Aplikasi Algoritma *Boosted Regression Tree* dalam Analisis Keberadaan Belut *Anguilla australis* di Selandia Baru

Ardysatrio Fakhri Haroen
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13517062@std.stei.itb.ac.id

Abstrak— **Perkembangan teknologi, khususnya di bidang *machine learning*, telah sangat membantu perkembangan riset di ilmu-ilmu pengetahuan lainnya, yang diantaranya adalah ilmu Ekologi. Ilmu ini sangat terbantu dengan adanya algoritma *machine learning* yang cukup canggih untuk memprediksi secara lebih akurat dan disaat yang sama cukup intuitif untuk dimengerti. Pada makalah ini akan dianalisis bagaimana algoritma machine learning, spesifiknya *Boosted Regression Tree* telah membantu penelitian ekologi, yakni dalam mempelajari korelasi antara frekuensi penangkapan belut *Anguilla australis* dengan variabel-variabel yang mendeskripsikan sungai habitatnya.**

***Keywords—Machine Learning*, Pohon, Aplikasi, Distribusi, *Anguilla au*stralis**

I. PENDAHULUAN

Ilmuwan ekologi seringkali menggunakan pemodelan matematika untuk mempermudah pekerjaan mereka. Contohnya, menganalisis berat burung dan korelasinya terhadap jenis kelamin, ras, dan umur burung tersebut; menganalisis jumlah tupai di suatu tempat dan korelasinya dengan temperatur, kelembaban, dan faktor-faktor lingkungan lainnya terkait ekosistem tertentu. Secara spesifik, mereka banyak sekali menggunakan model regresi untuk hal tersebut, karena model tersebut dilihat mudah dan intuitif untuk menginferensi korelasi antara variabel.

Namun, model-model regresi yang terdahulu terlalu simplistik untuk diaplikasikan ke berbagai kasus nyata. Seiring perkembangan zaman, ilmu statistik pun ikut berkembang. Perkembangan tersebut banyak yang diaplikasikan ke dalam ilmu ekologi. Diantaranya *generalized linear models* (GLM; McCullagh & Nelder 1989) dan *generalized additive models* (GAM; Hastie & Tibshirani 1990).

Seiring perkembangan statistik pula, bidang ilmu komputer juga terus berkembang. Bidang ini membawa banyak sekali terobosan dalam analisis data, diantaranya dengan munculnya *machine learning* bersama dengan komputerisasi algoritma-algoritma analisis yang lebih canggih. Diantaranya *ensemble tree*, *neural networks*, dan *support vector machines.* Namun, para ilmuwan ekologi jarang menggunakan model-model berkaitan dengan *machine learning* seperti diatas karena dianggap kurang intuitif dan kurang bisa ditafsirkan.

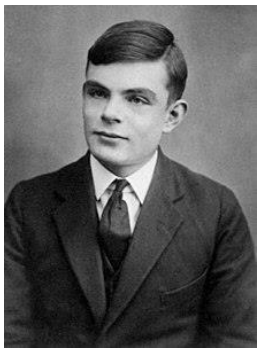
Pada makalah ini, kita akan mencoba menganalisis kerja sebuah algoritima *machine learning* yang keluar lebih belakangan dibanding model-model diatas, yakni *Boosted Regression Tree(BRT).* Metode ini (dari sisi ilmu ekologi), bisa sangat membantu karena menggabungkan beberapa keunggulan algoritma statistik dan algoritma *machine learning*, yang kemudian mengomplemen antara satu sama lain.

II. DASAR TEORI

A. Pembelajaran Mesin/ Machine Learning

Pembelajaran mesin, atau biasa dikenal dengan istilah *machine learning*, adalah sebuah metode analisis data yang bisa mengotomatisasi pembentukan model matematis.

Bidang ini adalah salah satu cabang dari kecerdasan buatan (*Artificial Intelligence)* yang didasarkan pada ide bahwa dapat dibuat sebuah sistem yang bisa belajar dari data, mengenali pola dan mengambil keputusan dengan campur tangan minim dari manusia.



