF PARTICULAR EVENT. ⇒ $\frac{p}{(1-p)}$, p = PROBABILITY OF POSITIVE EVENT.
    POSITIVE EVENT LABEL → $y=1$ ⇒ $p := p(y=1|x)$
    ↳ EVENT WE WANT TO PREDICT. DOESN'T NECESSARILY MEAN "GOOD".
  - LOGIT FUNCTION ⇒ $\text{logit}(p) = \log\frac{p}{(1-p)}$ ⇒ log = NATURAL LOG
    ↳ LOG OF ODDS
    ↳ TAKES INPUT VALUE IN THE RANGE 0 TO 1.
    → LOGIT FUNCTION MAPS PROBABILITY TO REAL-NUMBER RANGE
    ↳ CONSIDER INVERSE OF FUNCTION → MAP BACK TO [0,1] RANGE.

$$\text{logit}(p) = w_1x_1 + \dots + w_mx_m + b = \sum_{i=j} w_jx_j + b = w^Tx + b$$

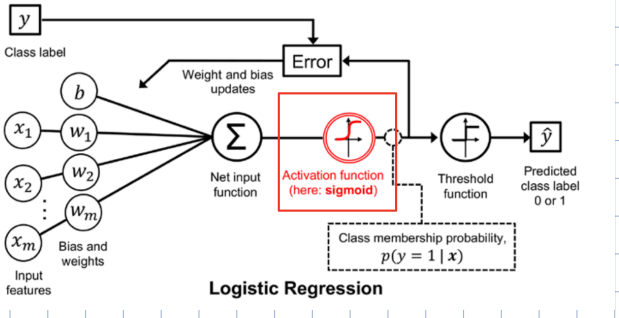
- INVERSE OF LOGIT FUNCTION → LOGISTIC SIGMOID FUNCTION → ABB. SIGMOID FUNCTION → S SHAPE

$$\sigma(z) = \frac{1}{1+e^{-z}}$$ → z NET INPUT → $z = w^Tx + b$





- $\sigma(z) \to 1$ IF $z \to \infty$ BECAUSE e^{-z} BECOMES VERY SMALL FOR LARGE VALUES.
- $\sigma(z) \to 0$ FOR $z \to -\infty$ AS A RESULT OF AN INCREASINGLY LARGE DENOMINATOR.
- CONCLUSION → SIGMOID TAKES REAL-NUMBER VALUES AS INPUT AND TRANSFORMS THEM INTO VALUES IN RANGE [0,1], INTERCEPT $\sigma(0) = 0.5$



- E.G. → $\sigma(z) = 0.8$ = 80 % IS A IRIS-VERSICOLOR

$$p(y=0|x,w,b) = 1 - p(y=1|x;w,b) = 0.2 \to 20\,\%$$

$$\hat{y} \begin{cases} 1 & \text{IF } z \geq 0.0 \\ 0 & \text{OTHERWISE} \end{cases}$$

PREDICTED PROBABILITY CAN THEN SIMPLY BE CONVERTED INTO BINARY OUTCOME.

- LOGISTIC REGRESSION USED IN WEATHER FORECASTING. ALSO POPULAR IN MEDICINE.

MODEL WEIGHTS VIA LOGISTIC LOSS FUNCTION

- MINIMIZE NSE TO LEARN PARAMETERS FOR ADALINE CLASSIFICATION MODEL.

$$L(w,b|x) = p(y|x;w,b) = \prod_{i=1}^{n} p(y^{(i)};w,b) = \prod_{i=1}^{n} (\sigma(z^{(i)}))^{y^{(i)}} (1-\sigma(z^{(i)}))^{1-y^{(i)}}$$

↳ IN PRACTICE → LOG-LIKELIHOOD

$$l(w,b|x) = \log L(y|x;w,b) = \sum_{i=1}^{} [y^{(i)} \log(\sigma(z^{(i)})) + (1+y^{(i)}) \log(1-\sigma(z^{(i)}))]$$

