Pengembangan Perangkat Lunak Prediktor Kanker Payudara Menggunakan Metode *Elastic* SCAD SVM dan Data DNA *Microarray*

R. Risky Dwi Listyo Firmansyah, Handayani Tjandrasa dan Isye Arieshanti
Jurusan Teknik Informatika, Fakultas Teknologi Informasi, Institut Teknologi Sepuluh Nopember (ITS)
Jl. Arief Rahman Hakim, Surabaya 60111
E-mail: handatj@its.ac.id

***Abstrak*—Kanker payudara (*Carcinoma Mammae*) merupakan salah satu penyakit kanker dengan angka kematian terbesar di dunia. Prediksi kanker payudara tentunya dapat membantu para penderitanya untuk menghindari berbagai akibat negatif yang dapat ditimbulkannya. Di sisi lain, data DNA *Microarray* ternyata dapat digunakan untuk diagnosa dini penyakit kanker payudara. Data DNA *Microarray* mengandung informasi dari DNA yang kemudian direpresentasikan dalam data vektor berdimensi tinggi. Untuk menangani permasalahan prediksi data berdimensi tinggi, *Support Vector Machine* (SVM) adalah salah satu metode yang cukup handal. Namun, sayangnya SVM tidak dapat mendukung proses seleksi fitur. Padahal, dengan adanya seleksi fitur, proses prediksi data dapat berjalan lebih cepat. Informasi tentang fitur-fitur penting dari suatu data juga dapat diperoleh dengan adanya seleksi fitur. Oleh karena itu, ada sebuah studi lain yang menggabungkan SVM dengan *elastic* SCAD (*penalization method*). Pada studi ini dikembangkan perangkat lunak untuk memprediksi kanker payudara berdasarkan model *elastic* SCAD SVM yang telah diusulkan oleh studi lain tersebut. Berdasarkan uji coba, perangkat lunak yang dikembangkan mampu melakukan prediksi kanker payudara. Hal ini ditunjukkan dengan nilai akurasi sebesar 95,4%. Fitur yang terpakai pun berkurang dari 1213 atribut menjadi 1193 atribut.**

***Kata Kunci*—DNA *Microarray*, *Elastic SCAD*, Kanker Payudara, Prediksi, *Support Vector Machine*.**

I. PENDAHULUAN

DIAGNOSA dini penyakit kanker payudara penting dilakukan mengingat pertumbuhan penderitanya yang semakin tinggi setiap tahunnya. Diagnosa penyakit kanker payudara biasanya diperoleh dari data citra ultrasonografi organ payudara tersebut. Namun, diagnosis berdasarkan citra tersebut menunjukkan bahwa sudah terdapat daging tumbuh berupa tumor bagi penderitanya. Jika demikian, penderita tersebut dapat dikatakan telah terjangkit penyakit kanker payudara stadium lanjut dan penanganannya sedikit terlambat. Oleh karena itu, diperlukan suatu cara baru untuk melakukan diagnosa dini tehadap penyakit kanker payudara ini.

DNA *Microarray* merupakan sebuah teknologi yang menampung hasil transkripsi dari *messenger*-RNA[1]. *Messenger*-RNA sendiri merupakan hasil dari transkripsi DNA. Dan DNA itu mengandung sifat dan informasi suatu makhluk hidup. Van't Veer[2] pun mengatakan bahwa kanker payudara dapat diprediksi dari data ekspresi DNA, sehingga data DNA *Microarray* ini merupakan salah satu alternatif yang dapat digunakan untuk diagnosa dini penyakit kanker payudara.

Data DNA *Microarray* yang kemudian direpresentasikan dalam bentuk vektor, merupakan sebuah data berdimensi tinggi. Permasalahan prediksi data berdimensi tinggi ini dapat ditangani dengan baik oleh metode *Support Vector Machine*. Namun, dalam beberapa kasus tidak hanya diperlukan hasil yang akurat saja, tetapi juga informasi tentang fitur-fitur mana sajakah yang berpengaruh terhadap model prediksi. Itulah salah satu fungsi dari proses seleksi fitur. Selain itu, seleksi fitur juga dapat mempercepat proses komputasi prediksi data[3]. Namun, sayangnya SVM tidak dapat melakukan hal itu.