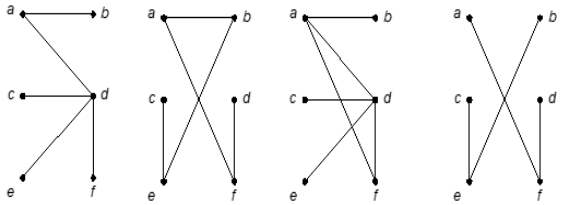
Gambar 2.A.1, Alan Turing salah satu pionir *machine learning*
Sumber: https://en.wikipedia.org/wiki/Alan_Turing Diakses pada 9 Desember 2018, 21.59

Pembelajaran mesin mempunyai sejarah yang cukup panjang, karena memang perkembangannya berjalan perlahan seiring berkembangnya teknologi komputasi. Akan tetapi tidak bisa dipungkiri bahwa Alan Turing adalah diantara ilmuwan yang menjadi salah satu pionir dalam bidang ini maupun dalam ilmu komputer secara umum. Salah satu penemuannya yang sangat berpengaruh adalah *Turing's Learning Machine*, yang pada akhirnya menginspirasi lahirnya *Genetic Algorithm.*

B. Algoritma dan Struktur Data yang Digunakan

B.1 Pohon

Pohon sebagai salah satu struktur diskrit didefiniskan sebagai graf tak-berarah terhubung yang tidak mengandung sirkuit.



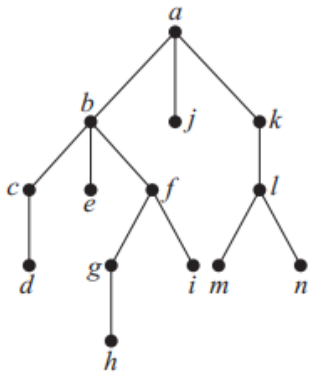
Gambar 2.B.1 Contoh Pohon
Sumber:http://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2 017-2018/Pohon%20(2013).pdf/ Diakses pada 9 Desember 2018, Pukul 07.19

Berikut adalah beberapa contoh Graf yang Pohon dan Bukan pohon. Graf yang paling kanan bukan pohon karena tidak terhubung. Sedangkan yang kedua dari kanan karena mengandung sirkuit.

Dalam matematika diskrit, pohon memiliki banyak jenis. Namun, yang akan dibahas disini adalah aplikasi dari pohon berakar, yakni pohon keputusan.

B.2 Pohon Berakar

Didefinisikan sebagai pohon yang suatu simpulnya sebagai akar dan sisi-sisinya diberi arah menjauh dari akar.



Gambar 2.B.2 Pohon Berakar
Sumber: Discrete Mathematics and Its Applications, Kenneth H. Rosen, Page 753

1. Akar
Diartikan sebagai simpul yang memiliki derajat masuk 0. Artinya akar merupakan simpul permulaan dari sebuah pohon.

2. Anak (*child*) dan Orangtua (*parent*)
Jika ada simpul x dan y, maka y dikatakan anak dari x jika ada sisi dari x ke y. Begitupula x dikatakan orangtua dari y.

3. Saudara kandung (*sibling*)
Dua buah simpul atau lebih dikatakan saudara kandung jika mempunyai orangtua yang sama.

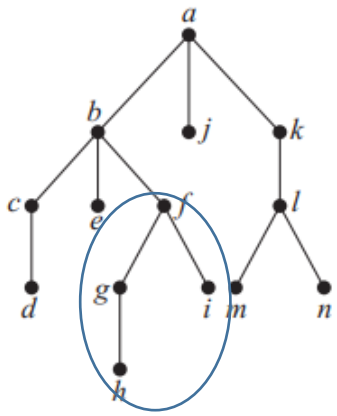
4. Daun (*leaf*)
Simpul yang mempunyai derajat keluar 0.

5. Simpul Dalam (*Internal Node*)
Simpul pada pohon yang mempunyai anak dan bukan merupakan akar.

6. Lintasan (*Path)*
Lintasan dari simpul x_1 ke x_n adalah runtutan simpul-simpul dari x_1, x_2, ... ,x_n sedemikian sehingga x_i adalah orangtua dari x(i+1).

7. Keturunan (*descendant*) dan Leluhur (*ancestor)*
Jika terdapat lintasan dari simpul x ke y maka x merupakan leluhur dari y dan y merupakan keturunan dari x.

8. Upapohon (*subtree*)
Jika x merupakan sebuah simpul di dalam pohon T, maka upapohon dengan menjadikan simpul x sebagai akarnya merupakan sebuah upagraf *T' = (V', E')* sedemikian sehingga *V'* mengandung x dan *E'* mengandung seluruh lintasan yang berasal dari x. Dalam satu pohon banyak sekali upapohon yang dapat dibentuk.



