کتابخانه‌های مورد نیاز

ابتدا کتابخونه‌های لازم توی برنامه را اضافه میکنیم

```
# Import the required libraries
import numpy as np
from sklearn.neighbors import NearestNeighbors
import pandas as pd
import matplotlib.pyplot as plt
from random import sample
from numpy.random import uniform

# Visualasation: Perform PCA on the original data (so that 4-D data can be visualised in 2-D)
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
```

روش کار الگوریتم

- یه داده اولیه داریم به اندازه n و به اسم X
- بدون عمل جایگذاری حدود m تعداد از داده اولیه رو که کوچکتر از اندازه کل داده است (m < n) انتخاب میکنیم، در اینجا ما حدود ۵٪ رو انتخاب کردیم.
- به صورت تصادفی یه سری دیتای جدید رو تولید میکنیم و اسمش رو میزاریم Y که به صورت یکنواخت توزیع شدند.
- دو روش فاصله رو تعریف میکنمی

  - U که فاصله یک نقطه در Y از نزدیک ترین همایسه های اون نقطه در X

  - W که قاصه یک نقطه از X از نزدیک ترین همسابه های اون نقطه در X
- اگر داده ها d بعدی باشند. Hopkins statistic به صورت زیر تعریف میشه

$$H = \frac{\sum_{i=1}^{m} u_i^d}{\sum_{i=1}^{m} u_i^d + \sum_{i=1}^{m} w_i^d}$$

Hopkins statistic: روشی برای محاسبه و اندازه گیری cluster tendency مجموعه داده ها.

```python
def hopkins_statistic(X, k = 2):

    # Convert dataframe to numpy array
    X = X.values;

    # Get sample size from data (5%)
    sample_size = int(X.shape[0]*0.05)

    # A uniform random sample in the orifinal data space
    urs = uniform(X.min(axis = 0), X.max(axis = 0), (sample_size, X.shape[1]))

    # A random sample of size sample_size from the original data X
    random_indices = sample(range(0, X.shape[0], 1), sample_size)
    samples = X[random_indices]

    # Initialise unsupervised learner for implementing neighbor searches
    neighbors = NearestNeighbors(n_neighbors = k)
    nbrs = neighbors.fit(X)

    # u_distances = nearest neighbour distances from uniform random sample
    u_distances, u_indices = nbrs.kneighbors(urs , n_neighbors = k)
    # distance to the first (nearest) neighbour
    u_distances = u_distances[: , 0]

    # w_distances = nearest neighbour distances from a sample of points from original data X
    w_distances , w_indices = nbrs.kneighbors(samples , n_neighbors = k)
    # distance to the second nearest neighbour (as the first neighbour will be the point itself, with distance = 0)
    w_distances = w_distances[: , 1]


    u_sum = np.sum(u_distances)
    w_sum = np.sum(w_distances)

    # Compute and return hopkins' statistic
    H = u_sum/ (u_sum + w_sum)
    return H
```

توضیح الگوریتم بالا داده شد. طبق فرمول داره محاصبه میشه. یه داده X رو میگیره . بعد مشخص میکنه 5% از اندازه داده چقدر میشه، مثلا اگر 100 تا داده داریم، میشه 5 تا، بعد میاد به اون تعداد از داده اصلی جدا میکنه . بعدش هم با استفاده از knn فاصله ها رو حساب میکنه و فرمولی که عکسش بالا اومده رو محاسبه میکنه.

اطلاعات دیتاست

- 150 تا نمونه گل توی سه تا دسته (برای هر دسته 50 تا)
- 4 تا ویژگی عددی و 1 دونه هم پیشبینی (کلا 5 تا)
- داده missing value هم جزو ویژگی ها نداریم

اطلاعات ویژگی ها

- زنبق نوک زبر (Iris setosa)
- زنبق رنگارنگ (Iris versicolor)
- زنبق ویرجینیا (Iris virginica)

نمایش دیتاست روی نمودار و پلات

ایجاد داده تصادفی و نمایش روی نمودار و پلات