Studi Kasus – Klasifikasi Hutan

Dr. rer. nat. Hendri Murfi

Intelligent Data Analysis (IDA) Group

Departemen Matematika, Universitas Indonesia – Depok 16424

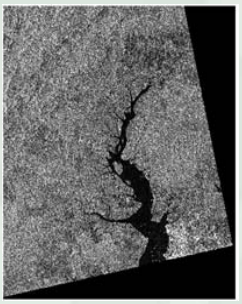
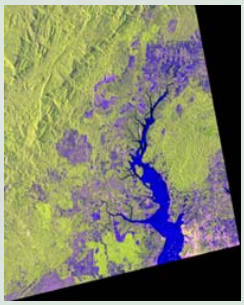
Telp. +62-21-7862719/7863439, Fax. +62-21-7863439, Email. hendri@ui.ac.id

ALOS PALSAR

- Data ALOS PALSAR adalah data berupa gambar permukaan bumi yang direkam dari satelit ALOS (*advanced land observing satellite*) dengan menggunakan sensor PALSAR (*phased arrayed L-band synthetic aperture radar*)
- Data ALOS PALSAR dapat digunakan untuk pembuatan DEM, Interferometry untuk mendapatkan informasi pergeseran tanah, kandungan biomass, monitoring kehutanan, pertanian, tumpahan minyak (oil spill), soil moisture, mineral, pencarian pesawat dan kapal yang hilang, dll.

ALOS PALSAR

- **Keistimewaan:** sensor PALSAR adalah dapat menembus awan, serta dapat digunakan baik malam maupun siang hari
- **Permasalahan:** diberikan gambar ALOS PALSAR dari suatu permukaan bumi, bagaimana kita menentukan bagian mana merupakan hutan dan non hutan [1][2] ?

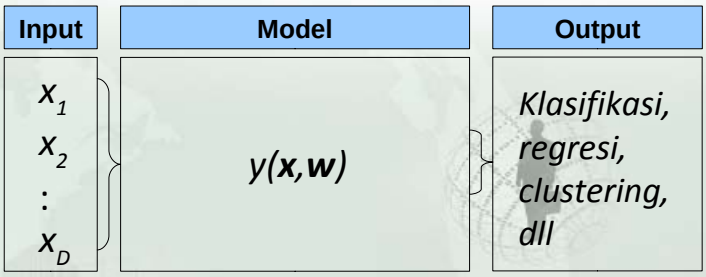
Sumber = [1][2]

Perangkat Lunak

- Perangkat lunak yang digunakan pada simulasi ini adalah *scikit-learn* [3], yaitu paket *machine learning* pada Python.
- Untuk pemodelan SVM, *scikit-learn* menggunakan pustaka *libSVM* [4].

```
from sklearn import preprocessing
from sklearn.svm import SVC
from sklearn.grid_search import GridSearchCV
from sklearn.cross_validation import StratifiedKFold
```

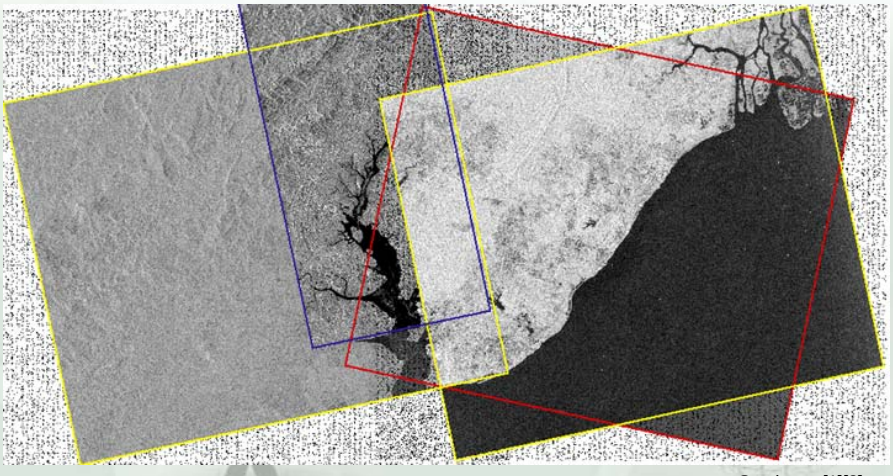
Machine Learning



- *Preprocessing*: ektraksi fitur dan representasi data, misal dalam bentuk vektor $\mathbf{x}_i = (x_1, x_2, .., x_D)^T$
- *Training*: pemilihan model dan penentuan parameter model, misal **w**, berdasarkan data pelatihan (*training data*)
- *Testing*: pengujian metode dengan data penguji (*testing data*) yang tidak sama dengan data pelatihan, sehingga didapat nilai estimasi untuk kapabilitas generalisasi dari model.

