# LANDASAN TEORI

Pada bab ini dijelaskan mengenai semua teori yang digunakanuntuk membuat aplikasi *analytics tools* dengan fitur *recommend* metode prediksi. Beberapa teori utama yang digunakan untuk membuat aplikasi ini adalah *Classification, Regression, Business Intelligence, Git*. Berikut adalah penjelasan mengenai teori-teori tersebut.

## *Business Intelligence*

### Pengertian *Business Intelligence*

*Business Intelligence* adalah sebuah sistem terintegrasi yang memberikan fakta / informasi dari data yang telah diolah untuk kepentingan pengambilan keputusan. Sistem ini bertugas untuk memberikan dukungan keputusan (*Decision Suport*)untuk tujuan yang spesifik pada sebuah proses bisnis. Fondasi dari *Business Intelligence* adalah data yang telah diolah menjadi informasi yang berguna untuk proses pendukungan keputusan. *Business Intelligence* menggunakan bergabai metode untuk melakukan ekstraksi informasi dari data yang ada. *Business* *Intelligence* memberikan informasi pada waktu yang tepat, kepada orang yang tepat dalam bentuk yang mudah dipahami pula.

### Tugas / *Task* dalam *Business Intelligence*

Untuk mencapai tujuan dalam melakukan analisa ada beberapa tugas / *task* yang harus dilalui. Beberapa tugas / *task* itu adalah:

1. Data *Task*

Tugas ini adalah bagaimana mendapatkan data dan apa pengertian / informasi mengenai data tersebut. Informasi yang didapatkan saat melakukan *Business Intelligence* semua berasal dari data, maka tugas ini perlu dilakukan untuk mengetahui sumber data.

1. *Business and Data Understanding*

Tugas ini dilakukan untuk memahami dan mengerti apa yang dimaksut dengan data yang dimiliki. Selain itu pengertian mengenai proses bisnis juga diperlukan untuk memahami bagaimana data tergenerasi.

1. *Modeling Task*

Tugas ini dilakukan untuk mengetahui model analisa yang ingin dilakukan. Model analisa ini digunakan untuk menghasilkan informasi yang dibutuhkan.

1. *Analysis Task*

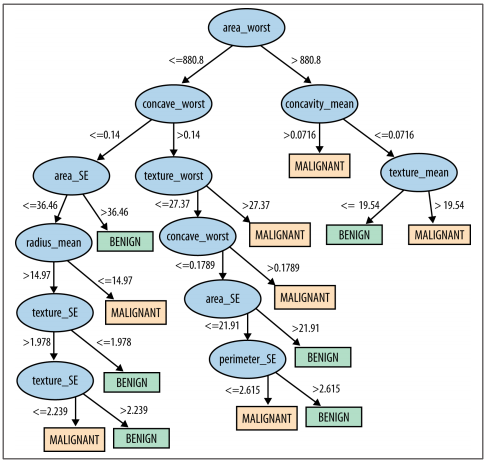
Tugas ini adalah implementasi dari *Modeling Task* dimana algoritma digunakan untuk menghasilkan informasi. Tugas ini adalah tugas utama dari *Business Intelligence* untuk menghasilkan fakta / pengetahuan mengenai bisnis.

1. Evaluation & Reporting Task

Tugas ini adalah tugas terakhir dalam *Business Intelligence* tugas ini dilakukan untuk melakukan evaluasi mengenai performa yang dihasilkan model yang dibuat. Pada tahap ini visualisasi dari data juga diperlukan untuk proses  *reporting.* Visualisasiyang baik dapat menghasilkan informasi yang mudah untuk dipahami.

## *Classification*

*Classification* merupakan metode yang mencoba untuk melakukan prediksi dengan melakukan klasifikasi data. Klasifikasi mencari data tersebut milik *class* yang mana. Salah satu contoh pertanyaan *classification* adalah “Dari semua *customer* siapa yang akan merespon penawaran yang akan diberikan?”. Dalam kasus ini terdapat 2 *class* yaitu respon dan tidak respon. Salah satu contoh metode yang dapat melakukan *classification* adalah *decision tree. Decision tree* merupakan metode yang memodelkan data menjadi bentuk sebuah *tree* dimana data baru yang yang masuk akan berjalan dan menghasilkan sebuah alur yang menyatakan pada *class* mana data tersebut terklasifikasi. (Provost & Fawcett, 2013)



1. Decision Tree

## *Regression*

*Regression* atau *value estimation* mencoba untuk melakukan estimasi atau meprediksi sebuah nilai *numeric* variabel pada sekumpulan data. Salah satu contoh pertanyaan *regression*  adalah “Berapa banyak *customer* yang akan menggunakan layanan kita”. Properti variabel yang ingin diprediksi adalah jumlah penggunaan layanan dan model analisa dapat terbuat dengan melihat variable lain yang sejenis.