Gambar 2.B.2 Upapohon yang dilingkari
Sumber: Discrete Mathematics and Its Applications, Kenneth H. Rosen, Page 753

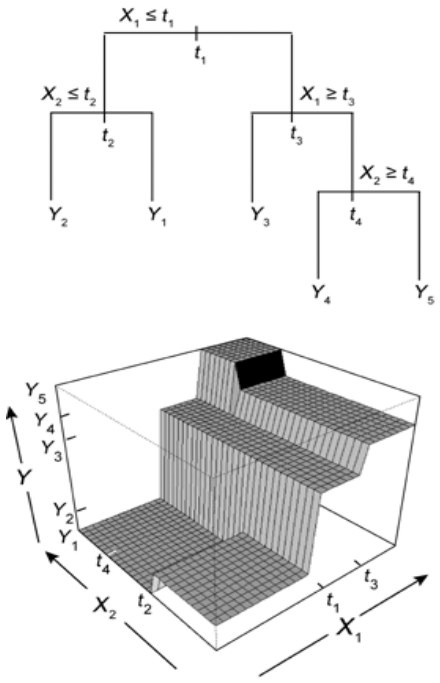
9. Derajat (*degree*)
Jumlah upapohon dari suatu simpul

10. Aras, Level, atau Tingkat
Aras dari akar adalah 0. Untuk selain akar maka arasnya adalah 1 + panjang lintasan dari akar ke simpul tersebut.

B.3 Pohon Keputusan (Decision Tree)

Merupakan sebuah pemodelan data yang berbasiskan struktur data pohon berakar. Secara matematis, model ini membagi-bagi ruang prediktor menjadi berbentuk kotak-kotak, berdasarkan aturan-aturan matematis yang menentukan kesesuaiannya dengan ruang prediktor.

Contoh penggunaan pohon keputusan sebagai berikut: Misalkan kita ingin melakukan pemodelan terhadap pengaruh ekosistem terhadap berat hewan dewasa dari sebuah spesies. Kita misalkan ada variable prediktor X_1 dan X_2 yang kita anggap sebagai prediktor, dan variable Y sebagai variabel respon yang menyatakan rata-rata berat hewan dewasa pada suatu spesies. Titik Y_1, Y_2, dst. dianggap sebagai daun pada pohon, dan variabel t_1, t_2, dst. dianggap sebagai titik pemisah (*split points*). *Split point* adalah sebuah variable yang kita hitung untuk menentukan anak-anak dari sebuah simpul (untuk membuat upapohon baru dari yang sudah ada). Representasi visual ada dibawah.



Gambar 2.B.3 Visualisasi Pohon Keputusan dan Representasi Geometrisnya
Sumber: https://besjournals.onlinelibrary.wiley.com/doi/full/10.1111/j.1365-2656.2008.01390.x Diakses pada 9 Desember 2018, 17.11

Pohon keputusan mempunyai beberapa kelebihan. Mereka mudah untuk divisualisasi dan mudah dimengerti bagi orang-orang yang tidak mendalami ilmu statistik dan matematika. Selain itu, ketika bekerja dengan pohon keputusan kita juga tidak perlu melakukan banyak *pre-processing* (Pekerjaan awal terhadap data), karena pohon keputusan bisa digunakan untuk berbagai macam prediktor, baik yang bentuknya numerik, biner, maupun kategorikal.

Namun disamping kelebihan diatas, pohon keputusan juga memiliki kekurangan, yakni relatif kurang akurat prediksinya dibandingkan dengan algoritma-algoritma seperti GLM dan GAM. Selain itu, pohon keputusan juga sulit untuk memodelkan fungsi-fungsi yang "mulus", bahkan fungsi semisal $x = y$. Berbagai kekurangan tersebut sangat mengurangi performa analisis pohon keputusan dan menambahkan ketidakpastian yang cukup signifikan dalam pemakaiannya.

B.4 Algoritma Boosting

Algoritma ini digunakan untuk meningkatkan akurasi suatu model, dan sangat cocok untuk mengomplemen kekurangan dari algortima pohon keputusan.

Cara kerja algoritma ini dapat dijabarkan dalam beberapa langkah berikut:

1. Membuat model dari sebagian acak dari data yang dimiliki

2. Mengetes akurasi dari model tersebut ke data yang ingin kita ujikan.

3. Setelah mendapatkan hasilnya, algoritma tersebut melihat ke data uji yang ia prediksi secara buruk.

4. Algoritma ini kembali membangun model dengan memfokuskan kepada perbaikan di bagian data uji yang ia berperforma buruk.