Preprocessing

Ekstraksi Fitur



Sumber = [1][2]

Keterangan:

- Biru : mode quadpol, polarisasi HH, HV, dan VV (ALOS ALPSRP222967170), Level 1,1.
- Kuning : mode FBD, polarisasi HH dan HV (ALOS ALPSRP250537160, ALPSRP253017160), level 1,5.
- Merah : mode FBS, polarisasi HH (ALOS ALPSRP136683640), level 1,5.

Preprocessing

Ekstraksi Fitur

- Fitur [1][2]:
  - Mode quadpol, polarisasi HH, HV, dan VV (ALOS ALPSRP222967170), Level 1,1

    --> quad_HH, quad_HV, quad_VV
  - Mode FBD, polarisasi HH dan HV (ALOS ALPSRP250537160, ALPSRP253017160), level 1,5.

    --> FBD2505_HH, FBD2505_HV, FBD2530_HH, FBD2530_HV
  - Mode FBS, polarisasi HH (ALOS ALPSRP136683640), level 1,5.

    --> FBS_HH
- Data [1][2]: 973
  - Hutan : 679
  - Non Hutan : 294

Preprocessing

Representasi Vektor

- Sumber data, misal dalam format *csv*, ditransformasi ke dalam bentuk vektor

```
data_file = csv.reader(open('alos_palsar.csv'))
data_file.next()
n_samples = 973
n_features = 8
feature_names = np.array(['quad_HH', 'quad_HV', 'quad_VV', 'FBD2505_HH',
'FBD2505_HV', 'FBD2530_HH', 'FBD2530_HV', 'FBS_HH'])
target_names = np.array(['hutan','non hutan'])
data = np.empty((n_samples, n_features))
target = np.empty((n_samples,), dtype=np.int)

for i, ir in enumerate(data_file):
    data[i] = np.asarray(ir[2:-1], dtype=np.int)
    target[i] = np.asarray(ir[-1], dtype=np.int)
```

Preprocessing

Scaling

- Normalisasi atau penskalaan pada masing-masing data fitur sangat direkomendasikan sebelum diproses oleh SVM, yaitu dalam interval [-1,+1] atau [0,1] [3].
  - Menghindari dominasi fitur dengan nilai besar terhadap fitur dengan nilai kecil
  - Menghindari kesulitan secara numerik selama proses perhitungan
- Kita harus menggunakan metode yang sama dalam penskalaan data training dan data testing.

Preprocessing

Scaling

- Penskalaan pada masing-masing data fitur

```
X = data
scaler = preprocessing.Scaler().fit(X)
X = scaler.transform(X)

y = target
```

Learning

Pemilihan Model : Fungsi Kernel

- Secara umum, fungsi kernel RBF direkomendasikan sebagai pilihan utama [3].

$$k(x_i, x_j) = \exp\left\{-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right\} = \exp\left\{-\gamma\|x_i - x_j\|^2\right\}, \gamma > 0$$

  - Fungsi ini dapat memetakan secara tidak linear data sampel ke ruang dimensi lebih tinggi sehingga diharapkan dapat menangani kasus dimana relasi antara fitur dan kelas adalah tidak linear .
  - Alasan kedua adalah jumlah hyperparameters yang mempengaruhi kompleksitas dari pemilihan model. Fungsi kernel polynomial dan sigmoid memiliki lebih banyak hyperparameters dari pada fungsi kernel RBF
  - Alasan berikutnya adalah fungsi kernel RBF memiliki kesulitan secara numerik yang lebih sedikit, yaitu $0 < k(i,j) \leq 1$

Learning

Pemilihan Model : *Grid-Search*

- Ada dua parameter pada SVM berbasis fungsi kernel RBF, yaitu C dan γ
- Nilai C dan γ yang terbaik untuk suatu masalah yang diberikan tidak dapat ditentukan didepan. Sehingga, pencarian parameter terbaik tersebut harus dilakukan
- *Grid-search* adalah metode pencarian nilai pasangan (C, γ) terbaik, yaitu nilai pasangan (C, γ) yang membuat SVM dapat mengklasifikasikan data testing dengan tingkat akurat terbaik

Learning

Pemilihan Model : *Grid-Search*

- Karena untuk melakukan g*rid-search* secara lengkap membutuhkan waktu yang lama, maka direkomendasikan untuk melakukannya dalam dua tahap [3], yaitu:
  - Pertama-tama lakukan *loose grid-search* pada C = 2^{-5}, 2^{-3}, ..., 2^{15} dan $\rho = 2^{-15}$, 2^{-13}, ..., 2^{3}
  - Setelah diketahui daerah „kandidat“, lakukan *fine grid-search* pada sekitar daerah „kandidat“ tersebut.