*Regression* berhubungan dengan *Classification* tapi bukan merupakan sesuatu hal yang sama. *Classification* mencoba untuk memprediksi apakah sesuatu akan terjadi atau tidak, sedangkan *Regression* mencoba untuk memprediksi berapa banyak yang akan dihasilkan. (Provost & Fawcett, 2013)

Rumus *Regression* adalah:

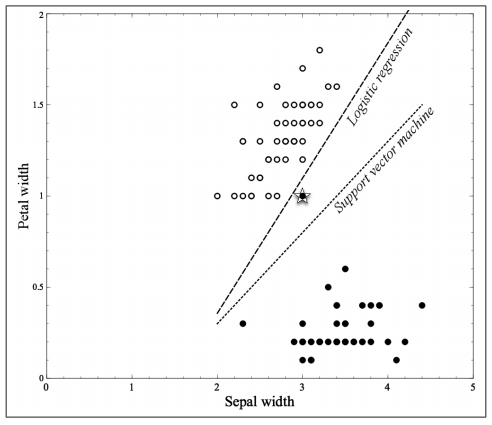
**Y1 = X11 β1 + X12 β2 + … + X1k βk + ε1**

**Y2 = X21 β1 + X22 β2 + … + X2k βk + ε2**

**…**

**Yn = Xn1 β1 + Xn2 β2 + … + Xnk βk + εn**

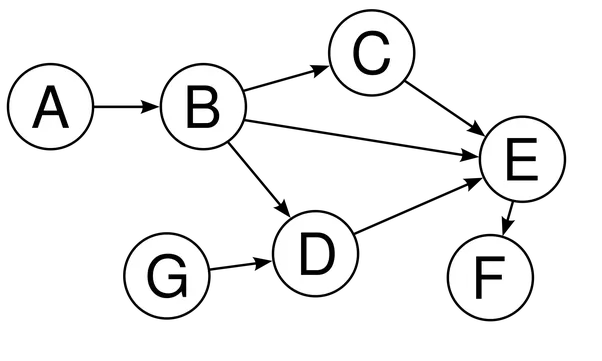
*Regression* dapat digambarkan dalam bentuk grafik garis linear.



1. Linear Regression

## *Directed Acyclic Graph*

*Directed Acyclic Graph* adalah *graph* yang tidak memiliki siklus. Pada implementasinya, tidak ada *node* yang menuju ke *node* sebelumnya. Karena tidak ada node yang kembali menuju *node* sebelumnya, maka pada graph ini tidak bida terjadi siklus sebuah node kembali menuju *node* sebelumnya kemudian kembali menuju *node* itu sendiri. (Allison, 1999) *Directed Acyclic Graph* (DAG) dapat digunakan untuk menggambarkan mata kuliah yang diambil oleh mahasiswa atau dapat menggambarkan managemen proyek, proses yang harus di lakukan untuk menghasilkan hasil akhir.



1. Gambar *Directed Acyclic Graph*

## *Breath First Search*

*Breath First Search* (BFS) merupakan algoritma *graph transversal* yang dimana tujuan dari algoritma ini dalah menelusuri *graph* dan melewati setiap *node* dengan urutan tertentu. Urutan yang dijalankan oleh BFS adalah dimulai dari *root* kemudian *node* dimasukan kedalam *queue* kemudian queue mulai di proses dengan mencari *node* yang dapat dikunjungi oleh *root*, kemudia *node – node* tersebut dimasukan kedalam *queue*. Proses ini berjalan sampai *queue* tersebut kosong. (Garg, 2016)

