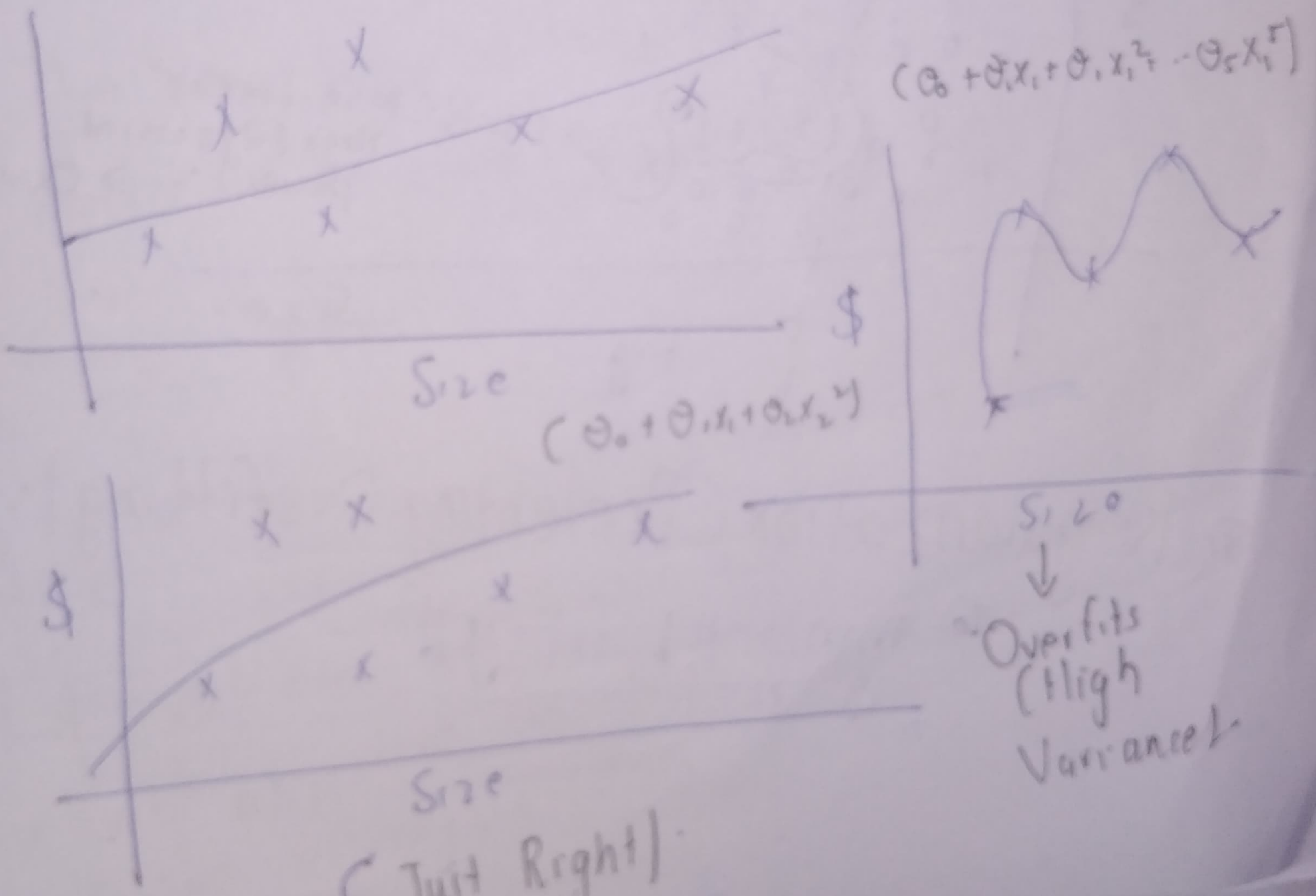
Lecture-8

⊛ Bias|Variance:-

$(\theta_0+\theta_1x_1)$

← Underfits (High Bias).



