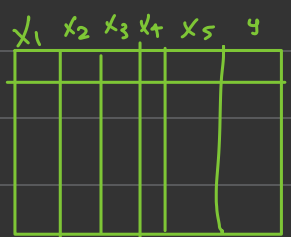
ML Ch2 Rev. ML2 18/02/2023

Agenda:

1. PCA
2. Clustering
3. Model Selection
4. Regularization
5. Boosting

PCA: - Unsupervised ML

- solve problem of 'Curse of Dimensionality'
- v v. high #features → model → Overfits
- ∴ Go for PCA → Captures variance in the IV.
- Statistical transformation technique
- ∴ accepts only numerical data & that too only X



PCA is applied on $X_1 \cdots X_5$.

- always apply PCA on scaled data.

Steps: 1. read csv 2 X, y 3. Train Test Split
↑ Scale X

- fit_transform (X_train)
- transform (X_test)

5. PCA
   - fit_transform (X_train)
   - transform (X_test)
6. Modeling
7. Evaluating
8. Deployment

Q1. What is Curse of Dim?

Q2. Describe pc.

Q3. Can PCA be used in Feature Selection? No.
↳ is Feature Reduction

Q4. Comment on PC1.

Q5. Disadv. of Dimensionality reduction? (Original features lost)

Q6. Do we scale data b4 PCA.

Q7. Can we use it on Large datasets?

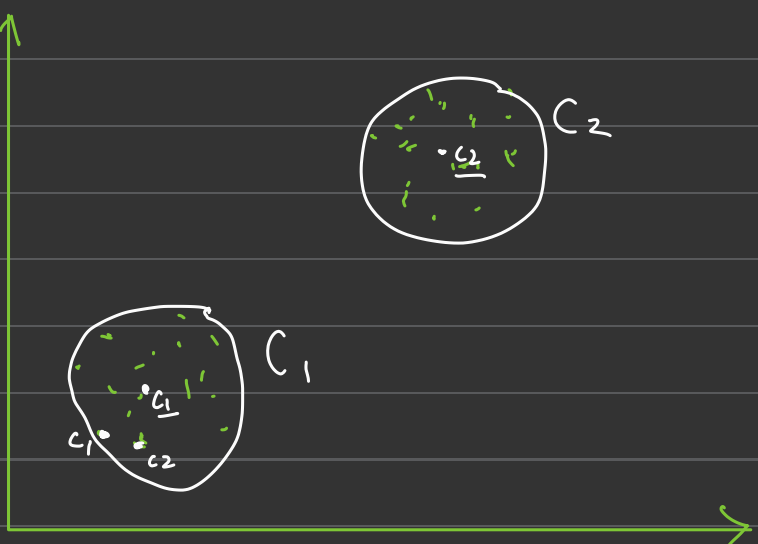
Clustering:

1. Euclidean Distance

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

$$= \sqrt{(5-2)^2 + (2-1)^2}$$

2. Math behind



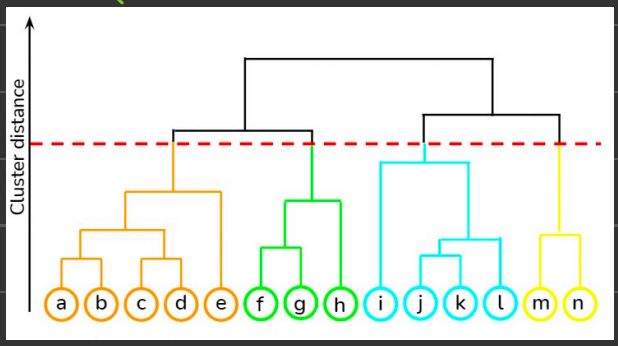
3. Stopping Criteria – No change in centroid values

4. Diff b/w kmeans & kNN

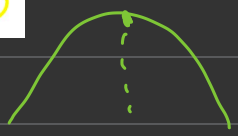
kmeans — Unsupervised — Only X

kNN — Supervised — X & y

5. Hierarchical Clustering vs KMeans Clustering

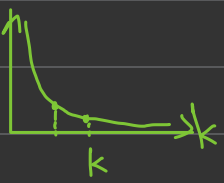


6. What is Silhouette Score?



7. Significance of Hopkins test.

8. Elbow method: wcss



Choosing right value of k.

