Aplikasi Pohon pada Statistik dan Machine Learning

Jon Felix Germinian 13518025[1]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]*jonfelix1@gmail.com*

***Abstract*—Machine Learning adalah sesuatu *buzzword* yang semua orang pernah dengar di abad ini. Dari naiknya social media yang menggunakan machine learning seperti Tik-Tok hingga algoritma rekomendasi Youtube. Perkembangan teknologi juga diikuti satu konsekuensi yang tidak dapat dihindari yaitu berkembangnya kebutuhan data kita. Data dapat digunakan untuk berbagai hal, salah satu contoh aplikasinya yang paling terlihat adalah untuk digunakan dalam konteks Machine learning. Di makalah ini, saya akan membahas tentang salah satu bentuk dari supervised machine learning yaitu Random Tree Forest.**

***Keywords*—Decision Tree, Random Tree Forest, Pohon.**

I. PENDAHULUAN

Belakangan ini, banyak kontroversi seperti pengkoleksian data yang dilakukan oleh facebook, google dan perusahaan teknologi lainnya. Apa yang mereka bisa lakukan dengan data yang telah mereka kumpulkan?

Tidak bisa dipungkiri, perkembangan teknologi komputasi akan diikuti dengan membesarnya data yang dihasilkan oleh manusia. Tahun lalu, volume data total manusia adalah 33 Zettabyte. Angka tersebut diproyeksikan untuk naik hingga 175 Zettabyte pada tahun 2025 [1].

Dengan data sebanyak itu, industri terdorong untuk memakai data tersebut untuk membuat pilihan-pilihan yang didorong oleh data. Untuk memproses sebuah data, dibutuhkan model untuk menjelaskan data tersebut. Model dalam konteks data science adalah diagram deskriptif yang menjelaskan hubungan antara berbagai variabel di dalam sebuah dataset.

Machine learning adalah topik hangat di industri dan dunia riset. Kecepatan perkembangan dan kompleksitas dari bidang ini susah diikuti bahkan untuk para profesional di dalam bidang ini. Machine learning secara umum dapat dibagi kedalam dua kategori. Kategori pertama adalah *supervised learning* yang biasa dipakai jika yang inginkan adalah model yang dapat dijelaskan dengan bahasa manusia. Unsupervised learning digunakan untuk menciptakan model yang lebih kompleks dan lebih abstrak.

Salah satu model yang sederhana dan sering digunakan di data science adalah model *Random Tree Forest*. Model ini senang dipakai oleh para *data scientist* karena modelnya mudah dijelaskan dan performanya yang relatif cukup bagus untuk data diskrit.

Meskipun model *Random Tree Forest* paling efektif digunakan untuk dataset yang tipe datanya diskrit, model ini juga dapat digunakan untuk tipe data kontinu.

Makalah ini akan membahas implementasi teori graf dalam model *Decision Tree, Adaptive Boosting, Random Tree Forest* dan menjelaskan aplikasinya dalam dunia nyata.

II. TEORI DASAR

A. Graf

Graf adalah salah satu cara untuk merepresentasikan objek-objek diskrit dan relasi/hubungan diantara objek-objek tersebut. Salah satu contoh graf yang kita temui setiap hari adalah peta busway seperti yang ada di TransJakarta. Dalam kasus peta busway, stasiun busway merepresentasikan objek diskrit dan rute merepresentasikan relasi. Graf mengandung simpul dan sisi. Simpul adalah objek yang dihubungkan dengan simpul lain oleh sebuah sisi.

