Classification & Regression

(classical ML)

List

KNN

Linear Regr

Logistic Regr

Naive Bayes

Bagging : RF

Boosting : GBDT; CatBoost
Adaboost
LightGBM

Stacking, Cascading

SVM → SVR, LR, Kernel

SMOTE

extra Trees

MLE, MAPE

Regularization : L1, L2, elastic Net

Calibration : Platt-scaling, Isotonic reg

RANSAC

bias-variance

Loss fns

Misc...

∀ each models:

- ✓ imbalanced data
- ✓ Multi-class
- ✓ Interpretability
- Prob. class labels
- C.I on regression
- Pros & cons
- Outliers

Pre-processing (standardization)

Post-processing (calibration)

- Good feature encoding
- bias-variance tradeoff
- Latency @ run-time
- Loss fn ...
- Data-drift ...

Obj
- recap
- applied
- Interviews

Linear-regr:

Loss: sq-loss (MSE) → MLE: $y_i \sim N(w^T x_i + b, \epsilon_i)$, $\epsilon_i \to N(0, \sigma^2)$

MAP: function of σ^2

bias-var: hyper-param

sq loss + λ · reg

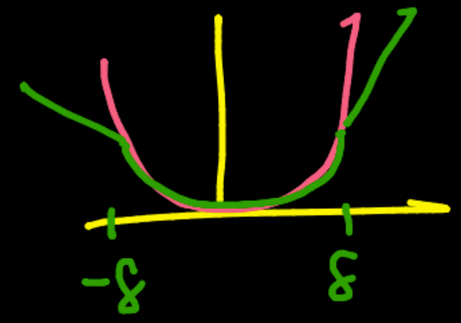
$\lambda \uparrow$: underfit

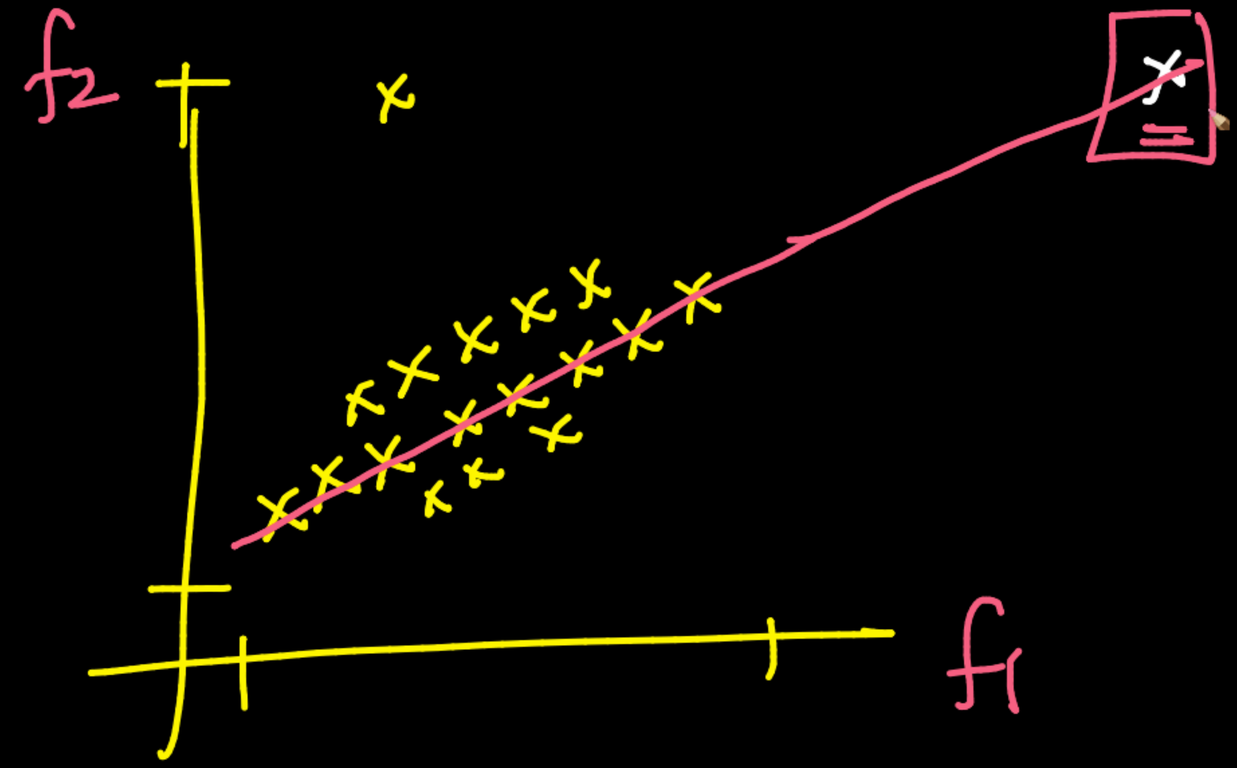
$\lambda \downarrow$: overfit

Outliers: impacted

fixes: RANSAC, MAE, Huber

$y_i - \hat{y}_i$




f2
f1

Decision Surface: π^d

↳ quadratic & higher order ply → Curve or Surface in d-dim

95% C.I on y_i → bootstrapping → disadv → Train time

→ Dist of errors

Stat. assumptions

$y_q \sim N(\underbrace{W^T x_q + b}_{\text{Model}}, \sigma^2)$

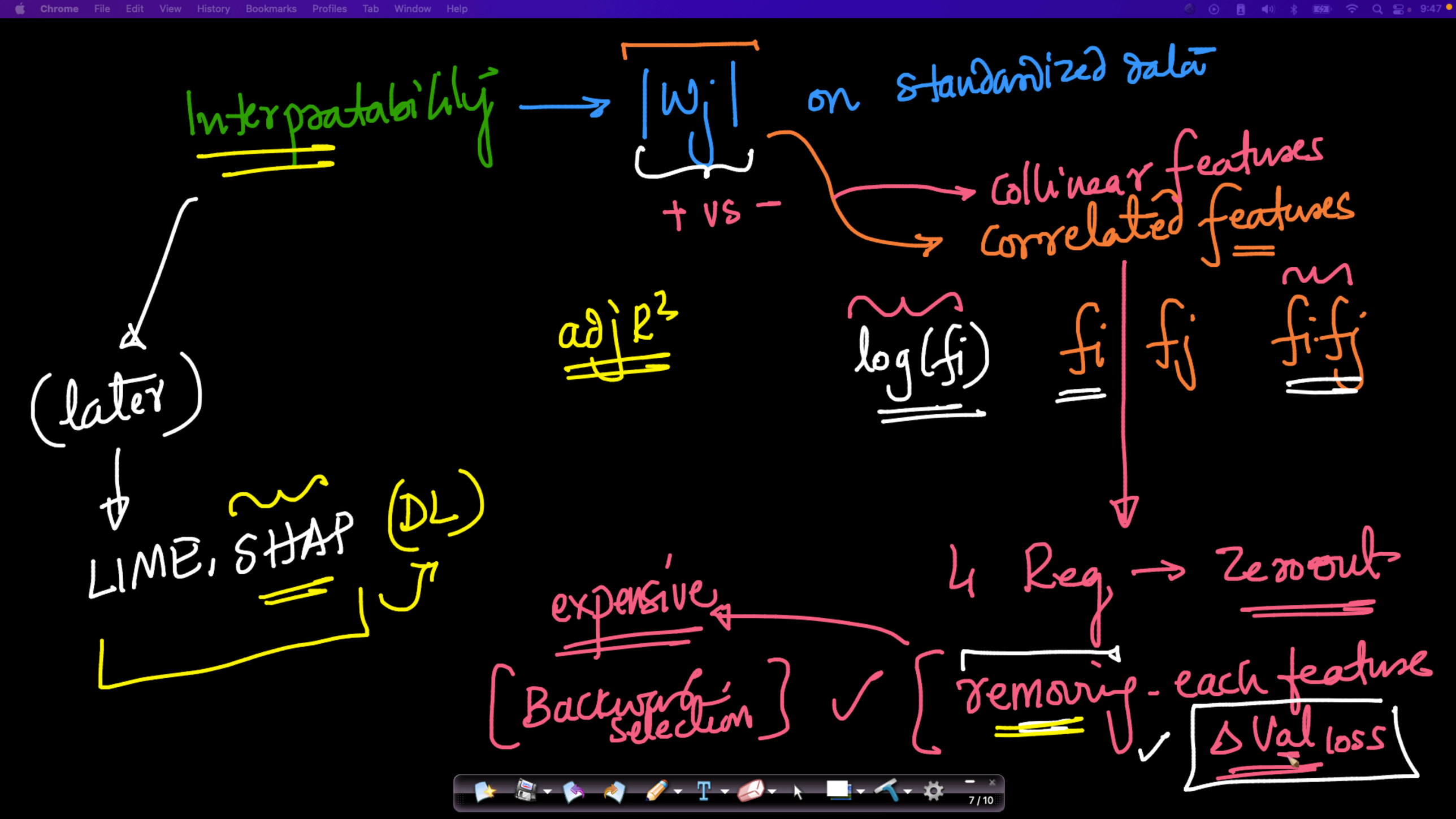
$N(0, \sigma^2)$

$\hat{\sigma}^2 = MSE_{train}$

$W^T x_q + b \pm 1.96\hat{\sigma}$ ✓ ✓

x_q → $M_1, M_2 \ldots M_{K=1000}$

[eval time is large]


Interpretability
|wj|
on standardized data
+ vs −
Collinear features
Correlated features
adj R²
log(fi)
fi
fj
fi·fj
(later)
LIME, SHAP (DL)
expensive
[Backward selection] ✓
4 Reg → Zero out
[removing - each feature
Δ Val loss]
7 / 10

Pre-processing

↳ std. data → interpretability of w_i's

→ optimization (SGD)

Post-processing : $\hat{y}_i$; check $\epsilon_i \sim$ Normal or not

↳ C.I which can be trusted...