Penggabungan SVM dengan proses seleksi fitur akan sangat berguna ke depannya. Salah satunya adalah *Penalized* SVM, yang merupakan gabungan antara metode SVM dan *penalization method* yang mendukung proses seleksi fitur. Beberapa fungsi yang merupakan *penalization method*, antara lain: LASSO, *Ridge Penalty*, dan SCAD. *Elastic* Net merupakan kombinasi dari fungsi *Ridge Penalty* dan LASSO. *Elastic* SCAD merupakan kombinasi dari fungsi *Ridge Penalty* dan SCAD. Kemudian, studi yang dilakukan Becker dan rekan mengusulkan model yang menggabungkan metode prediksi SVM dan fungsi *elastic* SCAD yang disebut dengan *elastic* SCAD SVM[3]. Dalam studi ini, penulis mengembangkan perangkat lunak berdasarkan model yang diajukan oleh Becker dan rekan tersebut[3]. Perangkat lunak dibangun dalam bahasa pemrograman yang bersifat *open source*, yaitu *Java*.

Sistematika penulisan artikel ini dapat dijelaskan sebagai berikut. Bab 2 akan membahas tentang metodologi yang digunakan artikel ini, yaitu *elastic* SCAD SVM. Kemudian akan dibahas mengenai desain dari perangkat lunak pada bab 3 dan dilanjutkan dengan implementasinya pada bab 4. Pengujian terhadap perangkat lunak yang dibuat akan dibahas pada bab 5. Kemudian, selanjutnya akan didapatkan kesimpulan-kesimpulan dari pengujian terhadap perangkat lunak yang telah dibuat.

II. METODE ELASTIC SCAD DENGAN SUPPORT VECTOR MACHINE

A. Support Vector Machine

Konsep dasar *Support Vector Machine* adalah mencari *hyperplane* terbaik dengan nilai margin tertinggi. *Hyperplane* adalah garis batas pemisah antara posisi data-data berlainan kelas. Margin adalah jarak antara *hyperplane* dengan data terdekat dari masing-masing kelas. Data yang terletak pada margin disebut sebagai *support vector*[4].

Diketahui sebuah permasalahan prediksi dengan data atribut $x_i \in R^p$ dan kelas label $y_i \in \{-1,+1\}$, $i = 1,...,n$. Fungsi *hyperplane* SVM didefinisikan sebagai berikut[4],

$$f(x) = w.x + b = 0 . \quad (1)$$

Nilai $w = (w_1, w_2, ..., w_p)$ merupakan koefisien vektor hyperplane dengan batasan $\|w\|_2 = 1$ dan nilai *b* adalah nilai bias vektor[4]. Mencari nilai margin maksimum sama dengan mencari nilai selisih garis batas data kelas +1 dan -1. Dengan memasukkan nilai +1 dan -1 pada (1), maka didapatkan nilai selisih sebesar $\frac{2}{\|w\|}$. Memaksimalkan nilai $\frac{2}{\|w\|}$ sama dengan[4],

$$\min \tfrac{1}{2}\|w\|^2 \quad (2)$$

dengan batasan $y_i(b + w.x_i) \geq 1$.

Persamaan (2) dapat diselesaikan dengan menggunakan teknik optimisasi *convex* yang disebut *langrange multiplier*. Teknik optimisasi ini menghasilkan solusi unik dari parameter *w*[4],

$$\hat{w} = \sum_{i=1}^{n} \sim_i . y_i . x_i \quad (3)$$

dengan batasan $\sum_{i=1}^{n} \sim_i . y_i . = 0$.

B. Elastic SCAD

Elastic SCAD merupakan kombinasi dari *Ridge Penalty* dan SCAD. SCAD sendiri merupakan perkembangan dari LASSO. Untuk nilai data yang kecil, SCAD menyusutkan nilainya hingga 0. Sebaliknya, SCAD memberlakukan nilai *penalty* untuk nilai data yang besar[5].

$$pen_{\alpha}(w) = \sum_{j=1}^{p} \circ scad_{\alpha}(w_j) \quad (4)$$

$$\rho scad_{\lambda}(w_j) = \begin{cases} \lambda.|w_j| & if |w_j| \leq \lambda \\ -\dfrac{|w_j|^2 - 2.a.\lambda.|w_j| + \lambda^2}{2.(a-1)} & ,if\ \lambda < |w_j| \leq a.\lambda \\ \dfrac{(a+1).\lambda^2}{2} & ,if |w_j| > a.\lambda \end{cases} .$$

Dalam (4) terdapat dua buah nilai parameter *tuning*, $a > 2$ dan $\alpha > 0$. SCAD *penalty* tidak dapat menentukan nilai dari parameter *tuning* yang paling optimal, sehingga peneletian yang telah dilakukan sebelumnya menyebutkan bahwa nilai terbaik dari paramater $a = 3{,}7$ [5].

