

ML Experiment 4

The image shows a Google Colab notebook interface. At the top, the browser address bar displays the URL: colab.research.google.com/drive/1VgKOLPjBjBfIFBw8f-ovB53Kp97gJtm5. The notebook's title bar indicates it is named 'KNN.ipynb'. The interface includes a menu bar with options like File, Edit, View, Insert, Runtime, Tools, and Help. Below the menu, there are tabs for '+ Code' and '+ Text'. The main area of the notebook contains a single code cell with the following Python code:

```
df = [(80, 12, 'Not Hired'),
      (75, 8, 'Not Hired'),
      (85, 10, 'Hired'),
      (90, 5, 'Hired'),
      (78, 9, 'Not Hired'),
      (82, 7, 'Hired'),
      (65, 11, 'Not Hired'),
      (92, 4, 'Hired'),
      (71, 9, 'Not Hired'),
      (89, 6, 'Hired'),
      (85, 13, 'Not Hired'),
      (79, 11, 'Not Hired'),
      (87, 9, 'Hired'),
      (93, 3, 'Hired'),
      (77, 7, 'Not Hired'),
      (83, 10, 'Hired'),
      (86, 14, 'Not Hired'),
      (94, 4, 'Hired'),
      (72, 8, 'Not Hired'),
      (88, 5, 'Hired'),
      (82, 15, 'Not Hired'),
      (76, 13, 'Not Hired'),
      (86, 6, 'Hired'),
      (91, 3, 'Hired'),
      (75, 7, 'Not Hired'),
      (89, 11, 'Not Hired'),
      (84, 9, 'Hired'),
      (96, 2, 'Hired'),
      (74, 10, 'Not Hired'),
      (90, 6, 'Hired'),
      (81, 8, 'Not Hired'),
      (83, 14, 'Not Hired'),
      (89, 4, 'Hired'),
      (95, 5, 'Hired'),
      (73, 9, 'Not Hired'),
      (88, 7, 'Hired'),
      (69, 12, 'Not Hired'),
      (86, 3, 'Hired'),
      (92, 4, 'Hired')]
```

The code defines a list of tuples, each representing a data point with three values: a numerical value, another numerical value, and a categorical label ('Not Hired' or 'Hired'). The list is assigned to the variable 'df'.

[illegible]

```
colabresearch.google.com/drive/1VgKOLPjBtFfIBw8i-ovB53Kp97gltm5

KNN.ipynb
File Edit View Insert Runtime Tools Help Last edited on 26 February
+ Code + Text
Connect Colab AI

k=5
dic = {
    'Hired':0,
    'Not Hired':0
}
max = -1
ans = ""
l = len(dist)
for i in range(k):
    dic[dist[i][1]] += 1
    if dic[dist[i][1]] > max:
        max = dic[dist[i][1]]
        ans = dist[i][1]
print(dic)
print(ans)

{'Hired': 0, 'Not Hired': 5}
Not Hired

[ ] def acc(tup, df):
    dist = []
    for i in range(len(df)):
        l = [math.sqrt((df[i][0]-tup[0])**2 + (df[i][1]-tup[1])**2), df[i][2]]
        dist.append(l)
    dist.sort()
    k=1
    dic = {
        'Hired':0,
        'Not Hired':0
    }
    max = -1
    ans = ""
    l = len(dist)
    for i in range(k):
        dic[dist[i][1]] += 1
        if dic[dist[i][1]] > max:
            max = dic[dist[i][1]]
            ans = dist[i][1]
    return ans==tup[2]
```

```
colabresearch.google.com/drive/1VgKOLPjBtFfIBw8i-ovB53Kp97gltm5

KNN.ipynb
File Edit View Insert Runtime Tools Help Last edited on 26 February
+ Code + Text
Connect Colab AI

[ ]
    max = -1
    ans = ""
    l = len(dist)
    for i in range(k):
        dic[dist[i][1]] += 1
        if dic[dist[i][1]] > max:
            max = dic[dist[i][1]]
            ans = dist[i][1]
    return ans==tup[2]

[ ] testdata = [
    (98, 2, 'Hired'),
    (76, 16, 'Not Hired'),
    (88, 8, 'Hired'),
    (93, 7, 'Hired'),
    (75, 11, 'Not Hired'),
    (82, 11, 'Hired'),
    (85, 14, 'Not Hired'),
    (69, 16, 'Not Hired'),
    (98, 5, 'Hired'),
    (77, 12, 'Not Hired'),
    (84, 10, 'Not Hired'),
    (96, 3, 'Hired'),
    (73, 13, 'Not Hired'),
    (89, 8, 'Hired'),
    (94, 7, 'Hired')
]
yes=0
no=0
for tup in testdata:
    if acc(tup, df):
        yes += 1
    else:
        no += 1
print(yes, no)
print("Accuracy: ", yes/(yes+no))

14 1
Accuracy: 0.9333333333333333
```

ML Experiment 4

DATE:

Kareena Shah

60004210243

0179

C'32

PSI

Aim : To Implement Principal Component Analysis (PCA)

Theory :

PCA works on the condition that while the data in a higher dimensional space is mapped to data in lower dimension space, the variance of the data in the lower dimensional space should be maximum.

PCA is a statistical procedure that uses an orthogonal transformation that converts a set of correlated variables to a set of uncorrelated variables.

It is an unsupervised learning algorithm used to examine the iterations among a set of variables.

Steps

(I) Standardization

$$z = \frac{x - \mu}{\sigma}$$

(II) Covariance Matrix Computation

$$\text{cov}(x_1, x_2) = \frac{\sum_{i=1}^n (x_{1i} - \bar{x}_1)(x_{2i} - \bar{x}_2)}{n-1}$$

DATE:

(iii) Compute Eigen values & eigen vectors of covariance matrix to identify principal components

$$AX = \lambda X$$

$$AX - \lambda X = 0$$

$$(A - \lambda I)X = 0$$

Conclusion : Thus, we implemented PCA.