VIF → identify x_i non-linear rel. (partially)

λ · L1-reg

↓ Val-loss ✓

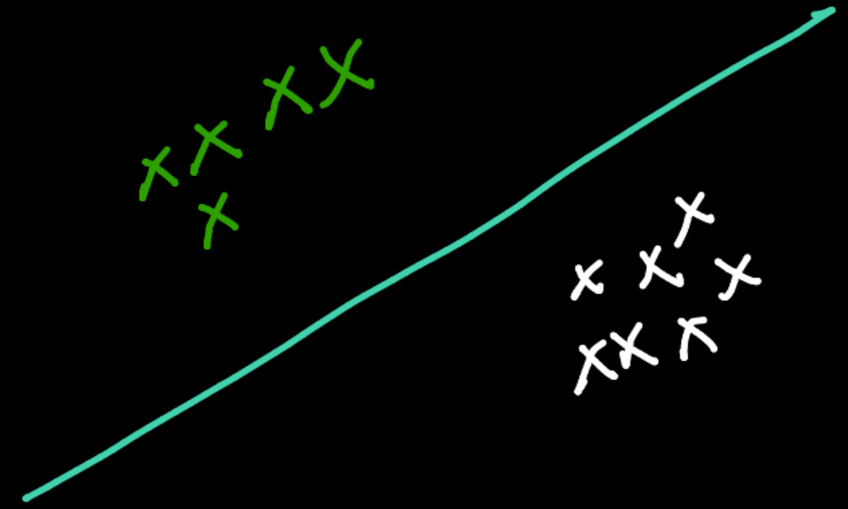
Good feature encoding

- cat-feat → target ; OHE → dim increasing → more "zoom" or flexibility to fit a TD → L1 reg to avoid overfitting
- Num-feat → std.

f_i : cat-feature

$C_1 C_2 \cdots C_{20}$ (categories)

Curse of dim



Kernel trick

Logistic reg:

Imbalanced → data

Logistic reg:

Imbalanced data → Upsampling; SMOTE; own Sampling; Class-weight (hyper-param)

Multi-class Classfn → OVR; softmax (DL...)

Loss-fn: → log-loss (BCE)

Interpretability → same as lr. reg

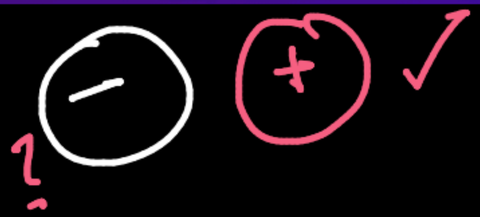
feat. encoding → " " "

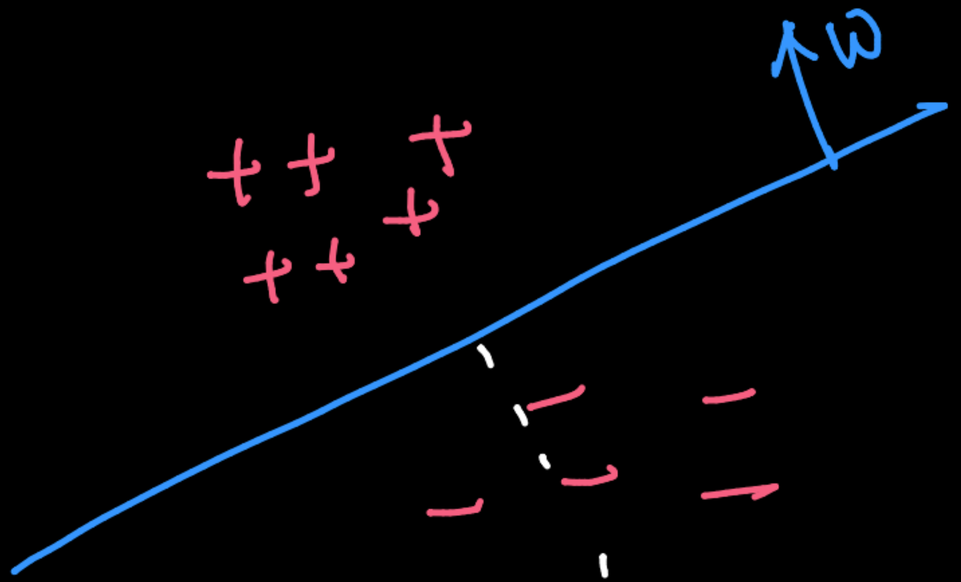
bias-var → " " "

Outliers → RANSAC; Change-loss fn (research..)

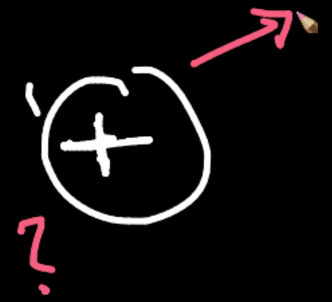
{ log-loss: $y_i \log p_i + (1-y_i) \log (1-p_i)$

↳ $\hat{y}_i$





Sigmoid



Prob. class label → Yes

$\hat{y}_i = P(y_i = 1 \mid x_i, w)$

→ failure: highly class imbalance

$P(y=1) = 0.001$

$\hat{y}_i$ are not calibrated

(*) Always plot Calib-plot

Isotonic regr

Post-processing → Confusion-Matrix (*)

Test-time: $O(d')$

↳ # non-zero features

- cons of lr-models
  - ↳ cannot automatically discover non-linear relationships

Lr. SUM ✓

→ Very similar to linear models

→ [Hinge-loss + L2-reg] Primal

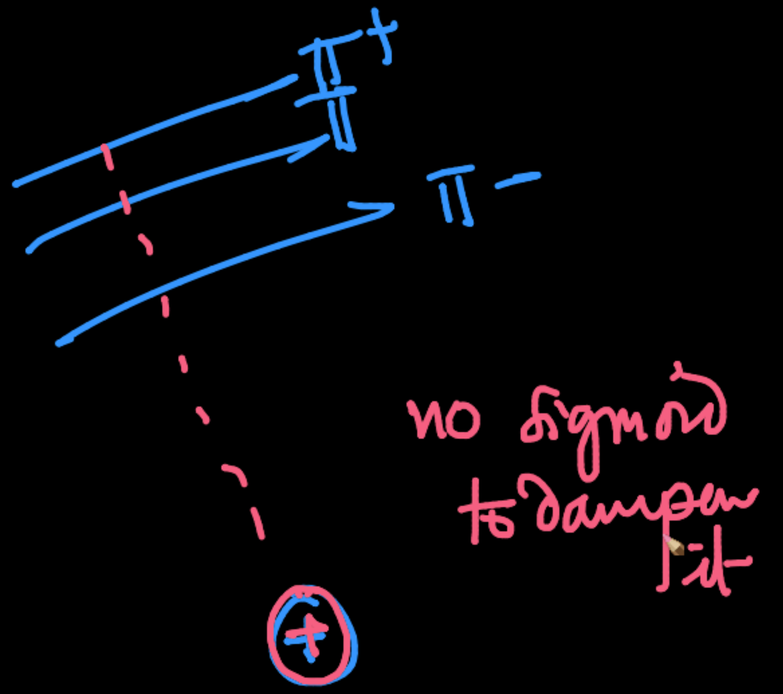
→ similar perf as linear models

→ Interpretability → w_j's (solving in Primal)

→ Prob. class. labels
↳ $P(y_i = 1 \mid w, x_i)$

→ feat. enc
↳ same as linear-model

→ outliers → " " "



multi-class → OVR

bias-var → "C" = 1/λ

Suggestion

↓

- think of all the cases .... } ✓
- whatsapp ... (weekend)

Minimizing **(2)** can be rewritten as a constrained optimization problem with a differentiable objective function in the following way.

For each $i \in \{1, \ldots, n\}$ we introduce a variable $\zeta_i = \max\left(0, 1 - y_i(\mathbf{w}^\mathsf{T}\mathbf{x}_i - b)\right)$. Note that ζ_i is the smallest nonnegative number satisfying $y_i(\mathbf{w}^\mathsf{T}\mathbf{x}_i - b) \geq 1 - \zeta_i$.

Thus we can rewrite the optimization problem as follows

$$\begin{aligned} &\text{minimize } \frac{1}{n}\sum_{i=1}^{n}\zeta_i + \lambda\|\mathbf{w}\|^2 \\ &\text{subject to } y_i\left(\mathbf{w}^\mathsf{T}\mathbf{x}_i - b\right) \geq 1 - \zeta_i \text{ and } \zeta_i \geq 0, \text{ for all } i. \end{aligned}$$

This is called the *primal* problem.