Sedangkan, *Ridge Penalty* tidak menyusutkan nilai data hingga bernilai 0. *Ridge Penalty* berfungsi untuk mengontrol nilai varians dari data yang dimasukkan[3].

$$pen_{\alpha}(w) = \alpha . \sum_{j=1}^{p} w_j^2 . \quad (5)$$

Elastic SCAD SVM

Bentuk optimasi dari *Support Vector Machine* ekivalen dengan permasalahan *penalization menthod* yang memiliki nilai *loss and penalty*[3].

$$\min_{b,w} \tfrac{1}{n} \sum_{i=1}^{n} l(y_i, f(x_i)) + pen_{\alpha}(w) . \quad (6)$$

Nilai *loss* didefinisikan dari bentuk penjumlahan fungsi *hinge loss* $l(y_i, f(x_i)) = [1 - y_i . f(x_i)]_+ = \max(1 - y_i . f(x_i), 0)$. Jadi, bentuk optimisasi *elastic* SCAD SVM adalah dengan memasukkan (1), (4), dan (5) pada (6)[3].

$$\min_{b,w} \tfrac{1}{n} \sum_{i=1}^{n} [1 - y_i . f(x_i)]_+ + \sum_{j=1}^{p} \circ scad_{\alpha_1}(w_j) + \alpha_2 . \|w\|^2 . \quad (7)$$

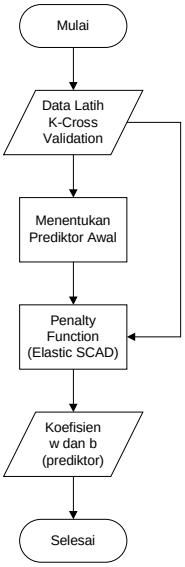
Bentuk optimisasi (7) dapat diselesaikan dengan bantuan fungsi kuadratik dan meminimalisasi nilainya dapat dilakukan secara iteratif. Setelah dilakukan penyederhanaan fungsi, maka (7) dapat dijabarkan kembali sebagai berikut[3],

$$A(b,w) \approx -\frac{1}{2n} \sum_{i=1}^{n} y_i (b + w.x_i) + \frac{1}{2n} \sum_{i=1}^{n} \frac{y_i (b + w.x_i)}{|y_i - (b_0 + w_0.x_i)|} + \frac{1}{4n} \sum_{i=1}^{n} \frac{(b + w.x_i)^2}{|y_i - (b_0 + w_0.x_i)|} + \sum_{j=1}^{p} \frac{\rho scad'_{\lambda 1}(|w_{jo}|)}{2.|w_{jo}|} . (w_j^2) + \sum_{j=1}^{p} \lambda_2 w_j^2 . \quad (8)$$

dengan nilai $\alpha_1, \alpha_2 > 0$ adalah parameter *tuning*.

III. RANCANGAN PERANGKAT LUNAK

Dalam pengembangan perangkat lunak ini, penulis membuat dua modul utama, yaitu modul latih dan modul uji. Modul latih merupakan proses pembelajaran untuk mendapatkan model klasifikasi terbaik berupa nilai vektor *w* dan nilai bias *b*. Model ini nantinya akan digunakan pada modul uji. Hasil dari modul uji kemudian dihitung performanya.



Gambar 1. Gambar runtutan proses modul latih yang dilakukan oleh perangkat lunak.

Untuk menjalankan perangkat lunak ini, maka data DNA *Microarray* dibagi menjadi data latih dan data uji melalui proses *cross validation*. Proses ini dijalankan supaya data menyebar secara merata dan model yang terbentuk tidak ketat terhadap tipe data tertentu. Penggunaan nilai *k-fold* pada proses *cross validation* yang disarankan oleh artikel ini adalah *k=5* atau *k=10*. Data latih merupakan data acuan untuk proses terbentuknya model prediksi, sedangkan data uji merupakan data yang digunakan untuk menguji model tersebut.

