1) Angle b/w two vectors ⇒

$a\cdot b = |a|\cdot|b|\cos\theta$

⇒ $\cos\theta = \dfrac{a\cdot b}{|a|\cdot|b|}$

2) $a\cdot b = |a||b|\cos\theta$

proj(a,b) =

$|a|\cos\theta = |p|$

$a\cdot b = |a||b|\cos\theta$

⇒ $\dfrac{|p|}{\cancel{|a|}} = \dfrac{a\cdot b}{\cancel{|a|}|b|}$ ⇒ $|p| = a\cdot\hat{b}$

$a\cdot b = |a||b|\cos\theta$

4)

$3x-4y-5=0$

$\dfrac{-4k-5}{5} = \sqrt{9+(1-k)^2}$

$= \sqrt{9+1^2+k^2-2k}$

⇒ $\dfrac{(-4k-5)^2}{25} = 9+1+k^2-2k$

⇒ $16k^2+25+40k = (10+k^2-2k)25$

⇒ $16k^2+25+40k = 250+25k^2-50k$

⇒ $9k^2-90k+225=0$ ⇒ $k^2-10k+25=0$

⇒ $3k^2-30k+75=0$ ⇒ $k^2-5k-5k+25=0$

⇒ $k(k-5)-5(k-5)=0$

$\Rightarrow (k-5)(k-5)=0 \Rightarrow k=5$

5) $x=(1,1,4)$

$a=(4,1,1)$

$b=(2,-4,-4)$

$\mathrm{Proj}^{\wedge}(x, a) = x \cdot \dfrac{\vec{a}}{|a|}$

$$(1,1,4) \cdot \frac{(4,1,1)}{\sqrt{18}} = \frac{4+1+4}{\sqrt{18}}$$

$$= 9/\sqrt{18} = 3/\sqrt{2}\,\hat{a}$$

$$\mathrm{proj}^{\wedge}(x, b) = [1,1,4] \cdot \frac{[2,-4,-4]}{\sqrt{36}}$$

$$= 2-4-16/\sqrt{36} = -18/\sqrt{36} = -3\,\hat{b}$$

6) $f(x) = x^2 - 3x + 2$

$f'(x) = 2x - 3 = 0$

$\Rightarrow x = 3/2$

$f(x=3/2) = 9/4 - 9/2 + 2$

$= \dfrac{9-18+8}{4}$

$= -1/4$

$f''(x) = 2$

7) $f(x) = 1/1+e^{-x}$

quotient rule $\rightarrow$ $\dfrac{d\, a/b}{dx} = \dfrac{a'b - b'a}{b^2}$

$$\frac{d\ 1/1+e^{-x}}{dx} = \frac{d(1+e^{-x})^{-1}}{dx} = -1(1+e^{-x})^{-2} \cdot \frac{d(1+e^{-x})}{dx}$$

$$= -1/(1+e^{-x})^2 \cdot (-e^{-x})$$

$$= \frac{e^{-x}}{(1+e^{-x})^2}$$

Q) $f(x,y) = 2x^2 + 3y^2 + 6$ wrt $x + y - 4 = 0$

$$f(x,y,\lambda) = 2x^2 + 3y^2 + 6 - \lambda(x + y - 4)$$

$\Rightarrow \dfrac{f'}{x} = 4x - \lambda \quad , \quad \dfrac{f'}{y} = 6y - \lambda$

$f'/\lambda = x + y - 4$ | $x = \lambda/4$, $y = \lambda/6$

$\lambda/4 + \lambda/6 - 4 = 0$

$= \dfrac{6\lambda + 4\lambda - 96}{24} = 0$

$\Rightarrow 10\lambda = 96 \Rightarrow \lambda = 96/10$.