Dual [edit]

By solving for the Lagrangian dual of the above problem, one obtains the simplified problem

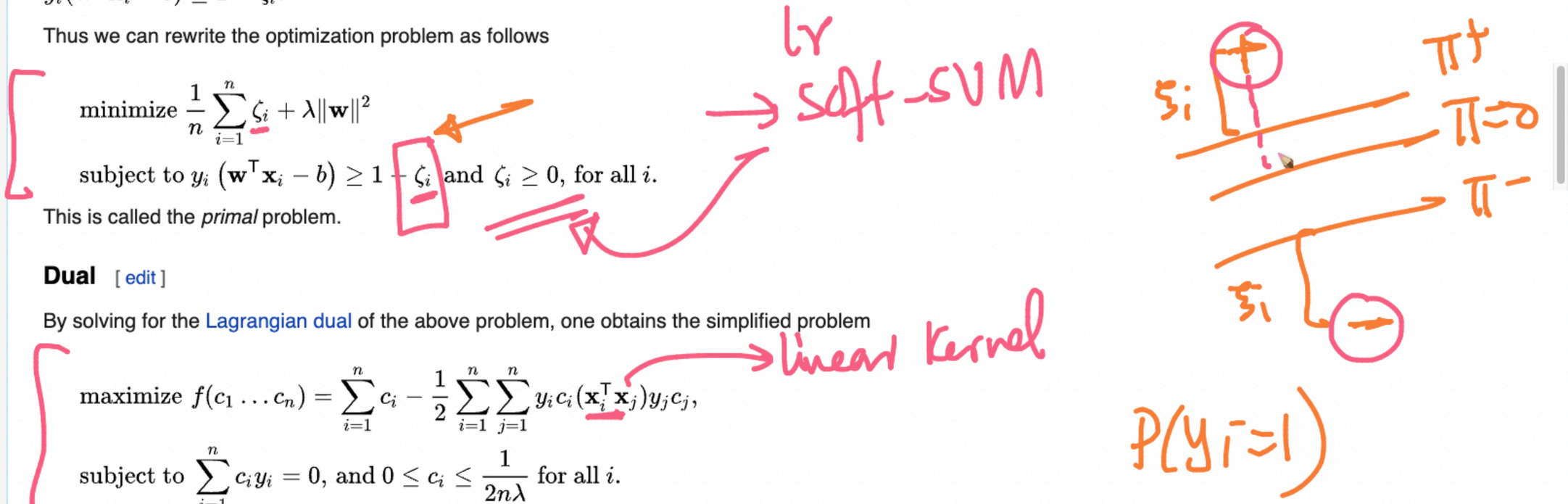
$$\begin{aligned} &\text{maximize } f(c_1 \ldots c_n) = \sum_{i=1}^{n} c_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} y_i c_i (\mathbf{x}_i^\mathsf{T}\mathbf{x}_j) y_j c_j, \\ &\text{subject to } \sum_{i=1}^{n} c_i y_i = 0, \text{ and } 0 \leq c_i \leq \frac{1}{2n\lambda} \text{ for all } i. \end{aligned}$$

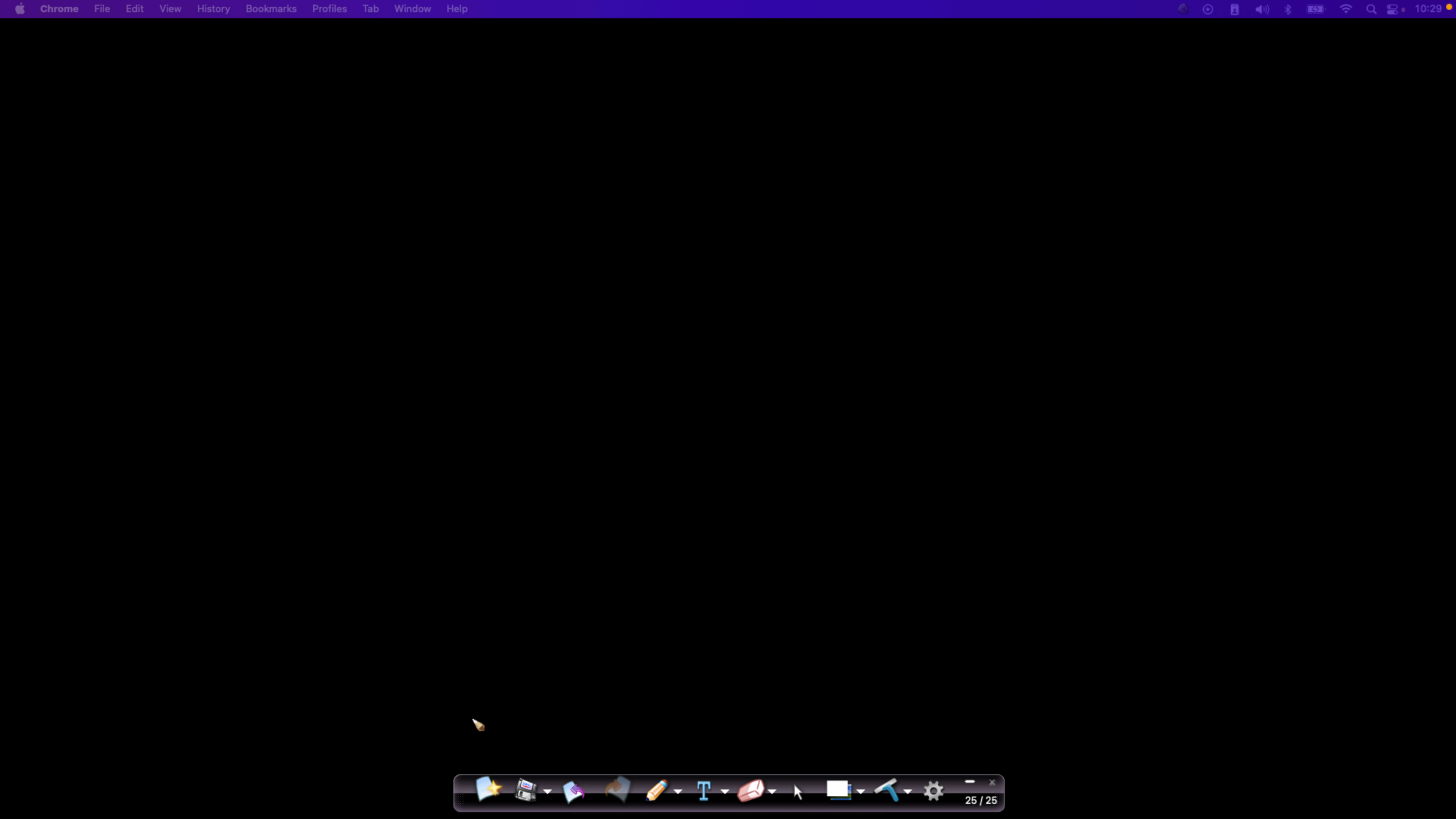
This is called the *dual* problem. Since the dual maximization problem is a quadratic function of the c_i subject to linear constraints, it is efficiently solvable by quadratic programming algorithms.

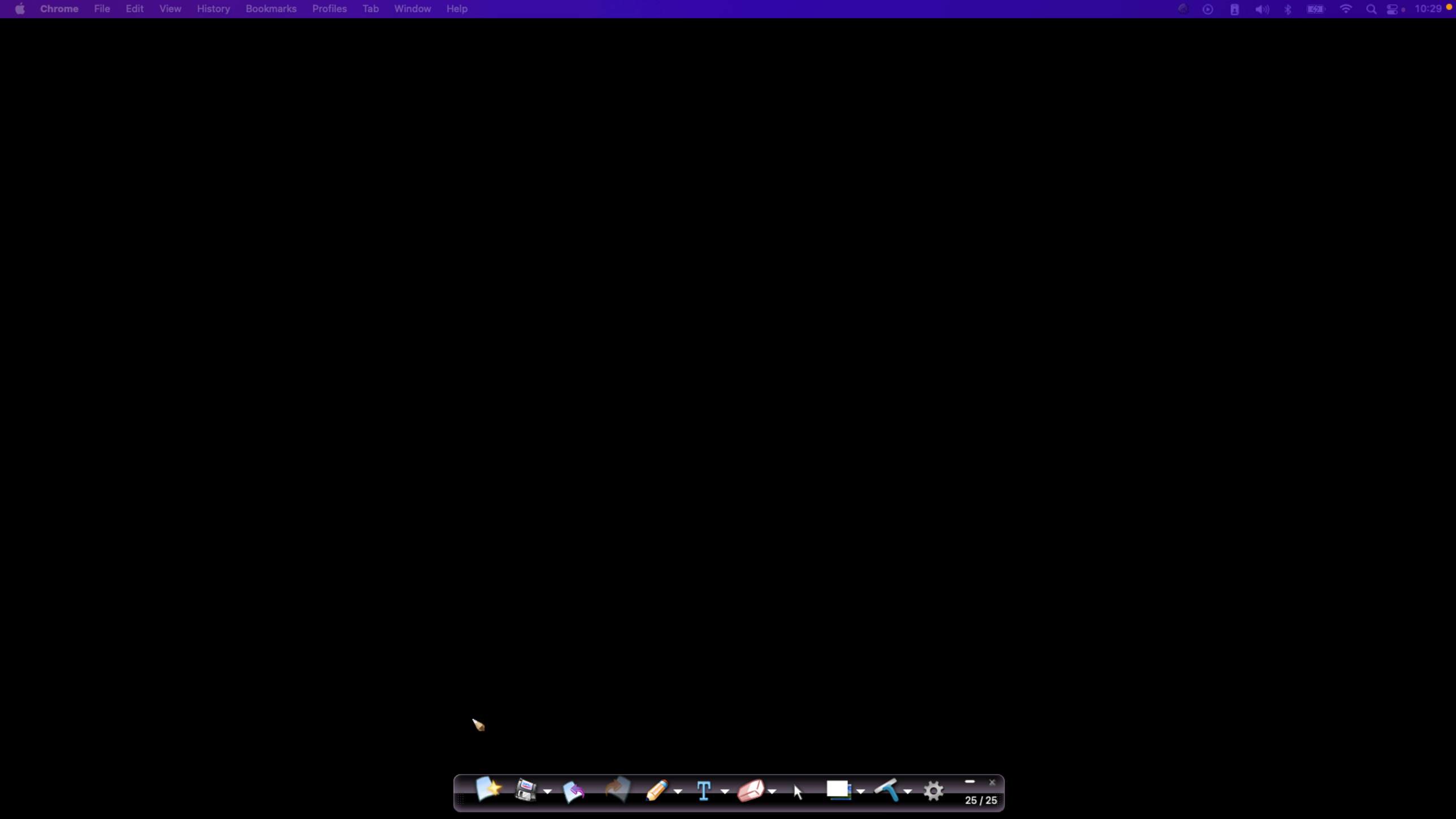
Here, the variables c_i are defined such that