LOGISTIC LOSS FUNCTION

$$L(\theta) = \prod_{i=1}^{n} P(y^{(i)}|x^{(i)};\theta) \longrightarrow \text{LOG-LIKELIHOOD}$$

$$\log L(\theta) = \sum_{i=1}^{n} \log P(y^{(i)}|x^{(i)};\theta)$$

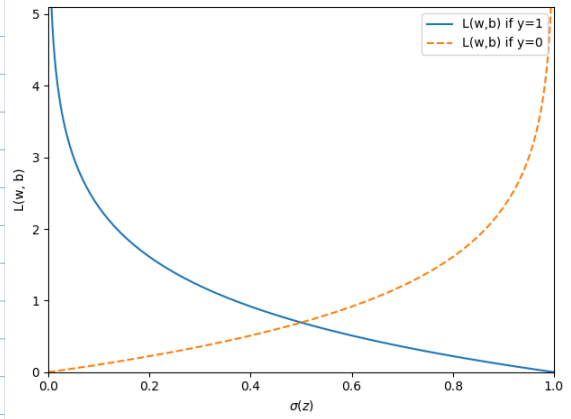
1. APPLYING LOG FUNCTION REDUCES POTENTIAL NUMERIC UNDERFLOW
   - WORKING WITH PROBABILITY IN RANGE 0-1.
     ↳ SOMETIMES PROBABILITIES CAN GET REALLY CLOSE TO 0 OR 1.
   } $\log(p)$ MAKES VERY TINY PROBABILITIES MANAGEABLE + NUMERICALLY STABLE.
   ↳ IMPROVES RELIABILITY FOR MODEL TRAINING PROCESS

2. CONVERT PRODUCT OF FACTORS INTO SUMMATION OF FACTORS. → EASIER TO COMPUTE DERIVATIVES.
   ↳ e.g. $\log(a \times b \times c) = \log(a) + \log(b) + \log(c)$
   ↳ TAKING LOG OF PRODUCTS ALLOWS YOU TO WRITE THE LOSS AS A SUM, MAKING GRADIENT CALCULATION FOR WEIGHTS SIMPLER + STABLE.

FOUNDATION OF LOGISTIC REGRESSION → LIKELIHOOD FUNCTION.
↳ GOAL ⇒ MAXIMISE THE LIKELIHOOD ⇒ DONE BY GRADIENT ASCENT.
↳ HOWEVER → CAN USE GRADIENT DESCENT WITH NEGATIVE LOG-LIKELIHOOD. → ML FRAMEWORKS DESIGNED FOR MINIMISING → COMMON TO SWITCH TO NEGATIVE.

$$L(w,b) = \sum_{i=1}^{} [-y^{(i)} \log(\sigma(z^{(i)})) + (1+y^{(i)}) \log(1-\sigma(z^{(i)}))]$$

$$l(\sigma(z),y;w,b) = -y\log(\sigma(z)) - (1-y)\log(1-\sigma(z)) \longrightarrow l(\sigma(z),y;w,b) = \begin{cases} -\log(\sigma(z)) & \text{if } y=1 \\ -\log(1-\sigma(z)) & \text{if } y=0 \end{cases}$$



MAIN POINT → WE PENALISE WRONG PREDICTIONS WITH INCREASINGLY LARGER LOSS.

CONVERTING ADALINE IMPLEMENTATION INTO ALGORITHM FOR LOGISTIC REGRESSION

↳ COMPUTE LOSS OF CLASSIFYING PER EPOCH } SWITCH THIS TWO → BECOMES LOGISTIC REGRESSION.
↳ SWAP LINEAR ACTIVATION WITH SIGMOID.
↳ FIT LOGISTIC REGRESSION MODEL → ONLY WORKS FOR BINARY CLASSIFICATION TASKS.

GRADIENT DESCENT LEARNING ALGORITHM FOR LOGISTIC REGRESSION

- ADALINE, WEIGHTS + BIAS DIDN'T CHANGE FOR LOGISTIC REGRESSION → GRADIENT DESCENT SIMILAR FOR LR.
- PARTIAL DERIVATIVE OF LOG-LIKELIHOOD WITH RESPECT TO j^{th} TERM:

$$\frac{\partial L}{\partial w_j} = \frac{\partial L}{\partial a}\frac{\partial a}{\partial z}\frac{\partial z}{\partial w_j} \quad \text{WHERE } a(z) = \frac{1}{1+e^{-z}}$$

$$\frac{\partial L}{\partial a} = \frac{a-y}{a\cdot(1+a)}$$

$$\frac{\partial a}{\partial z} = \frac{e^{-z}}{(1+e^{-z})^2} = a\cdot(1+a)$$

$$\frac{\partial L}{\partial z} = a-y$$

$$\frac{\partial z}{w_j} = x_j$$

$$\frac{\partial L}{\partial w_j} = (a-y)x_j = -(y-a)x_j$$

→ TAKE STEP OPPOSITE OF GRADIENT → FLIP $\frac{\partial L}{w_j} = -(y-a)x_j$ + LEARNING RATE η