$(\theta_0+\theta_1x_1+\theta_1x_1^2 \cdot\cdot \theta_5x_1^5)$

$(\theta_0+\theta_1x_1+\theta_2x_2^2)$

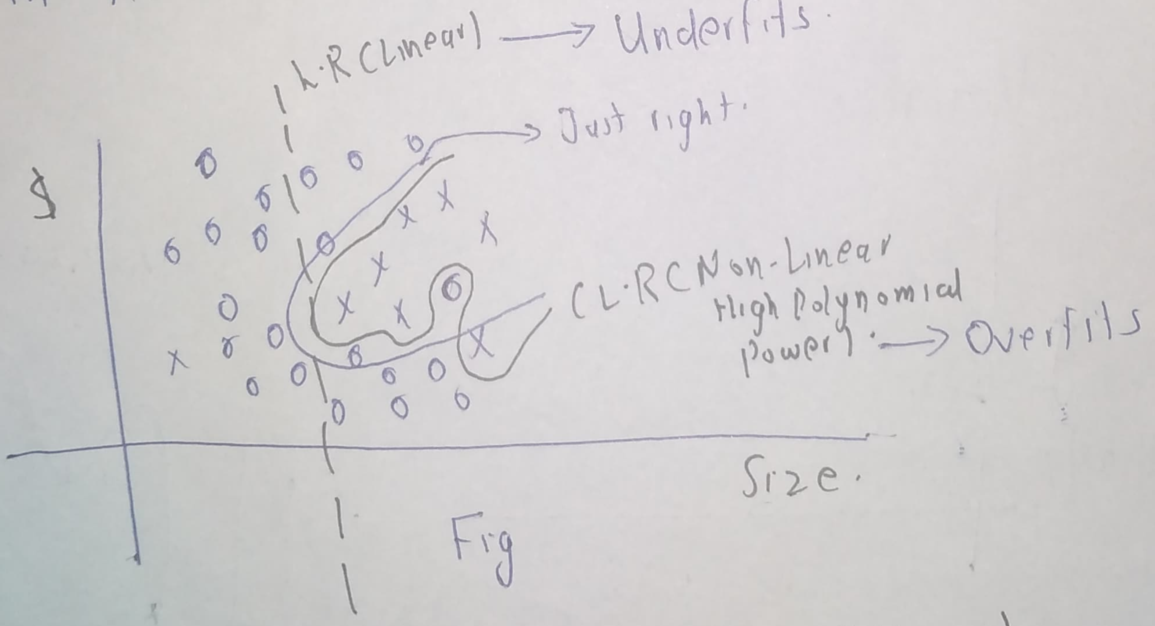
↓ Overfits (High Variance).

(Just Right)

(*) The term high bias comes from the fact that our model was ~~pr~~ Supposdely bias towards less compleaxity (linear) while actual data distribution was more complex (Quadratic). (High Bias).

(*) The term high ~~bias~~ Variance mean that everytime we fit our model on ~~stigth~~ slightly different data, our model would change to a large extent. Therefore our model fitting and predicting is highly varied given even a slight change in the distribution of data. (High Variance)

(*)

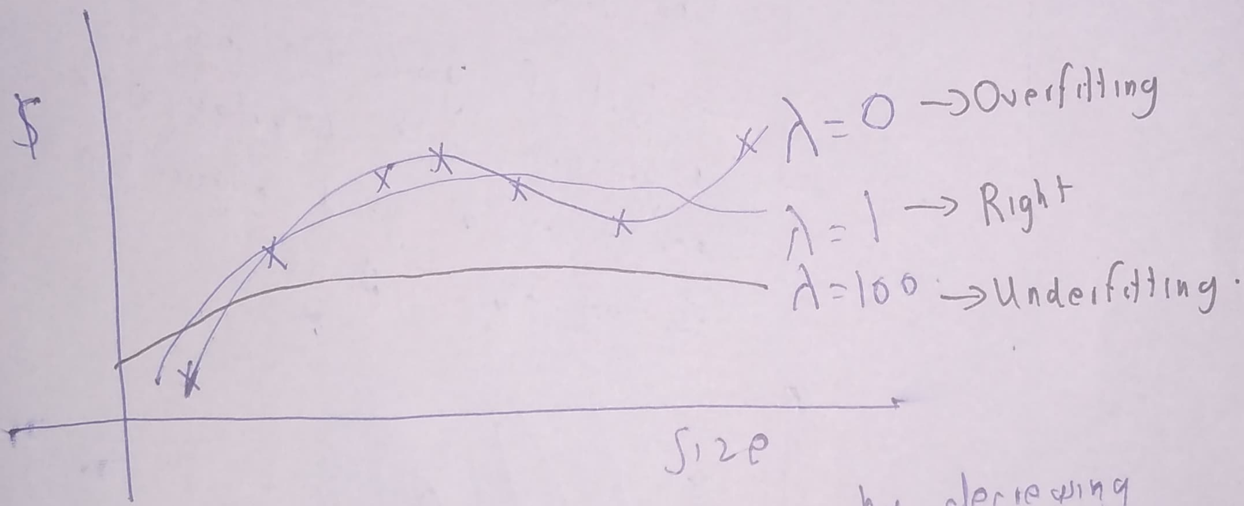

L.R (Linear) → Underfits.
→ Just right.
$
(L.R (Non-Linear High Polynomial power) → Overfits.
Size.