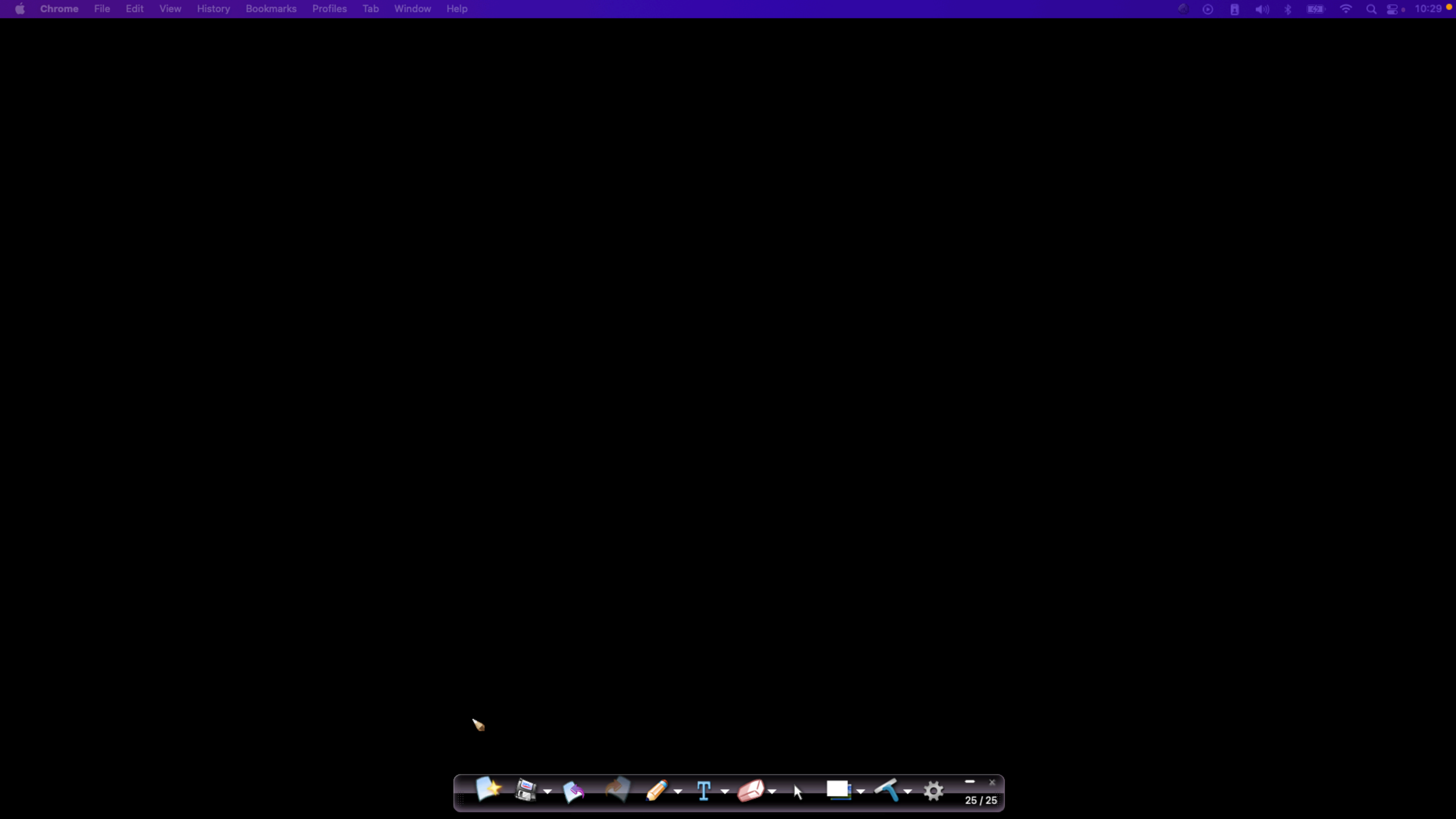
$$\mathbf{w} = \sum_{i=1}^{n} c_i y_i \mathbf{x}_i.$$

Moreover, $c_i = 0$ exactly when [illegible] on the margin's boundary. It follows that $\mathbf{w}$ can be written as a linear combination [illegible]




Chrome File Edit View History Bookmarks Profiles Tab Window Help
10:29
25 / 25

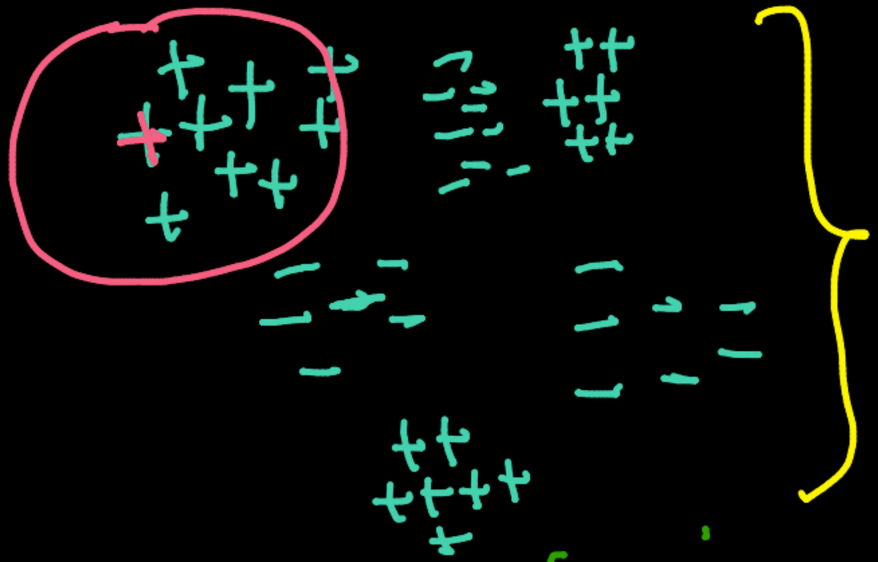

Chrome File Edit View History Bookmarks Profiles Tab Window Help
10:29
25 / 25


Chrome File Edit View History Bookmarks Profiles Tab Window Help
10:29
25 / 25

Kernel-SVM

Disadv:

- no Interpretability
- more time-complx.... → #SVs
- Custom-domain specific are needed for good perf...



→ Generic RBF-SUM

(σ)

{ σ ↑ k ↑ in KNN

same as the distribution of the length-squared of a random point in the *d*-cube. By the law of large numbers, this distribution concentrates itself in a narrow band around *d* times the standard deviation squared (σ^2) of the original derivation. This illuminates the chi-squared distribution and also illustrates that most of the volume of the *d*-cube concentrates near the boundary of a sphere of radius $\sigma\sqrt{d}$.

A further development of this phenomenon is as follows. Any fixed distribution on the real numbers induces a product distribution on points in $\mathbb{R}^d$. For any fixed *n*, it turns out that the difference between the minimum and the maximum distance between a random reference point *Q* and a list of *n* random data points $P_1,\ldots,P_n$ become indiscernible compared to the minimum distance:[12]

$$\lim_{d\to\infty} E\left(\frac{\mathrm{dist}_{\max}(d) - \mathrm{dist}_{\min}(d)}{\mathrm{dist}_{\min}(d)}\right) \to 0.$$

This is often cited as distance functions losing their usefulness (for the nearest-neighbor criterion in feature-comparison algorithms, for example) in high dimensions. However, recent research has shown this to only hold in the artificial scenario when the one-dimensional distributions $\mathbb{R}$ are independent and identically distributed.[13] When attributes are correlated, data can become easier and provide higher distance contrast and the signal-to-noise ratio was found to play an important role, thus feature selection should be used.[13]

More recently, it has been suggested that there may be a conceptual flaw in the argument that contrast-loss creates a curse in high dimensions. Machine learning can be understood as the problem of assigning instances to their respective generative process of origin, with class labels acting as symbolic representations of individual generative processes. The curse's derivation assumes all instances are independent, identical outcomes of a single high dimensional generative process. If there is only one generative process, there would exist only one (naturally occurring) class and machine learning would be conceptually ill-defined in both high and low dimensions. Thus, the traditional argument that contrast-loss creates a curse, may be fundamentally inappropriate. In addition, it has been shown that when the generative model is modified to accommodate multiple generative processes, contrast-loss can morph from a curse to a blessing, as it ensures that the nearest-neighbor of an instance is almost-surely its most closely related instance. From this perspective, contrast-loss makes high dimensional distances especially meaningful and not especially non-meaningful as is often argued.[14]

Nearest neighbor search [edit]

The effect complicates nearest neighbor search in high dimensional space. It is not possible to quickly reject candidates by using the difference in one coordinate as a lower bound for a distance based on all the dimensions.[15][16]

However, it has recently been observed that the mere number of dimensions does not necessarily result in difficulties,[17] since *relevant* additional dimensions can also increase the contrast. In addition, Irrelevant ("noise") dimensions, however, reduce the contrast in the manner described above. In time series analysis, where the data are inherently high-dimensional, distance functions also work reliably as long as the

same as the distribution of the length-squared of a random point in the *d*-cube. By the law of large numbers, this distribution concentrates itself in a narrow band around *d* times the standard deviation squared (σ^2) of the original derivation. This illuminates the chi-squared distribution and also illustrates that most of the volume of the *d*-cube concentrates near the boundary of a sphere of radius $\sigma\sqrt{d}$.

