Praktikum 7
Algoritma Klasifikasi SVM (lanjutan)

Tujuan

Setelah mengikuti praktikum ini, mahasiswa diharapkan mampu:

1. Memahami konsep klasifikasi multi kelas menggunakan SVM
2. Mengimplementasikan salah satu metode SVM untuk klasifikasi multi kelas

Dasar Teori

Algoritma klasifikasi SVM didesain untuk melakukan proses klasifikasi biner untuk data yang hanya memiliki dua kelas saja. Namun, SVM dapat dimodifikasi untuk menyelesaikan klasifikasi multi kelas dengan cara mengubah permasalahan multi kelas menjadi beberapa permasalahan klasifikasi biner. Dua metode umum untuk menyelesaikan permasalahan SVM multi kelas adalah **one-vs-one** dan **one-vs-rest**.

Metode **one-vs-rest** (atau biasa juga disebut one-vs-all atau one-against-all) membentuk sebuah model SVM untuk setiap kelas yang ada. Sebagai contoh, pada dataset Iris yang terdiri dari 3 kelas, maka akan terbentuk 3 model SVM sebagai berikut:

1. SVM1, berfungsi mengenali kelas Iris-setosa. Model ini dilatih menggunakan dataset Iris, yang mana data latih dengan kelas Iris-setosa diberi target/label 1 sedangkan kelas lainnya diberi target/label -1.
2. SVM2, berfungsi mengenali kelas Iris-versicolor. Model ini dilatih menggunakan dataset Iris, yang mana data latih dengan kelas Iris-versicolor diberi target/label 1 sedangkan kelas lainnya diberi target/label -1.
3. SVM3, berfungsi mengenali kelas Iris-virginica. Model ini dilatih menggunakan dataset Iris, yang mana data latih dengan kelas Iris-virginica diberi target/label 1 sedangkan kelas lainnya diberi target/label -1.

Strategi kedua adalah **one-vs-one** yang memecah permasalahan multi kelas menjadi permasalahan klasifikasi biner. Metode one-vs-one membentuk $K(K - 1)/2$ model, dimana K menyatakan banyaknya kelas pada data. Setiap model ditujukan untuk mengenali pasangan kelas tertentu. Dengan menggunakan contoh dataset Iris, maka akan terbentuk $3(3 - 1)/2 = 3$ model sebagai berikut :

1. SVM1 mengenali Iris-setosa dan Iris-versicolor
2. SVM2 mengenali Iris-setosa dan Iris-virginica
3. SVM3 mengenali Iris-versicolor dan Iris virginica.

Praktikum ini akan menggunakan metode one-vs-rest untuk melakukan proses klasifikasi data Iris. Metode one-vs-rest memanfaatkan klasifikasi SVM biner yang telah diimplementasikan pada praktikum sebelumnya.

Praktikum

1. Import Data

Unduh dataset yang akan digunakan pada praktikum kali ini. Anda dapat

menggunakan aplikasi wget untuk mendowload dataset dan menyimpannya dalam Google Colab. Jalankan cell di bawah ini untuk mengunduh dataset

```
! wget
https://gist.githubusercontent.com/netj/8836201/raw/6f9306ad21398e
a43cba4f7d537619d0e07d5ae3/iris.csv
```

Setelah dataset berhasil diunduh, langkah berikutnya adalah membaca dataset dengan memanfaatkan fungsi readcsv dari library pandas. Lakukan pembacaan berkas csv ke dalam dataframe dengan nama data menggunakan fungsi readcsv. Jangan lupa untuk melakukan import library pandas terlebih dahulu

```
import pandas as pd
import numpy as np
data = pd.read_csv('iris.csv')
```

Cek isi dataset Anda dengan menggunakan perintah **head()**

```
data.head()
```

2. Membagi Data Menjadi Data Latih dan Data Uji

Metode pembelajaran mesin memerlukan dua jenis data :

1. Data latih : Digunakan untuk proses training metode klasifikasi
2. Data uji : Digunakan untuk proses evaluasi metode klasifikasi

Data uji dan data latih perlu dibuat terpisah (mutualy exclusive) agar hasil evaluasi lebih akurat. Data uji dan data latih dapat dibuat dengan cara membagi dataset dengan rasio tertentu, misalnya 80% data latih dan 20% data uji.

Library Scikit-learn memiliki fungsi train_test_split pada modul model_selection untuk membagi dataset menjadi data latih dan data uji. Bagilah dataset anda menjadi dua, yaitu **data_latih** dan **data_uji**.

```
from sklearn.model_selection import train_test_split
data_latih,data_uji = train_test_split(data,test_size=0.2)
```

Parameter **test_size** pada fungsi **train_test_split** menunjukkan persentase data uji yang dihasilkan. Pada contoh ini, 20% data digunakan sebagai data uji.

