ML - Bias Variance,
Regularization
Hyperparameter Tuning
Cross Validation

Revision

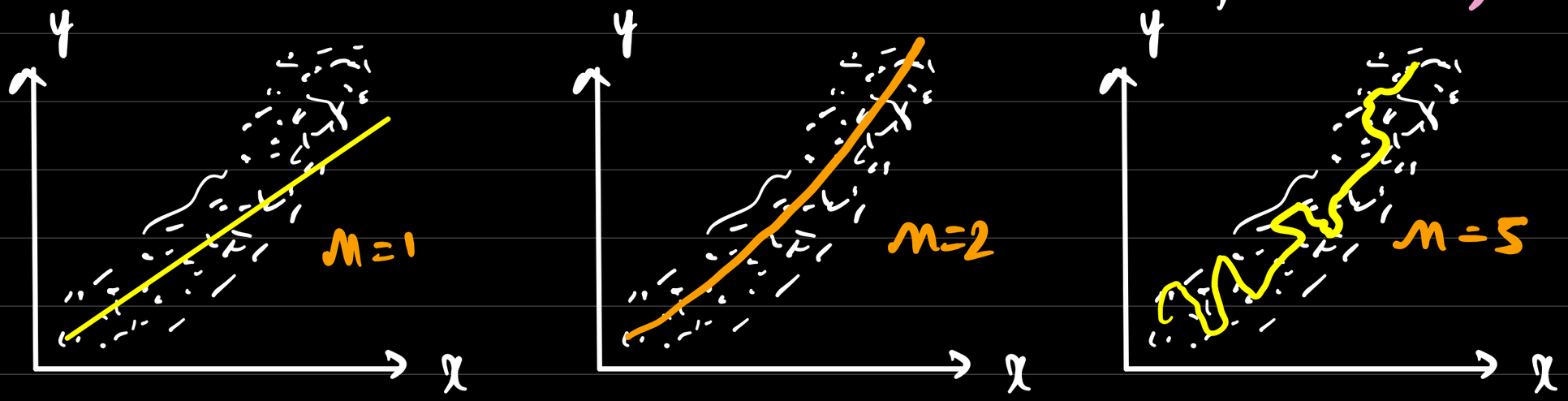
Polynomial Reg → $n > 1$
↓
select degree with highest R-sq. ← Training Performance
↓
Best degree poly eq may not work the best on Testing data. Why ??

↓

Overfitting to train data (learn patterns + noisy)

too

Underfitting - simple models (not even able to learn patterns)



Underfitting

Just right fit

Overfitted

Bias Variance Trade-off

Bias : Model is constantly 'biased' towards doing wrong predictions. - HIGH BIAS

Variance : Just with a slight change in X

$[1, 3, 4.4, 2] \rightarrow [1.1, 3, 4.5, 2.1]$

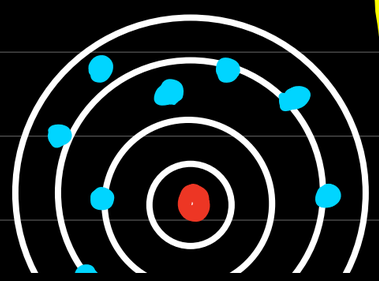
$x_i \rightarrow y$ $\quad$ $x_j \rightarrow y \pm \Delta$

predict is 'varying' a lot - HIGH VAR.