A further development of this phenomenon is as follows. Any fixed distribution on the real numbers induces a product distribution on points in $\mathbb{R}^d$. For any fixed *n*, it turns out that the difference between the minimum and the maximum distance between a random reference point *Q* and a list of *n* random data points $P_1,\ldots,P_n$ become indiscernible compared to the minimum distance:[12]

$$\lim_{d\to\infty} E\left(\frac{\mathrm{dist}_{\max}(d) - \mathrm{dist}_{\min}(d)}{\mathrm{dist}_{\min}(d)}\right) \to 0.$$

This is often cited as distance functions losing their usefulness (for the nearest-neighbor criterion in feature-comparison algorithms, for example) in high dimensions. However, recent research has shown this to only hold in the artificial scenario when the one-dimensional distributions $\mathbb{R}$ are independent and identically distributed.[13] When attributes are correlated, data can become easier and provide higher distance contrast and the signal-to-noise ratio was found to play an important role, thus feature selection should be used.[13]

More recently, it has been suggested that there may be a conceptual flaw in the argument that contrast-loss creates a curse in high dimensions. Machine learning can be understood as the problem of assigning instances to their respective generative process of origin, with class labels acting as symbolic representations of individual generative processes. The curse's derivation assumes all instances are independent, identical outcomes of a single high dimensional generative process. If there is only one generative process, there would exist only one (naturally occurring) class and machine learning would be conceptually ill-defined in both high and low dimensions. Thus, the traditional argument that contrast-loss creates a curse, may be fundamentally inappropriate. In addition, it has been shown that when the generative model is modified to accommodate multiple generative processes, contrast-loss can morph from a curse to a blessing, as it ensures that the nearest-neighbor of an instance is almost-surely its most closely related instance. From this perspective, contrast-loss makes high dimensional distances especially meaningful and not especially non-meaningful as is often argued.[14]

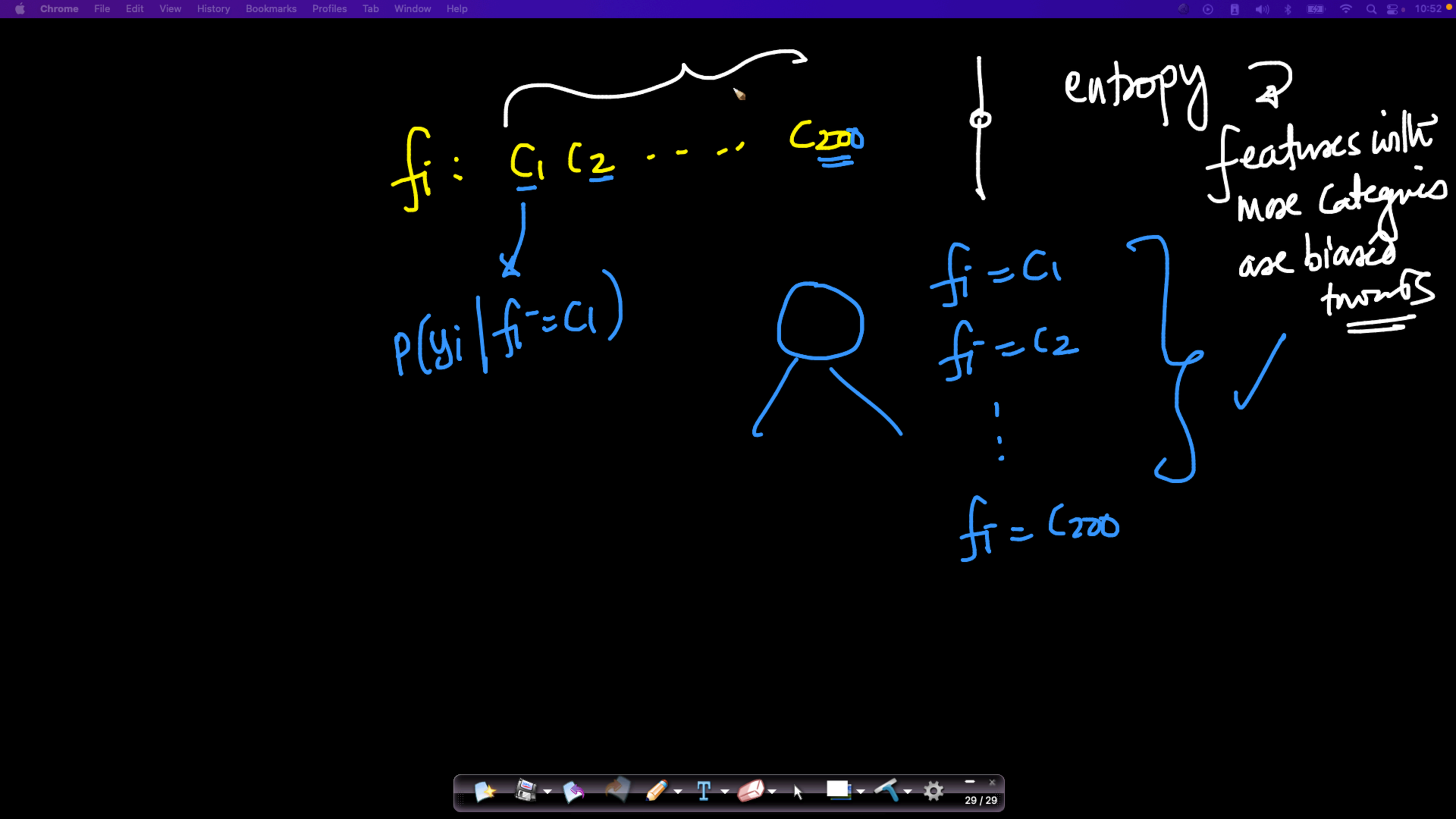
Nearest neighbor search [edit]

The effect complicates nearest neighbor search in high dimensional space. It is not possible to quickly reject candidates by using the difference in one coordinate as a lower bound for a distance based on all the dimensions.[15][16]

However, it has recently been observed that the mere number of dimensions does not necessarily result in difficulties,[17] since *relevant* additional dimensions can also increase the contrast. In addition, irrelevant ("noise") dimensions, however, reduce the contrast in the manner described above. In time series analysis, where the data are inherently high-dimensional, distance functions also work reliably as long as the

DT + RF + GBDT (Tree based)

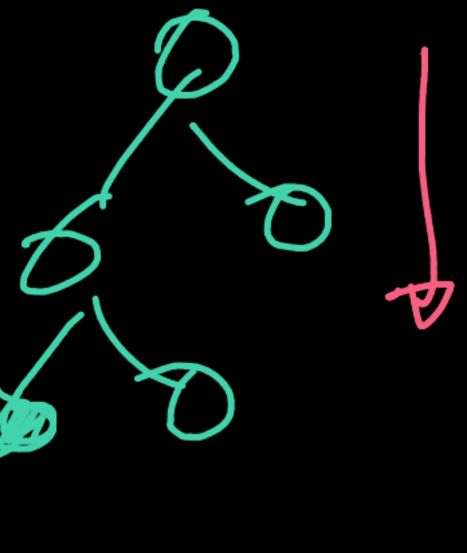
1. Pre-processing : No-standardization needed ...
2. Regression using Trees: → MSE
3. Feature-Imp: → weighted/normalized Info-gain
4. outliers: → not-much
5. Feature-encoding: → Cat-features → Target ; OHE ✗
   → Num-feat → as-is
   → bins ...


fi : C1 C2 . . . . C200
P(yi | fi = C1)
entropy ?
features with more categories are biased towards
fi = C1
fi = C2
fi = C200

C.I on x_q ⟶ bootstrapping

Prob. class. labels ⟶ Impurity in each leaf-node

(tricky)



