Prediksi Kesuksesan *Marketing Bank* pada Lembaga Perbankan Portugis dengan *Decision Tree* dan *Random Forest*

Dina Aprilia Aktarani[1, a)], Diva Kemala[1, b)], Fitriana Murtafiah[2, c)], Julius Satya Ratnand [1, d)], Maristy Widya Pangestika[1, e)], Tiara Yosianti Solekhah [1, f)]

Afiliasi Penulis

[1] *Mahasiswa S1 Program Studi Statistika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Gadjah Mada, Sekip Utara BLS 21 Yogyakarta, Indonesia*

[2] *Mahasiswa S1 Program Studi Aktuaria, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Gadjah Mada, Sekip Utara BLS 21 Yogyakarta, Indonesia*

Email Penulis

a)*dinaaktarani12@mail.ugm.ac.id*, b)*divakemala@mail.ugm.ac.id*, c) *fitriana.murtafiah@mail.ugm.ac.id*, d)*julius.satya.r@mail.ugm.ac.id*, f) *maristy.widya@mail.ugm.ac.id*, g)*tiara.y@mail.ugm.ac.id*

Abstract. Bank didefinisikan sebagai badan usaha yang menghimpun dana dari masyarakat dalam bentuk simpanan dan menyalurkannya kepada masyarakat dalam bentuk kredit atau bentuk lain untuk membantu masyarakat dalam meningkatkan taraf hidup. Dalam perbankan diperlukan teknik pemasaran, data pemasaran, dan lain-lain. Terdapat dua pendekatan utama bagi perusahaan untuk mempromosikan produk dan jasa yakni melalui kampanye massal yang menargetkan publik tanpa memandang bulu atau melalui pemasaran bertaget yang menargetkan serangkaian kontak tertentu. Seringnya terjadi persaingan internal dan krisis keuangan menyebabkan munculnya banyak tekanan pada bank-bank Eropa untuk meningkatkan aset keuangan. Untuk mengetahui pelanggan potensial untuk pinjaman agar tidak terjadi kredit bermasalah yang menimbulkan risiko utama bagi pemberi kredit, diperlukan adanya analisis klasifikasi perbandingan terhadap data marketing. Oleh karena itu, penelitian ini bertujuan untuk mengklasifikasi data marketing Bank pada lembaga perbankan Portugis sehingga memudahkan pihak bank dan bagian marketing dalam memprediksi nasabah yang akan berlangganan produk deposito berjangka atau tidak menggunakan data *Bank Marketing* dari situs *Kaggle*. Metode yang akan digunakan adalah metode *decision tree* dan *metode random forest*.

Kata Kunci : *marketing bank*, perbankan, *decision tree*, *random forest*

BAB I

PENDAHULUAN

A. Latar Belakang

Berdasarkan Undang-undang Perbankan No. 10 Tahun 1998, Bank didefinisikan sebagai badan usaha yang menghimpun dana dari masyarakat dalam bentuk simpanan dan menyalurkannya kepada masyarakat dalam bentuk kredit atau bentuk lain untuk membantu masyarakat dalam meningkatkan taraf hidup. Bank umum sendiri merupakan bank yang melakukan kegiatan usaha secara tradisional atau berdasarkan prinsip syariah dan memberikan jasa penyimpanan, pembayaran, dan peminjaman kepada masyarakat dalam kegiatannya. Dalam perbankan diperlukan teknik pemasaran, data pemasaran, dan lain-lain. Bank adalah salah satu jenis lembaga keuangan yang menyediakan jasa seperti simpanan, peminjaman, pemasaran, dan lain-lain.

