



המחלקה להנדסת חשמל ואלקטרוניקה
מערכות לומדות ולמידה עמוקה (31245)

Lab 4 report

פרנסיס עבוד

Prepared for: Dr. Amer Adler

Date: 27/04/2025

Ex.:

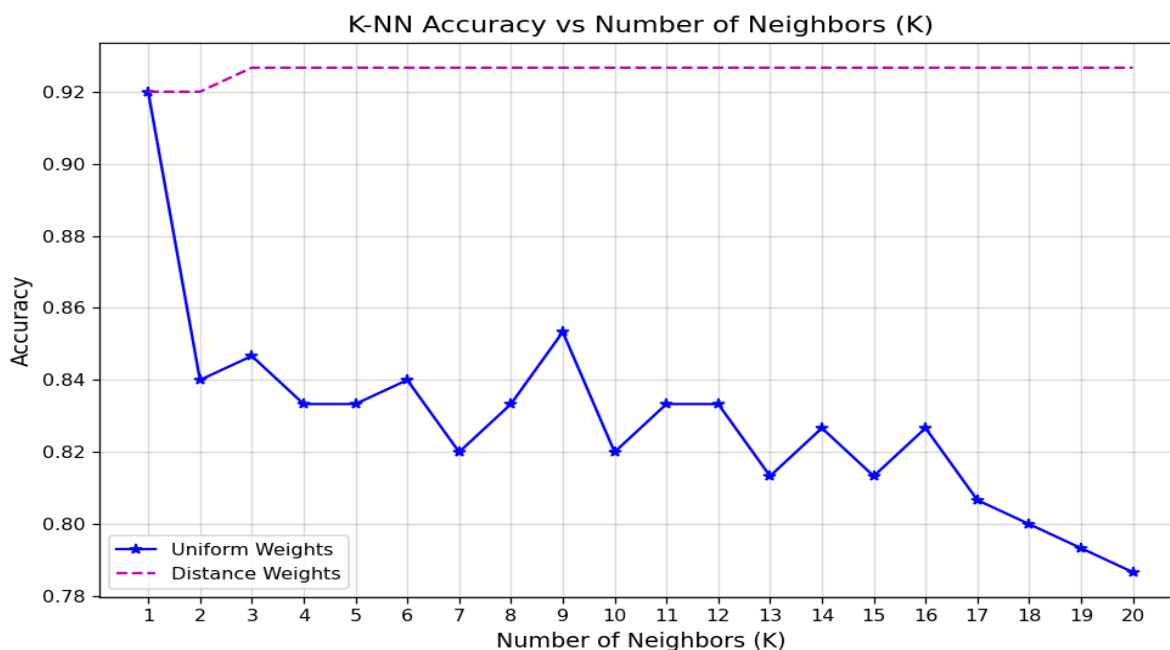
יש לכתוב קוד פייתון למימוש KNN עבור דוגמאות ה IRIS - בשני אופנים:

1. ללא משקול מרחק בין השכנים (uniform=weights) יש להציג את גרף ה Accuracy - בתלות בגודל K בטווח שבין 1 ל 20
2. עם משקול מרחק בין השכנים (distance=weights) יש להציג את גרף ה Accuracy - בתלות בגודל K בטווח שבין 1 ל 20

Solution:

```
1  # -*- coding: utf-8 -*-
2
3  """
4  K-Nearest Neighbors (K-NN) Experiment:
5  - Evaluate the accuracy of K-NN for different values of K (1 to 20).
6  - Compare two weighting strategies: 'uniform' and 'distance'.
7  - Visualize the accuracy trends for both strategies.
8  """
9
10 import numpy as np
11 import matplotlib.pyplot as plt
12 from sklearn.datasets import load_iris
13 from sklearn.neighbors import KNeighborsClassifier
14
15 # Load the Iris dataset and extract the first two features
16 data = load_iris()
17 features = data.data[:, :2] # Only the first two features
18 labels = data.target # Target labels
19
20 # Define the range of K values (number of neighbors)
21 k_range = range(1, 21)
22
23 # Initialize dictionary to store accuracy scores for each weighting method
24 accuracies = {
25     'uniform': [],
26     'distance': []
27 }
28
29 # Create a mapping for plot styles
30 style_map = {
31     'uniform': {'line': 'b*-', 'label': 'Uniform Weights'},
32     'distance': {'line': 'm--', 'label': 'Distance Weights'}
33 }
34
35 # Iterate over each weighting method
36 for weights in ['uniform', 'distance']:
37     # Iterate over each K value
38     for k in k_range:
39         # Create and fit K-NN model with the current weighting method
40         model = KNeighborsClassifier(n_neighbors=k, weights=weights)
41         model.fit(features, labels)
42
43         # Compute and store the accuracy score
44         accuracy = model.score(features, labels)
45         accuracies[weights].append(accuracy)
46
47 # Plot the accuracy results
48 plt.figure(figsize=(10, 6))
49
50 # Plot each weighting method's accuracy
51 for weights in accuracies:
52     plt.plot(k_range, accuracies[weights], style_map[weights]['line'],
53             label=style_map[weights]['label'])
54
55 plt.title('K-NN Accuracy vs Number of Neighbors (K)', fontsize=14)
56 plt.xlabel('Number of Neighbors (K)', fontsize=12)
57 plt.ylabel('Accuracy', fontsize=12)
58 plt.xticks(k_range)
59 plt.legend(loc='best')
60 plt.grid(alpha=0.5)
61 plt.show()
```

Output:



סיכום הניסוי ותוצאותיו

ניסוי זה בחן את השפעת מספר השכנים (K) על דיוק אלגוריתם K-NN בסיווג פרחי איריס, תוך השוואת שתי שיטות משקולות:

-משקולות אחידות ('uniform'): כל השכנים משפיעים באופן שווה על הסיווג
-משקולות מרחק ('distance'): שכנים קרובים יותר משפיעים יותר על הסיווג

תוצאות הניסוי מראות כי:

1. שיטת משקולות המרחק מניבה דיוק גבוה יותר באופן עקבי (כ-92%) לעומת שיטת המשקולות האחידות.
2. עבור משקולות אחידות, הדיוק הגבוה ביותר (כ-92%) מושג כאשר $K=1$ ויורד ככל שמספר השכנים גדל.
3. עבור משקולות מרחק, הדיוק נשאר יציב יחסית גם כאשר K גדל, מה שמצביע על עמידות גבוהה יותר לרעשים במידע.
4. באופן כללי, שיטת משקולות המרחק נמצאה כעדיפה לסט נתונים זה.

מסקנות:

-משקולות מרחק הינן בחירה עדיפה כאשר מפעילים אלגוריתם K-NN על מסד נתוני איריס.
-בחירת ערך K אופטימלי הינה קריטית להשגת ביצועים טובים, כאשר ערכים נמוכים של K (בין 1 ל-3) נותנים את התוצאות הטובות ביותר עבור נתונים אלו.
-הניסוי מדגים את החשיבות של בחירת הפרמטרים המתאימים באלגוריתמי למידת מכונה וכיצד הם משפיעים על ביצועי המודל.