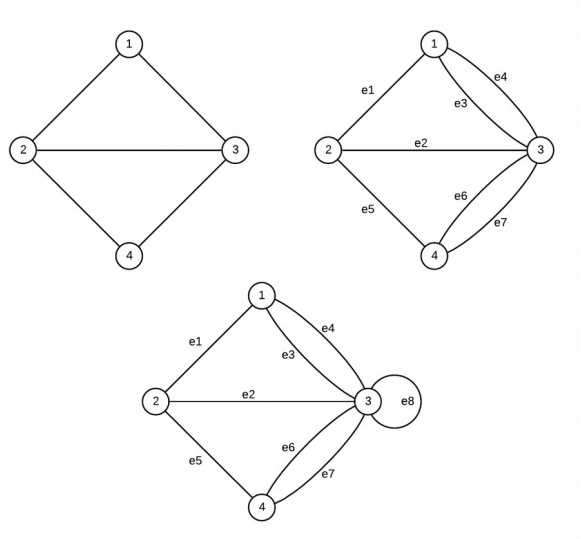


Diagram 1. a. Graf sederhana b. Graf ganda c. Graf semu

B. Pohon

Hutan di dalam matematika diskrit adalah kumpulan pohon yang saling lepas, atau graf tidak terhubung yang tidak mempunyai sirkuit sama sekali. Setiap komponen di dalam sebuah hutan adalah pohon.

Gambar 1. Ilustrasi Hutan yang terdiri dari 3 pohon (sumber : Matematika Diskit, Ed. 3 - Rinaldi Munir)

Pohon dalam matematika diskrit dapat dikategorikan lagi menjadi dua kategori yaitu pohon merentang dan pohon berakar.

Pohon berakar adalah pohon yang satu buah simpulnya diperlakukan sebagai sebagai akar dan sisi-sisinya diberi arah sehingga menjadi graf berarah[2]. Pohon n-ary adalah pohon berakar yang setiap simpul cabangnya mempunyai paling banyak n buah anak. Dalam aplikasinya, pohon biner adalah tipe pohon berakar yang paling sering dipakai karena kemudahan implementasinya. Pohon yang n-ary yang setiap simpul dalamnya mempunyai n-buah anak disebut dengan pohon lengkap.

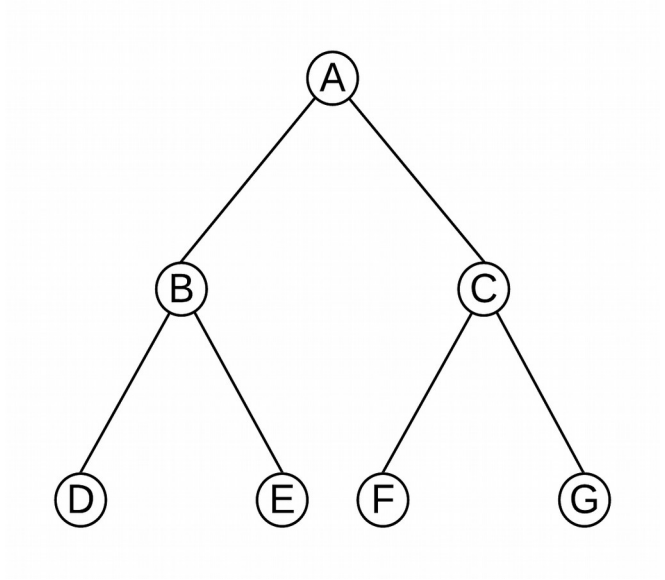
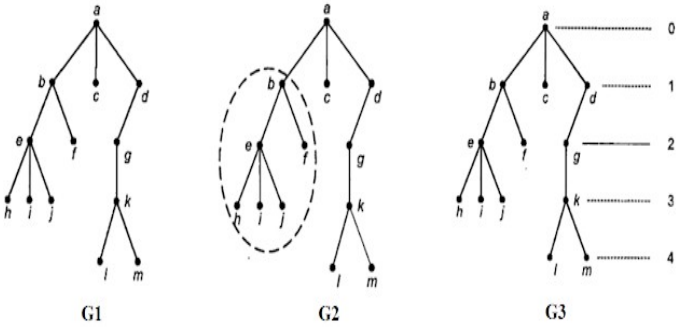


Diagram 2. Pohon biner penuh

Dalam kertas ini, ada beberapa terminologi yang harus diperhatikan :

