КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА

Факультет комп'ютерних наук та кібернетики

ЗВІТ
з лабораторної роботи №1
з курсу «Розпізнавання образів»
на тему «Основні поняття розпізнавання образів»

Кузьменко Володимир
група ПМ-1

Київ-2025

1. Постановка задачі

Напишіть програму для обчислення чутливості і специфічності, а також для побудови ROC – кривої по заданій дискримінантній функції, що розділяє дані про іриси Фішера.

2. Теоретичні відомості

Розпізнавання образів – розділ теорії штучного інтелекту, що вивчає методи класифікації об’єктів. За традицією об’єкт, що піддається класифікації, називається образом (pattern). Образом може бути цифрова фотографія (розпізнавання зображень), буква або цифра (розпізнавання символів), запис мови (розпізнавання мови) тощо. В межах теорії штучного інтелекту розпізнавання образів включається в більш широку наукову дисципліну — теорію машинного навчання (machine learning), метою якої є розробка методів побудови алгоритмів, що здатні навчатися. Існує два підходи до навчання: індуктивне і дедуктивне. Індуктивне навчання, або навчання за прецедентами, засноване на виявленні загальних властивостей об’єктів на підставі неповної інформації, отриманих емпіричним шляхом. Дедуктивне навчання передбачає формалізацію знань експертів у вигляді баз знань (експертних систем тощо). Кожний образ являє собою набір чисел, що описують його властивості і називаються ознаками (feature). Упорядкований набір ознак об’єкта називається вектором ознак (feature vector). Вектор ознак — це точка в просторі ознак (feature space). Класифікатор, або вирішальне правило (decision rule) — це функція, яка ставить у відповідність вектору ознак образу клас, до якого він належить. Задачу розпізнавання образів можна розділити на ряд підзадач.

1. Генерування ознак (feature generation) — вимірювання або обчислення числових ознак, що характеризують об’єкт.

2. Вибір ознак (feature selection) — визначення найбільш інформативних ознак для класифікації (в цей набір можуть входити не лише первинні ознаки, але й функції від них).

3. Побудова класифікатора (classifier construction) — конструювання вирішального правила, на підставі якого здійснюється класифікація.

4. Оцінка якості класифікації (classifier estimation) — обчислення показників правильності класифікації.

На практиці якість класифікації часто характеризують такими показниками як точність, чутливість, специфічність тощо. Припустимо, що тестова вибірка складається з P об'єктів класу A (позначення P означає positive – позитивні об'єкти) і N об'єктів класу B (позначення N означає negative – негативні об'єкти). Якщо об'єкт, про який заздалегідь відомо, що він належить до класу A, класифікується як позитивний, то результат називається *істинно позитивним*. Якщо об'єкт, про який заздалегідь відомо, що він належить до класу B, класифікується як негативний, результат називається *істинно негативним*. Відповідно, помилкові результати класифікації називаються *хибно позитивними* і *хибно негативними*. Кількість істинно позитивних результатів позначається як TP, кількість істинно негативних результатів як TN, кількість хибно позитивних результатів як FP і кількість хибно негативних результатів як FN.

Чутливість, або TPR (true positive rate), це доля істинно – позитивних результатів серед усіх позитивних результатів, тобто $TPR = TP / (TP + FN) = TP / P$.

Специфічність, або TNR (true negative rate), це доля істинно негативних результатів серед усіх негативних результатів, тобто $TNR = TN / (TN + FP) = TN / N$.

Точність – це доля правильних результатів серед усіх результатів класифікації, тобто $PR = (TP + TN) / (P + N)$.