Terdapat dua pendekatan utama bagi perusahaan untuk mempromosikan produk dan jasa yakni melalui kampanye massal yang menargetkan publik tanpa memandang bulu atau melalui pemasaran bertaget yang menargetkan serangkaian kontak tertentu (Ling dan Li, 1998). Saat ini, di dunia yang kompetitif secara global, respon positif terhadap kampanye massal biasanya sangat kecil yakni kurang dari 1% menurut studi yang sama. Pendekatan lain seperti fokus pemasaran terarah pada target yang diharapkan akan lebih tertarik pada produk/jasa tertentu, membuat kampanye jenis ini lebih menarik karena lebih efesien (Ou et al, 2003). Namun, pemasaran terarah memiliki kelemahan seperti menyebabkan sikap negatif terhadap bank karena gangguan privasi (Page dan Luding, 2003).

Karena persaingan internal dan krisis keuangan, terdapat banyak tekanan pada bank-bank Eropa untuk meningkatkan aset keuangan. Untuk menangani masalah ini dapat dengan menjalankan strategi dengan penawaran jangka panjang yang menarik permintaan deposit dengan tingkat bunga yang baik, terutama melalui kampanye pemasaran terarah. Terdapat kebutuhan untuk meningkatkan efisiensi seperti kontak yang lebih sedikit untuk dilakukan tetapi perkiraan jumlah keberhasilan bisa lebih besar.

Pembiayaan kredit adalah penyediaan dana berdasarkan kesepakatan antara bank dan nasabah, yang mewajibkan peminjam untuk membayar setelah jangka waktu tertentu. Bagian pemasaran harus memilih pelanggan potensial untuk pinjaman dengan beberapa pertimbangan. Hal ini diperlukan agar tidak terjadi kredit bermasalah yang menimbulkan risiko utama bagi pemberi kredit. Salah satu cara untuk menyiasatinya adalah dengan menggunakan model klasifikasi dalam data mining. Klasifikasi adalah proses mengubah record menjadi sekumpulan kelas yang sama (Umadevi dan Marseline, 2017).

Analisis klasifikasi data mining terdiri dari penentuan kumpulan data baru dalam salah satu dari beberapa kategori yang telah ditentukan, juga dikenal sebagai supervised learning. Metode yang dikembangkan peneliti untuk menyelesaikan kasus klasifikasi antara lain: Decision Tree, Naïve Bayes, Jaringan Syaraf Tiruan, Analisis Statistik, Algoritma Genetik,

Rough Sets, kNearest Neighbour, Metode Berbasis Aturan, Memory Based Reasoning, dan Support Vector Machine.

Klasifikasi Decision Tree adalah salah satu teknik yang paling terkenal dalam penambangan data dan salah satu metode paling populer untuk menentukan keputusan suatu kasus. Metode ini tidak memerlukan proses pengelolaan pengetahuan terlebih dahulu dan dapat menyelesaikan kasus dengan dimensi yang besar.

Metode Random Forest adalah pengembangan dari metode CART, yaitu dengan menerapkan metode bootstrap aggregating (bagging) dan random feature selection (Breiman 2001). Dalam Random Forest, banyak pohon ditumbuhkan sehingga terbentuk hutan (forest), kemudian analisis dilakukan pada kumpulan pohon tersebut. Menurut berbagai penelitian, Random Forest dapat digunakan untuk mengklasifikasikan data dalam jumlah besar yang tidak seimbang dengan memberikan hasil kinerja yang baik dengan waktu eksekusi yang cepat.

Berdasarkan uraian di atas, kami akan melakukan analisis klasifikasi perbandingan antara Decision Tree dan Random Forest terhadap data marketing pada lembaga perbankan Portugis. Analisis klasifikasi ini dilakukan dengan menggunakan data yang diperoleh dari situs kaggle, yakni data marketing pada lembaga perbankan Portugis dengan 16 atribut dan tidak ada missing value. Tujuan klasifikasi adalah untuk memprediksi apakah nasabah akan berlangganan deposito berjangka (variabel y). Data pada penelitian ini nantinya akan dikaitkan dengan kampanye pemasaran lembaga perbankan Portugis. Kampanye pemasaran didasarkan pada panggilan telepon. Data tersebut pertama kali digunakan pada paper berjudul *A Data-Driven Approach to Predict the Success of Bank Telemarketing* [Moro et al., 2014]. Data ini dikumpulkan pada periode bulan Mei 2008 hingga November 2010. Metode yang akan kami gunakan adalah metode *decision tree dan* metode *random forest.*

