Лабораторна робота №2.

Мета: засвоїти основні відомості про роботу з алгоритмом метод головних компонент(PCA) та дерева рішень.

Теоретичні відомості

Опис алгоритму дерева рішень

Дерево рішень (Decision tree) — це непараметричний контрольований алгоритм навчання, який можна застосовувувати для опрацювання як дискретних, так і безперервних даних. Він має ієрархічну структуру дерева, яка складається з кореневого вузла, гілок, внутрішніх вузлів і листових вузлів. Основне завдання алгоритму, це розподілити набір даних на підмножини на основі найважливішого атрибута у цих даних.

Основні сфери застосування DT:

- Класифікація даних
- Регресивний аналіз даних

Покроковий опис алгоритму обрахунку DT.

1. Підготуйте дані. Для усіх значень знайдіть значення Gini(D)

$$Gini(D) = 1 - \sum(pi)2$$

2. Знайти $Gini_k(D)$ для кожного атрибута k

$$Gini_k(D) = \sum (D_i / D) * Gini(D_i)$$

3. Розбити за мінімальними значеннями індексу Gini. Тобто елемент з найменшим значення Gini, буде коренем нашого дерева, а всі подальші гілки і листки додаватимуться за зростанням значення індексу Gini.

Опис методу головних компонент. Застосування.

Ме́тод головни́х компоне́нт (*PCA*) — метод аналізу в статистиці, який використовує ортогональне перетворення множини спостережень з можливо пов'язаними змінними (сутностями, кожна з яких набуває різних числових значень) у множину змінних без лінійної кореляції, які називаються **головними компонентами**.

Загалом

Основні сфери застосування PCA:

- Візуалізація даних
- Стиснення даних
- Зменшення шуму
- Анонімізація даних

Покроковий опис алгоритму обрахунку PCA.

1. Підготуйте дані. При потребі розділіть на залежні і незалежні змінні(Y і X). Нехай X дані мають розмірність d x n, де d - кількість ознак, а n - кількість рядків даних.
2. Відцентрувати дані. Порахуйте вектор середніх ознак який матиме розмірність d і міститиме середні значення усіх ознак. Віднімiть цей вектор від кожного рядка даних.

3. Якщо у вас є залежні ознаки(Y):
    a. Обрахуйте вектор стандартного відхилення усіх ваших ознак
    b. Розділіть кожну одиницю даних на цей вектор.
    c. Цю відцентровану і нормалізовану матрицю назвемо Z.
4. Обрахуйте матрицю коваріації $\mathbf{Z^TZ/(n-1)}$
5. Обрахуйте власні значення і власні вектори отриманої матриці
6. Посортуйте значення та вектори згідно спадання значень. Відсортовану матрицю векторів назвемо P*.
7. Обрахуйте нові ознаки. Z* = ZP*.

Завдання

1. Частина 1. Підготовка даних
    a. Завантажити дані згідно варіанту.
    b. Розділити на тренувальну, валідаційну та тестові вибірки в пропорції 70/15/15
    c. Обробити NaN значення - або видаленням, або заповненням середнім, або якимось іншим чином. Обраний варіант обгрунтувати.
    d. Нормалізувати дані - для тренувальних даних обрахувати значення mean(середнього) та standard deviation(стандартного відхилення) для кожної з ознак і перетворити дані віднявши обраховане середнє та поділивши на стандартне відхилення. Зауважте, що на відміну від Z-score тут немає модуля.
    e. Нормалізувати валідаційні та тестові даних на основі обрахованих mean та std для тренувальних даних
2. Частина 2. Побудова дерева рішень
    a. Побудувати 3 різні дерева рішень(Decision tree) з різною глибиною на основі даних згідно варіанту.
    b. Візуалізувати отримані дерева рішень. Порівняти точність отриманих дерев рішень на валідаційній та тренувальній вибірках.
3. Частина 3. Зашумлення.
    a. Добавити до тренувальних даних шум. Зробити це додаючи випадкове(uniform(рівномірний) розподіл в межах [-0.1,0.1]) зміщення до кожного елементу тренувальної вибірки датасету по кожній з ознак.
    b. Побудувати 3 різні дерева рішень(Decision tree) з різною глибиною на основі даних згідно варіанту.
    c. Візуалізувати отримані дерева рішень. Порівняти точність отриманих дерев рішень на валідаційній та тренувальній вибірках.
4. Частина 4. Знешумлення
    a. Обрахувати на оригінальній тренувальній вибірці PCA. Візуалізувати отримані компоненти.
    b. Обрахувати на зашумленій тренувальній вибірці PCA. Візуалізувати отримані компоненти.