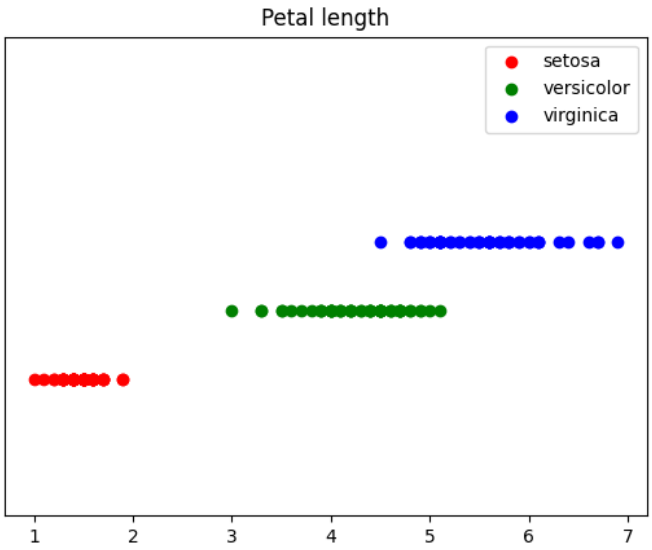
Якщо долю хибно позитивних результатів позначити як $FPR = FP / N$, то легко бачити, що $FPR + TNR = 1$.

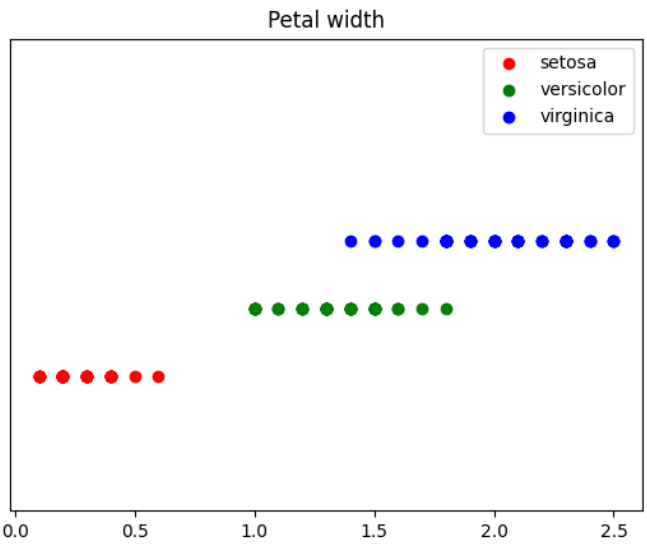
Крива залежності специфічності TPR від величини 1 – TNR при варіюванні параметрів вирішальної функції, називається ROC – кривою. Ця крива характеризує якість бінарної класифікації і називається також кривою помилок, а її аналіз називається *ROC* – аналізом.

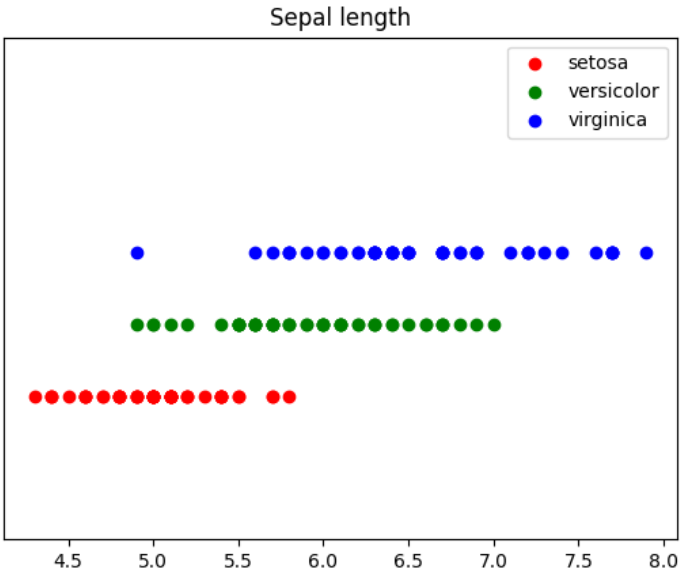
Якісна інтерпретація *ROC* – кривої виражається площею, обмеженою *ROC* – кривою і віссю, на якій відкладаються долі хибних позитивних результатів. Цей показник називається *AUC* (area under curve – площа під кривою). Якісні класифікатори мають більш високий показник *AUC*. Якість класифікатора в залежності від значення *AUC* визначається так: від 0.9 до 1.0 – відмінна, від 0.8 до 0.9 – дуже добра, 0.7 – 0.8 – добра, 0.6 – 0.7 – задовільна, 0.5 – 0.6 – незадовільна (де факто, випадковий результат).