B. Tujuan dan Manfaat

- Mengimplementasikan *supervised learning* dalam persoalan dunia nyata, yakni klasifikasi data marketing Bank pada lembaga perbankan Portugis dengan menggunakan Decision Tree dan Random Forest.
- Mengetahui klasifikasi data marketing bank pada lembaga perbankan Portugis sehingga memudahkan pihak bank dan bagian marketing dalam memprediksi nasabah yang akan berlangganan produk deposito berjangka atau tidak.

BAB II

LANDASAN TEORI

1. *Supervised Learning*

Supervised learning merupakan salah satu metode algoritma yang digunakan dalam pembelajaran mesin atau *machine learning*. *Supervised learning* digunakan pada *dataset* atau kumpulan data yang memiliki label untuk melatih algoritma untuk mengklasifikasikan data atau memprediksi hasil secara akurat. Metode pembelajaran ini menggunakan data latih untuk dipelajari oleh algoritma, dengan tujuan untuk menemukan hubungan atau struktur tertentu pada data masukan atau *input* agar memungkinkan algoritma tersebut untuk menghasilkan suatu model dan data keluaran atau *output* yang benar secara efektif. Data latih yang digunakan sudah mencakup *input* dan *output* yang benar, sehingga memungkinkan model untuk belajar dari waktu ke waktu. Algoritma tersebut juga dapat mengukur keakuratannya dengan fungsi kerugian, yang terus menyesuaikan hingga kesalahan yang terjadi dapat diminimalkan. *Supervised learning* dapat dibedakan menjadi dua jenis, yakni klasifikasi dan regresi.

1.1. Analisis Klasifikasi

Analisis klasifikasi adalah jenis *supervised learning* yang menggunakan algoritma untuk mengidentifikasi dan menetapkan data uji ke dalam kategori tertentu dengan tujuan memberikan hasil analisis yang lebih akurat. Algoritma dalam analisis klasifikasi belajar untuk mengenali entitas atau variabel tertentu dalam *dataset* dan mencoba menarik kesimpulan tentang bagaimana variabel tersebut diberi label atau didefinisikan. Contoh algoritma klasifikasi yang umum adalah *k-nearest neighbor (KNN), support vector machines (SVM), decision tree*, dan *random forest*.

1.2. Analisis Regresi

Analisis regresi adalah jenis *supervised learning* yang menggunakan algoritma untuk mengidentifikasi dan memahami hubungan antara variabel dependen dan variabel independen. Analisis regresi sering digunakan untuk membuat prediksi, seperti mengestimasi besarnya penjualan perusahaan berdasarkan beberapa faktor yang dianggap memiliki pengaruh. Contoh algoritma regresi yang populer adalah regresi linier dan regresi logistik.

2. *Decision Tree*

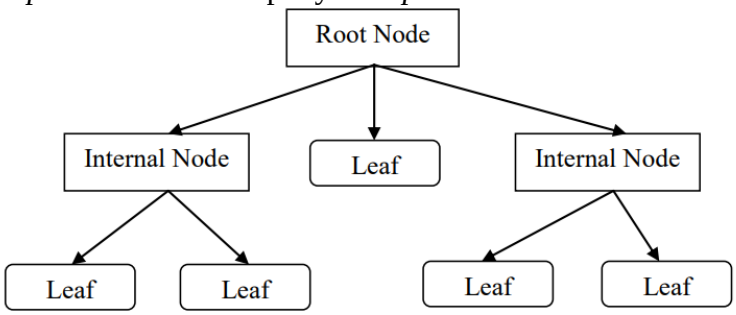
Decision tree atau yang biasa juga disebut pohon keputusan adalah sebuah struktur pohon dimana setiap cabangnya menunjukkan pilihan di antara sejumlah alternatif pilihan yang ada dan setiap daunnya menunjukkan keputusan yang dipilih. *Decision tree* sering digunakan untuk mendapatkan informasi dengan tujuan pengambilan keputusan. Salah satu kelebihan *decision tree* yaitu daerah pengambilan keputusan yang sebelumnya kompleks dan sangat luas, dapat diubah menjadi bentuk yang lebih simpel dan spesifik.

