Tugas Besar CLO 2 Simulasi Testing dan Methodhologi Penelitian

CSH4M3 - KOMPUTASI FINANSIAL

Jumat, 16 November 2018

Siti Saadah (SSD)

= Individu, Open Book and References Tracing =

= Dilarang keras melakukan tindakan plagiarisme! Jika dilakukan, maka nilai Akhir adalah E! =

Petunjuk:

- Pertanyaan sesuai dengan CLO yang hendak diujikan dengan TOTAL POIN 110.
- Jawablah semua pertanyaan di bawah ini dengan mengacu kepada 5 (lima) rangkuman paper yang telah dilakukan sebelumnya.

Nama Mahasiswa:	NIM:	Kls:	Ruang:	Nilai (Diisi Dosen):
I Putu Indra Aristya	1301154219	IF-ICM-39-01		

Salinlah pernyataan berikut:

Saya mengerjakan ujian ini dengan jujur dan mandiri. Jika saya melakukan pelanggaran, maka saya bersedia menerima sanksi.

.....

.....

Tanda Tangan Mahasiswa:

.....

Daftar Kompetensi:

[CLO 2] Memahami teori dasar penunjang financial engineering.

[CLO 3] Mampu menerapkan beberapa metode financial komputasi pada kasus nyata dalam dunia IT.

---

1. [Poin 5] Tuliskan JUDUL Tugas Besar yang kalian pilih.
   Jawab.

   Judul tugas besar yang saya ajukan adalah "*Bankcrupty Prediction using SVM and Multi-layer Perceptron*"

2. [Poin 10] Tuliskan hipotesa untuk judul yang kalian pilih.
   Jawab.

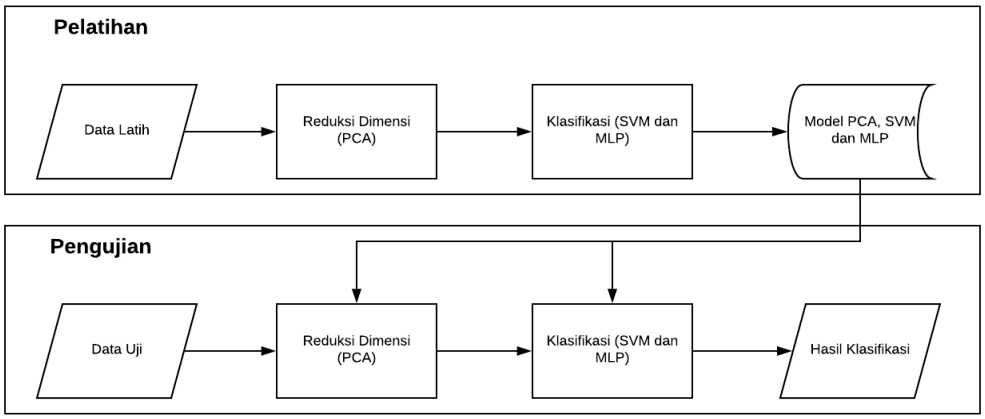
   Pada judul ini, saya menggunakan 2 metode klasifikasi untuk perbandingan, yaitu *Support Vector Machine* dan *Multi-layer Perceptron.* Menurut saya, dari kedua metode tersebut memungkinkan untuk mendapatkan akurasi lebih tinggi dengan metode MLP daripada SVM. Karena, pada penelitian ini untuk kasus SVM hanya memiliki 3 kemungkinan kernel yang cocok, yaitu *linear, rbf* atau *polynomial*. Berbeda dengan MLP yang bisa memiliki arsitektur bermacam-macam, karena banyak hal yang bisa dimodifikasi, seperti jumlah *hidden layer, neuron* dan atau *learning rate*.

3. Jelaskan rangkaian riset metodologi / implementasi desain berdasarkan poin di bawah ini.
   a. [Poin 15] Determining requirement dan Pemodelan Matematika Finansial
      Tuliskan pemodelan matematikan keuangan yang di gunakan. Kemudian tuliskan variabel penelitian / pengukuran akurasi / indikator pendukung penelitian lainnya.
      Jawab.