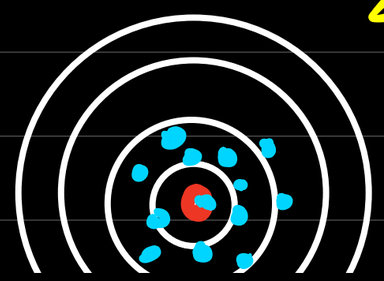
e.g.: You are learning archery 

- Prediction $\hat{y}$
- Ground Truth y

1



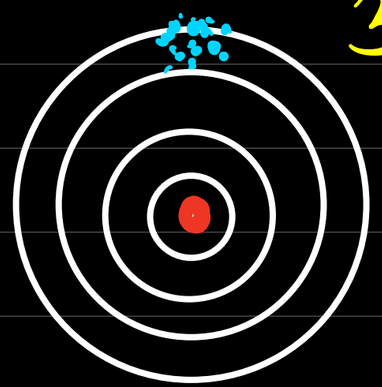
2



Overfitting

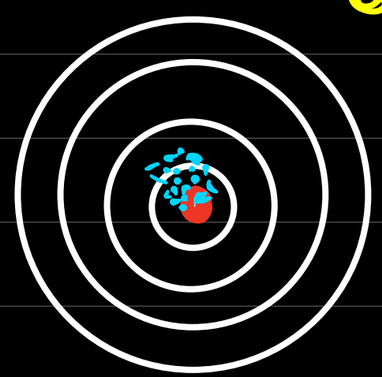
↑ BIAS
↑ VARIANCE

↓ BIAS
↑ VARIANCE



3

Underfitting



4

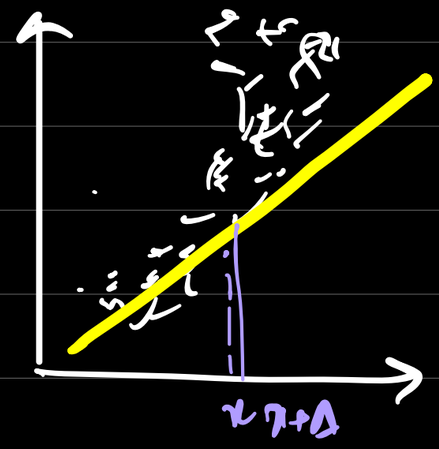
= Ideal

↑ BIAS
↓ VARIANCE

↓ BIAS
↓ VARIANCE

Ideally WANT ↓ Bias ↓ Variance.

Underfitting



Can't even learn the patterns in the data

↓

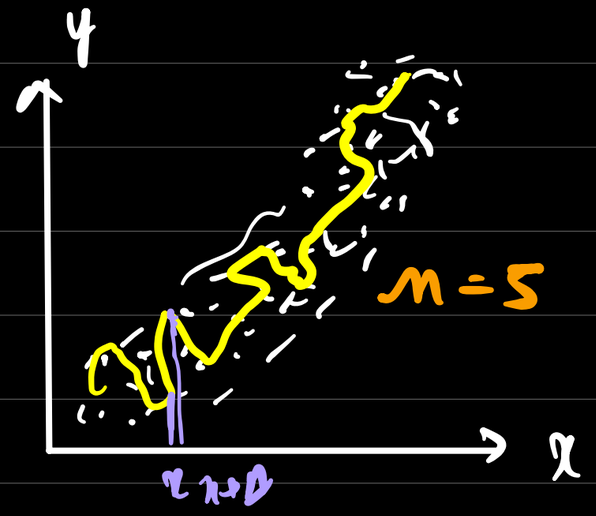
Bad predictions

Bias ↑↑

Variance ↓↓

Simple Models - Prob. of ↑ Bias.

Overfitting



Bias - Is model making bad pred. always?

NO Training Data - GOOD
Testing Data - May or not be good.

↓ Bias, ↑ Variance.

Practical tips for detecting U or V ?

Performance	Underfitting	Overfitting
Train	BAD	GOOD
Test	BAD	BAD or NOT AS GOOD

Still better than an underfitted model

Training

Bad Performance → Underfitting

↓

(Increase model complexity)

Good Performance → look at the testing performance

Testing

Bad Performance → Overfitting

↓

Reduce model complexity

Good Performance - DANCE

Deploy

just one right fit

- Train
- Test

Overfitted

Monotonically ↓ing

Underfit

Loss / Error / MSE

Model complexity

degree

Q Train - 90% Test - 90%

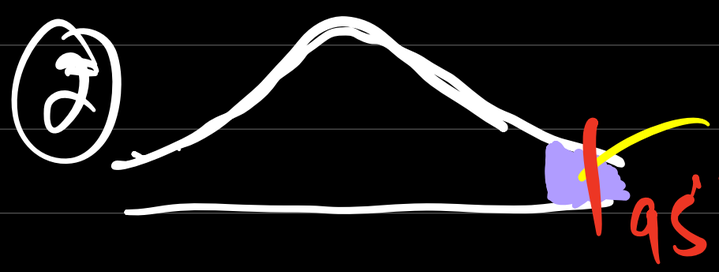
↓ ↓

~ Good Good - No overfitting

Test the right fit

Q Test Performance > Train Performance.

① Data Leakage (Reusing test data)

② 

sampling data from rare region.