c. Реконструювати дані використовуючи PCA обрахований на зашумлених даних для знешумлення. Для цього використати ознаки що пояснюють 95% variance.
d. Візуалізувати оригінальні, зашумлені і знешумлені дані

5. Частина 5
    a. Побудувати на основі зашумлених і знешумлених даних тренувальної вибірки дерева рішень. Порівняти їхню точність на валідаційній та тренувальних вибірках.
    b. Візуалізувати отримані дерева рішень

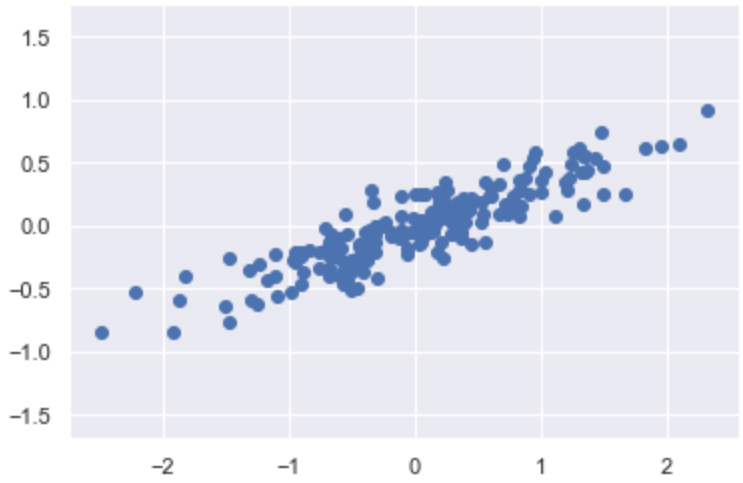
Варіанти. Варіант обираєте згідно таблиці і номер в документі розподілу на підгрупи.

Крім цього можна замість варіанту з таблиці обрати індивідуальний датасет. Список ресурсів де можна обрати індивідуальний варіант(узгодити з викладачем)
https://data.gov.ua/
https://www.kaggle.com/datasets

Варіант #	Датасет	Назва колонки яку потрібно передбачити
1	diabetes	‘Diabetes_012’
2	bankruptcy	‘Bankrupt’
3	planets	'mass',
4	tips	'tip'
5	diamonds	'clarity',
6	exercise	‘pulse’
7	titanic	'survived',
8	iris	'species'

Приклад реалізації:

```
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns; sns.set()
rng = np.random.RandomState(1)
X = np.dot(rng.rand(2, 2), rng.randn(2, 200)).T
plt.scatter(X[:, 0], X[:, 1])
plt.axis('equal');
```




```
from sklearn.decomposition import PCA
pca = PCA(n_components=2)
pca.fit(X)
print(pca.components_)
```