5. Proses 1-4 di iterasi selama berulang kali, bisa sampai ratusan hingga ribuan kali.

Pada algoritma *Boosted Regression Tree* yang digunakan pada makalah ini, algoritma *boosting* yang digunakan mempunyai perbedaan secara intuitif. Proses iteratif yang digunakan untuk memperbaiki model pada hakikatnya adalah *functional gradient descent.* Pada tiap iterasi, dilakukan hal sebagai berikut:

1. Dihitung *loss function* (sebuah ukuran deviasi) antara model yang ingin dibentuk dengan hasil yang didapat.

2. Dilakukan optimasi berupa penambahan sebuah pohon keputusan (yang dimaksud adalah dengan representasi secara fungsional) yang paling optimal dalam mengurangi *loss function.*

3. Dihitung residual dari *loss function* yang sebelumnya, kemudian digunakan untuk iterasi berikutnya.

Perlu diketahui bahwa proses ini berbasis tahap, bukan berbasis langkah. Jadi, semua perubahan atau optimasi yang dilakukan hanya menambah pada pohon yang sudah kita bangun, tidak merubah total pohon yang telah terbangun.

III. STUDI KASUS

A. Pengenalan Dataset

Data yang disediakan berupa variabel-variabel yang mendeskripsikan lingkungan hidup lele *Anguilla a.*,dengan variable target adalah probabilitas (antara 0 sampai 1) lele tersebut muncul di lingkungan yang memiliki variabel-variable tertentu.

Anguilla a. Adalah satu spesies lele air tawar yang berasal dari Australia Tenggara, Selandia Baru, dan Kepulauan Pasifik Barat. Habitatnya adalah sungai dangkal, rawa-rawa, arus yang lamban, dan daerah-daerah pastoral. Variabel-variabel terkait lingkungan yang akan dikaji dipaparkan dibawah ini:

1. LocSed, Rata-rata ketebalan sedimen tanah
(1 = Lumpur, 2 = Pasir, 3 = kerikil halus, 4 = kerikil kasar, 5 = bebatuan, 6 = perairan dengan batu besar, 7=lapisan tanah keras)
Mean = 3.77

2. SegSumT, Suhu udara pada musim panas (ºC)
Mean = 16.3, Range = 8.9 – 19.8

3. SegTSeas, Suhu udara pada musim dingin (ºC)
Mean = 0.36, Range = -4.2 – 4.1

4. SegLowFlow, Segmentasi *Low Flow*
Mean = 1.092, Range = 1.0 – 4.09

5. DSDist, Jarak ke Pesisir Pantai Terdekat (km)
Mean = 74, Range = 0.03 – 433.4

6. DSDam, Keberadaan obstruksi tertentu (biner)
Mean = 0.88, Range 0 – 1

7. DSMaxSlope, Kemiringan arus turun maksimum(º)
Mean = 3.1, Range = 0 – 29.7

8. USAvgT, Temperatur relatif terhadap segment
Mean = 0.38, Range = -7.7 – 2.2

9. USRainDays, Jumlah hari dalam satu bulan dimana curah hujan > 25mm.
Mean = 1.22, Range = 0.21-3.30

10. USSlope, Kemiringan arus menanjak rata-rata(º)
Mean = 14.3, Range = 0 – 41.0

11. USNative, Luas daerah dengan hutan alami (proporsi)
Mean = 0.57, Range = 0 – 1

12. Method, Metode memancing di daerah tersebut (kategorikal: *electric, net, spot, trap, mixture*)

Eksplorasi terhadap dataset dan pembentukan model yang akan dibahas dibawah menggunakan bahasa pemrograman R.

B. Perlunya Optimasi Model

Untuk seluruh masalah yang berikaitan dengan prediksi, ada kemungkinan terjadinya *overfitting*, yakni model yang dibuat sangat terikat dengan data yang dipunya sehingga tidak bisa dipakai untuk data yang bernilai lain.

Pada algoritma *Boosted Regression Tree*, *overfitting* sangat mungkin terjadi karena memang algoritmanya didesain untuk menyocokkan diri dengan seluruh data. Oleh karena itu, untuk mencegah terjadi *overfitting* diperlukan sebuah metode yang dinamakan *regularization*. Metode ini digunakan untuk menghentikan penyocokan algoritma di satu titik sehingga model dapat mencapai keseimbangan antara kemampuan prediktif dengan kecocokan pada model tersebut.