Fig

(*) Regularization (Preventing Overfitting):-

~~$(a)\ \min_{\theta} \frac{1}{2} \|h_\theta(x) - y\|^2$~~

$$\underbrace{\min_{\theta} \frac{1}{2}\sum_{i=1}^{m} \| y^{(i)} - \theta^{T} x^{(i)} \|^{2}}_{\text{Cost}} + \underbrace{\frac{\lambda}{2}\|\theta\|^{2}}_{\text{Regularization term.}}$$



↳ Over fitting preventing by decreasing the size of θ.

↳ Very large λ mean θ will be forced to be close to zero.

(*) ~~$\max_{\theta} \sum_{i=1}^{m}$ y to~~

(*) $\arg\max_{\theta} \sum_{i=1}^{m} \log P(y^{(i)} | x^{(i)}; \theta) - \lambda \|\theta\|^{2}$.

↳ $\boxed{\text{SVM}}$ does not overfit badly because have same $\min \|w\|^{2}$ objective function effective as regularization.

(*) Text classification

- $m = 100$

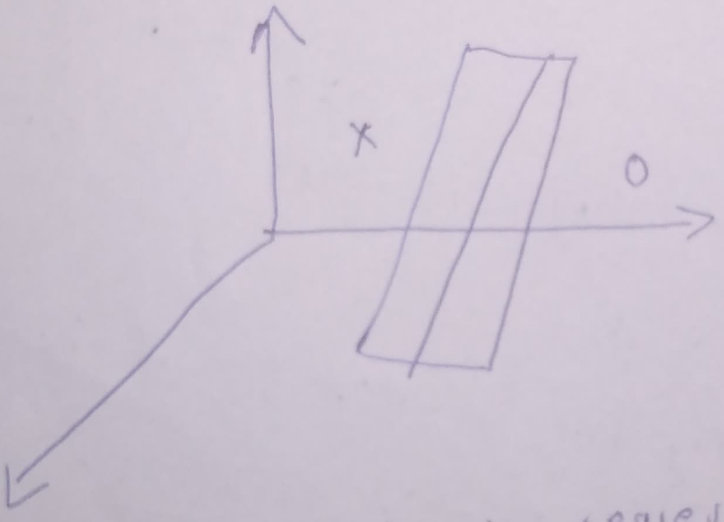
  $n = 10{,}000$

  $$X = \left[\begin{array}{c} 1 \\ 0 \\ \vdots \\ 1 \\ 0 \\ \vdots \end{array}\right] \begin{array}{l} a \\ adan \\ \vdots \\ \\ \\ \end{array}$$

→ if you fit logistic regression to such a data where $n >> m$ then it will overfit the data ~~because~~

→ But if you use logistic regression with regularization then it is a good algorithm for text classification.

(*)



⑦ More Example. mean logistic regression even without regularization won't overfit and would preform good provided $m > n$.

⑧ Features at different scales should be normalized on the same range so that these learning models trains faster and perform better.

→Regularization and θ prior.

⊛ $S = \{(x^{(i)}, y^{(i)})\}_{i=1}^{m}$

$$P(\theta \mid S) = \frac{P(S \mid \theta)\, P(\theta)}{P(S)}$$

⊛ $$\arg\max_{\theta} P(\theta \mid S) = \arg\max_{\theta} P(S \mid \theta) \cdot P(\theta)$$

$$= \arg\max_{\theta} \underbrace{\left(\prod_{i=1}^{m} P(y^{(i)} \mid x^{(i)}; \theta) \right)}_{\text{Generalized Linear model}} \cdot P(\theta).$$

$$P(\theta): \theta \sim \mathcal{N}(0, \tau^2 I)$$

$$P(\theta) = \frac{1}{\sqrt{2\pi}\,(\tau^2 I)^{1/2}} \exp\left(-\frac{1}{2}\theta^T (\tau^2 I)^{-1} \theta\right).$$

⊛ If this is the prior distribution for θ and you plug it in then it is equivulent to regularization.