      Dari yang saya dapat pahami, pada tugas ini tidak ada model keuangan yang digunakan. Namun, model matematika digunakan pada PCA, SVM dan MLP. Pada SVM dicari jarak margin paling besar dengan menghitung jarak *hyperplane* ke data yang terdekat dengan *hyperplane* tersebut. Dan pada MLP, digunakan fungsi aktivasi ReLU. Rumus ReLu pada proses *feedforward* adalah $R(x) = \max(0, x)$ dan pada proses *backpropagation* adalah $R(x) = \begin{cases} 0; x < 0 \\ 1; x \geq 0 \end{cases}$.
      Pada pengukuran performansi penelitian ini menggunakan akurasi dengan rumus
      $$Akurasi = \frac{Jumlah\ data\ benar}{Jumlah\ semua\ data}$$

      Variabel yang diuji pada penelitian ini adalah kernel dari SVM yang akan dicoba (*linear, rbf* dan *polynomial*) dan arsitektur MLP serta *learning rate*-nya.

   b. [Poin 20] Designing
      Gambar dan jelaskan mengenai step-by-step Design Metodologi umum untuk penelitian yang kalian gunakan. (hint: pemodelan dapat menggunakan UML / flowchart / DFD).
      Jawab.



Gambar 1 Gambaran Program

   c. [Poin 20] Implementation dan Testing
      Paparkan dan tampilkan pengujian tahap awal (bisa dalam bentuk barisan code program atau screen shoot hasil generate tampilan tracing program).
      Jawab.

      Pada tahap pengujian, saya mencoba menggunakan data tahun pertama sebagai data latih dengan jumlah data 7027 data dan data tahun kedua sebagai data validasi dengan jumlah data 10173 data. Pada percobaan tersebut dengan *principal components* adalah 30, didapatkan hasil akurasi menggunakan MLP adalah 93.85% dan SVM Linear adalah 95,43%.

```
clf_nn = MLPClassifier(hidden_layer_sizes=(80, 50, 30))
gnb_pred = clf_nn.fit(x_train, y_train).predict(x_valid)
acc_gnb = accuracy_score(y_valid, gnb_pred)
f1_gnb = f1_score(y_valid, gnb_pred,
                  average='macro')
report_mlp = classification_report(y_valid, gnb_pred, target_names=['0','1'])

lin_clf = svm.LinearSVC()
lin_clf.fit(x_train, y_train)
predict_label = lin_clf.predict(x_valid)
accs = accuracy_score(y_valid, predict_label)
f1s = f1_score(y_valid, predict_label,
               average='macro')
report_svm = classification_report(y_valid, predict_label, target_names=['0','1'])

# s = classification_report(y_valid, gnb_pred, target_names=[0,1]))
```

Gambar 2 Potongan code program untuk klasifikasi

```
MLP
Acc:  81.95222648186376 %
F1:  49.985250737463126 %
==============================
Report
             precision    recall  f1-score   support

          0       0.97      0.84      0.90      9773
          1       0.06      0.26      0.10       400

avg / total       0.93      0.82      0.87     10173
```

Gambar 3 Hasil akurasi dengan menggunakan MLP

```
SVM Linear
Acc:  94.93757986827877 %
F1:  50.20707077187146 %
==============================
Report
             precision    recall  f1-score   support

          0       0.96      0.99      0.97      9773
          1       0.06      0.02      0.03       400

avg / total       0.93      0.95      0.94     10173
```

Gambar 4 Hasil akurasi dengan menggunakan SVM Linear

Kemudian, dicoba juga dengan menggunakan data tahun pertama saja (membagi menjadi 6000 data latih dan 1027 dengan *cross validation* sebanyak 3 pembagian, didapatkan hasil rata – rata akurasi adalah sebesar 90.18% menggunakan MLP dan 95.55% menggunakan SVM Linear.

```
MLP
Acc:  [0.894534585824082, 0.9474807856532877, 0.8825789923142613]
Mean Acc:  0.9081981212638771
==============================
F1:  [0.49889769863730427, 0.49451019960517656, 0.5177804648864364]
Mean F1:  0.5037294543763057
```

Gambar 5 Hasil rata - rata akurasi menggunakan MLP

```
SVM Linear
Acc:  [0.9491887275832621, 0.955593509820666, 0.96199829205807]
Mean Acc:  0.955593509820666
==============================
F1:  [0.48696604600219057, 0.5071548821548821, 0.49031556039173013]
Mean F1:  0.49481216284960094
```

Gambar 6 Hasil rata - rata akurasi menggunakan SVM Linear

d. Data Collecting

a. [Poin 20] Sebutkan tata cara pre-processing data (data preparation-data validation) dalam studi kasus komputasi finansial yang kalian angkat.
Jawab.