- Test data as a nice/ideal/beautiful/no trouble subset - NO OUTLIERS
NO WEIRDNESS

Regularisation.

degree = 1, 2, 3, 4, 5, 6 7, 8, 10, 20, 30, 40

$y = 0.0002x^{20} + 1.8x^{18} + 20x + 40$

less imp. more imp.

Might have stopped at this

$$y = 20x + 40 + \ldots$$

GOAL: To select a higher degree or complex

YET

Somehow prevent overfitting

^

AUTOMATICALLY

Assume, chose d 30 model

$$y = w_0 + w_1 x_1 + w_2 x^2 + w_3 x^3 + \dots + w_{29} x^{29} + w_{30} x^{30}$$

Goal: Prevent overfitting

↓

Reduce w's of less imp features.

magnitude.

$\text{Loss}_{\text{UNREG}} = \text{MSE}$

↓

Training = Minimum loss.

= Minimizing errors. y $\hat{y}$

Training data.

∵ Chosen a high degree polynomial.

$\xrightarrow{d}$ Reduce error → Overfitting

Solution??

↓ overfitting

$$\text{loss}_{\text{LINREG}} = \text{MSE} + \frac{\quad}{\quad}$$

↑ ỹ overfitting

Q $LR = 1 + 1.5f_1 + 0.2f_2 + 0.f_3$

least imp feature - f_3

Reduce this

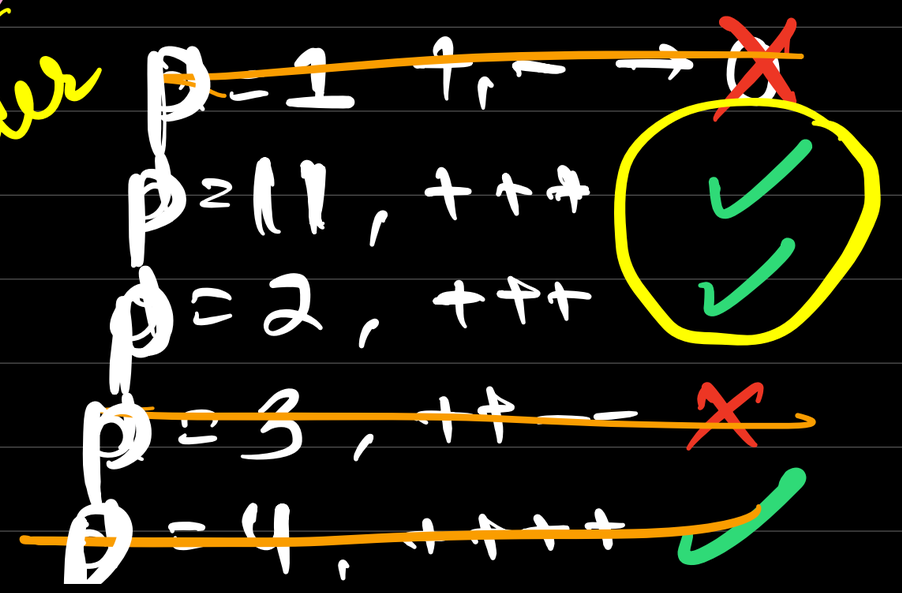
$$MSE = \frac{\sum_{i=1}^{m} e_i}{m} = \frac{\sum_{i=1}^{m} (y^i - \hat{y}^i)^2}{m}$$

Goal 1: $\hat{y}^i \longleftrightarrow y^i$

Goal 2: Reduce the w's for less imp features.

$$\sum_{j=1}^{d} (w_j)^{p}$$ any power

2, 4, 6, 8



Reduce irrelevant w_j's so that $\sum w_j$ will be less

Error minimization – three options

① MAE $\sum_{i=1}^{m} |e_i|$

② MSE $\sum_{i=1}^{m} |e_i|^2$

Why $\sum_{j=1}^{d} |w_j|$ $\sum_{j=1}^{d} (w_j)^2$ — To minimize.

Option 1

Option 2.

Loss update = MSE + Regularization loss.

Minimize

Reduce Errors

May ↑ overfitting

Reduce irrelevant W's

↓ Overfitting

TUG OF WAR.

Find equilibrium.