x_q

$P(y_q = 1) = \frac{5}{6}$

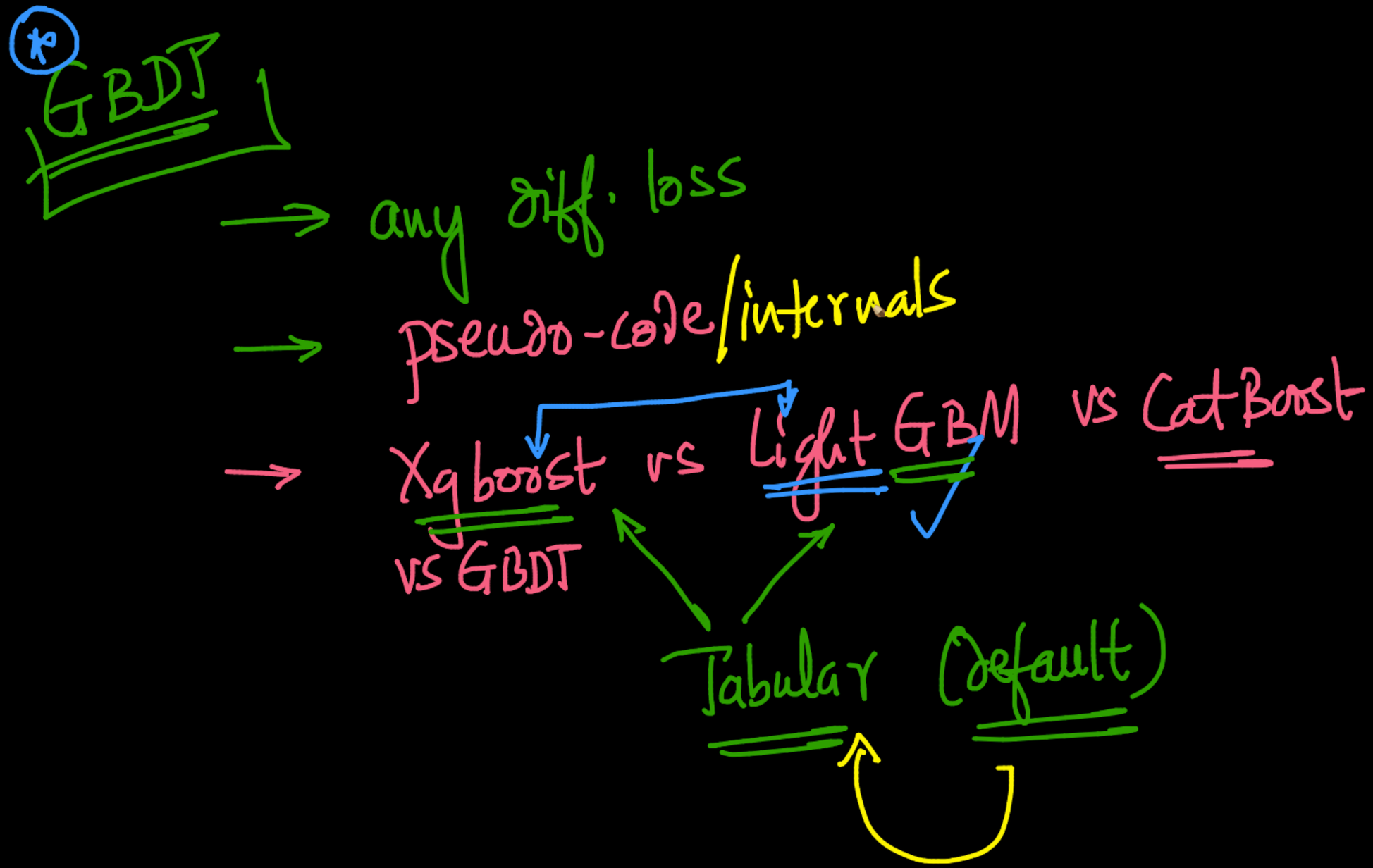
{5: +
1: -

Tr

Naïve Bayes:-

→ laplace - smoothing (bias - var)

→ assumptions: Conditional indep (prev. classes)



to find some function $\hat{F}(x)$ that best approximates the output variable from the values of input variables. This is formalized by introducing some loss function $L(y, F(x))$ and minimizing it in expectation:

$$\hat{F} = \arg\min_{F} \mathbb{E}_{x,y}[L(y, F(x))].$$

The gradient boosting method assumes a real-valued y. It seeks an approximation $\hat{F}(x)$ in the form of a weighted sum of M functions $h_m(x)$ from some class $\mathcal{H}$, called base (or weak) learners:

$$\hat{F}(x) = \sum_{m=1}^{M} \gamma_m h_m(x) + \text{const}.$$

We are usually given a training set $\{(x_1, y_1), \ldots, (x_n, y_n)\}$ of known sample values of x and corresponding values of y. In accordance with the empirical risk minimization principle, the method tries to find an approximation $\hat{F}(x)$ that minimizes the average value of the loss function on the training set, i.e., minimizes the empirical risk. It does so by starting with a model, consisting of a constant function $F_0(x)$, and incrementally expands it in a greedy fashion:

$$F_0(x) = \arg\min_{\gamma} \sum_{i=1}^{n} L(y_i, \gamma),$$

$$F_m(x) = F_{m-1}(x) + \arg\min_{h_m \in \mathcal{H}} \left[\sum_{i=1}^{n} L(y_i, F_{m-1}(x_i) + h_m(x_i)) \right],$$

for $m \geq 1$, where $h_m \in \mathcal{H}$ is a base learner function.

Unfortunately, choosing the best function h_m at each step for an arbitrary loss function L is a computationally infeasible optimization problem in general. Therefore, we restrict our approach to a simplified version of the problem.

The idea is to apply a steepest descent step to this minimization problem (functional gradient descent).

The basic idea behind the steepest descent is to find a local minimum of the loss function by iterating on $F_{m-1}(x)$. In fact, the local maximum-descent direction of the loss function is the negative gradient.[10]

Hence, moving a small amount γ such that the linear approximation remains valid:

$$F_m(x) = F_{m-1}(x) - \gamma \sum_{i=1}^{n}$$

$$f_m = \arg\min_{\gamma} \sum_{i=1} L(y_i, F_{m-1}(x_i)) \quad \gamma \nabla F_{m-1} L(y_i, F_{m-1}(x_i))).$$

In the discrete case however, i.e. when the set $\mathcal{H}$ is finite, we choose the candidate function h closest to the gradient of L for which the coefficient γ may then be calculated with the aid of line search on the above equations. Note that this approach is a heuristic and therefore doesn't yield an exact solution to the given problem, but rather an approximation. In pseudocode, the generic gradient boosting method is:[5][2]

Input: training set $\{(x_i, y_i)\}_{i=1}^{n}$, a differentiable loss function $L(y, F(x))$, number of iterations M.

Algorithm:

1. Initialize model with a constant value:

$$F_0(x) = \arg\min_{\gamma} \sum_{i=1}^{n} L(y_i, \gamma).$$

2. For m = 1 to M:
    1. Compute so-called *pseudo-residuals*:

$$r_{im} = -\left[\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)}\right]_{F(x)=F_{m-1}(x)} \quad \text{for } i = 1, \ldots, n.$$

    2. Fit a base learner (or weak learner, e.g. tree) closed under scaling $h_m(x)$ to pseudo-residuals, i.e. train it using the training set $\{(x_i, r_{im})\}_{i=1}^{n}$.
    3. Compute multiplier γ_m by solving the following one-dimensional optimization problem:

$$\gamma_m = \arg\min_{\gamma} \sum_{i=1}^{n} L\left(y_i, F_{m-1}(x_i) + \gamma h_m(x_i)\right).$$

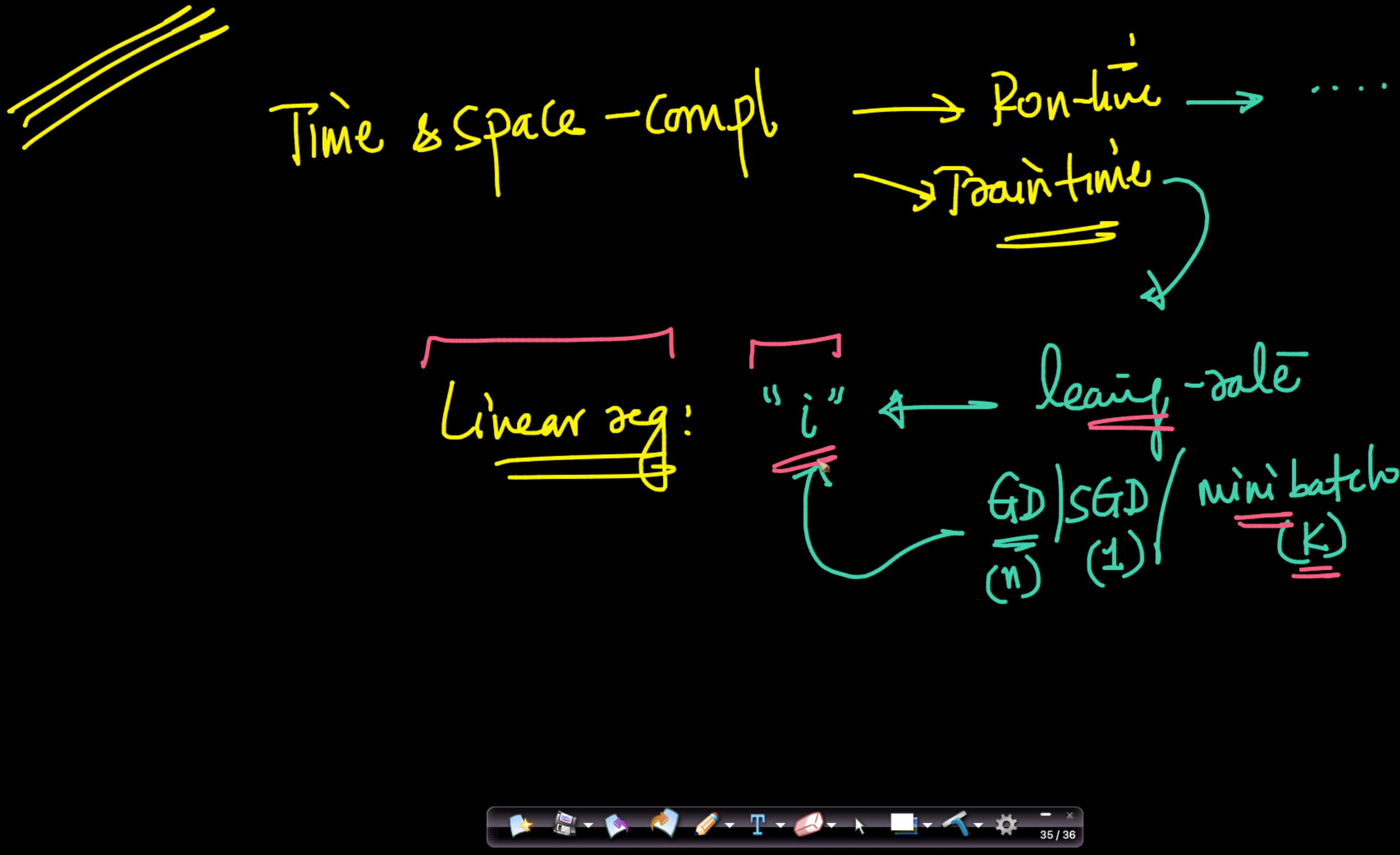
    4. Update the model:

$$F_m(x) = F_{m-1}(x) + \gamma_m h_m(x).$$

3. Output $F_M(x)$.

Gradient tree boosting [edit]