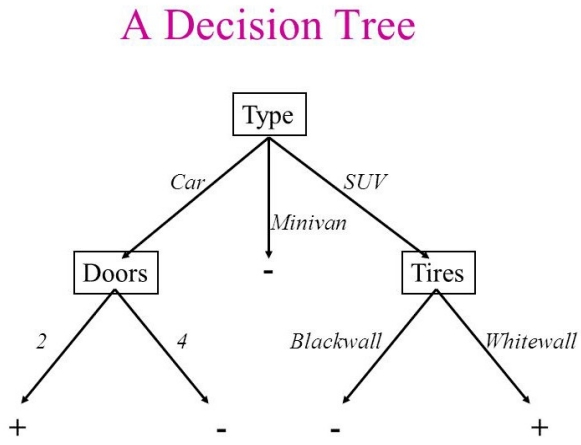
1. Simpul dalam : Simpul yang mempunyai anak dan bukan akar
2. Tinggi/Kedalaman : Level maksimum sebuah pohon
3. Derajat : Jumlah anak sebuah simpul
4. Daun : Simpul berderajat nol
5. Upagraf/Subpohon : Pohon yang diciptakan dengan mengambil suatu simpul menjadi akar
6. Lintasan : Simpul yang dilewati dari simpul orang tua ke simpul anak.
7. Anak dan orang tua : Pada diagram 2, B dan C adalah anak dari A. B adalah orang tua dari D dan E.



Gambar 2. Pohon (sumber : Matematika Diskit, Ed. 3 - Rinaldi Munir)

C. Decision Tree

Pohon berakar yang dipakai dalam *decision tree* adalah pohon biner. Secara tidak langsung, kita telah memakai *decision tree* di kehidupan kita. Salah satu contoh kasus adalah dalam pemilihan ketua himpunan. Kasus lain adalah kategorisasi sebuah produk.



Gambar 3. *Decision Tree* (sumber : https://miro.medium.com/max/840/0*DKVni_-q7dAKVel7.png)

Decision Tree juga merupakan salah satu model dalam machine learning. Model *Decision Tree* adalah salah satu model paling sederhana dan paling mudah digunakan. Model ini membuat prediksi berdasarkan observasi sebuah objek dan membungkusnya kedalam sebuah *decision tree*.

Model *decision tree* dapat dibagi kedalam kedua tipe.

1. Analisis *Classification tree* digunakan saat output yang diinginkan adalah sebuah data diskrit (contoh :

yes/no)

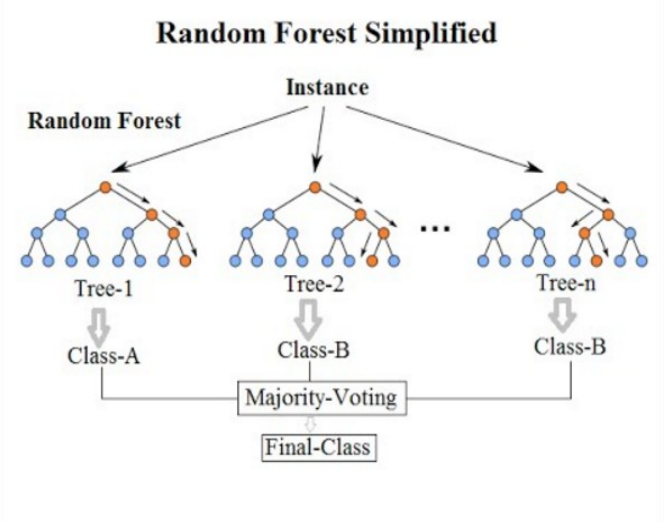
2. Analisis *Regression tree* digunakan saat hasil yang diinginkan adalah sebuah angka riil. (contoh : harga properti)

Pada dasarnya kedua tipe ini mirip, perbedaannya hanya ada di mana prosedur menentukan dimana untuk memecah nilai dalam simpul dalam[3].