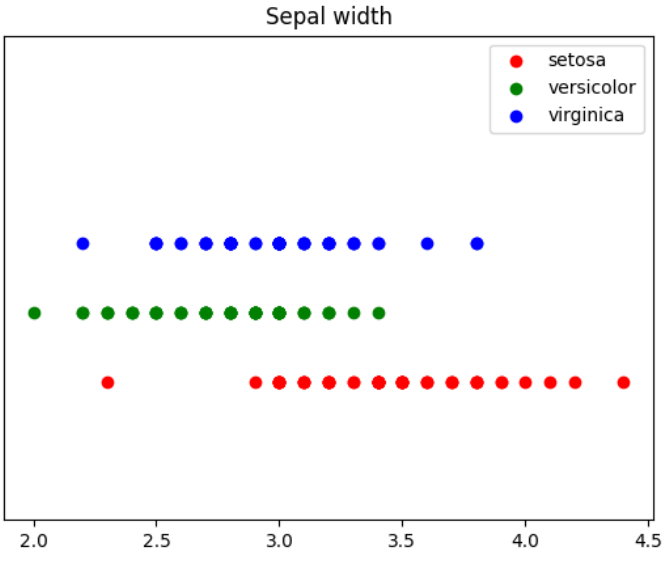
3. Результати

Для початку покажемо розподіл класів по кожній характеристиці окремо, щоб зрозуміти, які характеристики найкраще відокремлюють класи і як класи накладаються. В нашому випадку є три класи: *setosa, versicolor, virginica*. Ці класи мають 4 характеристики: *sepal length, sepal width, petal length, petal width.*