Gradient boosting is typically used with decision trees (especially CART trees) of a fixed size as base learners. For this special case, Friedman proposes a modification to gradient boosting method wh



Feat. Imp in Linear Reg

$d = 50$

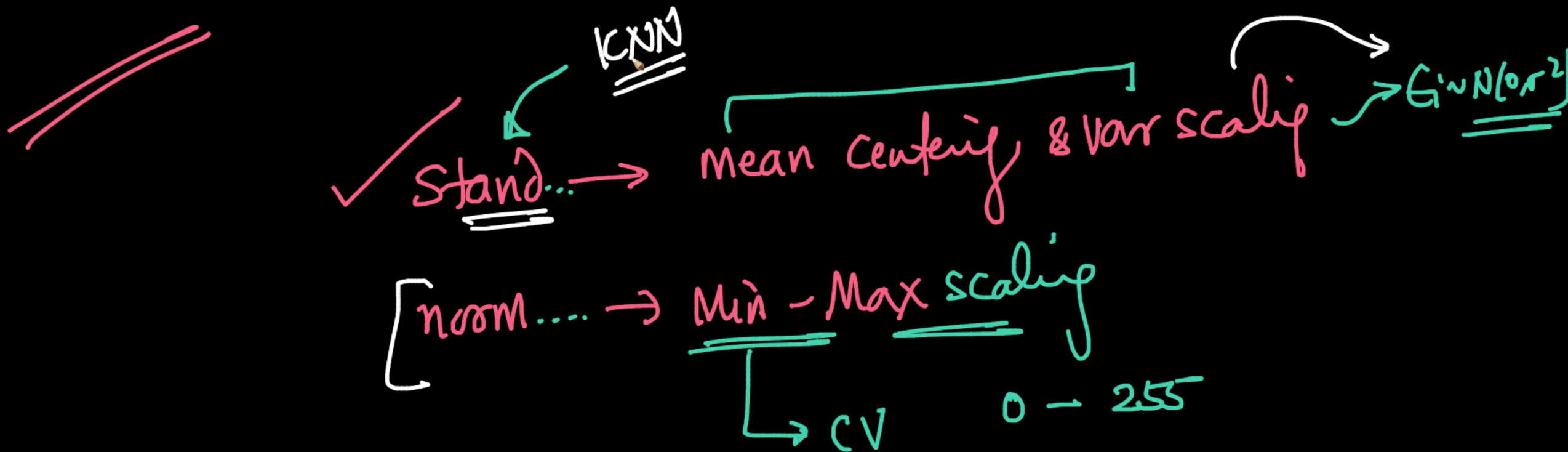
$\ldots f_{10} \ldots f_{15} \ldots f_{20}$

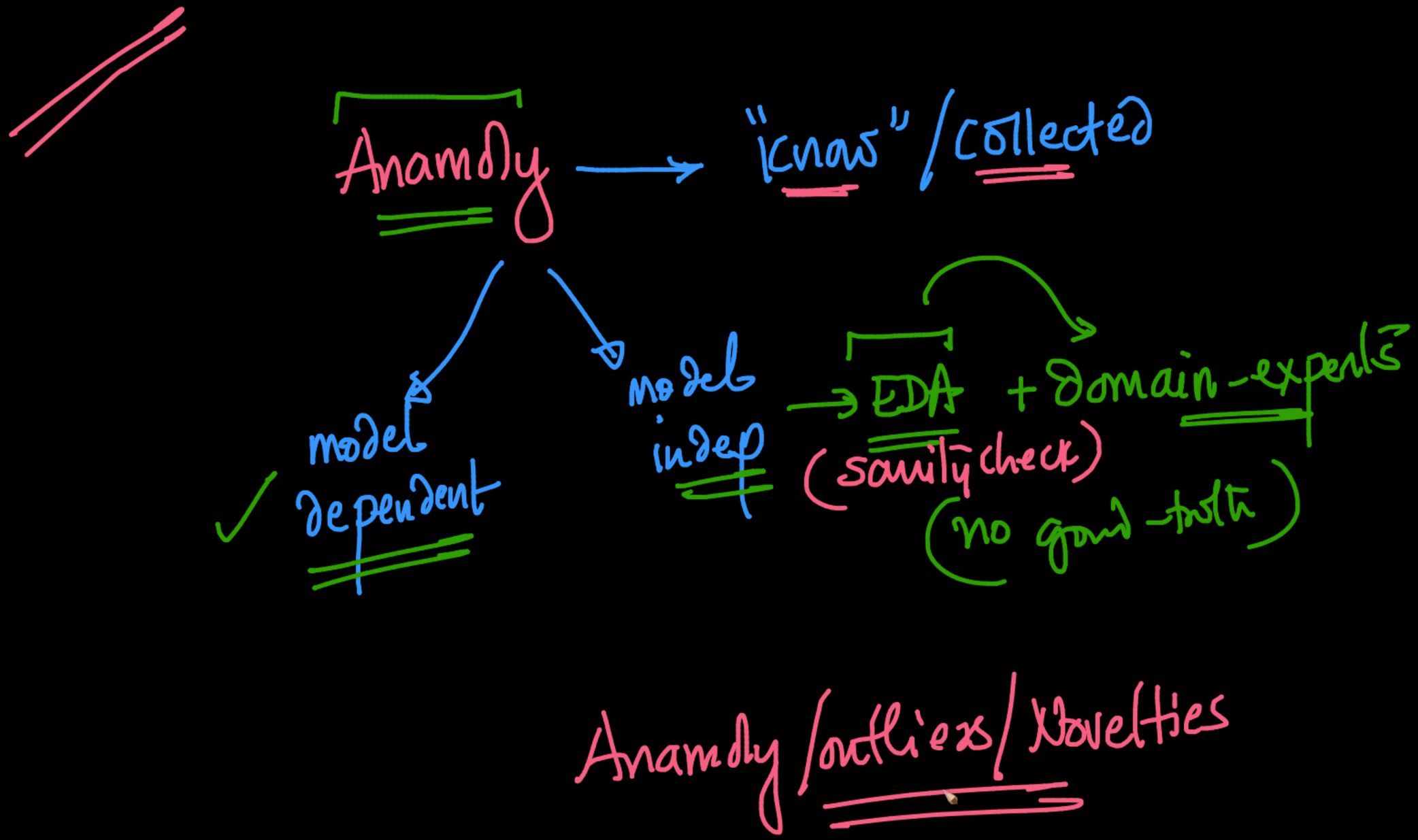
→ VIF can easily handle this

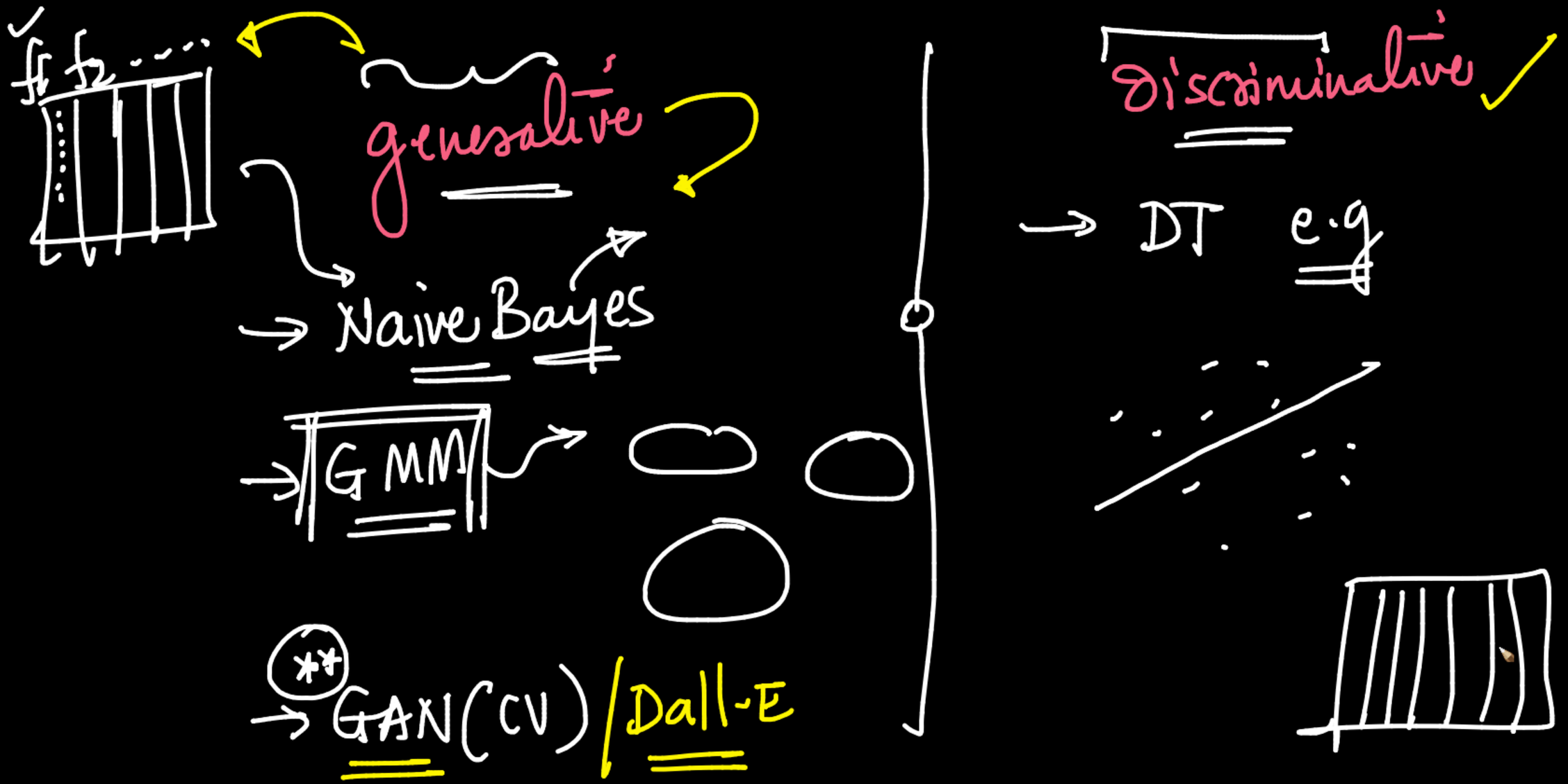
PCA → not preferred

(Q1) d PCA → d'