Untuk mayoritas metode pemodelan, *regularization* dicapai dengan mengendalikan jumlah batasan. Batasan disini didefiniskan sebagai jumlah variabel prediktor dan kompleksitas dari model yang sudah dicocokkan. Pengurangan jumlah batasan dilakukan karena memang model dengan batasan-batasan yang lebih sedikit akan lebih umum.

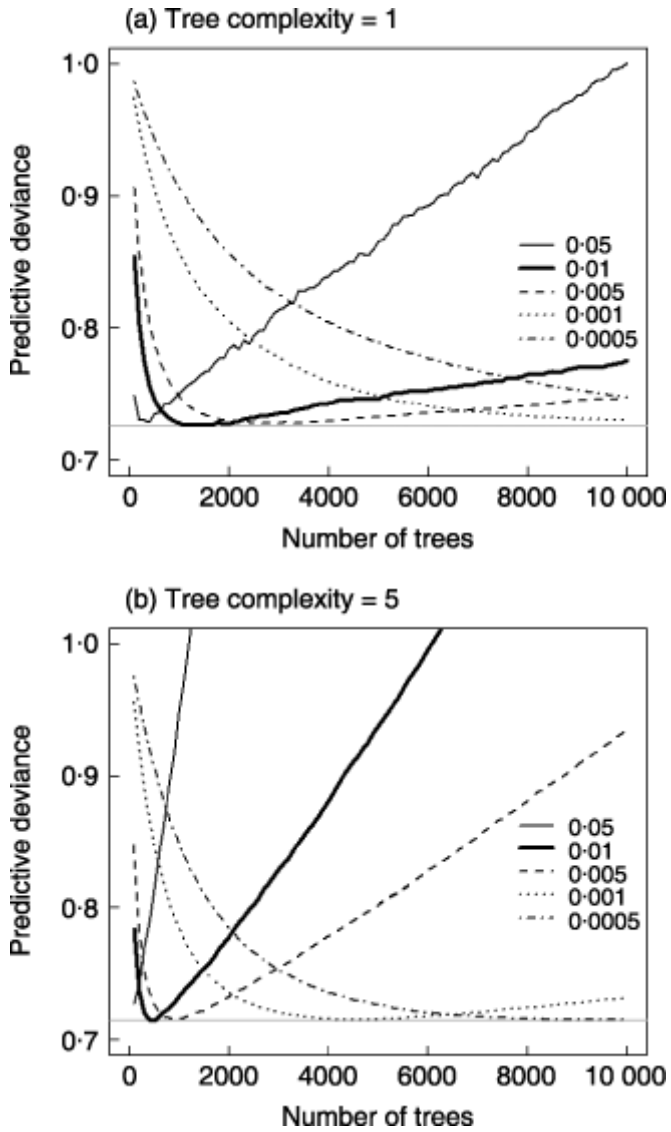
Namun, ada cara lain untuk melakukan *regularization*, yakni dengan metode *shrinkage.* Pada metode ini, jumlah batasan tidak dikurangi, namun diberikan *weight* (besar koefisien) yang lebih kecil sehingga tidak berpengaruh terlalu besar ke model secara keseluruhan. Secara umum, metode itulah yang dipakai pada *Boosted Regression Tree.* Secara spesifik, metode *regularization* pada *Boosted Regression Tree* melibatkan keseimbangan pemilihan *nt* (jumlah pohon), *lr* (laju belajar pohon), dan *tc* (kompleksitas model pohon).

C.Analisis Laju Belajar dan Jumlah Pohon Optimal

Laju belajar model (*lr*) digunakan untuk menentukan kontribusi dari setiap pohon baru seiring pohon tersebut ditambahkan ke pohon keseluruhan. Memperkecil *lr* akan memperkecil kontribusi tiap pohon, sehingga dengan *lr* yang lebih kecil, jumlah pohon (*nt*) yang dibutuhkan lebih banyak.

Secara umum, untuk algoritma *Boosted Regression Tree* kita mencari *lr* yang lebih kecil dan *nt* yang lebih banyak. Maka kita akan relatif memperkecil *lr* dari model kita.