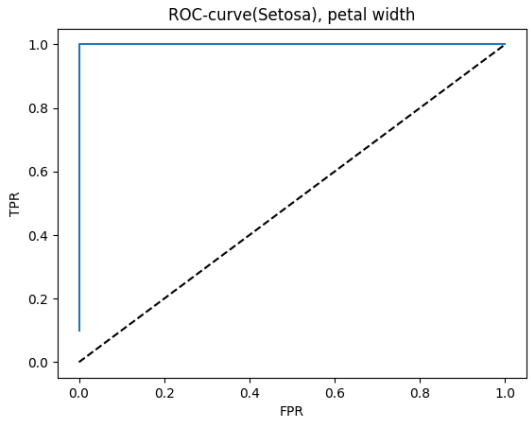


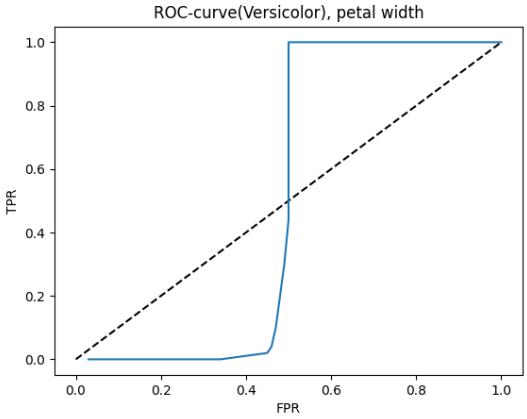


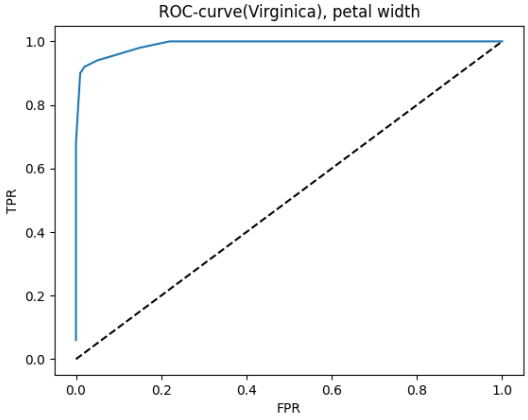


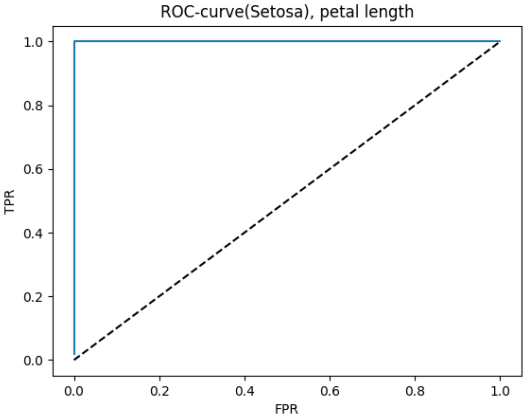


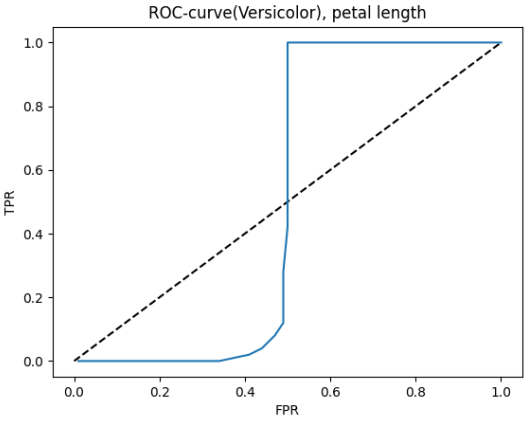
Далі будуть зображені всі ROC – криві для кожного класу і характеристики.