$$w_j := w_j + \eta(y-a)x_j$$

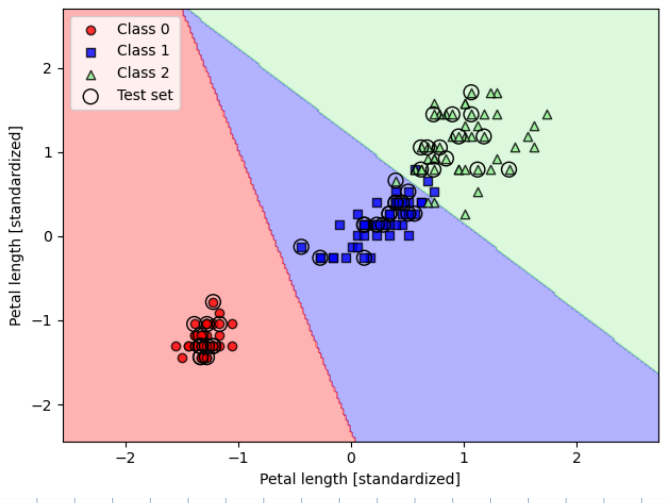
$$b := b + \eta(y-a)$$

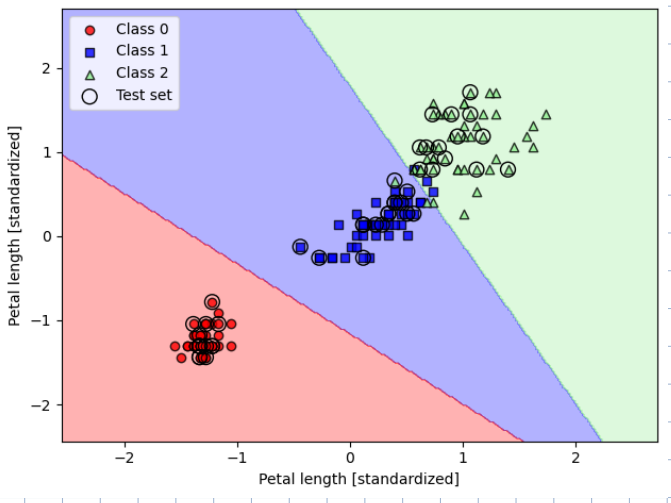
WEIGHT + BIAS = ADALINE WEIGHT + BIAS

LOGISTIC REGRESSION WITH SCIKIT-LEARN

- SUPPORTS MULTI-CLASS LOGISTIC REGRESSION OFF THE SHELF

DEFAULT FOR SCIKIT-LEARN LOGISTIC REGRESSION





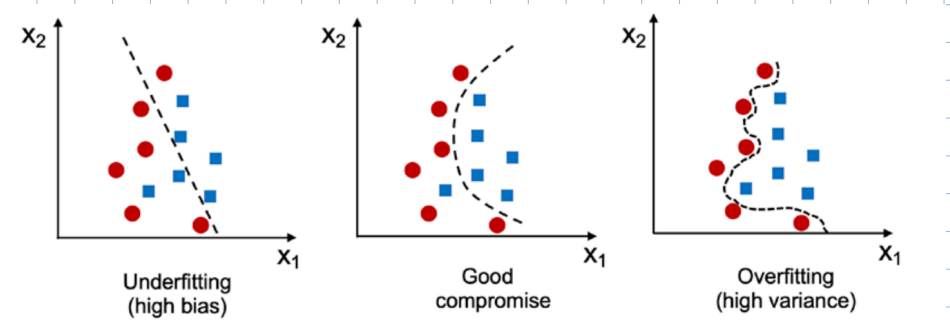
MULTI-CLASS = OvR

MULTI-CLASS = MULTINOMIAL

CONVEX OPTIMISATION → MINIMISING CONVEX LOSS FUNCTIONS (LOGISTIC REGRESSION) → RECOMMENDED TO USE MORE ADVANCED ALGO OVER SGD.

OVERFITTING VIA REGULARISATION

- COMMON PROBLEM FOR MACHINE LEARNING → DOESN'T GENERALISE WELL WITH UNSEEN DATA.
- OVERFITTING → HIGH VARIANCE MODEL → CAUSED BY TOO MANY PARAMETERS (TOO COMPLEX)
- UNDERFITTING → HIGH BIAS → MODEL NOT COMPLEX ENOUGH



BIAS-VARIANCE TRADE-OFF

WHAT IS BIAS?