Menggunakan data 1000 lokasi masing masing dengan variabel lingkungan tersendiri, kita akan memperkirakan nilai *rt* dan *nl* terbaik. Berikut ini visualisasi hasil penyocokan dengan berbagai nilai *rt* dan *nl.*



Gambar 3.C.1 Hasil *plotting* model BRT ketika dicocokkan dengan berbagai nilai *nt* dan lr

Sumber: https://besjournals.onlinelibrary.wiley.com/doi/full/10.1111/j.1365-2656.2008.01390.x Diakses Tanggal 10 Desember 2018

Pada data tersebut, di plot kurva yang merepresentasikan *lr* pada sumbu x yang merepresentasikan *nt* dan sumbu y yang merepresentasikan deviasi terhadap prediksi. Terdapat 5 nilai *lr* yang diujikan, sebagaimana terdapat 5 garis kurva, terdapat 2 variasi kompleksitas pohon yang diujikan, (kompleksitas = 1 dan kompleksitas = 5), dan nilai *nt* yang diujikan ada pada rentang [0 – 10.000]. Meninjau nilai *lr* terbesar dan terkecil dapat ditarik kesimpulan sebagai berikut:

1. Untuk *lr=0.05 dan lr=0.01* (Nilai terbesar), deviasi meningkat sangat cepat setelah mencapai deviasi minimum di *nt = 1000*. Hal ini terjadi baik di kompleksitas=1 maupun pada kompleksitas=5.

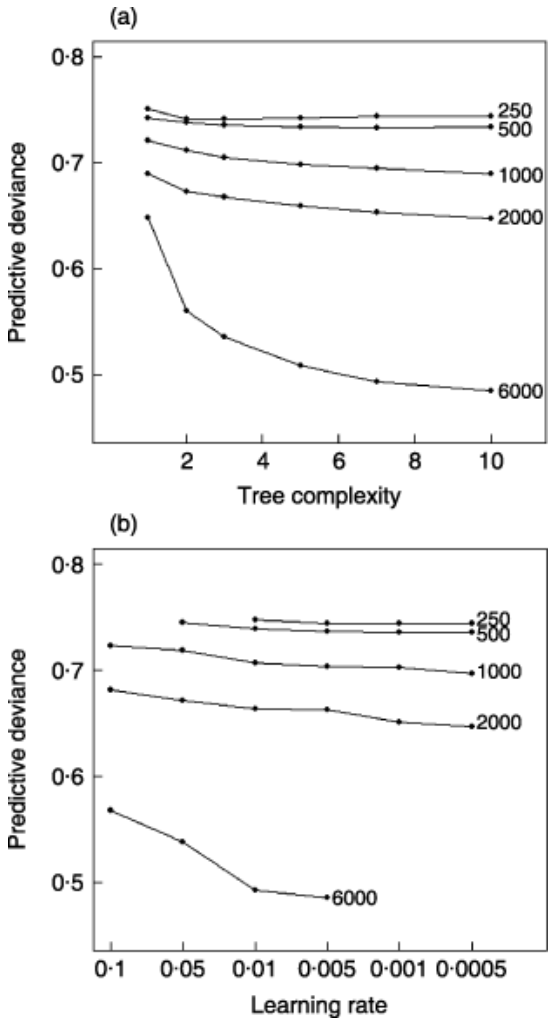
2. Untuk *lr=0.0005 dan lr=0.001*, deviasi minimum dicapai dalam waktu yang sangat lama, karena butuh nilai *nt* yang sangat besar agar bisa mencapai kondisi tersebut.

Meninjau pada data yang variatif, kita perlu mempertimbangkan penggunaan banyak pohon (nilai *nt* besar). Oleh karena itu, kembali dibuktikan pernyataan pada subbab sebelumnya bahwa pada algoritma *Boosted Regression Tree* lebih tepat digunakan *lr* yang kecil dan konsekuensinya, *nt* yang besar.

D. Kompleksitas Pohon

Kompleksitas pohon/*tree complexity* (*tc)* didefinisikan sebagai jumlah node dalam satu pohon. Nilai *tc* sangat menentukan nilai *nt*, juga *lr*. Seiring nilai *tc* naik, maka pohon akan semakin kompleks sehingga jumlah pohon yang dibutuhkan untuk memodelkan seluruh data pun ikut berkurang. Maka seiring nilai *tc* ditingkatkan, nilai *lr* perlu dikurangi.