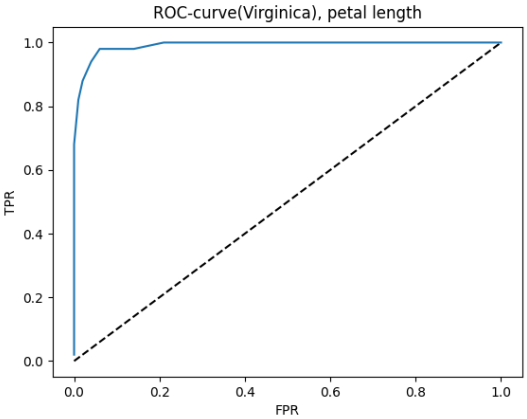

ROC-curve(Setosa), petal width
TPR
FPR
0.0
0.2
0.4
0.6
0.8
1.0

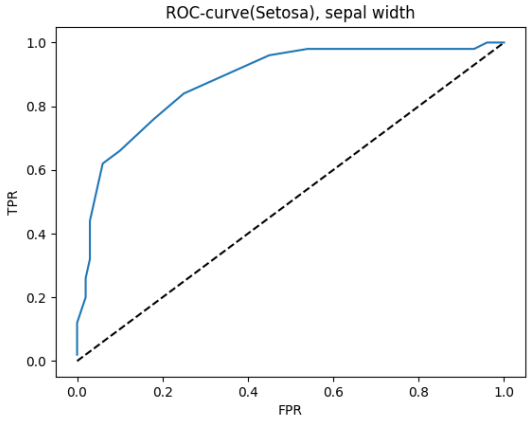

ROC-curve(Versicolor), petal width
TPR
FPR
0.0
0.2
0.4
0.6
0.8
1.0

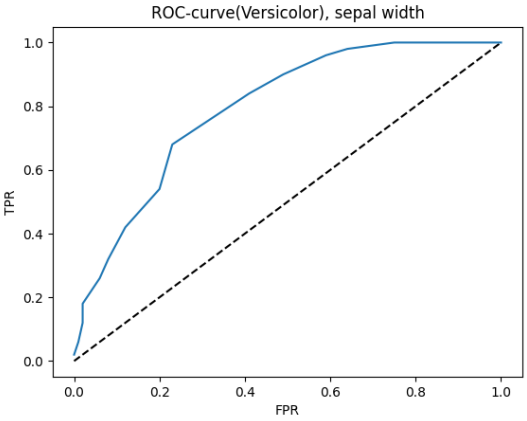

ROC-curve(Virginica), petal width
TPR
FPR
0.0
0.2
0.4
0.6
0.8
1.0

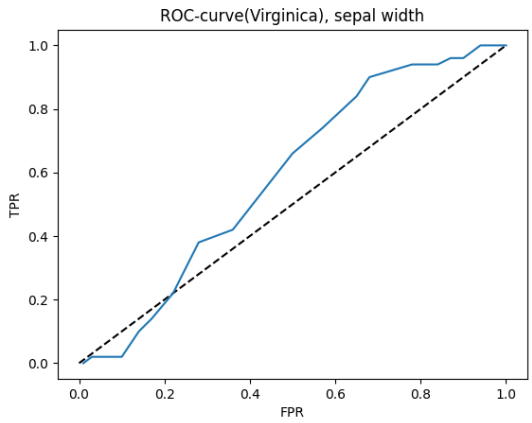

ROC-curve(Setosa), petal length
TPR
FPR
0.0
0.2
0.4
0.6
0.8
1.0

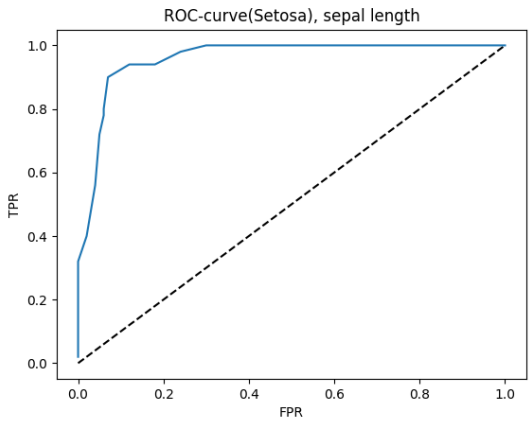

ROC-curve(Versicolor), petal length
TPR
FPR
0.0
0.2
0.4
0.6
0.8
1.0

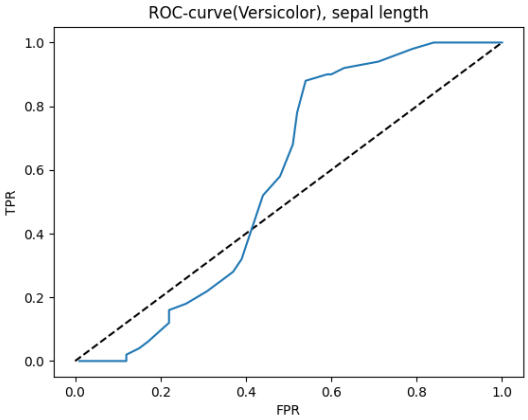

ROC-curve(Virginica), petal length
TPR
FPR
0.0
0.2
0.4
0.6
0.8
1.0

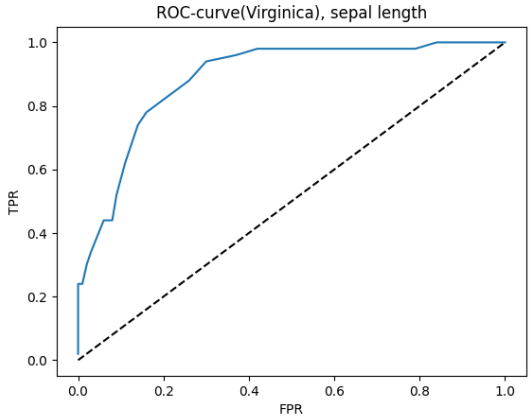

ROC-curve(Setosa), sepal width
TPR
FPR
0.0
0.2
0.4
0.6
0.8
1.0


ROC-curve(Versicolor), sepal width
TPR
FPR
0.0
0.2
0.4
0.6
0.8
1.0









З цих рисунків можна зробити висновок, що чим більш відокремлені класи за певною характеристикою, тим більше значення AUC, а отже класифікація за порогом в такому випадку є доволі ефективною.