A. *Modul Latih*

Proses berjalannnya modul latih dapat dilihat pada gambar 1. Proses modul latih dimulai dengan menentukan nilai prediktor awal w_0 dan b_0. Proses penentuan nilai prediktor awal ini menggunakan bantuan *library libsvm*[6] yang dapat diunduh gratis di http://www.csie.ntu.edu.tw/~cjlin/LibSVMtools. Untuk mendapatkan nilai w_0 dan b_0, tipe SVM yang digunakan adalah *C_SVC* dan fungsi yang dijalankan pada *library libsvm* adalah *svm_train.main()* dengan parameter tipe kernel *linear*. Fungsi ini akan menghasilkan sebuah *file* yang di dalamnya terdapat informasi-informasi mengenai *hyperplane* yang terbentuk. Nilai b_0 didapatkan dari nilai angka berlabel *rho*. Dan nilai vektor w_0 didapatkan dengan menyelesaikan (3). Data yang dibutuhkan untuk menyelesaikan (3) ada pada *file* tersebut.

Nilai w_0 dan b_0 yang dihasilkan pada proses sebelumnya, kemudian menjadi data masukan untuk proses *elastic* SCAD. Inti dari proses *elastic* SCAD SVM adalah menyelesaikan (8). Untuk menuliskan (8) ke dalam bahasa pemrograman, maka diperlukan pendefinisian beberapa matriks terlebih dahulu[3].

$y = [y_1,...,y_n]^T$, $w = [w_1,...,w_p]^T$,

$£ = [£_1,...,£_n]^T$, dengan nilai $£_i = y_i - (b_0 + w_0.x_i)$,

$$X = \begin{bmatrix} 1 & x_{11} & x_{np} \\ 1 & x_{21} & x_{np} \\ 1 & x_{n1} & x_{np} \end{bmatrix},\ r = \left[\frac{y_i}{|£_i|},...,\frac{y_n}{|£_n|} \right]^T,$$

$$D_0 = \tfrac{1}{2n} diag\left[\tfrac{1}{|£_1|},...,\tfrac{1}{|£_n|}\right],$$

$$Q_1 = diag\left[0, \frac{°scad^{a}1(w_{10})}{|w_{10}|},...,\frac{°scad^{a}1(w_{p0})}{|w_{p0}|}\right],$$

$$Q_2 = diag[0,2^{a}{}_1,...,2^{a}{}_2],$$

$$P = \tfrac{1}{2n}(y + r)^T.X$$

$$Q = X^T D_0.X + Q_1 + Q_2.$$

Nilai *w* dan *b* didapatkan dengan menyelesaikan persamaan berikut ini.

$$Q\begin{pmatrix} \hat{b} \\ \hat{w} \end{pmatrix} = P^T. \quad (9)$$

Pencarian nilai *w* dan *b* akan berjalan secara iteratif hingga perangkat lunak memenuhi kondisi-kondisi berikut ini:

1) Konvergen.
   Perangkat lunak dianggap konvergen ketika selisih nilai prediktor yang dihasilkan dengan prediktor sebelumnya mendekati 0.
2) Ditemukannya nilai *NaN*.
3) Semua prediktor *w* bernilai 0.
4) Iterasi melebihi batas iterasi maksimal.

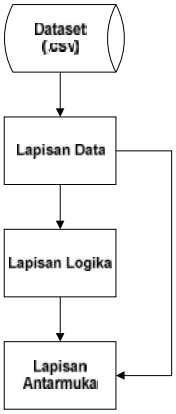
Ketika perangkat lunak memenuhi kondisi-kondisi di atas, maka nilai *w* dan *b* yang diambil adalah nilai *w* dan *b* yang didapatkan pada iterasi sebelumnya.

B. *Modul Uji*

Modul latih menghasilkan data keluaran sebuah model prediksi berupa nilai *w* dan *b*. Model inilah yang kemudian diujikan terhadap data uji yang telah dibuat sebelumnya.

Pengujian terhadap model prediksi menggunakan fungsi *classifier* SVM. Fungsi *classifier* SVM mencari nilai *significant* dari (1). Jika nilai yang dihasilkan dari (1) positif, maka kelas prediksi dari data tersebut adalah +1. Begitu juga sebaliknya[4].