$4x - 96/10 = 0$

$\Rightarrow 40x - 96 = 0$

$\Rightarrow 40x = 96$

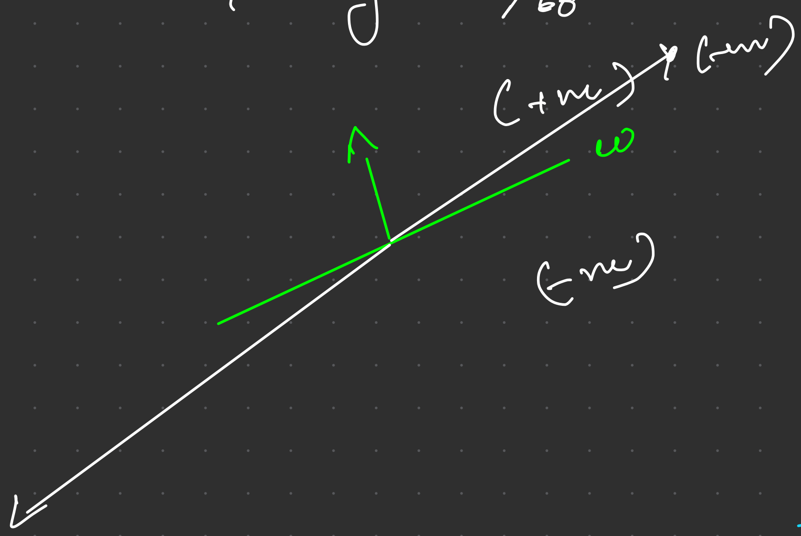
$\Rightarrow x = 96/40$

$= 2.4$

$6y - 96/10 = 0$

$\Rightarrow 60y - 96 = 0$

$\Rightarrow y = 96/60 = 1.5$



$w = w + (x_i) \cdot y_i$

$$\begin{array}{r} 97 \\ 31 \\ \hline 66 \end{array}$$

8 — 11:30 (3 hrs) leetcode + Scaler revision

12 — 3 (3 hrs) gym + refresh

3:30 — 7 (3.5hrs) Scaler + web dev

8 — 11 (3 hrs) web dev

1	2	3	4	5	6	7	8	9	10
2	8	14	29	31	49	65	79	88	97

ans	runs
7th	65 - 49 = 16
6th - 10th	97 - 31 = 66
10th	97 - 88 = 9
3rd - 6th	49 - 8 = 41
4th - 9th	88 - 14 = 74

[10, 42, 48, 60, 80, 8]

Recommendation System

Content based

Cosine similarity

$$\cos\theta = \frac{A \cdot B}{|A| \cdot |B|}$$

Collaborative.

Pearson Correlation.

→ User based → suggest according to similar user.

→ Item based →

Apriori algorithm →

1) **Support** → Prob. of occurrence of an item.

- Filter out less frequent items from the data.
- Combine the selected items & form permutations.

$P(a), P(b), P(c) \ldots P(n)$

a, b, c ← only three items are selected.

→ $P(a|b), P(b|c)\ P(a|c)$

2) **Confidence** →

Pd.notna(data)
— 1 if data notna.
— 0 if data is nan.

Is $P(a|b)$ better than $p(b/a)$

a → 23 times bought.
b → 2 times bought.

$P(b/a) > P(a/b)$

→ when b is bought a is recommended & not vice-versa.

→ Given a pair of items determine which item should be recommended after purchase of the first item.

This is a fountain pen. Very sensitive to pressure & very good for handwriting.
Can be a little rough sometimes.

Association Rule Mining →

$A \to 30,\quad B \to 20,\quad (A \cap B) = 15$

$P(B/A)$? → Confidence.

$$P(B/A) = \frac{P(A \cap B)}{P(A)} = 15/30 = 1/2$$

how likely is $P(B/A)$ going to happen?

$$\text{lift}(B/A) = \text{confidence}(B/A) / \text{support}(B)$$

$$= 1/2 \cdot 1/0.2 = 1/0.4 = 2.5$$

given A, B is 2.5 times more likely to happen.

$$P(B|A) = \frac{P(A \cap B)}{P(A)} \rightarrow \frac{Count(A \,\&\, B)}{Count(A)}$$

GPE → Station.

Cardinal & len(text) > 2 → Train #

ordinal → Coach

text.lower() in [water, ac, cleanliness, toilet] → Service.

else → other.

$[2, 3, 4, 10, 11] = 30$ $\quad$ $[12, 20, 25, 30] = 87$

C $\quad\quad$ 21.75

WCSS $\quad\quad$ M = 13

$= 4^2 + 3^2 + 2^2 + 4^2 + 5^2$

$= 16 + 9 + 4 + 16 + 25$

$= 70$

1) Calculate distance b/w points & centroids.
& Assign each point to the nearest centroid.

2) Recompute the centroid for each group of points.

3) Rinse & Repeat

x	y	dc_1	dc_2 ...	dc_n	C#

```
for iter in len(max iterations)
  // Compute distances from each centroid, for each point &
     assign it to the min distance centroid.
  for point in points:
      dist = ∞
      for j, centroid in enumerate(centroids)
          tdist = euclidean(point, centroid)
          if( tdist < dist)
              dist = tdist
              df.iloc(point)[cluster] = j
  //recompute clusters
gdf = df.groupby('cluster').agg({'points': mean})

      Centroids = gdf.points.toarray()
```

def kmean(points, k, centroids, maxiter = 5)

// calculate dist b/w each point & centroid & assign the
// point to the nearest centroid.

mp → (x, y) → cluster # [0 − k−1]

for i, point in enumerate(points)

dist = −1

for j, cluster in enumerate(centroids)

$$\begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ 2 & 3 & 4 & 4 & 2 & 3 & 1 & 5 & 2 \end{bmatrix}$$

$$\mu = 24/9 = 2.66$$

min ———————————— max

diff = nmax − nmin

$$\frac{\text{diff} \times (x - \min)}{\max - \min} + n\min$$

→ Proportion of range taken by x.