Learning

Pemilihan Model : *Cross-Validation*

- *K-fold cross-Validation* adalah prosedur yang direkomendasikan dalam pemilihan model, yaitu nilai pasangan (C, γ) [3] :
  - Bagi data menjadi *k* bagian dengan ukuran yang sama
  - Gunakan masing-masing bagian sebagai data testing, dimana k-1 bagian lainnya dijadikan sebagai data training

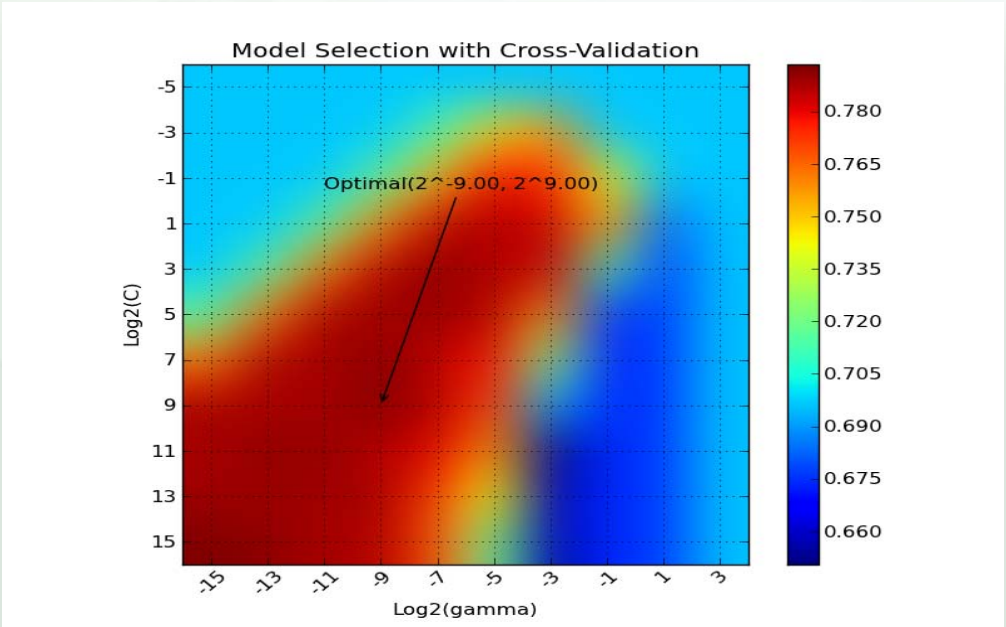
Learning

Pemilihan Model : *Loose Grid-Search* dan *Cross-Validation*

```
C_power = np.arange(-5, 16, 2)
gamma_power = np.arange(-15, 4, 2)

C_range = 2.0 ** C_power
gamma_range = 2.0 ** gamma_power

param_grid = dict(gamma=gamma_range, C=C_range)

grid = GridSearchCV(SVC(), param_grid=param_grid, cv=StratifiedKFold(y=y, k=5))
grid.fit(X, y)

best_C = np.log2(grid.best_params_.values()[0])
best_gamma = np.log2(grid.best_params_.values()[1])
```


