26-10-2025:

Agenda:

- → optimal value of k in kNN
- → practical of cross-validation
- → practical of SVM (linear, polynomial, RBF)
- → Hyper parameter tuning using GridSearch
- → Naive Bayes ML Algorithm.

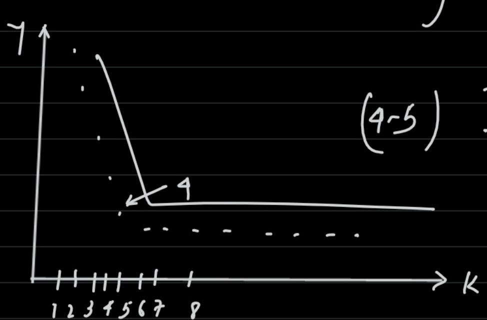
Cross-validation:

Data: A, B, C, D, E, F

k-fold: 3

k-fold	Train	Test
1	C, D, E, F	A, B
2	A, B, E, F	C, D
3	A, B, C, D	E, F

Elbow method → k-means clustering → inertia → kNN → centroid ~~X~~



(4-5) → 4
→ 5

Cross-validation for optimal value of k.

Step 0:

kNN [k=1 , k=2
↓ ↓
mean.acc mean.acc

k-fold = 10 (1)

dataset (2)
1 → train, test → (3) 98%.
2
3
⋮
10 train, test → 86%.

} average accuracy (4) → highest score will get us optimal k. (5)

```
for k in range (1, 20)
knn = KNN(k)
knn.fit(x_train, y_train)
ypred = knn.predict(x_test)

acc = accuracy_score(y_test, y_pred)
accuracy.append(acc)
```

```
for k in range (1, 20)
knn = KNN(k)
for fold in k-folds:

    knn.fit(x_train_fold  y_train_fold)
    y_pred_fold = knn.predict(x_test_fold)
acc_fold_k = accuracy_score(y_test_fold, y_pred_fold)
```

train, test

1 → knn(k) → acc

k → CV fold data → 2 → knn(k) → acc

3 → knn(k) → acc

k → single accuracy

→ mean accuracy of k-folds

k-fold = 3

k(KNN) → k-fold accuracy

1 → avg → mean accuracy

2

3

Hyper parameter Tuning :

```
def add(x, y, z):
```

arguments

$w_{new} = w_{old} - \eta$ gradient

→ $\boxed{\text{learning} \rightarrow \eta}$

→ $\boxed{w, b = [0, 0]}$

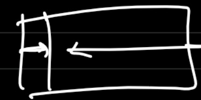
line

hyper parameter

SVM → C
→ gamma
→ kernel

} affect the performance

↓
parameter

brighting : 
↓
Tuning

SVM (parameter)

C=1 , C=0.1 , C=100

→ gamma : scale , auto

→ kernel : linea

accu

→ best model

model_train

m1
m2
m3
m4
m5
⋮
m100

-1

multithre & proc