L1 Regularisation - $\sum_{j=1}^{d} |w_j|$ - lasso

L2 Regularisation - $\sum_{j=1}^{d} (w_j)^2$ - Ridge.

Q Should we give equal weightage to MSE and Regions?

Analogy

$$W_{new} = W_{old} + \alpha(-Gradient)$$

Why

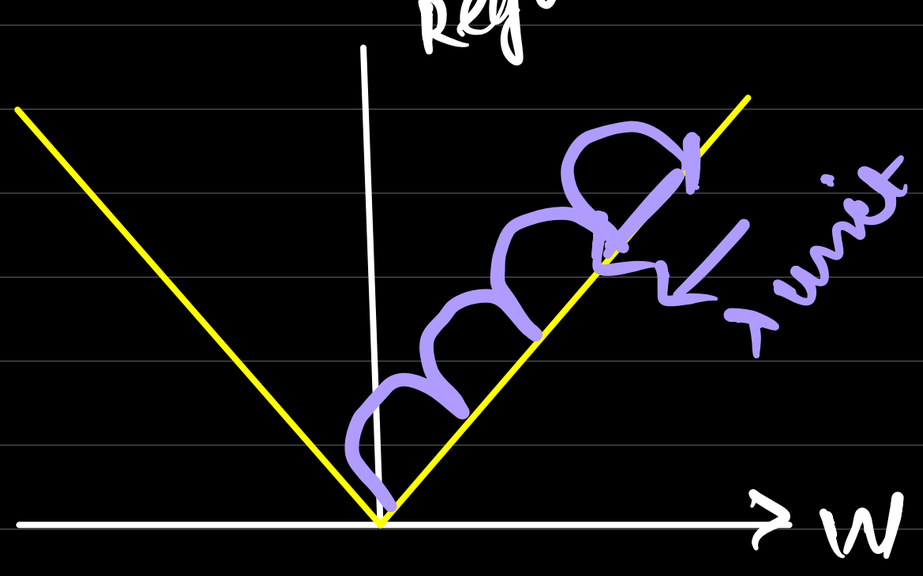
$$L_{NEW} = MSE + \lambda \text{ Regularisation } L$$

Regularisation Rate

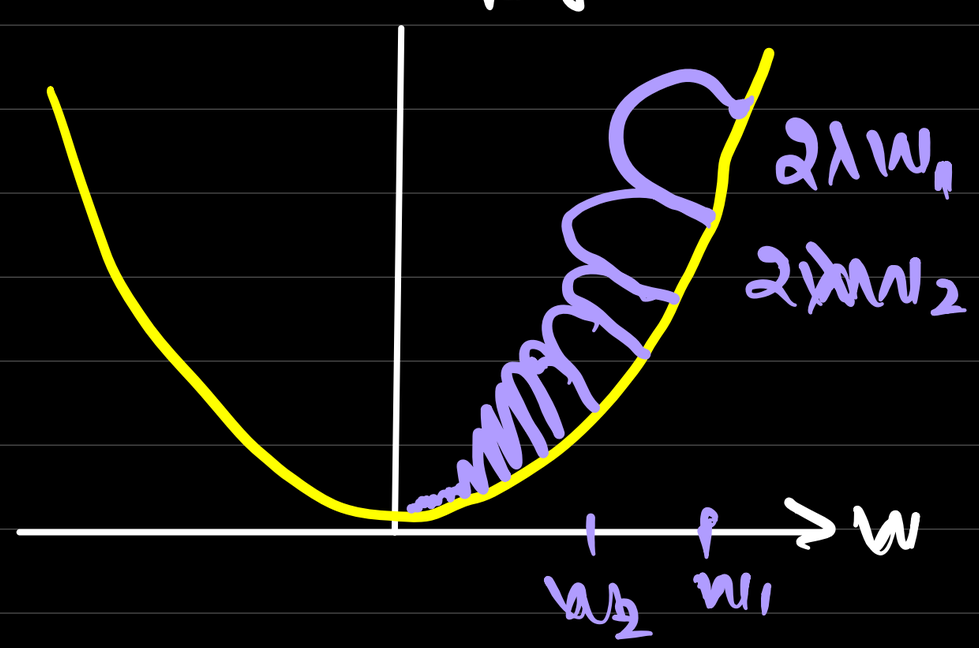
Remember

$$\text{Weight Update} = \text{Calculated } \frac{\partial L}{\partial w} = \frac{\partial \text{MSE}}{\partial w} + \lambda \frac{\partial \text{Reg } L}{\partial w}$$

L1 Regularization



L2 Regularization



$w > 0 \quad \frac{\partial L}{\partial w} = 1$

$w < 0 \quad \frac{\partial L}{\partial w} = -1$

$w = 0 \quad \frac{\partial L}{\partial w} = 0$

$\frac{\partial L}{\partial w} = -2w$

In case of L2, steps towards an update will become smaller & smaller.