Decision tree juga berguna untuk mengeksplorasi data, menemukan hubungan tersembunyi antara sejumlah calon variabel *input* dengan sebuah variabel *output*. *Decision tree* memadukan eksplorasi data dan pemodelan, sehingga cocok digunakan sebagai langkah awal dalam proses pemodelan atau dijadikan sebagai model akhir dari beberapa teknik lain. Kelebihan lainnya, metode ini mampu mengeliminasi perhitungan atau data-data yang kiranya tidak diperlukan. Sebab, sampel yang ada biasanya hanya diuji berdasarkan kriteria atau kelas tertentu saja. Kekurangan dari *decision tree* ini adalah bisa terjadi *overlap*, terutama ketika kelas dan kriteria yang digunakan sangat banyak yang dapat meningkatkan waktu pengambilan keputusan sesuai

dengan jumlah memori yang dibutuhkan. Dalam hal akumulasi, *decision tree* juga seringkali mengalami kendala *error* terutama dalam jumlah besar. Selain itu, terdapat pula kesulitan dalam mendesain *decision tree* yang optimal, mengingat kualitas keputusan yang didapatkan dari metode *decision tree* sangat tergantung pada bagaimana pohon tersebut didesain. Meskipun memiliki banyak kekurangan, *decision tree* tetap banyak digunakan dalam berbagai macam pengolahan data.

Pada *decision tree* terdapat 3 jenis *node* atau jaringan, yaitu :

a. *Root Node*, merupakan *node* yang terletak paling atas. Pada *node* ini tidak ada *input* dan bisa tidak mempunyai *output* atau mempunyai *output* lebih dari satu.
b. *Internal Node*, merupakan *node* percabangan. Pada *node* ini hanya terdapat satu *input* dan mempunyai minimal dua *output*.
c. *Leaf Node* atau *terminal node*, merupakan *node* paling akhir. Pada *node* ini hanya terdapat satu *input* dan tidak mempunyai *output*.



Gambar alur pembentukan model decision tree

Dalam membangun sebuah model *decision tree*, tahap awal yang dilakukan adalah melakukan evaluasi semua variabel yang ada menggunakan suatu ukuran statistik. Tujuannya adalah untuk mengukur efektivitas suatu variabel dalam mengklasifikasikan suatu kumpulan sampel data. Ukuran statistik yang biasa digunakan adalah *information gain* atau nilai *gain*. Variabel yang diletakkan pada *root node* adalah variabel yang memiliki nilai *gain* terbesar. Penghitungan nilai *gain* dapat dilakukan dengan konsep *entropy*, *gini index*, dan *classification error*. Penggunaan *information gain* digunakan untuk mencari satu variabel dari kumpulan data yang dimiliki untuk dijadikan *root node*. Sesudah menemukan variabel yang menjadi *root node*, proses berlanjut dengan melakukan perhitungan nilai *gain* kembali, untuk mencari variabel yang menjadi cabang (*internal node*) hingga menemukan *leaf node* yang merupakan label kelas. Berikut adalah rumus penghitungan *information gain* dengan tiga konsep yang ada.

a. *Information Gain* dengan nilai *entrophy*
   Entropy adalah formula untuk menghitung homogenitas variabel (A) dari sebuah sampel data (S).