Learning

Pemilihan Model : *Loose Grid-Search* dan *Cross-Validation*



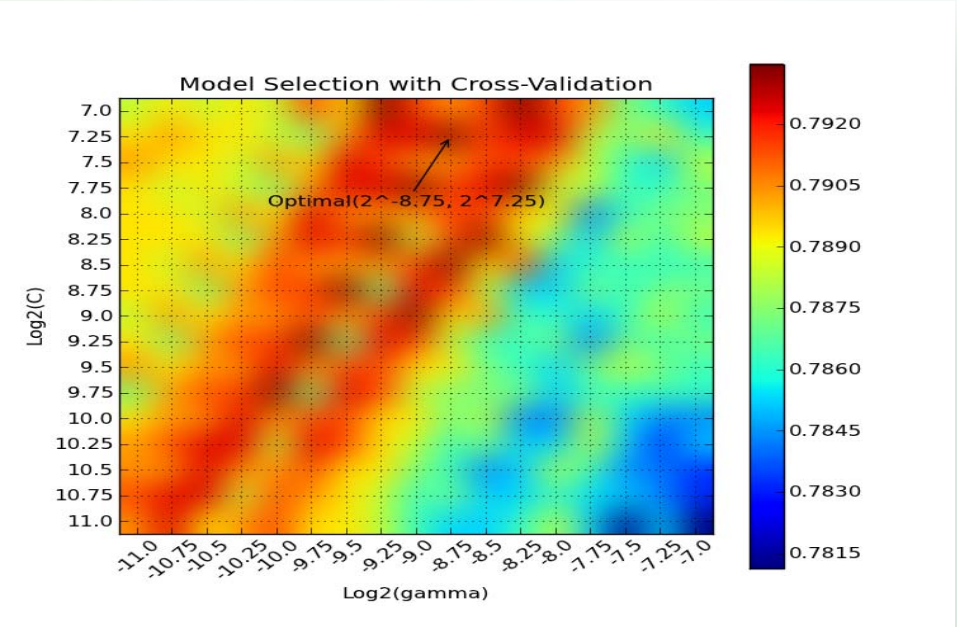
Learning

Pemilihan Model : *Fine Grid-Search* dan *Cross-Validation*

```
C_power = np.arange(7, 11.25, 0.25)
gamma_power = np.arange(-11, -6.75, 0.25)

C_range = 2.0 ** C_power
gamma_range = 2.0 ** gamma_power

param_grid = dict(gamma=gamma_range, C=C_range)

grid = GridSearchCV(SVC(), param_grid=param_grid, cv=StratifiedKFold(y=y, k=5))
grid.fit(X, y)

best_C = np.log2(grid.best_params_.values()[0])
best_gamma = np.log2(grid.best_params_.values()[1])
```


Learning

Pemilihan Model : *Fine Grid-Search* dan *Cross-Validation*



Testing

Cross-Validation

- Setelah diperoleh model yang optimal, yaitu nilai pasangan (C, γ), tahap selanjutnya adalah melakukan pelatihan ulang SVM berdasar model tersebut.
- Setelah diperoleh model SVM akhir, *k-fold cross-Validation* adalah prosedur yang juga direkomendasikan untuk mengestimasi kapabilitas generalisasi dari model tersebut [3].
  - Bagi data menjadi k bagian dengan ukuran yang sama
  - Gunakan masing-masing bagian sebagai data testing, dimana k-1 bagian lainnya dijadikan sebagai data training

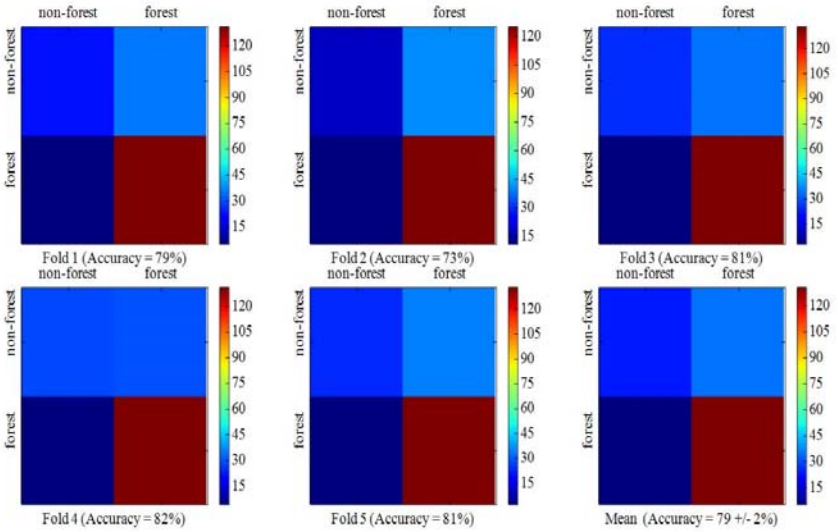
Testing

Kapabilitas Generalisasi

```
clf = SVC(kernel='rbf', C=2**12.5, gamma=2**(-9.25), probability=True)
cv = cross_validation.StratifiedKFold(y, 5)

# Compute Cross Validation Scores
# -------------------------------
scores = cross_validation.cross_val_score(clf, X, y, cv=cv)
print "Cross Validation per Fold : %s" % scores
print "Cross Validation Accuracy : %0.2f +/- %0.2f" % (scores.mean(), scores.std()/2)
```