Setelah semua data diujikan terhadap model, kemudian dihitung nilai performanya. Penghitungan yang dilakukan meliputi nilai *sensitivity*, *specificity*, dan *youden index*. *Youden index* merupakan hasil penjumlahan nilai *sensitivity* dan *specificity*, kemudian dikurangkan dengan 1. Hal ini membuat nilai *youden index* sangat ketat. Ketimpangan hasil antara *sensitivity* dan *specificity* membuat nilai *youden index* tidak begitu baik, walaupun salah satunya bernilai sempurna. Oleh karena itu, digunakan pula nilai *accuracy* untuk menilai performa perangkat lunak. Selain itu, banyaknya fitur yang terseleksi juga menjadi parameter apakah berjalan dengan benar atau tidak. Semakin besar fitur yang tereduksi dan nilai prediksi, berarti model yang dibentuk juga semakin baik.



Gambar 2. Gambaran umum perangkat lunak. Terdiri dari tiga lapisan, yaitu lapisan data, lapisan logika, dan lapisan antarmuka.

IV. IMPLEMENTASI

Secara umum, perangkat lunak terdiri dari tiga lapisan, yaitu lapisan data, lapisan logika, dan lapisan antarmuka. Gambaran umum perangkat lunak dapat dilihat pada gambar 2.

A. Lapisan Data

Dataset yang diterima oleh perangkat lunak adalah sebuah *file* berekstensi *.csv* dengan kolom pertama merupakan kolom kelas data (berupa nilai 1 atau -1), dan kolom selanjutnya adalah nilai atribut dari data tersebut. Tujuan utama lapisan ini supaya pengolahan data menjadi lebih mudah dan efisien.

Lapisan data terdiri dari dua kelas, yaitu kelas *DataSingle* dan kelas *Data*. Kelas *DataSingle* merupakan representasi dari sebuah *record* yang di dalamnyaterdapat informasi kelas label dan atribut dari *record* tersebut. Kelas *Data* merupakan sebuah *linked list* dari kelas *DataSingle*. Kelas *Data* ini merepresentasikan sebuah *dataset* yang memiliki banyak *record*.

B. Lapisan Logika

Algoritma utama pada perangkat lunak berjalan pada lapisan ini. Ada dua modul utama pada lapisan ini, yaitu modul latih dan modul uji. Modul latih melakukan proses penentuan nilai prediktor awal dan *elastic* SCAD. Modul uji melakukan prediksi data uji menggunakan fungsi *classifier* SVM dan kemudian menghitung performa prediksinya.

Penentuan Prediktor Awal

Seperti yang sudah dibahas sebelumnya bahwa penentuan nilai prediktor awal ini menggunakan bantuan *library libsvm*. Kemudian perangkat lunak akan memroses hasil yang dikeluarkan *libsvm* untuk mendapatkan nilai w_0 dan b_0.

```
public void getFromFileWb0() {
   //compute w0
   Data dataTemp = DatasetReader.readModelLibsvmFile(file);
   LinkedList<Object> y = dataTemp.getDataColumn(1);
   for(int i = 2; i < dataTemp.getAttr().size(); i++) {
      double woi = 0;
      LinkedList<Object> xj = dataTemp.getDataColumn(i);
      for(int j = 0; j < xj.size(); j++) {
         double ayi =
Double.parseDouble(String.valueOf(y.get(j))); //get alpha*y
         double xij =
Double.parseDouble(String.valueOf(xj.get(j))); //get x
         woi += (ayi*xij); //get w
      }
      wb0.getW().set(i-2, 0, woi);
   }
   //compute b0
   BufferedReader br = new BufferedReader(new
FileReader(file));
   String str = "";
   while((str = br.readLine()) != null) {
      String[] countSpace = str.split(" ");
      if(countSpace[0].equals("rho")) { //find label rho
         double tempB = Double.parseDouble(countSpace[1]);
         wb0.setB(tempB);
         break;
      }
   }
}
```

Elastic SCAD

Setelah proses pertama berjalan, maka proses selanjutnya adalah proses *elastic* SCAD SVM. Proses ini berjalan secara iteratif hingga memenuhi kondisi-kondisi yang telah dijelaskan sebelumnya.

```
public void doElasticScad() {
   getFromFileWb0(); //get w0 and b0 (first predictor)
   iter = 1; diff = 1;
   computeX();

   while(diff >= tolerance) {
      //stop iteration when iter > max_iter (user input)
      if(iter >= max_iter) {
         break; //get previous w and b
      }
      computeE();   computeR();   computeDo();
      computeQ1();     computeQ2();
      computeP();   computeQ();
      //Compute new w and b, Q*(b, w)' = P'
      try {
         Matrix temp = Q.solve(P.transpose());
         Matrix w = temp.getMatrix(1,
(temp.getRowDimension()-1), 0, 0);
         wb.setB(temp.get(0, 0));
         for(int i = 0; i < w.getRowDimension(); i++) {
            double tempW = w.get(i, 0);
            wb.getW().set(i, 0, tempW);
         }
      } catch (Exception ex) {
         break; //get previous w and b
      }
      if(wb.isNan()) {
         break; //get previous w and b
      }
      diff = setDifference(wb, tempWb);
      emptyness();
      if(wb.isEmpty()) {
         break; //get previous w and b
      }
   }
}
```