$$Entropy\ (S) = \sum_{i=1}^{n} - p_i * log_2\ p_i$$

Dengan :
S = Himpunan kasus dalam *dataset*
n = Jumlah partisi S
pi = Proporsi dari Si terhadap S

Kemudian dapat dihitung nilai *gain*.

$$Gain\ (S, A) = Entropy\ (S) - \sum_{i=1}^{n} \frac{|S_i|}{|S|} * Entropy\ (S_i)$$

Dengan :
S = Himpunan kasus dalam *dataset*
n = Jumlah partisi variabel A
$|S_i|$ = Jumlah kasus pada partisi ke-i
$|S|$ = Jumlah kasus dalam S

b. *Information Gain* dengan nilai *gini index*

$$Gini(S) = 1 - \sum_{i=1}^{C} (p_i)^2$$

Dengan :
C = Jumlah nilai dari masing-masing variabel
Pi = Jumlah masing-masing variabel berdasarkan label.

Kemudian dapat dihitung nilai *gain*.

$$Gain\ (S, A) = Gini\ (S) - \sum_{i=1}^{n} \frac{|S_i|}{|S|} * Gini\ (S_i)$$

Dengan :
S = Himpunan kasus dalam *dataset*
n = Jumlah partisi variabel A
$|S_i|$ = Jumlah kasus pada partisi ke-i
$|S|$ = Jumlah kasus dalam S

c. *Information Gain* dengan nilai *classification error*

$$Error(S) = 1 - max\ \{P_i\}$$

Dengan :
Pi = Jumlah masing-masing variabel berdasarkan label

Kemudian dapat dihitung nilai *gain*.

$$Gain\ (S, A) = Error\ (S) - \sum_{i=1}^{n} \frac{|S_i|}{|S|} * Error\ (S_i)$$

Dengan :
S = Himpunan kasus dalam *dataset*
n = Jumlah partisi variabel A
$|S_i|$ = Jumlah kasus pada partisi ke-i
$|S|$ = Jumlah kasus dalam S

Secara garis besar, alur pembentukan *decision tree* adalah sebagai berikut.

a. Pohon dibentuk dengan menentukan *root node* yang merepresentasikan semua data.
b. Data yang terdapat pada *root node* kemudian diukur dengan nilai *information gain* untuk menentukan variabel mana yang menjadi variabel cabang (*internal node*)

c. Setelah dipilih sebuah cabang, data didistribusikan ke cabang masing-masing. Algoritma ini akan terus menggunakan proses yang sama (rekursif) untuk dapat membentuk sebuah *decision tree*. Ketika sebuah variabel telah dipilih menjadi cabang, maka variabel tersebut tidak disertakan lagi dalam penghitungan nilai *information gain*.
d. Proses berulang tersebut akan berhenti jika salah satu dari kondisi di bawah ini terpenuhi :
    1. Semua data dari anak cabang telah termasuk dalam kelas yang sama.
    2. Semua variabel telah dipakai, tetapi masih tersisa data dalam kelas yang berbeda. Dalam kasus ini, diambil data yang mewakili kelas yang terbanyak untuk menjadi label kelas pada *leaf node*.
    3. Tidak terdapat data pada anak cabang yang baru. Dalam kasus ini, *leaf node* akan dipilih pada cabang sebelumnya dan diambil data yang mewakili kelas terbanyak untuk dijadikan label kelas.

3. *Random Forest*

Random forest adalah algoritma dalam *machine learning* yang digunakan untuk pengklasifikasian data set dalam jumlah besar. *Random forest* bisa digunakan untuk banyak dimensi dengan berbagai skala dan performa yang tinggi. Algoritma ini menggunakan *decision tree* untuk melangsungkan proses seleksi, di mana *tree* atau pohon *decision tree* akan dibagi secara rekursif berdasarkan data pada kelas yang sama. Dalam hal ini, penggunaan *tree* yang semakin banyak akan mempengaruhi akurasi yang didapat menjadi lebih optimal. Penentuan klasifikasi dengan *random forest* dilakukan berdasarkan hasil *voting* dan *tree* yang terbentuk.