Data yang saya dapatkan adalah data dengan 64 atribut termasuk 1 atribut kelas (bangkrut atau tidak). Pada data tersebut terdapat beberapa atribut yang tidak memiliki nilai, sehingga nilai pada atribut tersebut saya isi dengan nilai 0. Setelah itu, data tersebut langsung saya reduksi dimensinya menggunakan PCA, dan hasil tersebut saya klasifikasi menggunakan SVM dan MLP. Saya menggunakan 2 cara pada pembagian data validasi, yaitu data validasi diperoleh dari data latih yang saya bagi secara manual seberapa banyak dan dengan *cross validation*.

b. [Poin 20] Kemudian berikan tampilan hasil generate code-program per fungsi/procedure yang kalian dapatkan, sehingga hasil per langkah dari desain implementasi dapat terlihat.
Jawab.

Code program saya lampirkan di halaman belakang.

```python
from scipy.io import arff
import numpy as np
from sklearn.metrics import accuracy_score, recall_score, precision_score, f1_score, classification_report
from sklearn.neural_network import MLPClassifier
from sklearn import svm
from sklearn.decomposition import PCA


data_path = './data/1year.arff'
data_path2 = './data/2year.arff'
data_path3 = './data/3year.arff'
data_path4 = './data/4year.arff'
data_path5 = './data/5year.arff'


data, meta = arff.loadarff(data_path)
data2, meta2 = arff.loadarff(data_path2)
data3, meta3 = arff.loadarff(data_path3)
data4, meta4 = arff.loadarff(data_path4)
data5, meta5 = arff.loadarff(data_path5)

# Menggunakan data tahun pertama sebagai data latih dan data tahun kedua sebagai data validasi
arr = np.array(data)
randomize = np.arange(len(arr))
np.random.shuffle(randomize)
arr = arr[randomize].tolist()

arr_v = np.array(data2)
randomize = np.arange(len(arr_v))
np.random.shuffle(randomize)
arr_v = arr_v[randomize].tolist()

train_data = arr
valid_data = arr_v

x_train = []
y_train = []
x_valid = []
y_valid = []

# Memisahkan atribut (X) dengan label atau kelas dari setiap data (y)
for i in range(0,len(train_data)):
    y = np.array(train_data[i],dtype='float32')
    y = np.nan_to_num(y).tolist()
    y_train.append(str(y[64]))
    y.pop()
    x_train.append(y)
x_train = np.array(x_train)

for i in range(0,len(valid_data)):
    y = np.array(valid_data[i],dtype='float32')
    y = np.nan_to_num(y).tolist()
    y_valid.append(str(y[64]))
    y.pop()
    x_valid.append(y)
x_valid = np.array(x_valid)
```

```python
# Mereduksi dimensi atribut (X) menjadi 30 dimensi atribut
pca = PCA(n_components=30)
pca.fit(x_train)

# Mengimplementasikan hasil reduksi dimensi kepada data latih dan data validasi
x_train = pca.transform(x_train)
x_valid = pca.transform(x_valid)

# Melakukan pelatihan dan prediksi menggunakan MLP dengan 3 hidden layer dan pada setiap layer memiliki neuron sebanyak 80, 50 dan 30
clf_nn = MLPClassifier(hidden_layer_sizes=(80, 50, 30))
gnb_pred = clf_nn.fit(x_train, y_train).predict(x_valid)
acc_gnb = accuracy_score(y_valid, gnb_pred)
f1_gnb = f1_score(y_valid, gnb_pred,
                  average='macro')
report_mlp = classification_report(y_valid, gnb_pred, target_names=['0','1'])

# Melakukan pelatihan dan prediksi menggunakan SVM Linear
lin_clf = svm.LinearSVC()
lin_clf.fit(x_train, y_train)
predict_label = lin_clf.predict(x_valid)