Selain terkait dengan kedua variable diatas, *tc* juga sangat terkait dengan besar data sampel. Berikut ini hasil *plotting* antara jumlah data sampel, kompleksitas pohon, dan deviasi prediksi



Gambar 3.D.1 Plotting kurva yang merupakan representasi besar data sampel terhadap sumbu x yang merupakan laju belajar (*lr*), dan sumbu y sebagai deviasi

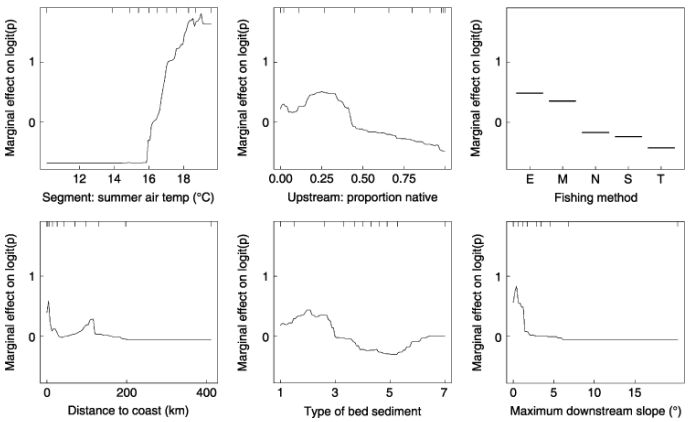
Sumber: https://besjournals.onlinelibrary.wiley.com/doi/full/10.1111/j.1365-2656.2008.01390.x Diakses Tanggal 10 Desember 2018

Telah terbukti bahwa performa prediksi sangat ditentukan oleh jumlah data sampel, sebagaimana *plot* diatas. Dapat dipahami secara intuitif pula bahwa data yang lebih banyak berarti lebih banyak variasi kasus yang dapat dipelajarkan ke model sehingga model bisa lebih umum dan lebih tahan terhadap variasi data.

Sebagai pedoman umum, seiring nilai *lr* diturunkan, nilai *tc* harus dinaikkan. Biasanya hubungan keduanya secara invers. Contohnya bila nilai *tc* dinaikkan 2 kali lipat maka nilai *lr* perlu diturunkan menjadi setengah dari nilai awal untuk menjaga nilai *nt* yang sama.

E. Plot Dependensi Parsial

Setelah optimasi dan pencocokan model dilakukan, kita bisa melihat efek suatu variabel prediktor terhadap respons yang kita inginkan (dalam hal ini, probabilitas adanya belut). Hal ini dapat dilakukan dengan menghitung efek rata-rata dari semua variabel. Berikut ini disajikan plot *plot dependensi parsial* dari beberapa variabel pada data.



Gambar 3.E.1
Plot dependensi parsial probabilitas adanya belut terhadap beberapa variabel prediktor.
Sumber: https://besjournals.onlinelibrary.wiley.com/doi/full/10.1111/j.1365-2656.2008.01390.x Diakses Tanggal 10 Desember 2018

Dari data diatas dapat kita ambil kesimpulan:

1. Suhu pada musim panas sangat berpengaruh ke probabilitas adanya belut, jika suhu sekitar 18ºC

2. Kemiringan tanjakan sungai tidak terlalu berpengaruh ke probabilitas adanya belut, namun probabilitas paling besar ada ketika kemiringannya 0.25º

3. Metode penangkapan yang paling berpengaruh ke keberadaan belut adalah penangkapan dengan elektrik

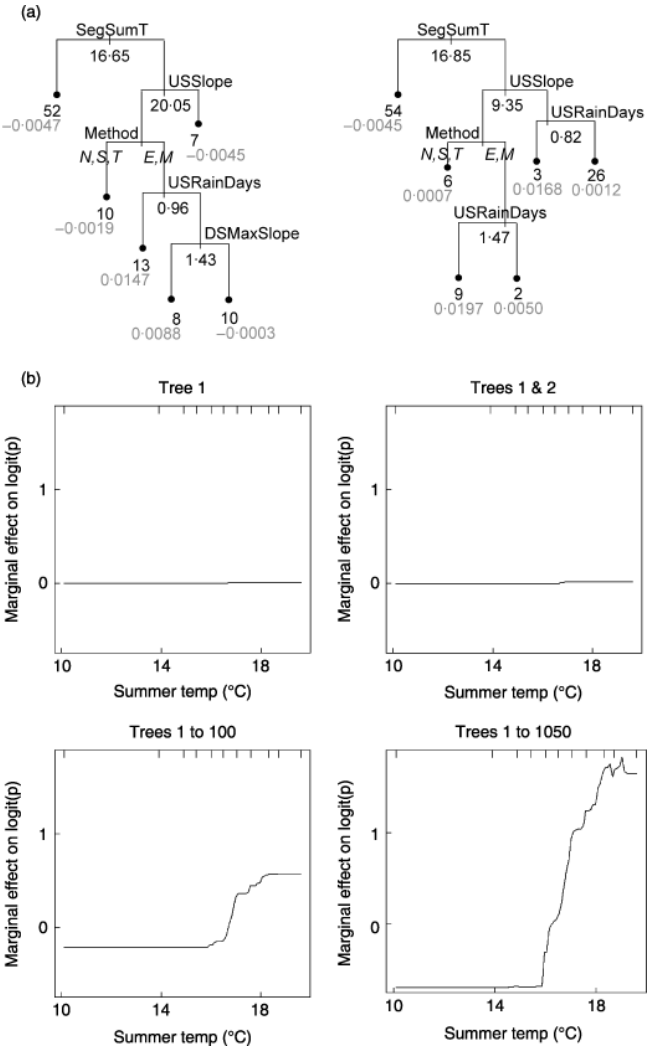
4. Jarak habitat belut relatif terhadap pantai cukup sama untuk jarak lebih dari 175km, namun mayoritas berada di sekitar pantai (antara 0 – 100km)