Dua konsep yang menjadi dasar dari *random forest* adalah membangun *ensemble* dari *tree* via *bagging* dengan *replacement* dan penyeleksian fitur secara acak untuk tiap *tree* yang dibangun. Pertama, setiap sampel yang diambil dari kumpulan data untuk *training tree* bisa dipakai lagi untuk *training tree* yang lain. Kedua, fitur yang digunakan pada saat *training* untuk tiap *tree* merupakan subset dari fitur yang dimiliki oleh kumpulan data tersebut (Wibowo et al., 2016).

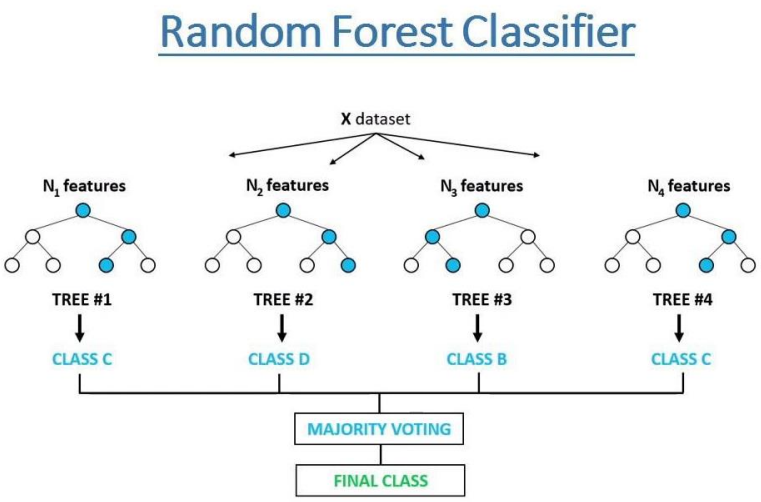
Klasifikasi berbasis *ensemble* akan mempunyai performa yang maksimal jika antar *basic learner* mempunyai korelasi yang rendah. Sebuah *ensemble* harus membangun *basic learner* yang lemah, karena *learner* yang kuat kemungkinan besar akan mempunyai korelasi yang tinggi dan biasanya juga menyebabkan *overfit*, sedangkan *random forest* meminimalkan korelasi serta mempertahankan kekuatan klasifikasi dengan cara melakukan pengacakan pada proses *training*, yaitu dengan memilih sejumlah fitur secara acak dari semua fitur yang ada pada setiap melakukan *training tree*, kemudian menggunakannya menggunakan fitur-fitur yang terpilih untuk mendapatkan percabangan *tree* yang optimal. Berbeda dengan proses *training tree* pada *decision tree* biasa, proses *training tree* yang menjadi bagian dari *random forest* tidak menggunakan proses *pruning* akan tetapi percabangan akan terus dilakukan sampai ukuran batas *leaf* tercapai (Wibowo et al., 2016).

Kelebihan dari metode *random forest* adalah menghasilkan error yang relatif rendah, memberikan hasil yang lebih akurat dalam klasifikasi, dapat mengatasi data latih dalam jumlah sangat besar secara efisien, efektif untuk mengestimasi hilangnya data, dapat menentukan variabel apa saja yang penting dalam klasifikasi, dan mampu mendeteksi interaksi variabel. Sementara beberapa kekurangan dari metode ini adalah, membutuhkan waktu pemrosesan yang cukup lama karena ukuran data yang besar dan membangun *tree* dalam jumlah yang

banyak, interpretasi yang relatif sulit, dan tidak terlalu efektif memprediksi di luar kisaran data latih.