Побудова розподілів по окремим характеристикам

```
import matplotlib.pyplot as plt
import numpy as np
from sklearn import datasets

iris = datasets.load_iris()

data = iris.data
target = iris.target
target_names = iris.target_names

setosa = data[target == 0]
versicolor = data[target == 1]
virginica = data[target == 2]

setosa_sepal_length = setosa[:, 0]
setosa_sepal_width = setosa[:, 1]
setosa_petal_length = setosa[:, 2]
setosa_petal_width = setosa[:, 3]

versicolor_sepal_length = versicolor[:, 0]
versicolor_sepal_width = versicolor[:, 1]
versicolor_petal_length = versicolor[:, 2]
versicolor_petal_width = versicolor[:, 3]

virginica_sepal_length = virginica[:, 0]
virginica_sepal_width = virginica[:, 1]
virginica_petal_length = virginica[:, 2]
virginica_petal_width = virginica[:, 3]

'''lenght'''

x_1 = setosa_sepal_length
y_1 = [1 for i in range(50)]

x_2 = versicolor_sepal_length
y_2 = [1.01 for i in range(50)]

x_3 = virginica_sepal_length
y_3 = [1.02 for i in range(50)]

plt.scatter(x_1, y_1, label='setosa', color='red')
plt.scatter(x_2, y_2, label='versicolor', color='green')
plt.scatter(x_3, y_3, label='virginica', color='blue')
plt.yticks([])
plt.ylim(0.98, 1.05)
plt.legend()
plt.title('Sepal length')
plt.show()

'''width'''

x_1 = setosa_sepal_width
y_1 = [1 for i in range(50)]

x_2 = versicolor_sepal_width
y_2 = [1.01 for i in range(50)]

x_3 = virginica_sepal_width
y_3 = [1.02 for i in range(50)]
```

```
plt.scatter(x_1, y_1, label='setosa', color='red')
plt.scatter(x_2, y_2, label='versicolor', color='green')
plt.scatter(x_3, y_3, label='virginica', color='blue')
plt.yticks([])
plt.ylim(0.98, 1.05)
plt.legend()
plt.title('Sepal width')
plt.show()

'''lenght'''

x_1 = setosa_petal_width
y_1 = [1 for i in range(50)]

x_2 = versicolor_petal_width
y_2 = [1.01 for i in range(50)]

x_3 = virginica_petal_width
y_3 = [1.02 for i in range(50)]

plt.scatter(x_1, y_1, label='setosa', color='red')
plt.scatter(x_2, y_2, label='versicolor', color='green')
plt.scatter(x_3, y_3, label='virginica',color='blue')
plt.yticks([])
plt.ylim(0.98, 1.05)
plt.legend()
plt.title('Petal width')
plt.show()

'''width'''

x_1 = setosa_petal_length
y_1 = [1 for i in range(50)]

x_2 = versicolor_petal_length
y_2 = [1.01 for i in range(50)]

x_3 = virginica_petal_length
y_3 = [1.02 for i in range(50)]

plt.scatter(x_1, y_1, label='setosa', color='red')
plt.scatter(x_2, y_2, label='versicolor', color='green')
plt.scatter(x_3, y_3, label='virginica', color='blue')
plt.yticks([])
plt.ylim(0.98, 1.05)
plt.legend()
plt.title('Petal length')
plt.show()
```

Побудова ROC – кривих

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn import datasets

def classifier_1(threshold, characteristic_list, classification_list,
Non_list):
    classified_positive = []
```