⊛ Regularization is equivelent to assuming a prior θ Gaussian distribution.

~~Fre~~

Frequenist vs Bayesian Approach:

→ In Frequenist Approach we want to find the value of θ which makes ~~that~~ the data as likely as possible

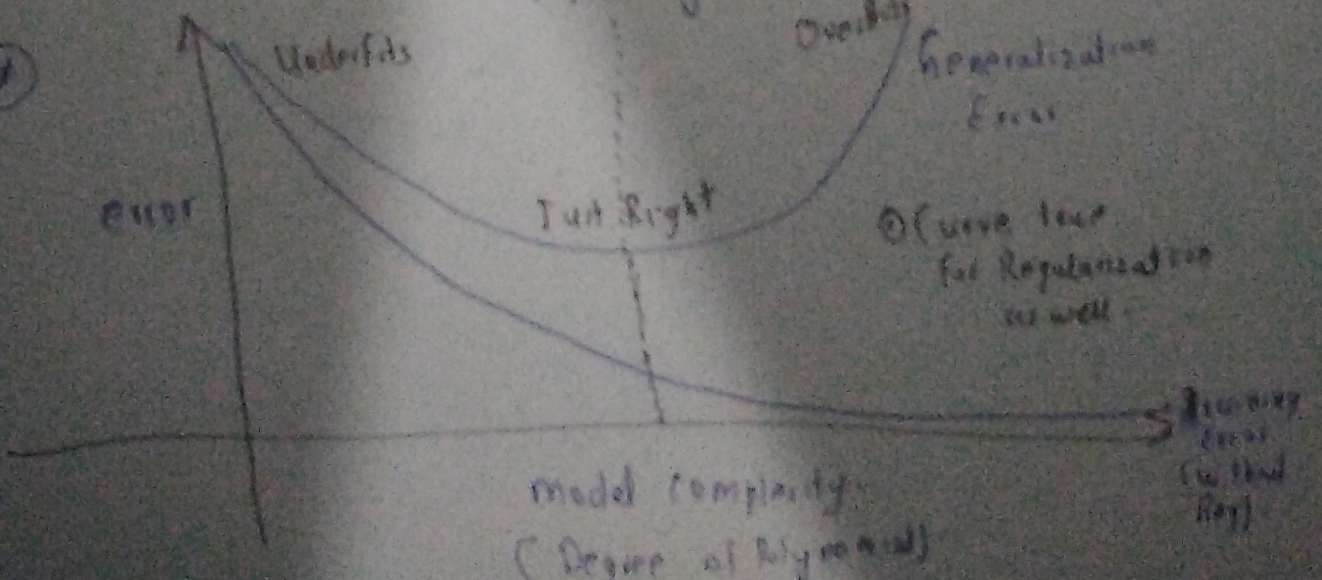
$$\boxed{\underset{\theta}{\mathrm{argmax}}\ P(S|\theta) — \mathrm{MLE}}$$ (Maximum Likelihood Estimate)

→ In Baysenia approach you already have some prior belief about θ in form of P(θ) (Gaussian prior)

↳ Our goal is to find

$$\text{value } \boxed{\underset{\theta}{\mathrm{argmax}}\ P(\theta|S)} \rightarrow \text{MAP (Maximum a Positerioi)}$$

↳ Adding Regularization



(Curve true for Regularization as well

(*) Traming / Dev / Test

→ Spliting the Dataset into different subset:

→ 10000 examples

E.g $\theta_0 + \theta_1 x$

$\theta_0 + \theta_1 x + \theta_2 x^2$

or choosing value λ

or τ

Different ← Hyperparameter needed to be choose

→ Splity your training data:

① S_{train} : $S_{dev(development)}$: S_{test}

(*) Train each model(option for different hyperparameters) on S_{train}.

↳ Get some hypothesis h.

(*) Measure error on S_{dev} for new developed hypothesis h.

(*) Pick h with lowest error on S_{dev}.

(*) If You want to check the unbiased performance of your model then evaluate on S_{test} since S_{dev} has alrady been optimized upon by your model.