accs = accuracy_score(y_valid, predict_label)
f1s = f1_score(y_valid, predict_label,
               average='macro')
report_svm = classification_report(y_valid, predict_label, target_names=['0','1'])

print("MLP")
print("Acc: ",str(acc_gnb*100),'%')
print("F1: ",str(f1_gnb*100),'%')
print("============================")
print("Report")
print(report_mlp)

print("SVM Linear")
print("Acc: ",str(accs*100),'%')
print("F1: ",str(f1s*100),'%')
print("============================")
print("Report")
print(report_svm)


# Cross Validation using Year 1 - Data
data, meta = arff.loadarff(data_path)
arr = np.array(data)
randomize = np.arange(len(arr))
np.random.shuffle(randomize)
arr = arr[randomize].tolist()

cross_val = 3 # isi pembagian cross validasi yang diinginkan
arr = np.array(arr).tolist()
x_train = []
y_train = []
acc = []
f1 = []
s = []
ss = []
```

```
acc_svm = []
f1_svm = []

# Memisahkan atribut (X) dengan label atau kelas dari setiap data (y)
for i in range(0,len(arr)):
    y = np.array(arr[i],dtype='float32')
    y = np.nan_to_num(y).tolist()
    y_train.append(str(y[64]))
    y.pop()
    x_train.append(y)
x_train = np.array(x_train)

x_dt = x_train
y_dt = np.array(y_train)


# Melakukan shuffle agar urutan data menjadi acak
randomize = np.arange(int(len(y_dt)))
np.random.shuffle(randomize)
x_dt = x_dt[randomize]
y_dt = y_dt[randomize]

cv = int(len(y_dt)/cross_val)

# Melakukan reduksi dimensi dengan PCA, pelatihan dan pengujian menggunakan SVM dan MLP sebanyak pembagian cross validasi
for i in range(1,cross_val+1):
    x_train = []
    y_train = []

    j = i-1
    if (i == 1):
        x_valid = x_dt[:i*cv]
        y_valid = y_dt[:i*cv]

        x_train = x_dt[i*cv:]
        y_train = y_dt[i*cv:]
    else:
        x_valid = x_dt[j*cv:i*cv]
        y_valid = y_dt[j*cv:i*cv]

        x_train = np.concatenate((x_dt[:j*cv],x_dt[i*cv:]))
        y_train = np.concatenate((y_dt[:j*cv],y_dt[i*cv:]))

    pca = PCA(n_components=30)
    pca.fit(x_train)

    x_train = pca.transform(x_train)
    x_valid = pca.transform(x_valid)

    clf_nn = MLPClassifier(hidden_layer_sizes=(80, 50, 20))
    gnb_pred = clf_nn.fit(x_train, y_train).predict(x_valid)
    acc_gnb = accuracy_score(y_valid, gnb_pred)
    f1_gnb = f1_score(y_valid, gnb_pred,
                      average='macro')
    acc.append(acc_gnb)
    f1.append(f1_gnb)
```

```
    s_mlp = classification_report(y_valid, gnb_pred, target_names=['0','1'])
    s.append(s_mlp)

    lin_clf = svm.LinearSVC()
    lin_clf.fit(x_train, y_train)
    predict_label = lin_clf.predict(x_valid)
    accs = accuracy_score(y_valid, predict_label)
    f1s = f1_score(y_valid, predict_label,average='macro')
    acc_svm.append(accs)
    f1_svm.append(f1s)
    s_svm = classification_report(y_valid, predict_label, target_names=['0','1'])
    ss.append(s_svm)

print("MLP")
print("Acc: ",str(acc))
print("Mean Acc: ",str(np.mean(acc)))
print("==============================")
print("F1: ",str(f1))
print("Mean F1: ",str(np.mean(f1)))

print("SVM Linear")
print("Acc: ",str(acc_svm))
print("Mean Acc: ",str(np.mean(acc_svm)))
print("==============================")
print("F1: ",str(f1_svm))
print("Mean F1: ",str(np.mean(f1_svm)))
```