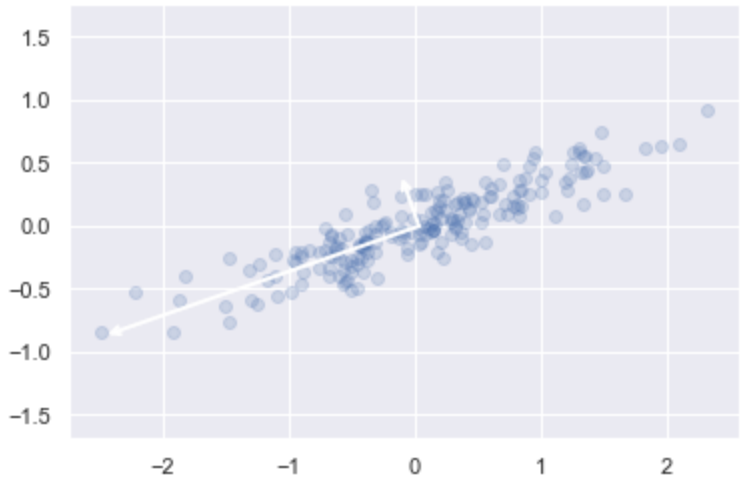
```
[[-0.94446029 -0.32862557]
 [-0.32862557  0.94446029]]
```

print(pca.explained_variance_)

```
[0.7625315 0.0184779]
```

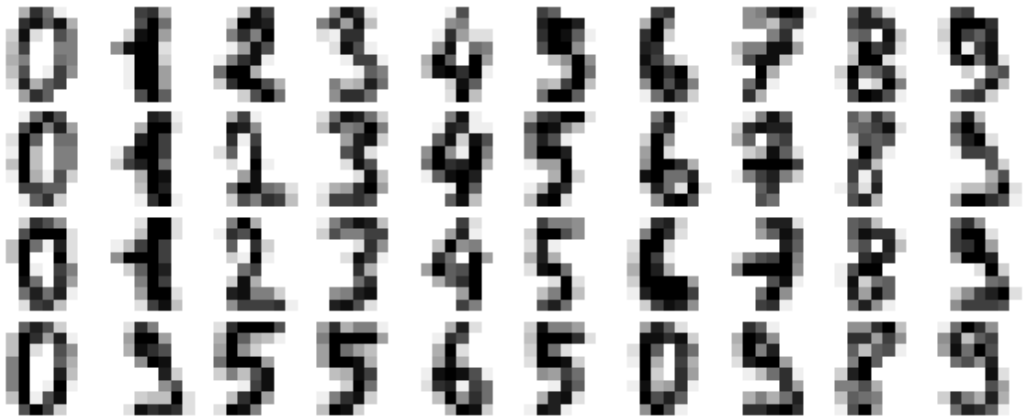
```
def draw_vector(v0, v1, ax=None):
    ax = ax or plt.gca()
    arrowprops=dict(arrowstyle='->',
                    linewidth=2,
                    shrinkA=0, shrinkB=0)
    ax.annotate('', v1, v0, arrowprops=arrowprops)

# plot data
plt.scatter(X[:, 0], X[:, 1], alpha=0.2)
for length, vector in zip(pca.explained_variance_, pca.components_):
    v = vector * 3 * np.sqrt(length)
    draw_vector(pca.mean_, pca.mean_ + v)
plt.axis('equal');
```



Приклад реалізації знешумлення:

```
def plot_digits(data):
    fig, axes = plt.subplots(4, 10, figsize=(10, 4),
                             subplot_kw={'xticks':[], 'yticks':[]},
                             gridspec_kw=dict(hspace=0.1, wspace=0.1))
    for i, ax in enumerate(axes.flat):
        ax.imshow(data[i].reshape(8, 8),
                  cmap='binary', interpolation='nearest',
                  clim=(0, 16))
plot_digits(digits.data)
```



```
np.random.seed(42)
noisy = np.random.normal(digits.data, 4)
plot_digits(noisy)
```

```
pca = PCA(0.50).fit(noisy)
components = pca.transform(noisy)
filtered = pca.inverse_transform(components)
plot_digits(filtered)
```

Питання для самоконтролю:

- Що таке дерева прийняття рішень?
- Яким чином відбувається розбиття на піддерева/листки в деревах прийняття рішень?
- Що таке Ентропія та Information gain? Яку роль вони грають в деревах прийняття рішень?
- Що таке PCA?
- Які особливості нових ознак що генерує PCA?
- Завдяки чому PCA можна використовувати для знешумлення даних?
- Поясніть чому на вашу думку ви отримали кращі/гірші результати на зашумлених та знешумлених даних