- MEASURES HOW FAR OFF THE PREDICTIONS ARE FROM THE CORRECT VALUES.
- HIGH-BIAS MODEL MAKES STRONG ASSUMPTIONS ABOUT DATA BUT UNDERFITS - CAN'T FIGURE OUT THE UNDERLYING PATTERN WELL.

WHAT IS VARIANCE?

- MEASURES CONSISTENCY OF MODEL PREDICTION
- HIGH-VARIANCE → MODEL IS SENSITIVE TO SPECIFIC TRAINING DATA (FITS NOISE + SMALL DATA FLUCTUATION)

THE TRADE-OFF

- GOTTA FIND THE SWEET SPOT BETWEEN THE TWO.
- ONE WAY TO FIND GOOD BIAS-VARIANCE TRADEOFF → TUNE COMPLEXITY OF MODEL VIA REGULARISATION.

REGULARIZATION → VERY USEFUL FOR HANDLING COLLINEARITY, FILTERING OUT NOISE FROM DATA + EVENTUALLY PREVENTING OVERFITTING.

↳ ADD ADDITIONAL INFORMATION TO PENALISE EXTREME PARAMETER VALUES.

↳ MOST COMMON → L2-REGULARISATION

REGULARISATION → FEATURE SCALING → STANDARDISATION → IMPORTANT.

↳ FOR THIS TO WORK → FEATURES MUST BE ON COMPARABLE SCALES.

$$\frac{\lambda}{2n}\|w\|^2 = \frac{\lambda}{2n}\sum_{j=1}^{m} w_j^2$$

λ = REGULARISATION PARAMETER

2 = SCALING FACTOR → CANCELS COMPUTING LOSS FUNCTION.

n = SAMPLE SIZE ADDED TO SCALE REGULARISATION TERM SIMILAR TO LOSS.

LOSS FUNCTION FOR LOGISTIC REGRESSION + REGULARISATION:

$$L(w,b) = \sum_{i=1}^{n} \left[-y^{(i)} \log\left(\sigma(z^{(i)})\right) + \left(1 - y^{(i)}\right) \log\left(1 - \sigma(z^{(i)})\right) \right] + \frac{\lambda}{2n} \|w\|^2$$

PARTIAL DERIVATIVE OF UNREGULARISED LOSS:

$$\frac{\partial L(w,b)}{\partial w_j} = \left(\frac{1}{n} \sum_{i=1}^{n} \left(\sigma\left(w^T x^{(i)}\right) - y^{(i)} \right) x_j^{(i)} \right)$$

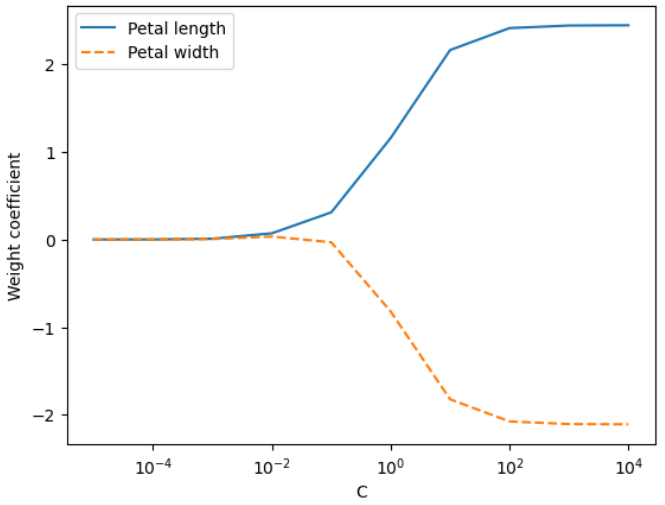
ADDING REGULARISATION TO DERIVATIVE OF LOSS:

$$\Longrightarrow \quad \frac{\partial L(w,b)}{\partial w_j} = \left(\frac{1}{n} \sum_{i=1}^{n} \left(\sigma\left(w^T x^{(i)}\right) - y^{(i)} \right) x_j^{(i)} \right) + \frac{\lambda}{n} w_j$$

CAN CONTROL HOW CLOSELY WE FIT THE DATA WITH λ → INCREASE λ, INCREASE REGULARISATION STRENGTH.

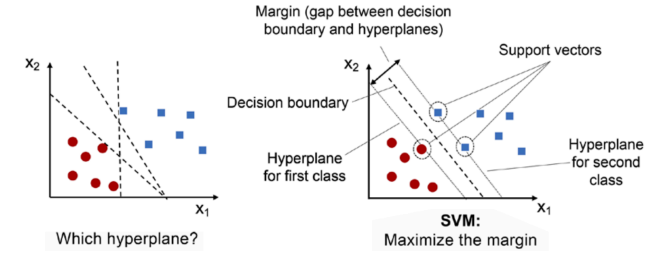
WHY DO WE NOT STRONGLY REGULARISE ALL MODELS IF IT REDUCES OVERFITTING?