Secara garis besar langkah kerja algoritma *random forest* adalah sebagai berikut.

a. Algoritma memilih sampel *random* dari kumpulan data yang dimiliki. Tahapan ini disebut sebagai tahapan *bootstrap*.
b. Dengan menggunakan hasil dari *bootstrap*, algoritma akan membangun pohon yang merupakan *decision tree* untuk setiap sampel yang dipilih. Apabila diinginkan hutan dengan *k* pohon, maka proses poin (a) dan (b) akan dilakukan berulang sebanyak *k*.
c. Hasil prediksi akan didapatkan dari setiap *decision tree* yang dibuat. *Voting* kemudian dilakukan untuk setiap hasil prediksi. Umumnya, pada masalah klasifikasi akan digunakan sistem *majority voting*. Sistem *voting* ini akan memilih hasil prediksi akhir berdasarkan suara terbanyak yang diperoleh.
d. Algoritma akan memilih hasil prediksi akhir dengan sistem *majority voting*.



Gambar alur kerja pembentukan random forest

BAB III

METODE PENELITIAN

A. Data Penelitian

Data yang digunakan dalam penelitian ini merupakan data sekunder atau data yang diperoleh dari sumber yang sudah tersedia. Digunakan *dataset Bank Marketing* yang diperoleh dari situs *kaggle* pada laman https://www.kaggle.com/datasets/henriqueyamahata/bank-marketing. Data tersebut pertama kali digunakan pada paper berjudul *A Data-Driven Approach to Predict the Success of Bank Telemarketing* [Moro et al., 2014]. Data terdiri dari empat kumpulan data. Kumpulan data yang digunakan dalam penelitian ini memiliki jumlah observasi sebanyak 41188, variabel bebas sebanyak 20, dan 1 variabel terikat. Data ini dihimpun pada periode bulan Mei 2008 hingga November 2010.

B. Variabel Penelitian

Dalam penelitian ini akan digunakan 21 variabel yang terdapat pada kumpulan data yang dimiliki, terdiri dari 20 variabel bebas dan 1 variabel terikat. Penjabaran untuk masing-masing variabel adalah sebagai berikut.

Variabel Bebas

- Age : usia nasabah (numerik)
- Job : jenis pekerjaan nasabah (kategorik)
- Marital : status perkawinan nasabah (kategorik)
- Education : pendidikan terakhir nasabah (kategorik)
- Default : apakah nasabah memiliki kredit default (kategorik)
- Housing : apakah nasabah memiliki pinjaman perumahan (kategorik)
- Loan : apakah nasabah memiliki pinjaman pribadi (kategorik)
- Contact : tipe komunikasi dengan nasabah (kategorik)
- Month : bulan dilakukannya komunikasi terakhir dengan nasabah (kategorik)
- Day of week : hari dilakukannya komunikasi terakhir dengan nasabah (kategorik)
- Duration : durasi saat dilakukan komunikasi terakhir, dalam satuan detik (numerik)
- Campaign : banyaknya komunikasi yang dilakukan selama kampanye *marketing* (numerik)
- Pdays : jumlah hari setelah nasabah terakhir kali dihubungi dari kampanye sebelumnya (numerik)
- Previous : banyaknya komunikasi yang dilakukan sebelum kampanye ini (numerik)
- Poutcome : hasil dari kampanye *marketing* sebelumnya (kategorik)
- Emp.var.rate : tingkat variasi pekerjaan, dengan indikator triwulanan (numerik)
- Cons.price.idx : indeks harga konsumen, dengan indikator bulanan (numerik)
- Cons.conf.idx : indeks kepercayaan konsumen, dengan indikator bulanan (numerik)
- Euribor3m : tarif *euribor* 3 bulan, dengan indikator harian (numerik)
- Nr.employed : jumlah karyawan, dengan indikator triwulanan (numerik)

Variabel Terikat

- y : apakah nasabah berlangganan produk deposito berjangka (kategorik : ‘yes’ dan ‘no’)

C. Langkah Analisis

Langkah-langkah analisis yang dilakukan dalam penelitian ini adalah sebagai berikut.