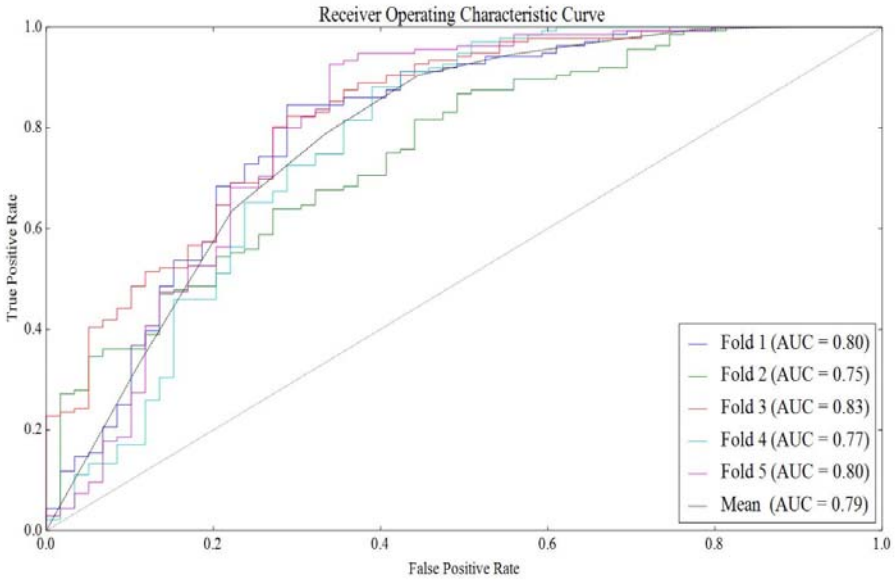
Testing

Kapabilitas Generalisasi : *Confusion Matrix*



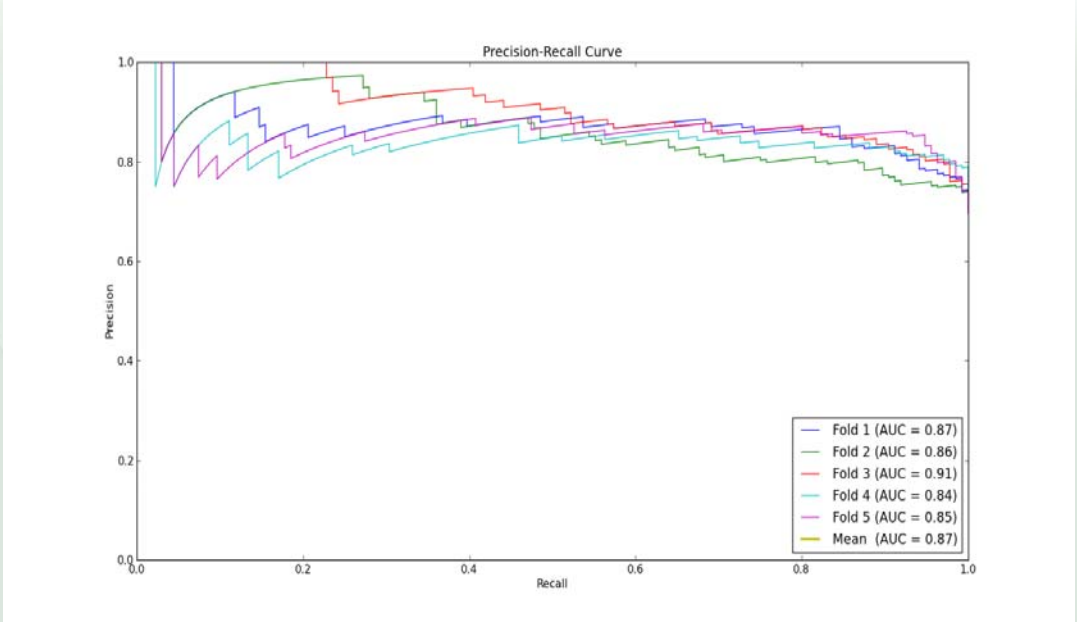
Testing

Kapabilitas Generalisasi : *ROC Curve*



Testing

Kapabilitas Generalisasi : *Precission-Recall Curve*



Referensi

(1) Rokhmatuloh, H. Murfi, R. Tateishi. *Support Vector Machine for Forest Cover Change Identification Derived from Microwave data*. The 33rd Asian Conference on Remote Sensing, Pattaya, Thailand, 2012

(2) Rokhmatuloh, H. Murfi, Ardiansyah. *A Method to Derive Optimal Decision Boundary in SVM for Forest and non-Forest Classification in Indonesia*. The 34th Asian Conference on Remote Sensing, Denpasar, Indonesia, 2013

(3) Pedregosa, et al. Scikit-learn: machine learning in python. *Journal Machine Learning Research, vol*. 12, pp. 2825-2830, 2011.

(4) C.-C. Chang and C.-J. Lin. LIBSVM : a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1--27:27, 2011

(5) C.-W. Hsu, C.-C. Chang, C.-J. Lin. *A practical guide to support vector classification.* http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf, April 5, 2013