Prediksi Data Uji

Setelah model terbentuk, maka dilakukan pengujian terhadap model. Pengujian ini menggunakan fungsi *classifier* SVM.

```
public void compute() {
   for(int i = 0; i < dataTesting.getData().size(); i++) {
      DataSingle ds = dataTesting.getData().get(i);
      double temp = 0;
      for(int j = 0; j < ds.getValue().size(); j++) {
         double temp1 = ds.getValue().get(j);
         double temp2 = hyperplane.getW().get(j, 0);
         temp += (temp1*temp2);
      }
      temp += hyperplane.getB();
      observed.add(ds.getLabel());
      if(temp < 0) {
```

```
        predicted.add("-1");
      }
      else if(temp > 0) {
        predicted.add("+1");
      }
      else {
        predicted.add("0");
      }
    }
}
```

Performa Prediksi

Setelah semua data uji terprediksi, kemudian dihitung nilai performa prediksi berupa *sensitivity*, *specificity*, *youden index*, dan *accuracy*. Penghitungan ini dilakukan untuk menilai apakah perangkat lunak telah bekerja dengan benar atau tidak.

C. Lapisan Antarmuka

Lapisan antarmuka merupakan jembatan antara pengguna dengan perangkat lunak. Pada lapisan ini didefinisikan *user interface* untuk mempermudah pengguna memakai perangkat lunak yang telah dibuat.

V. UJI COBA

A. Data Uji Coba

Pengujian perangkat lunak dilakukan dengan menggunakan data kanker payudara yang dikeluarkan oleh Van't Veer pada tahun 2002[2]. Data tersebut kemudian dilakukan pemetaan ulang (*subclass mapping*) oleh Hoshida pada tahun 2007[7]. Hasil *subclass mapping* mengatakan bahwa ada 51 pasien dinyatakan bebas kanker dan 36 pasien dinyatakan sel kankernya telah menyebar ke organ tubuh lainnya. 51 pasien tersebut kemudian diberi label +1 (normal) dan 36 pasien lainnya diberi label -1 (kanker). Data ini merupakan data berdimensi tinggi dengan total 1213 atribut.

B. Skenario Uji Coba

Pengujian perangkat lunak prediksi kanker payudara yang menggunakan metode *elastic* SCAD SVM ini, terbagi ke dalam beberapa skenario. Masing-masing skenario memiliki tujuan yang berbeda dalam menilai performa perangkat lunak.

Perbandingan Hasil Prediksi

Untuk mengetahui apakah metode *elastic* SCAD SVM dapat menangani permasalahan data berdimensi tinggi, maka dilakukan pengujian perangkat lunak dengan membandingkan performanya antara *elastic* SCAD SVM dan SVM.

Berdasarkan hasil yang tertera pada tabel 1, dapat diketahui bahwa nilai *accuracy elastic* SCAD SVM lebih baik daripada SVM itu sendiri. Hal itu juga berarti bahwa jumlah data *miss-classification error* dengan metode *elastic* SCAD SVM lebih sedikit daripada SVM. Hal ini tentu sangat baik karena dalam dunia kedokteran, hasil akurasi yang tinggi sangatlah penting.

Dari uji coba ini juga didapatkan informasi bahwa metode *elastic* SCAD SVM menggunakan 1193 atribut untuk membangun model prediksi. Ini berarti bahwa metode *elastic* SCAD SVM mampu menyeleksi 20 atribut yang dianggap tidak penting. Dengan hasil *accuracy* yang tinggi, bisa dikatakan bahwa *elastic* SCAD SVM mampu menyeleksi fitur dengan benar.

Selain itu, gambar 2 menunjukkan bahwa hampir di semua *fold* perangkat lunak mengalami konvergensi. Hal ini dapat ditunjukkan dengan grafik nilai beda yang semakin mendekati 0. Namun pada *fold* 0, grafik nilai beda tidak mendekati 0. Hal ini bisa saja disebabkan karena pesebaran datanya yang kurang merata.