5. Tipe sedimen yang paling besar probabilitasnya menjadi habitat belut tsb. adalah pasir.

6. Kebanyakan habitat *Angrilla a.* tidak mempunyai kecuraman turunan yang besar. Artinya mereka tidak suka berada di tempat yang arus airnya cukup kuat.

F. Pembentukan Pohon Keputusan

Setelah melalui optimasi terkait berbagai variabel seperti *nt, tc,* dan *lr;* juga setelah dilakukan analisis korelasi antar variabel, kita dapat memvisualisasi pohon keputusan yang sudah dibuat, sebagaimana pada gambar berikut:



Gambar 3.F.1
(a) Visualisasi Pohon keputusan yang telah dibuat dari optimasi sebelumnya. Nama-nama variabel yang tercantum pada pohon dapat dilihat di BAB 3.A
(b) Korelasi variabel variabel lingkungan (dalam sumbu x) terhadap probabilitas adanya belut *Anguilla a.*
Sumber: https://besjournals.onlinelibrary.wiley.com/doi/full/10.1111/j.1365-2656.2008.01390.x Diakses Tanggal 10 Desember 2018

V. KESIMPULAN

Pembahasan diatas hanyalah sebuah contoh penggunaan algoritma *boosted regression tree* sebagai salah satu algoritma *machine learning* yang intuitif. Optimasi yang dilakukan tentunya perlu dieksperimentasikan karena semua data punya keunikan masing-masing. Selain itu, tentunya kita perlu terbuka terhadap kemungkinan munculnya algoritma-algoritma lebih canggih di kemudian hari. Selain pada ekologi, banyak aspek kehidupan dan bidang akademis lain yang bisa terbantu dengan adanya teknologi tersebut. Terutama dengan kemajuan teknologi baik secara kecanggihan maupun distribusi, semakin banyak orang yang bisa berkontribusi terhadap kemajuan umat manusia.

VII. UCAPAN TERIMA KASIH

Puji syukur kepada Allah *ta'ala* yang sudah memberikan banyak sekali karuniaNya sehingga saya bisa menyelesaikan makalah ini. Kemudian terimakasih yang tak henti-hentinya kepada kedua orangtua saya yang selalu memberikan kasih sayang dan berbagai dukungannya. Terimakasih yang tak terkira juga bagi tim dosen Matematika Diskrit, terkhusus Ibu Harlili sebagai pengajar di kelas saya. Kemudian terimakasih bagi J.Elith, J.R Leathwick, dan T.Hastie yang telah menulis makalah mengenai *Boosted Regression Tree* yang banyak saya ambil sebagai referensi, pula bagi kakak-kakak tingkat yang telah bersedia menjadikan makalah mereka sebagai referensi bagi saya.

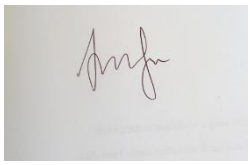
REFERENCES

[1] https://besjournals.onlinelibrary.wiley.com/doi/full/10.1111/j.1365-2656.2008.01390.x
Diakses pada: Sabtu, 8 Desember 2018, 13.05
[2] https://www.sas.com/en_id/insights/analytics/machine-learning.html
Diakses pada: Sabtu, 8 Desember 2018, 20.49
[3] http://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2017-2018/Makalah2017/Makalah-Matdis-2017-036.pdf
Diakses pada: Minggu, 9 Desember 2018, 06.02
[4] Rosen, Kenneth H. Discrete Mathematics and Its Applications

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 10 Desember 2018



Ardysatrio Fakhri Haroen
13517062