↳ IF REGULARISATION → TOO HIGH → WEIGHTS COEFFICIENTS APPROACH 0 → THEN MODEL WILL UNDERFIT.



MAXIMUM MARGIN CLASSIFICATION WITH SUPPORT VECTOR MACHINES

- POWERFUL + WIDELY USED ALGORITHM → SUPPORT VECTOR MACHINE (SVM) → CONSIDERED EXTENSION OF PERCEPTRON.
- SVM - MAXIMISE MARGIN → DISTANCE BETWEEN THE SEPARATING HYPERPLANE (DECISION BOUNDARY) AND TRAINING EXAMPLES CLOSEST TO HYPERPLANE.
  ↳ SUPPORT VECTORS

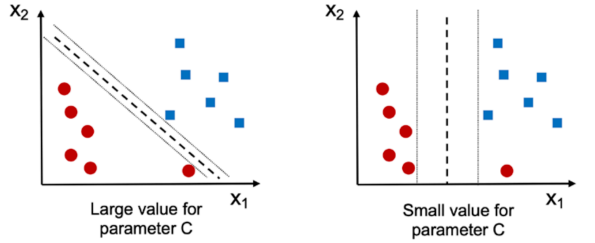


MAXIMUM MARGIN INTUITION

- LARGER MARGINS → TEND TO HAVE A LOWER GENERALIZATION ERROR.
- SVM LOOKS SIMPLE → MATH IS COMPLEX → NEED TO KNOW CONSTRAINED OPTIMISATION.

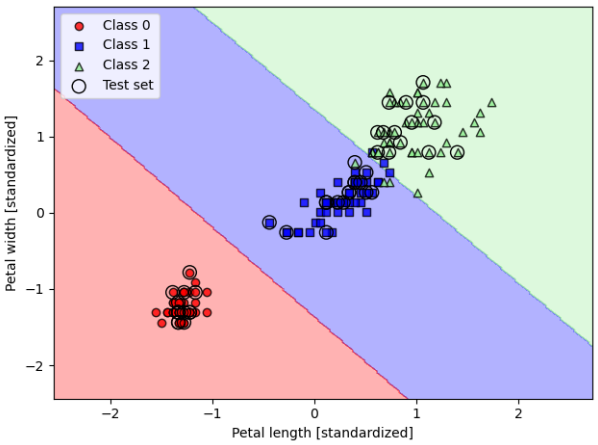
SLACK VARIABLE

- WON'T GO INTO MATHEMATICAL EXPLANATION ⇒ LOOK INTO THIS + SVM.
- INTRODUCED → VLADIMIR VAPNIK (1995) → LED TO SOFT-MARGIN CLASSIFICATION
  ↳ STRICT CONSTRAINTS → FORCE SVM TO FIND A HARD BOUNDARY → FAILS IF DATA ISN'T CLEANLY SEPARABLE.
  ↳ RELAXING CONSTRAINTS → ADDING SLACK VARIABLES → SOME POINTS CAN BE MISCLASSIFIED → BUT PENALTY FOR EACH MISTAKE.
  ↳ SVM OPTIMIZATION SEEKS BALANCE → WIDE MARGIN (GOOD SEPARATION) + FEW MISCLASSIFICATION (LOW SLACK PENALTY)
- TO USE: INTRODUCE VARIABLE C → HYPERPARAMETER FOR CONTROLLING PENALTY FOR MISCLASSIFICATION.



- LARGER C ⇒ LARGE ERROR PENALTIES
- SMALLER C ⇒ LESS STRICT ABOUT MISCLASSIFICATION ERROR
- C CONTROLS WIDTH OF MARGIN → TUNES BIAS-VARIANCE TRADEOFF

THREE DECISION REGIONS OF SVM.



LOGISTIC REGRESSION VS. SVM

- LOGISTIC REGRESSION + SVM IN PRACTICAL CLASSIFICATION YIELD SIMILAR RESULTS.
- LOGISTIC REGRESSION → MAXIMISE CONDITIONAL LIKEHOOD
- SVM → CARES ABOUT POINTS CLOSEST TO DECISION BOUNDARY (SUPPORT VECTORS)
- LOGISTIC REGRESSION → SIMPLER MODEL + EASY TO IMPLEMENT + EASILY UPDATEABLE.