Pengaruh Parameter Tuning

Metode *elastic* SCAD SVM memiliki beberapa paramater *tuning* di dalamnya, antara lain nilai a_1 yang digunakan pada SCAD *penalty*, nilai a_2 yang digunakan pada *Ridge Penalty*, dan tipe kernel SVM yang digunakan untuk menentukan nilai prediktor awal. Uji coba ini dilakukan untuk mengetahui bagaimana pengaruh parameter-parameter *tuning* tersebut terhadap model prediksi yang terbentuk.

Berdasarkan tabel 2, dapat diketahui bahwa dengan memperbesar nilai a_1, jumlah fitur yang terseleksi menjadi

Tabel 1.
Hasil uji coba perbandingan hasil prediksi.

Jenis Prediksi	Support Vector Machine	Elastic SCAD SVM
Sensitivity	90,2%	98,04%
Specificity	100 %	91,67%
Youden	90,2%	89,7%
Accuracy	94,25%	95,4%

Tabel 2.
Hasil uji coba perubahan parameter tuning a_1.

Parameter tuning a_1	Sensitivity	Specificity	Youden Index	Accuracy	Fitur
0,001	98,04%	91,67%	89,71%	95,4%	1190
0,01	98,04%	91,67%	89,71%	95,4%	1193
0,1	90,2%	86,11%	76,31%	88,51%	1211
1	100%	91,67%	91,67%	96,55%	1206
10	98,04%	91,67%	89,71%	95,4%	773
100	100%	91,67%	91,67%	96,55%	437

Tabel 3.
Hasil uji coba perubahan parameter tuning a_2.

Parameter tuning a_2	Sensitivity	Specificity	Youden Index	Accuracy	Fitur
0,001	92,16%	88,89%	81,65%	90,8%	1209
0,01	98,04%	91,67%	89,71%	95,4%	1193
0,1	98,04%	91,67%	89,71%	95,4%	1201
1	100%	91,67%	91,67%	96,55%	1200

Tabel 4.
Hasil uji coba perubahan parameter tipe kernel SVM.

Tipe Kernel SVM	Accuracy	Fitur Terpakai
Linear	95,4%	1193
Polynomial deg 2	95,4%	1200
Polynomial deg 3	96,55%	1198
Polynomial deg 4	95,4%	1200

Polynomial deg 5	95,4%	1198
RBF gamma 0,001	96,55%	1200
RBF gamma 0,01	96,55%	1199
RBF gamma 0,1	95,4%	1199
RBF gamma 1	95,4%	1199

semakin sedikit. Seharusnya semakin banyak fitur yang terseleksi, nilai prediksi tentunya akan semakin baik. Namun ketika nilai $a_1 = 0{,}1$, hasil performa prediksi tidak baik. Hal ini bisa terjadi karena perangkat lunak salah menyeleksi fitur. Fitur yang penting justru malah terseleksi. Hal-hal semacam itu sangat mempengaruhi hasil prediksi. Oleh karena itulah, penentuan nilai a_1 yang optimal sangat diperlukan.

Di sisi lain, hasil uji coba pada tabel 3 tidak menunjukkan pengaruh nilai a_2 terhadap jumlah fitur yang terseleksi. Secara teori, nilai a_2 memang digunakan untuk mengontrol proses seleksi fitur pada SCAD *penalty* yang ketat.

Tabel 4 menunjukkan bahwa masing-masing tipe kernel menunjukkan hasil prediksi yang tidak jauh berbeda dan sama-sama baik. Selisih kesalahan hanya satu buah data berlabel +1 yang diprediksi oleh perangkat lunak pada label -1. Hal ini bisa terjadi karena data uji yang digunakan pada skenario ini sudah berdimensi tinggi. Fungsi kernel pada SVM ini memetakan data yang berdimensi rendah yang tidak bisa dipisah secara linier, menjadi data berdimensi lebih tinggi, sehingga bisa dipisah secara linier. Karena data uji yang digunakan adalah data berdimensi tinggi, maka tipe kernel tidak begitu berpengaruh terhadap hasil prediksi.