```
    classified_negative = []
    for i in range(len(characteristic_list)):
        if characteristic_list[i] <= threshold:
            classified_positive.append(i)
        else:
            classified_negative.append(i)
    TP = len(np.intersect1d(classified_positive, classification_list))
    P = len(classification_list)
    TPR = TP / P
    FP = len(np.intersect1d(classified_positive, Non_list))
    N = len(Non_list)
    TN = len(np.intersect1d(classified_negative, Non_list))
    FPR = FP / N
    TNR = TN / N
    return FPR, TPR

def classifier_2(threshold, characteristic_list, classification_list,
Non_list):
    classified_positive = []
    classified_negative = []
    for i in range(len(characteristic_list)):
        if characteristic_list[i] >= threshold:
            classified_positive.append(i)
        else:
            classified_negative.append(i)
    TP = len(np.intersect1d(classified_positive, classification_list))
    P = len(classification_list)
    TPR = TP / P
    FP = len(np.intersect1d(classified_positive, Non_list))
    N = len(Non_list)
    TN = len(np.intersect1d(classified_negative, Non_list))
    FPR = FP / N
    TNR = TN / N
    return FPR, TPR

data = datasets.load_iris()
X = data.data

Sepal_length = []
Sepal_width = []
Petal_length = []
Petal_width = []

for i in range(len(X)):
    Sepal_length.append(X[i][0])
    Sepal_width.append(X[i][1])
    Petal_length.append(X[i][2])
    Petal_width.append(X[i][3])

Setosa = [i for i in range(0, 50)]
Versicolor = [i for i in range(50, 100)]
Virginica = [i for i in range(100, 150)]

Non_setosa = np.hstack([Versicolor, Virginica])
Non_versicolor = np.hstack([Setosa, Virginica])
Non_virginica = np.hstack([Setosa, Versicolor])

'''Setosa petal width'''
x_1 = []
y_1 = []
tresholds_1 = np.linspace(min(Petal_width), max(Petal_width), 1000)
```

```
for i in tresholds_1:
    x, y = classifier_1(i, Petal_width, Setosa, Non_setosa)
    x_1.append(x)
    y_1.append(y)

plt.plot([0, 1], [0, 1], 'k--')
plt.plot(x_1, y_1)
plt.xlabel('FPR')
plt.ylabel('TPR')
plt.title('ROC-curve(Setosa), petal width')
plt.show()

'''Versicolor petal width'''
x_2 = []
y_2 = []
for i in tresholds_1:
    x, y = classifier_2(i, Petal_width, Versicolor, Non_versicolor)
    x_2.append(x)
    y_2.append(y)

plt.plot([0, 1], [0, 1], 'k--')
plt.plot(x_2, y_2)
plt.xlabel('FPR')
plt.ylabel('TPR')
plt.title('ROC-curve(Versicolor), petal width')
plt.show()

'''Virginica petal width'''
x_3 = []
y_3 = []
for i in tresholds_1:
    x, y = classifier_2(i, Petal_width, Virginica, Non_virginica)
    x_3.append(x)
    y_3.append(y)

plt.plot([0, 1], [0, 1], 'k--')
plt.plot(x_3, y_3)
plt.xlabel('FPR')
plt.ylabel('TPR')
plt.title('ROC-curve(Virginica), petal width')
plt.show()

'''Setosa petal length'''
x_4 = []
y_4 = []
tresholds_2 = np.linspace(min(Petal_length), max(Petal_length), 1000)
for i in tresholds_2:
    x, y = classifier_1(i, Petal_length, Setosa, Non_setosa)
    x_4.append(x)
    y_4.append(y)

plt.plot([0, 1], [0, 1], 'k--')
plt.plot(x_4, y_4)
plt.xlabel('FPR')
plt.ylabel('TPR')
plt.title('ROC-curve(Setosa), petal length')
plt.show()