$\Rightarrow$ L1 Regularisation leads to a sparse solution i.e., it has tendency to make less relevant

feature to go off by making
$w_j = 0$

$\Rightarrow$ L2 has property of making mag. of w's overall small but doesn't make anyone of them as zero.

$\Downarrow$

Feature Selection - To remove all

less imp features.

Use L1 Reg

Care about only performance

and don't want remove even less imp features

Use L2 Reg

Use both the L1 and L2

$$\text{Loss} = \text{MSE} + \lambda_1 \sum_{j=1}^{d} |w_j| + \lambda_2 \sum_{j=1}^{d} (w_j)^2$$

Elastic Net Regularisation.

Hyper-parameter Tuning

Parameters - $w_0, w_1, w_2, w_3 \ldots w_d$

Things which model learns during training

Hyper-parameters - learning rate (eta, α)
Regularization Rate
Degree of polynomial

Values set all set by DS training the model.
Not learnt from training, but from diff experiments

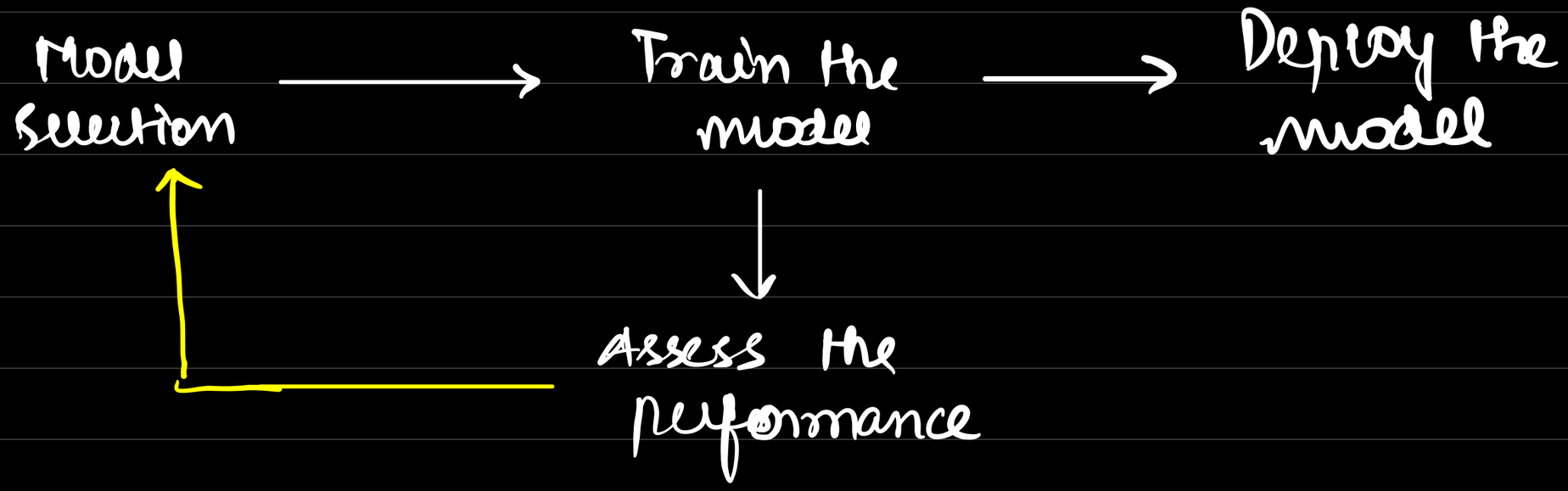
The process of selecting the right values of Hyperparameter - Hyperparameter Tuning

d = [1, 2, 3, 4, 5, 10, 20, 30]

for deg in d:

Train a model for degree = (deg)
Check the train & test performance

Select whichever has best test perf - use that deg.



Similar job for 'd', 'k', 'α'