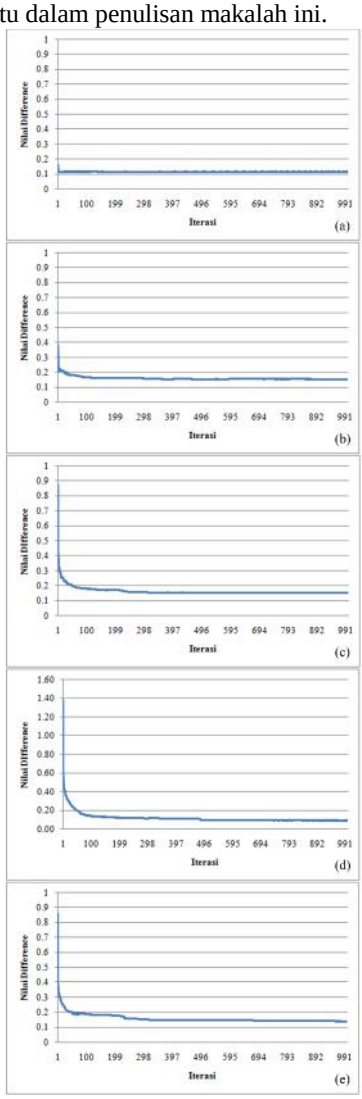
VI. KESIMPULAN

Berdasarkan perangkat lunak prediksi kanker payudara yang telah dibuat beserta uji coba yang telah dilakukan, maka dapat ditarik beberapa kesimpulan sebagai berikut:

1) Perangkat lunak yang dikembangkan berdasarkan model *elastic* SCAD SVM terbukti mampu menangani prediksi data berdimensi tinggi, khususnya data penyakit kanker payudara yang berasal dari DNA *Microarray*. Hal ini dapat disimpulkan berdasarkan nilai *accuracy* yang dihasilkan mencapai 95,4% dan dapat menyeleksi atribut hingga 1193 atribut.
2) Berdasarkan uji coba, perangkat lunak yang dikembangkan berdasarkan model memiliki hasil prediksi yang lebih baik daripada perangkat lunak yang menggunakan SVM. SVM menghasilkan nilai *accuracy* sebesar 94,25% dan *elastic* SCAD SVM lebih baik 1,15%, yaitu 95,4%.
3) Semakin besar nilai a_1, semakin banyak fitur yang tereduksi. Namun, hal itu dibatasi dengan adanya nilai a_2. Nilai *accuracy* terbaik didapatkan ketika $a_1 = 1$ dan $a_2 = 1$.
4) Berdasarkan uji coba, performa dari perangkat lunak yang menggunakan kernel *non-linear* (*Polynomial* dan *RBF*) tidak jauh berbeda dengan dengan kernel *linear*.

UCAPAN TERIMA KASIH

Para penulis mengucapkan terima kasih kepada Allah SWT atas limpahan rahmat dan hidayah-Nya dan semua pihak yang telah membantu dalam penulisan makalah ini.



Gambar 3. Grafik konvergensi pada setiap *fold*. Data *fold* 0 ditunjukkan pada gambar (a), *fold* 1 pada gambar (b), *fold* 2 pada gambar (c), *fold* 3 pada gambar (d), dan gambar (e) menunjukkan data *fold* 4.

DAFTAR PUSTAKA

[1] Sanchez, A., & de Villa, M. C., *A Tutorial Review of Microarray Data Analysis.* Barcelona: Universitat de Barcelona, (2008).
[2] Van't Veer, L. J., Dai, H., van de Vijver, M. J., He, Y. D., Hart, A. A., Mao, M., et al., Gene expression profiling predicts clinical outcome of breast cancer. *Nature* , 415: (2002)530-536.
[3] Becker, N., Toedt, G., Lichter, P., & Benner, A., Elastic SCAD as a novel penalization method for SVM classification tasks in high-dimensional data. *BMC Bioinformatics* , (2011)12:138.
[4] Gunn, S. R, *Support Vector Machine for Classification and Regression.* University of South Hampton, (1998).
[5] Fan, J., & Li, R. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of American Statistical Association* , 96: (2001)1348-1360.
[6] Chang, C.-C., & Lin, C.-J, *LIBSVM: A Library for Support Vector Machines.* Library Documentation, Taiwan, (2001).
[7] Hoshida, Y., Brunet, J.-P., Tamayo, P., Golub, T. R., & Mesirov, J. P. Subclass Mapping: Identifying Common Subtypes in Independent Disease Data Sets. *PLoS One* , (2007).