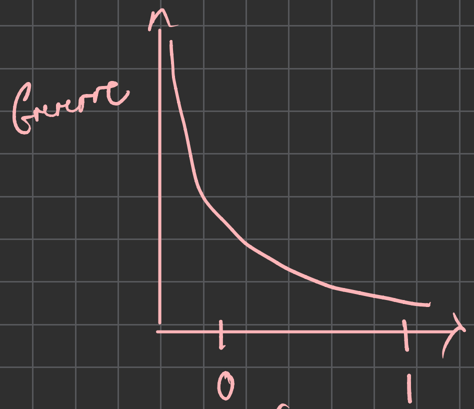
Multiplying this with diff ← gives the proportion of the new range taken by x.

→ nmax − nmin.

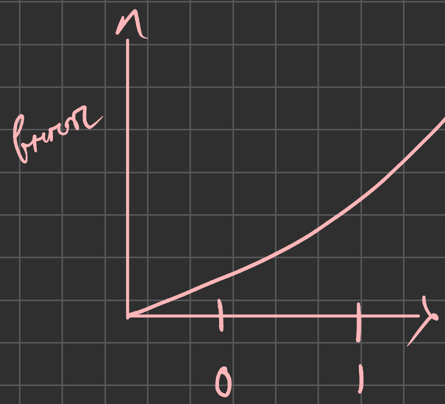
→ Adding this up with nmin brings it to the desired range.

Negative-log loss → Loss for logistic regression.

$$\text{neg-log loss}(y, \hat{y}) = -\sum_{i=1}^{n} y_i \log(\hat{y}_i) + (1-y_i)\log(1-\hat{y}_i)$$



$y_i \log(\hat{y}_i)$



$(1-y_i)\log(1-\hat{y}_i)$

Q:- Implement the model →

Server waiting latency (x)

0
| 1
80
| 0
500

Customer tipped (y)

0
| 1
40
| 0
80
| 1
380
| 0
500

prob = 50% ⇒ $z = 0$

$0.05 \times x + 1 \times 3.5 - 6 = 0$

$0.05x = 2.5$

$\Rightarrow x = \frac{250}{5} = 50$ ms.

D-36

Confusion Matrix, Precision & Recall →

		model P	model F
actual	P	TP	FN
	F	FP	TN

K-Mean →

```
def kmean (k, points, iters)        (x,y)

    centroids = points.sample(k , repeat=false)
    clusters = [-1]*n
    for itr in range(iters):

        // assign clusters to points
        for i, point in enumerate(points)

            distances = [euclid(point, centroid)
                            for _, centroid in centroids.iterrows()]
            cluster = np.argmin(distances)
            points.loc[i, 'cluster'] = cluster


        // update centroids

        centroids = points.groupby('cluster')[['x', 'y']]
                                              .mean()
```

or

```
for j, cent in enumerate(centroids)
    tdist = euclid(cent, point)
    if( tdist < dist) dist = tdist
        cluster[i] = j
```

KNN →

```
def knn ( x , points)
    distances = [euclid(x, point) for point in points]
    sort( (points, distances) )
```

SVM →

$$Y = mx + c$$

$$w_1 x + w_2 y = 1$$

$$Ax + By + C = 0$$

$\Rightarrow By + C = 0 \qquad Ax + C = 0$

$\Rightarrow y = -C/B \qquad \Rightarrow x = -C/A$

$$Y = mx + c$$

$x = 0$, $Y = c$ | $Y = 0$
$mx + c = 0$
$\Rightarrow x = -c/m$

$\Rightarrow \frac{x}{a} + \frac{y}{b} = 1 \Rightarrow -\frac{mx}{c} + \frac{y}{c} = 1$

$Y = mx + c$ | $Ax + By + C_g = 0$

$By = -Ax - C_g$

$\Rightarrow y = -\frac{A}{B}x - \frac{C_g}{B}$

$m = -w_1/w_2, \quad c = -C_g/w_2$

$$w_1 x + w_2 y + w_0 = \pm K$$

$\Rightarrow w_1 x + w_2 y = \pm k$

$\Rightarrow d_1 + d_2 = \frac{2k}{|w|}$

Perceptron Training →

→ true, but classified as −ve.

if(misclassification)

$w = w + (-y * x);$

$y = +ve, \quad x = -ve$

$x - y = 0$

$1 - 2 = -1$ ✓

$2 - 4 = -2$

$6 - 7 = -1$ ✓

$9 - 10 = -1$ ✓

$11 - 8 = 2$

$2 - 0 = 2$

$4 - 3 = 1$ ✓

$2 - 1 = 1$ ✓

$11 - 20 = 9$

$6 - 10 = -4$

$5 - 1 = 4$

$1 - 0 = 1$ ✓