D. Random Tree Forest

Random Tree Forest pada dasarnya adalah sekumpulan *decision tree*. Secara formal, *Random Tree Forest* adalah model *supervised machine learning* yang digunakan untuk klasifikasi dan regresi dengan membuat banyak *decision tree* yang hasilnya akan dirata-ratakan menjadi sebuah hasil prediksi.

Setiap *decision tree* yang ada di *random tree forest* lahir dari sampel dataset sehingga kita dapat memprediksi kesalahan prediksi berdasarkan data yang tidak dipakai. Mencari rata-rata dari kesalahan prediksi ini akan memberi estimasi kesalahan yang dinamakan *out-of bag error estimate.*



Gambar 4. Ilustrasi Random Tree Forest (sumber : https://miro.medium.com/max/592/1*i0o8mjFfCn-uD79-F1Cqkw.png)

Pada umumnya, pembuatan *random tree forest* dibuat dengan *greedy algorithm*. *Greedy algorithm* adalah algoritma pemecahan masalah yang mencari hasil local paling optimal untuk mencari hasil global optimum. Algoritma ini tidak selalu menghasilkan global optimum, tetapi memiliki performa baik saat diimplementasikan.

III. RANDOM TREE FOREST

Pada Bab ini akan dibahas *Random Tree Forest* secara teknis, mulai dari pembuatan model *Decision Tree,* hingga

A. Decision Tree

Decision Tree adalah alat yang dipakai untuk klasifikasi dan prediksi. *Decision tree* adalah pohon n-ary dengan setiap daun merepresentasikan sebuah kelas atau nilai (sesuai dengan tipe *Decision tree* yang digunakan). Setiap simpul dalam merepresentasikan sebuah komparasi seperti (apakah umur > 15). Sebagai contoh, misalkan ada simpul yang melihat apakah hari itu cerah apa tidak, jika ya maka lintasan yang dilewati adalah anak di kanan simpul komparasi tersebut, jika tidak maka akan melewati anak di kiri simpul komparasi tersebut.

Pembuatan *decision tree* dimulai dengan membagi dataset yang merupakan simpul akar pohon menjadi dua subset yang merupakan anak dari simpul tersebut. Pembagian didasarkan pada seperangkat aturan pemisahan berdasarkan fitur-fitur yang terdapat di dalam dataset. Pembagian ini dilakukan terus menerus hingga tercipta pure subset dimana semua data didalam subset menghasilkan kelas/nilai tertentu. Proses induksi pohon keputusan inilah contoh dari *greedy algorithm* karena hanya memperhatikan hasil lokal yang paling optimal.

Sebagai contoh, misal kita punya dataset apakah Mahasiswa misterius X pergi berlari di saraga sebagai berikut (1 menyatakan mahasiswa berlari di saraga, 0 menyatakan mahasiswa tidak pergi berlari) :

Hari	Cuaca	Kelembapan	Angin	Lari
1	Hujan	Tinggi	Lemah	0
2	Mendung	Tinggi	Kuat	1
3	Mendung	Normal	Kuat	1
4	Cerah	Normal	Kuat	1
5	Hujan	Tinggi	Lemah	0
6	Cerah	Normal	Kuat	1
7	Cerah	Normal	Lemah	1
8	Mendung	Tinggi	Lemah	1
9	Hujan	Tinggi	Kuat	0
10	Hujan	Tinggi	Lemah	0
11	Hujan	Normal	Kuat	0
12	Mendung	Normal	Lemah	1
13	Cerah	Tinggi	Lemah	0
14	Cerah	Tinggi	Lemah	0

Tabel 1 : Data lari mahasiswa misterius X

Dari data tersebut, dapat dibagi menjadi 3 subset sesuai dengan cuaca di hari tersebut. *Decision Tree* yang dihasilkan adalah sebagai berikut:

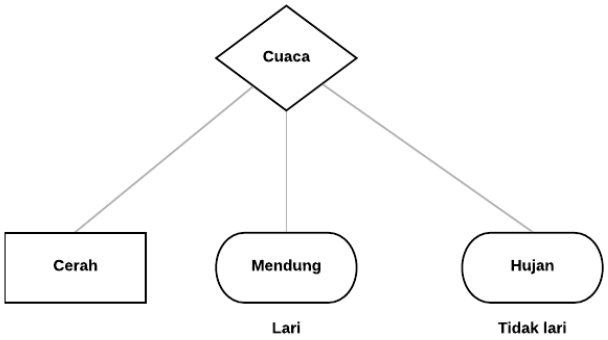


Diagram 3 : *Decision Tree* step 1

Dari pembagian data didapatkan bahwa setiap hari mendung, mahasiswa misterius X berlari di saraga dan tidak jika hari tersebut hujan. Kedua subset ini adalah contoh dari subset murni karena tidak perlu ada pembagian lagi. Untuk cuaca cerah masih perlu ada pembagian lagi. Subset cuaca cerah setelah pembagian pertama adalah sebagai berikut :

Hari	Cuaca	Kelembapan	Angin	Lari
4	Cerah	Normal	Kuat	1
6	Cerah	Normal	Kuat	1
7	Cerah	Normal	Lemah	1
13	Cerah	Tinggi	Lemah	0
14	Cerah	Tinggi	Lemah	0

Tabel 2 : Subset dari dataset pertama

Dari subset cuara cerah dapat dibagi lagi menjadi dua subset sesuai dengan kelembapan di hari tersebut. *Decision Tree* yang dihasilkan adalah sebagai berikut:

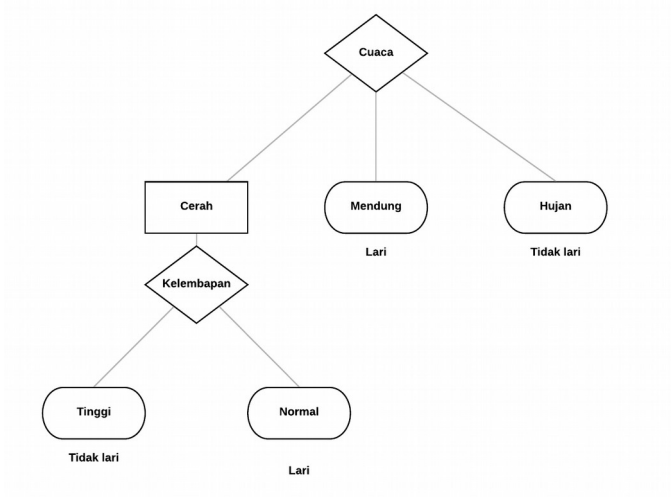
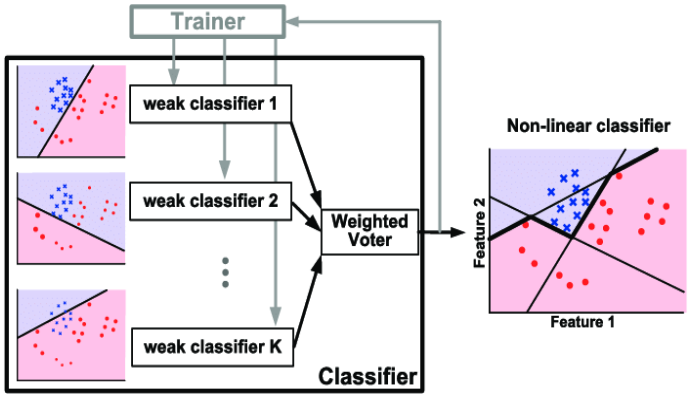


Diagram 4 : *Decision Tree* step 2

Setelah pembagian di subset cuaca cerah semua subset yang dihasilkan adalah subset murni dan tidak perlu dilakukan pembagian lebih lanjut. Dengan *Decision Tree* tersebut, kita dapat memprediksi bahwa mahasiswa misterius X akan pergi berlari di hari 15 dengan cuaca mendung, kelembaban tinggi dan angin kuat.

B. Boosted Tree

Salah satu teknik yang digunakan dalam pembuatan model. Di dalam model ini, komputer membuat *decision tree* dan memperbaikinya secara bertahap dengan memperhatikan kesalahan yang dibuat dalam pembuatan *decision tree* sebelumnya saat membuat *decision tree* berikutnya.