Regularization: avoid Overfitting

$$y = 15 + 1.2x_1 + 20x_2 + 39x_3$$

$$\downarrow \text{reg}$$

$$y = 0.9 + 1.2x_1 + 2x_2 + 5x_3$$

Penalty

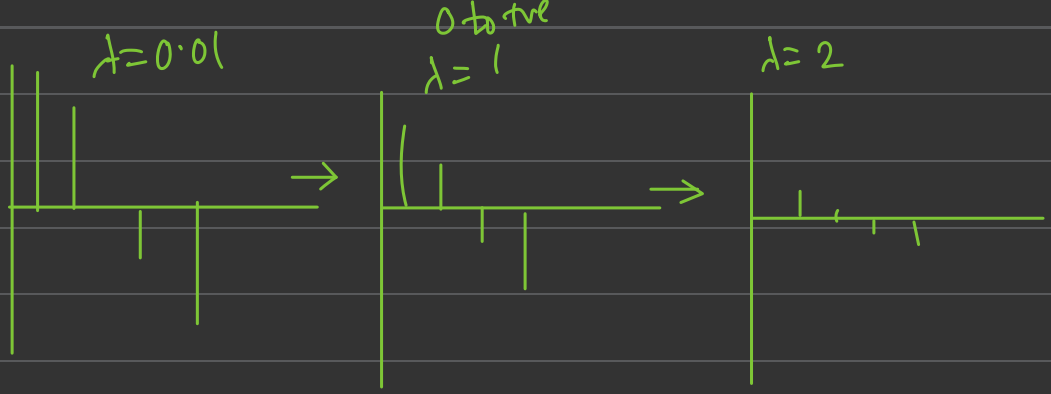
1. Ridge Reg: Loss + $\lambda |m|^2$

or L2 Reg: $(y-\hat{y})^2$

vector of coefficient

Hyperparameter

0 to +ve



$\lambda \uparrow \rightarrow$ Lesser value of coeff.

Use Ridge: when u encounter colinearity

2. LassoReg: Lasso= Loss + $\lambda |m|$

or L1 reg: Penalty

$y = 15 + 1.2x + 20x_2 + 39x_3$

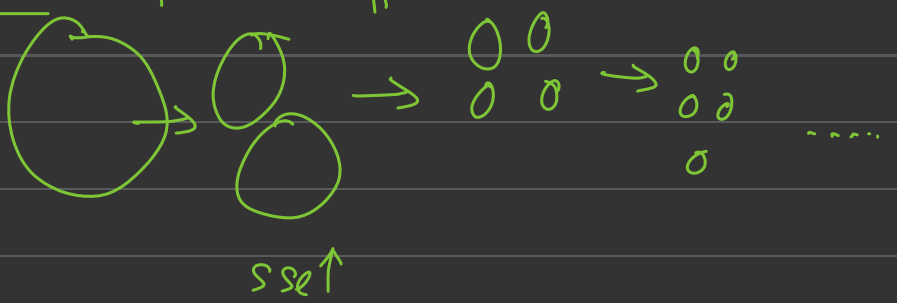
↓ Lasso Reg.

$y = 0.9 + 0x_1 + 0x_2 + 5x_3$

Use: Feature Selection, more robust to Outliers

Hierarchical Cl.: Agglomerative & Diursive:

Diursive: Top-Down App.



Agglomerative: Bottom up app.

Boosting: Combine individual models into a strong learner

- Sequential learning
- Stump
- Avoid Overfitting

1. Is XGBoost faster than RF?

→ XGBoost is a better performer

2. Tick adv. of XGBoost?

a. Lot of Hyperparameters

b. Can handle missing values

c. Distributed computing (Parallel)

3. Disadv. of XGB:

a. Sensitive to Outliers

b. Manually create dummies.

4. Imp. hyperparameters of XGBoost?

https://www.analyticsvidhya.com/blog/2016/03/complete-guide-parameter-tuning-xgboost-with-codes-python/

↳ Only point 1 & 2

5. XGBoost vs LightGBM.

XGBoost	LightGBM
Slower	faster
powerful	relatively less powerful

6. How XGB handles miss values?

gblinear booster → fill miss. val. with 0.

7. XGBoost vs Adaboost?

XGBoost: learns from errors in prediction in order to minimize error.

Adaboost: learns from previous stumps & predictions

Model Selection:

1. Bias & Variance
2. Precision vs Accuracy
3. Bias Variance Tradeoff → Overfitting.
4. Cross Validation: KFold, Leave one out

https://www.javatpoint.com/cross-validation-in-machine-learning

https://www.analyticsvidhya.com/blog/2021/05/4-ways-to-evaluate-your-machine-learning-model-cross-validation-techniques-with-python-code/

5. Correlation
6. P-value Significance
7. VIF [0-5]

$$VIF_i = \frac{1}{1 - R_i^2}$$

8. Wrapper Method: Step fwd, Step bwd, RFE

9. Tree Pruning

10. Grid Search CV vs Randomized SearchCV