## *Colaborative Analytics*

*Colaborative Analytics*  adalah sebuah metode yang digunakan untuk meningkatakan hasil akurasi berdasarkan banyak *predictive model.* Ketika data training siap diproses, data di proses dengan lebih dari 1 *predictive model*. Sebagai contoh, jika ada sebuah data yang ingin di proses dengan metode *classification*, maka data tersebut diproses dengan banyak *classifier*. *Classifier* terbaik akan dipilih sebagai *classifier* yang akan digunakan untuk mem-prediksi data *training*. (Chong, et al., 2012)

## JSON (*Javascript Object Notation*)

JSON merupakan sebuah format data yang ringan dan mudah dipahami, biasanya digunakan untuk melakukan pertukaran data antar mesin. JSON mudah dipahami oleh manusia dan mudah di-*parse* oleh mesin. JSON dibangun dengan struktur:

* Kumpulan pasangan nama / nilai. Dalam berbagai bahasa, ini diwujudkan sebagai objek, catatan, struct, kamus, tabel hash, daftar kunci, atau array asosiatif.
* Daftar nilai yang terurut. Dalam kebanyakan bahasa, ini diwujudkan sebagai array, vektor, daftar, atau urutan.

## *Git*

*Git* merupakan sebuah sistem pengontrolan versi terdistribusi. Tujuan utama dari git adalah memberikan sebuah tempat (disebut *repository*) dimana pengguna dapat mengerjakan sebuah proyek secara bersama – sama. Pada aplikasi ini ketika *user /* pengguna ingin melakukan analisa maka *user* akan membuat sebuah *project* baru dimana *project*  ini dalam dibagi (*share*)dengan pengguna lainya. Git adalah solusi yang dipakai untuk mengatasi masalah tersebut.

*Version Control* atau pengaturan versi merupakan sebuah sistem untuk memberikan versi pada setiap file di *repository*. Keunggulan *Version Control* adalah pengguna dapat kembali pada versi yang lebih rendah lalu melanjutkan menuju versi yang lebih baru (*Downgrade*). Git merupakan *Version Control* yang terdistribusi dimana pengguna yang menjalankan *project* tidak hanya 1, melainkan lebih dari 1.



1. Git Distributed Version Control

Beberapa fitur yang digunakan pada git adalah:

1. *Clone*

Merupakan fitur yang digunakan untuk melakukan duplikat penuh terhadap *repository* yang ada pada server.

1. *Push*

Merupakan kegiatan untuk mengirimkan data baru / hasil kegiatan yang dilakukan pada komputer lokal menuju server.

1. *Pull*

Merupakan kegiatan yang dilakukan untuk mengambil / melakukan duplikat perubahan yang terjadi pada server menuju *project* pada komputer lokal.

1. *Branch*

Merupakan fitur yang digunakan untuk mengatur perubahan yang dilakukan oleh kolabolator *project.* Ketika kolaborator melakukan perubahan maka pemilik *project* dapat memutuskan untuk menerima perubahan itu dan mengubah total sesuai dengan perubahan yang terjadi atau menambahkan perubahan itu pada *project* yang ada sekarang atau bahkan menolak perubahan tersebut.

## *Gitlab*

Gitlab merupakan aplikasi berbasis web yang digunakan untuk membuat sebuah *server* menjadi *git server*. Aplikasi ini memiliki kumpulan REST API (*Representational State Aplication Program Interface*) yang dapat digunakan untuk melakukan integrasi dengan program lainya. Aplikasi ini berjalan *self-hosted* atau menggunakan *hosting* sendiri. Aplikasi Gitlab dapat di-*install* pada ubuntu server dan berjalan sebagai website (Port 80).

## *Flask Framework*

*Flask* merupakan *microframework* yang dibangun dengan menggunakan bahasa pemrograman *Python*. *Flask* digunakan untuk me-*develop* sebuah aplikasi web. *Flask* merupakan *microframework* yang artinya *flask* membuat sebuah pengerjaan aplikasi web menjadi mudah dan *simple* karena dapat menjalankan sebuah web hanya dengan menggunakan 1 file *Python*. *Flask* membuat susunan kerja yang ringan, dan mudah tetapi juga dapat dikembangkan dengan mudah. (Flask, 2010) Beberapa keuntungan menggunakan *flask* adalah:

* Mudah / *Simple* dalam penngkodean
* Pengembangan aplikasi tergolong mudah
* Menggunakan Jinja2 *templating* dimana desain html dapat menjadi lebih leluasa dan mudah, karena dapat memanggil fungsi *Python* kedalam *template* html.