(*) S_{test} gives the real performance of your model. Since S_{dev} is a biased estimate as model hyperparameter are adapted to perform on S_{dev}.

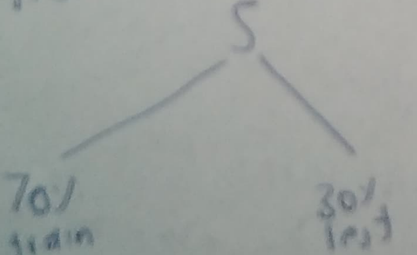
(*)

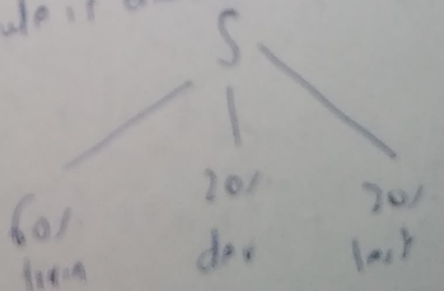
degree	S_{dev} Error	S_{test} Error
1	10	10
2	51	5
3	5.0	5.0
4	49 → Biased estimate	5.0 → Actual Performance.
5	47	7
6	10	10
	⋮	⋮

(*) Sometimes you can ignore S_{test} in real world product but in publishing paper you need to have S_{test}.

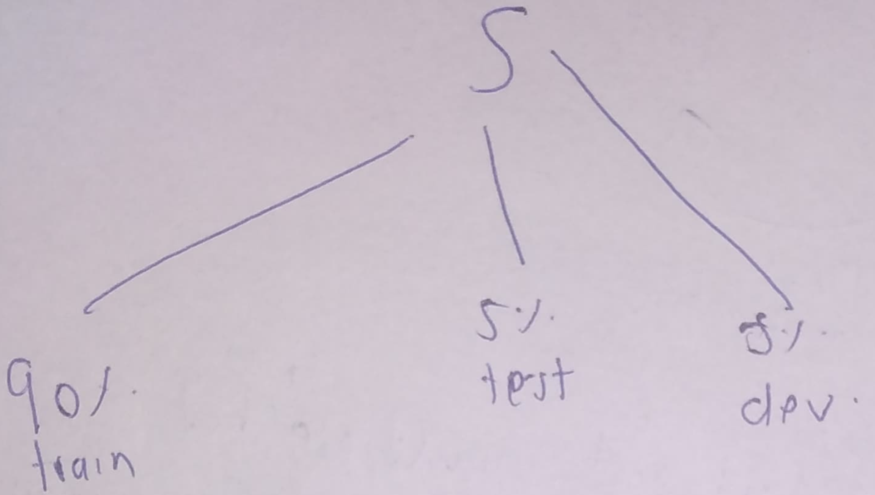
(*) How much data to each split!

→ Historical:- (Decent thumrule if dataset is small eg thousands examples).





⊗ Era of Big-data (10 million example):-



⊗

⊗ But, Sometimes you need a largest test dataset if you the perfoimance improvement between models is really small.

↳ ⊗ Choose test and dev big enough to make meaningful comparision between different models.

⊗ This Process of holding dev set from train set is called hold-out cross-validation (Simple hold-out cross validation).

↳ Development set = Cross validation set
(Dev)

⊛

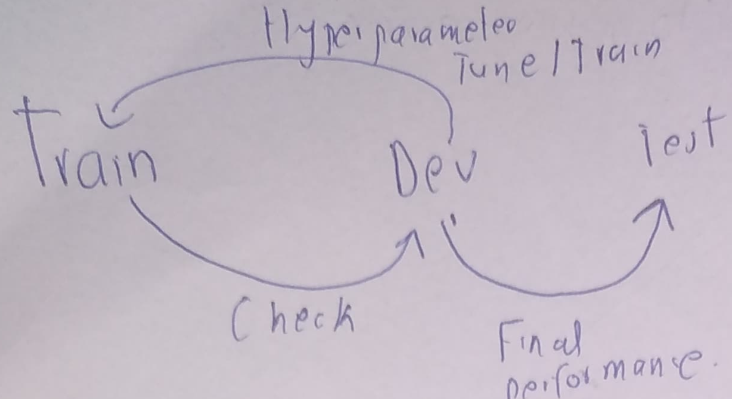


Fig.