Tampilkan banyaknya data pada **data_latih** dan **data_uji**. Seharusnya **data_latih** terdiri dari 120 data, dan **data_uji** terdiri dari 30 data

```
print(data_uji.shape[0])
print(data_latih.shape[0])
```

Pisahkan label/kelas dari data uji menjadi sebuah variabel bernama **label_uji**

```
label_uji = data_uji.pop('variety')
```

3. Pembentukan Data Latih one-vs-rest

Metode one-vs-rest memerlukan tiga jenis data latih yang diperlukan untuk melatih

tiga SVM yang berbeda pada dataset Iris. Fungsi **buat_trainingset** digunakan untuk membentuk tiga dataset tersebut.

```
def buat_trainingset(dataset):
  trainingset = {}
  kolom_kelas = dataset.columns[-1]
  list_kelas = dataset[kolom_kelas].unique()
  for kelas in list_kelas:
    data_temp = dataset.copy(deep=True)
    data_temp[kolom_kelas]=data_temp[kolom_kelas].map({kelas:1})
    data_temp[kolom_kelas]=data_temp[kolom_kelas].fillna(-1)
    trainingset[kelas]=data_temp
  return trainingset
```

Fungsi **buat_trainingset** menghasilkan dictionary dengan key berupa kelas-kelas yang terdapat pada data. Value pada dictionary adalah data latih dengan nilai kelas yang dimodifikasi. Kelas yang sama dengan key akan diberi nilai 1, selain itu diberi nilai -1.

Gunakan fungsi **buat_trainingset** untuk membentuk data latih dengan nama variabel **training yang akan digunakan pada proses training.

```
trainingset = buat_trainingset(data_latih)
```

Tampilkan isi trainingset agar Anda dapat memahami struktur dari variabel tersebut.

```
print(trainingset)
```

4. Pembentukan SVM Biner

Metode one-vs-rest pada SVM memanfaatkan SVM biner yang telah dipraktekkan pada praktikum sebelumnya. Pembentukan SVM biner tidak dijelaskan pada praktikum kali ini karena metodenya sama persis dengan praktikum sebelumnya.

5. Proses Training

Proses training dilakukan dengan memanggil fungsi **sgd** berulang kali sesuai banyaknya kelas yang ada pada data. Dengan demikian, proses training menghasilkan bobot sebanyak kelas yang ada pada dataset. Buatlah fungsi bernama **training** yang digunakan untuk melakukan proses training one-vs-rest.

```
def training(trainingset):
  list_kelas = trainingset.keys()
  w = {}
  for kelas in list_kelas:
    data_latih = trainingset[kelas]
    label_latih = data_latih.pop(data_latih.columns[-1])
    w[kelas] = sgd(data_latih,label_latih)
  return w
```

Hasil dari proses training berupa dictionary dengan key berupa kelas dan value berupa bobot dari SVM yang mengenali kelas tersebut.

Lakukan proses training dengan memanggil fungsi **training** dan menempatkan hasilnya pada variabel **W**

```
W = training(trainingset)
```

Tampilkan isi variabel **W**

```
print(W)
```

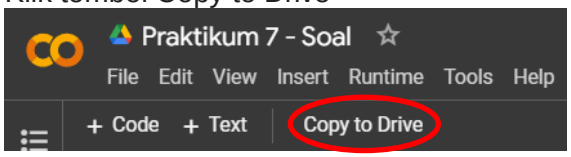
6. **Proses Testing Biner**

Proses testing biner menjadi dasar dari proses testing pada metode one-vs-rest. Proses testing biner sama persis dengan proses testing pada praktikum sebelumnya.

Tugas

Pada tugas kali ini Anda mengimplementasikan metode testing pada metode one-vs-rest. Lengkapilah kerangka source code pada notebook yang tersedia dan jawablah pertanyaan yang ada.

Petunjuk pengerjaan soal:

1. Klik link berikut, pastikan Anda login menggunakan **akun student UB**.
   https://colab.research.google.com/drive/1TjWo9T-h3Uo4Aq_B8FHF0vGfObNaSpeD?usp=sharing
2. Klik tombol Copy to Drive
   

3. Beri nama file **BAB 7-Nama-NIM.ipynb**
4. Lengkapi nama, NIM, dan kelas Anda di bawah judul bab
5. Isilah cell yang kosong
6. Download file *.ipynb dengan cara klik **File -> Download .ipynb**
7. Kumpulkan file *.ipynb ke asisten (Google Classroom)