Gambar 5 : Ilustrasi Boosted Tree (sumber : https://www.researchgate.net/figure/Illustration-of-AdaBoost-algorithm-for-creating-a-strong-classifier-based-on-multiple_fig9_288699540)

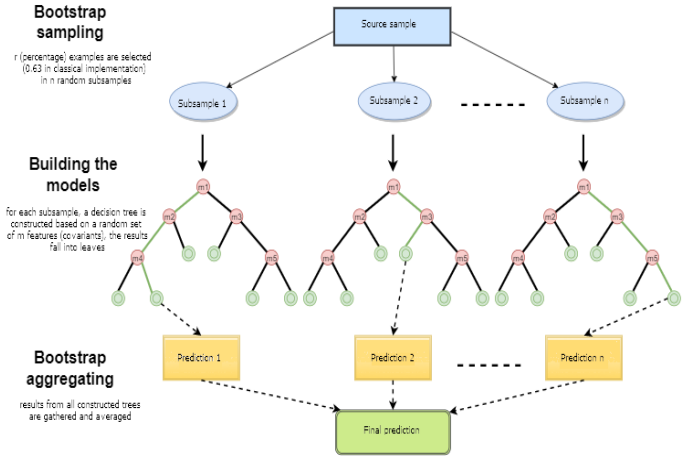
Salah satu contoh dari implementasi teknik ini adalah *Adaptive Boosting* atau disingkat *AdaBoost* yang diciptakan Yoav Freund dan Robert Schapire. Model ini dapat digunakan dengan model-model machine learning lain untuk meningkatkan performa. Keluaran dari model-model lain digabung dengan jumlah berbobot yang merepresentasikan keluaran dari model *AdaBoost*. Kata *Adaptive* berasal dari kemampuan model ini untuk mengadaptasikan model-model yang memiliki performa lebih lemah menjadi model yang lebih kuat. Sayangnya model ini sensitif terdapat data yang memiliki presisi rendah dan pencicilan sehingga dapat menghasilkan model yang *overfit* terhadap suatu dataset. Model-model di dalam *AdaBoost* mungkin lemah, tetapi saat digabung ke dalam *AdaBoost,* model itu akan konvergen ke model yang lebih kuat.

Overfitting adalah kesalahan pemodelan saat sebuah model hanya bagus untuk dataset tertentu dan bukan untuk data secara umum.

C. Random Tree Forest

Sama seperti *AdaBoost*, *Random Tree Forest* adalah teknik pembuatan model yang didasari oleh *decision tree*. Berbeda dengan *AdaBoost* dalam perihal melainkan menggunakan hasil

dari banyak *decision tree* untuk menghasilkan keluaran dengan merata-ratakan hasil dari semua *decision tree*.



Gambar 6 : Ilustrasi *Random Tree Forest* (sumber : https://c.mql5.com/2/33/image1__1.png)

Random tree forest pertama diciptakan oleh Tin Kam Ho menggunakan metode ruang bagian acak[4]. Metode *Random Tree Forest* pertama kali diusulkan oleh Ho pada tahun 1995.

Random Tree Forest pertama dibuat dengan memecah dataset menjadi sampel yang unik untuk setiap *decision tree* yang akan dibuat. Setelah itu, *estimator* (*decision tree* dalam konteks *random tree forest*) dibuat sesuai dengan sampel data yang unik untuk dirinya mereka sendiri. Sampel data tersebut dipilih secara random sehingga muncul kata *random* di dalam *random tree forest*. Setelah *estimator* selesai diciptakan, akan dilakukan *bootstrap aggregating*. *Bootstrap aggregating* atau yang biasa disebut *bagging/majority voting* adalah proses untuk mencari mean dari hasil prediksi masing estimator di dalam *random tree forest*. Secara matematis dapat ditulis sebagai berikut :

$$\hat{f} = \frac{1}{B}\sum_{b=1}^{B} f_b(x')$$

Dengan f_b *estimator* ke-b, B jumlah *estimator, x'* adalah dataset yang digunakan untuk melatih model.