'''Versicolor petal length'''
x_5 = []
y_5 = []
```

```
for i in tresholds_2:
    x, y = classifier_2(i, Petal_length, Versicolor, Non_versicolor)
    x_5.append(x)
    y_5.append(y)

plt.plot([0, 1], [0, 1], 'k--')
plt.plot(x_5, y_5)
plt.xlabel('FPR')
plt.ylabel('TPR')
plt.title('ROC-curve(Versicolor), petal length')
plt.show()

'''Virginica petal length'''
x_6 = []
y_6 = []
for i in tresholds_2:
    x, y = classifier_2(i, Petal_length, Virginica, Non_virginica)
    x_6.append(x)
    y_6.append(y)

plt.plot([0, 1], [0, 1], 'k--')
plt.plot(x_6, y_6)
plt.xlabel('FPR')
plt.ylabel('TPR')
plt.title('ROC-curve(Virginica), petal length')
plt.show()

'''Setosa sepal width'''
x_7 = []
y_7 = []
tresholds_3 = np.linspace(min(Sepal_width), max(Sepal_width), 1000)
for i in tresholds_3:
    x, y = classifier_2(i, Sepal_width, Setosa, Non_setosa)
    x_7.append(x)
    y_7.append(y)

plt.plot([0, 1], [0, 1], 'k--')
plt.plot(x_7, y_7)
plt.xlabel('FPR')
plt.ylabel('TPR')
plt.title('ROC-curve(Setosa), sepal width')
plt.show()

'''Versicolor sepal width'''
x_8 = []
y_8 = []
for i in tresholds_3:
    x, y = classifier_1(i, Sepal_width, Versicolor, Non_versicolor)
    x_8.append(x)
    y_8.append(y)

plt.plot([0, 1], [0, 1], 'k--')
plt.plot(x_8, y_8)
plt.xlabel('FPR')
plt.ylabel('TPR')
plt.title('ROC-curve(Versicolor), sepal width')
plt.show()

'''Virginica sepal width'''
x_9 = []
y_9 = []
```

```
for i in tresholds_3:
    x, y = classifier_1(i, Sepal_width, Virginica, Non_virginica)
    x_9.append(x)
    y_9.append(y)

plt.plot([0, 1], [0, 1], 'k--')
plt.plot(x_9, y_9)
plt.xlabel('FPR')
plt.ylabel('TPR')
plt.title('ROC-curve(Virginica), sepal width')
plt.show()

'''Setosa sepal length'''
x_10 = []
y_10 = []
tresholds_4 = np.linspace(min(Sepal_length), max(Sepal_length), 1000)
for i in tresholds_4:
    x, y = classifier_1(i, Sepal_length, Setosa, Non_setosa)
    x_10.append(x)
    y_10.append(y)

plt.plot([0, 1], [0, 1], 'k--')
plt.plot(x_10, y_10)
plt.xlabel('FPR')
plt.ylabel('TPR')
plt.title('ROC-curve(Setosa), sepal length')
plt.show()

'''Versicolor sepal length'''
x_11 = []
y_11 = []
for i in tresholds_4:
    x, y = classifier_2(i, Sepal_length, Versicolor, Non_versicolor)
    x_11.append(x)
    y_11.append(y)

plt.plot([0, 1], [0, 1], 'k--')
plt.plot(x_11, y_11)
plt.xlabel('FPR')
plt.ylabel('TPR')
plt.title('ROC-curve(Versicolor), sepal length')
plt.show()

'''Virginica sepal length'''
x_12 = []
y_12 = []
for i in tresholds_4:
    x, y = classifier_2(i, Sepal_length, Virginica, Non_virginica)
    x_12.append(x)
    y_12.append(y)

plt.plot([0, 1], [0, 1], 'k--')
plt.plot(x_12, y_12)
plt.xlabel('FPR')
plt.ylabel('TPR')
plt.title('ROC-curve(Virginica), sepal length')
plt.show()
```