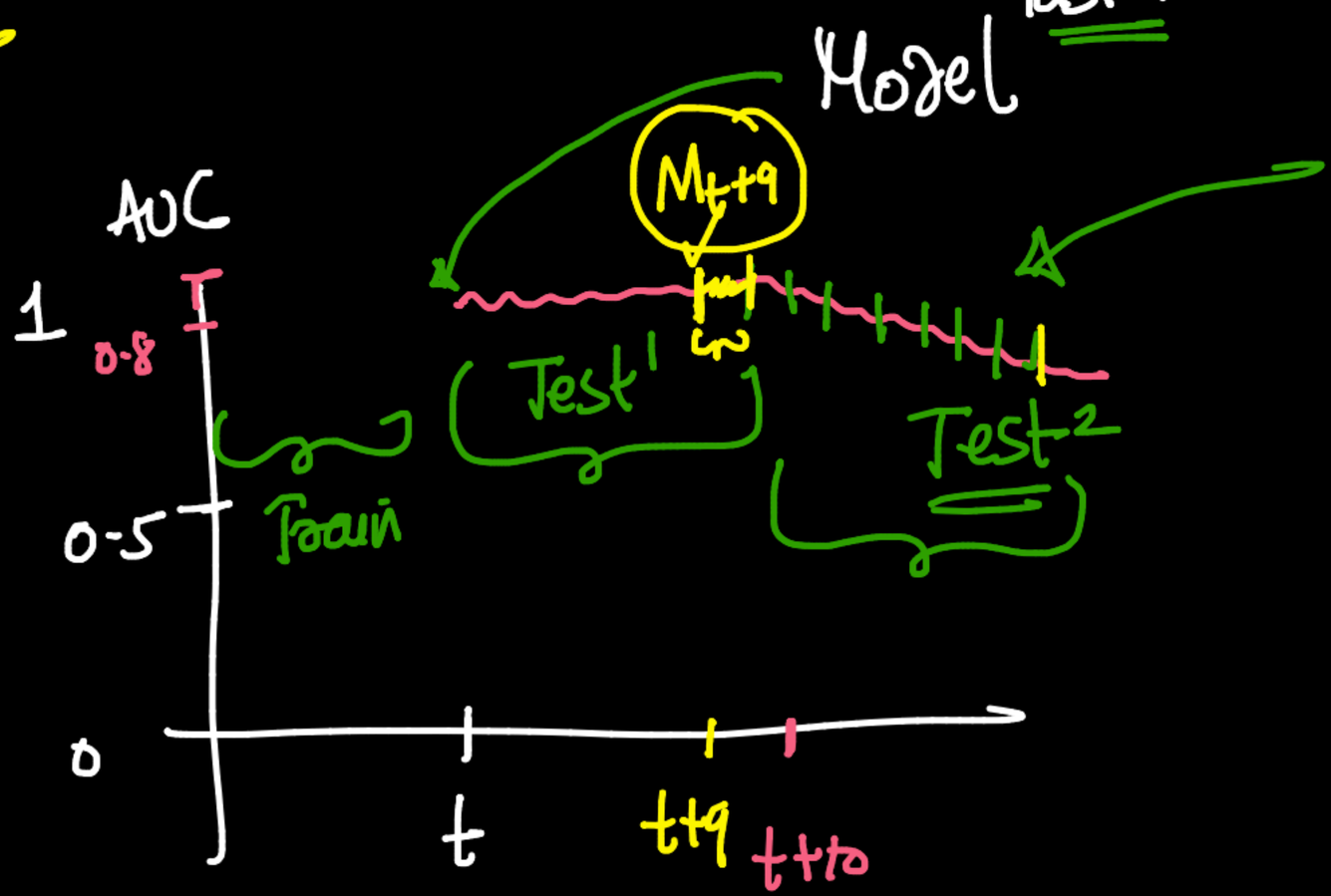
(Q2)


KNN
Stand.....→
Mean Centering & Var scaling
→ G'N N(0,1²)
[norm.....→ Min - Max scaling
CV
0 - 255


Anamdy
"known" / collected
model dependent
model indep
EDA
+ domain-experts
(sanity check)
(no ground-truth)
Anamdy / outliers / Novelties


generative
Naive Bayes
GMM
GAN(CV) / Dall-E
discriminative
DT e.g.

Data drift



Data drift

retrain model (nightly)

prev. model: init

→ high weightage to recent data

$D^1 \ldots D^{100} \rightarrow w = 1$

$M_{1-100} \rightarrow d_{100}$

$\rightarrow d_{101}$

$\sim D^{101}$ → oversample
→ SMOTE

$w = 10$

re-Train

init : M_{1-100}

update : $D^{101}_{upsampled}$

Chunks / modules:

1. Programming
2. Prob - Stats
3. SQL
4. ML-1 — 2
5. ML-2 — 2
6. NN (later)

7. Foundational Math for ML

{ 3 weekends + 1 weekend (ML-1 & ML-2)

Chrome File Edit View History Bookmarks Profiles Tab Window Help

Curse of dimensionality - Wiki | Support-vector machine - Wiki | Gradient boosting - Wikipedia | Computational complexity of m | 3blue 1 brown fourier tranfomrs

youtube.com/results?search_query=3blue+1+brown+fourier+tranfomrs

3blue 1 brown fourier tranfomrs

Home
Explore
Shorts
Subscriptions

Library
History
Watch later
Liked videos

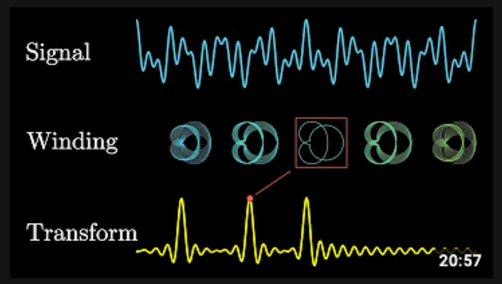
SUBSCRIPTIONS
Music
Sports
Gaming
Movies

EXPLORE
Movies
Gaming
Live
Fashion & Beauty
Learning
Sports

MORE FROM YOUTUBE

FILTERS

Did you mean: ***3 blue 1 brown fourier transform***



But what is the Fourier Transform? A visual introduction.

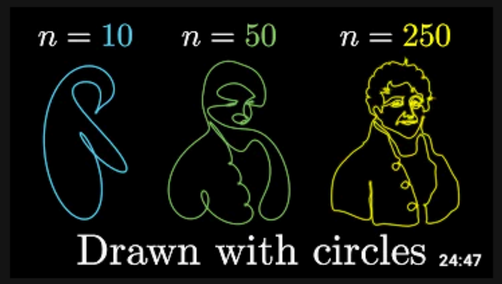
7.8M views • 4 years ago

3Blue1Brown

If you want to check it out, I feel compelled to warn you that it's not the most well-documented tool, and has many other quirks you ...

CC

What's that? | "Almost" Fourier transform? | Inverse Fourier? 3 moments



But what is a Fourier series? From heat flow to drawing with circles | DE4

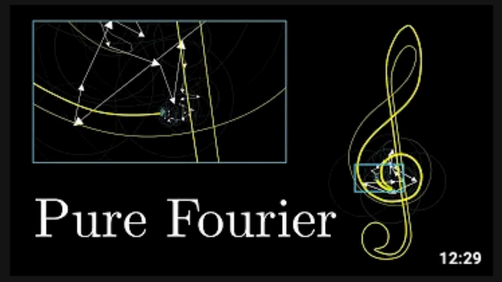
14M views • 3 years ago

3Blue1Brown

Small correction: at 9:33, all the exponents should have a pi^2 in them. If you're looking for more **Fourier** Series content online, ...

CC

Oooh, circle drawings | The heat equation | Interpreting the infinite sum | To the complex plane |... 7 chapters



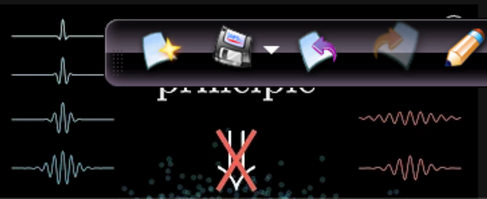
Pure Fourier series animation montage

602K views • 3 years ago

3Blue1Brown

If you're curious about the number of vectors used for each animation: - Eighth note: 100 - Sigma: 200 - Britain: 500 - **Fourier** ...

CC



ier transforms

43 / 43

3Blue1Brown

There's a key way in which the description I gave of the trade-off in Doppler radar differs from reality. Since the speed of light is so ...

CC