→ San Evaluate on test set multiple time but don't make decision or tune hyper-parameter based on the performance of test set which is supposed to be an unbiased estimator benchmark

↳ Dev set evaluation is used for hyper-parameter tunning

⊛ Small datasets:- (K-Fold - CV-Cross-Validation).

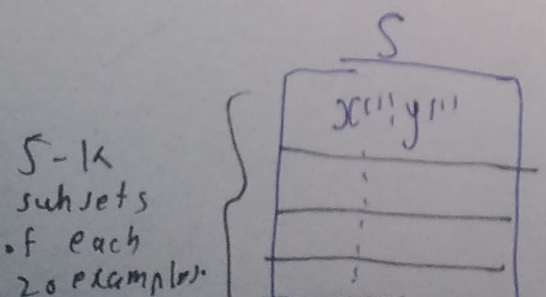
⊙ m=100 (Expensive Examples).

70% - S_{train}, 30% - S_{dev}.

⊙ This next procedure should be used only with small dataset:-

⊙ K-Fold CV (Cross-Validation

Divide/split data into five different subset



For i, ... k:-

Train on k-1 subsets

Test on remaing 1 piece.

→ You will cross-validation for different hyperparameter and calculate average of metric for the cross-validations and then see which hyper-parameter performed the best.

→ Final Optional Step:-

→ Refit the Model with chousen hyperparameter on Complete dataset.

→ This procedure make efficent use of data but is computationally very expensive.

⊗ Extreme version of ~~CV~~ K-Fold CV:- (For really small dataset):-

m=20 (20 example)

⑥ Leave - one - out CV:-

K=m → Divide your data into as many pieces as your training instances and leave only instance for evaluation at each iteration of validation.

(*) Feature Selection:-

- Small subset of features that are most useful for your task
- Most Important
- Depends on domain & context.

① Foward Search.

```
→ Start with $\mathcal{F} = \emptyset$
    Repeat {
        1) Try adding each feature $i$ to
           $\mathcal{F}$, which single feature addition
           most improve dev set performano
        2) Add that feature to $\mathcal{F}$
    }
```

Example

- $X_1 \cdots X_5$ (Five Features.

$\emptyset$ $\quad h(x) = \theta_0$ (No feature)

$$\left[\begin{array}{l} \emptyset_{+x_1} \\ \emptyset_{+x_2} \xrightarrow{\quad} h_{\mathcal{F}}(x) = \Theta + \Theta_2 X_2 \quad \mathcal{F} = \{x_2\}. \\ \vdots \\ \emptyset_{+x_5} \end{array}\right.$$

$$\left[\begin{array}{l} X_2 \;\; X_1 \rightarrow \Theta + \Theta_1 X_1 + \Theta_2 X_2 \quad \mathcal{F} = \{X_1, X_2\}. \\ X_2 \;\; X_3 \\ X_2 \;\; X_4 \\ X_2 \;\; X_5 \end{array}\right.$$

→ The above example shows that you ~~set~~ Start with empty array of features and then at ~~each~~ first iteration train five different model with only one feature respectively. See which model with one specific featuie perfoim the best and add that feature to your feature array

→ Then at second iteration you add an other feature one at a time to previously choosen feature and see which model perform best with these two feature (one → new, one → previously choosen)

↳ You continue until you have either added all features or performance jump by new features addition is minimal.

(*) Backward → Start with all features and remove features ~~to~~ one by one to see the performance degradation.

$f = (X_1, X_2, X_3, X_4, X_5)$.

```
Repeat {
    1) Remove Feature ith and train model with remaining features.
    2) Remove feature with most minimal decrease in performance when removed }
```

Example

$Z = (X_1 \ldots X_5)$ (Five Features).

$$\left[\begin{array}{l} X_1, X_2, X_3, X_4 \\ X_2, X_3, X_4, X_5 \\ X_3, X_1, X_2, X_5 \\ X_1\ X_2\ X_4\ X_5 \\ X_2\ X_3\ X_4\ X_5 \end{array}\right\} \rightarrow$$

Five different model and see whoose accuracy is still the greatest and remove that feature (Say feature 5).

$Z = (X_1, X_2\ X_3, X_4)$.

$$\left[\begin{array}{l} X_1, X_2, X_3 \\ X_2\ X_3\ X_4 \\ X_1\ X_2\ X_4 \\ X_2\ X_2\ X_4 \end{array}\right. \rightarrow$$

Repeat until one feature is left or performance decrease is same across all features