Proses ini dapat menghasilkan model yang lebih baik karena mengurangi variansi dalam model tanpa mengurangi bias. Berbeda dengan *AdaBoost* yang sensitif terhadap data yang memiliki presisi rendah karena yang sensitif hanyalah beberapa pohon di dalam *random tree forest* tetapi secara keseluruhan tidak sensitif asalkan setiap *estimator* di dalam *random tree forest* tidak berkorelasi antara satu sama lain.

Dari model yang didapatkan, ketidakpastian sebuah model dapat didapatkan dengan persamaan sebagai berikut :

$$\sigma = \sqrt{\frac{\sum_{b=1}^{B}(f_b(x') - \hat{f})^2}{B-1}}.$$

Random Subspace Method pada dasarnya adalah *random tree forest* yang dimodifikasi dengan implementasi algoritma pembelajaran yang memilih fitur secara random dengan pengganti untuk setiap *estimator*. Secara lokal, teknik ini membuat setiap *estimator* untuk tidak terlalu fokus terhadap fitur-fitur yang sangat deskriptif dan prediktif di dalam dataset pembelajaran dengan tujuan menghindari overfitting didalam *estimator* secara lokal.

Random Tree Forest adalah gabungan bagging dan *Random Subspace Method* menjadi sebuah model utuh.

D. Aplikasi dalam dunia nyata

Beberapa model yang sudah disebutkan didalam makalah ini dapat diaplikasikan kedalam beberapa hal. Salah satu contoh riil adalah pengesahan credit core sebuah bank kepada nasabahnya. Selain dalam modeling, model *decision tree* dan *random tree forest* sudah diimplementasikan kedalam library python yang bernama scikit learn. Model-model yang tergolong sederhana ini dapat digunakan sebagai langkah awal mempelajari model-model machine learning yang lebih kompleks

IV. KESIMPULAN

Di dalam dunia ini, banyak hal disimplifikasi menjadi objek diskrit untuk mempermudah pemrosesan. Salah satu aplikasi tersebut adalah model *Random Tree Forest* dan *AdaBoost* yang sering dipakai dalam dunia *data science*. Kedua model tersebut adalah aplikasi nyata dari teori graf dalam perihal pemodelan dan prediksi data.

V. UCAPAN TERIMAKASIH

Saya, sebagai penulis ingin mengucapkan terima kasih kepada :

1. Bapak Dr. Ir. Rinaldi Munir, M. T., Ibu Dra. Harlili S., M. Sc., Ibu Fariska Zakhralativa Ruskanda, S.T., M.T., atas bimbingannya dalam mata kuliah IF2120 Matematika Diskirt terutama untuk Pak Munir yang telah mengajar di K-01
2. Keluarga dan teman-teman yang turut membantu dan mendukung saya dalam menjalani perkuliahan
3. Ilmuwan lampau yang telah menciptakan fondasi machine learning sekarang.

REFERENCES

[1] https://www.seagate.com/files/www-content/our-story/trends/files/idc-seagate-dataage-whitepaper.pdf (Diakses Desember 2, 2019)

[2] Munir,Rinaldi.Bahan Ajar Mata KuliahIF2120 Matematika Diskrit, Program Studi Teknik Informatika, Institut Teknologi Bandung.Bandung.

[3] Breiman, Leo; Friedman, J. H.; Olshen, R. A.; Stone, C. J. (1984). *Classification and regression trees*. Monterey, CA: Wadsworth & Brooks/Cole Advanced Books & Software. ISBN 978-0-412-04841-8.

[4] Ho, Tin Kam (1995). *Random Decision Forests* (PDF). Proceedings of the 3rd International Conference on Document Analysis and Recognition, Montreal, QC, 14–16 August 1995. pp. 278–282

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, 6 Desember 2019

Jon Felix Germinian/13518025