1. Melakukan proses *importing* data
2. Melakukan prapemrosesan dan visualisasi data
3. Melakukan analisis dengan metode *decision tree*
4. Melakukan analisis dengan metode *random forest*
5. Menarik kesimpulan berdasarkan hasil analisis kedua metode

BAB IV

HASIL DAN PEMBAHASAN

Dalam kasus ini akan dilakukan prediksi kesuksesan marketing bank pada lembaga perbankan Portugis menggunakan Decision Tree dan Random Forest. Terlebih dahulu, dilakukan *import library* dan data yang diperlukan.

```
!pip install shap
```

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.metrics import confusion_matrix, classification_report
from sklearn.model_selection import train_test_split
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import StratifiedKFold

import shap
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
bank = pd.read_csv("/content/drive/MyDrive/bank-additional.csv")
```

Prapemrosesan dan Visualisasi Data

Dilakukan pengecekan tipe data terhadap masing-masing variabel sebagai berikut.

```
bank.dtypes
```

```
age               int64
job              object
marital          object
education        object
default          object
housing          object
loan             object
contact          object
month            object
day_of_week      object
duration          int64
campaign          int64
pdays             int64
previous          int64
poutcome         object
emp.var.rate    float64
cons.price.idx  float64
cons.conf.idx   float64
euribor3m       float64
nr.employed     float64
y                 object
dtype: object
```

Dapat dilihat bahwa terdapat beberapa variabel yang memiliki tipe *object* yaitu variabel *job, marital, education, default, housing, loan, contact, month, day_of_week, poutcome,* dan *y*. Sehingga perlu dilakukan pembentukan variabel *dummy*.

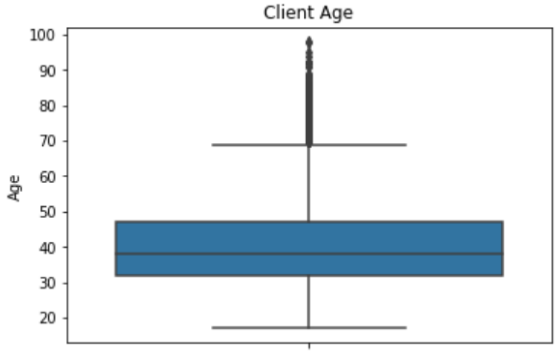
Selanjutnya, dilakukan pengecekan *missing value* pada data sebagai berikut.

```
bank.isnull().sum()
```

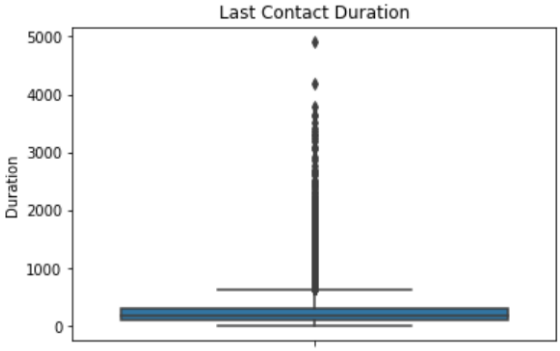
```
age               0
job               0
marital           0
education         0
default           0
housing           0
loan              0
contact           0
month             0
day_of_week       0
duration          0
campaign          0
pdays             0
previous          0
poutcome          0
emp.var.rate      0
cons.price.idx    0
cons.conf.idx     0
euribor3m         0
nr.employed       0
y                 0
dtype: int64
```

Dari *output* di atas terlihat bahwa pada data tidak terdapat *missing value* ditandai dengan angka 0 pada setiap variabel sehingga dapat dilanjutkan untuk analisis berikutnya.

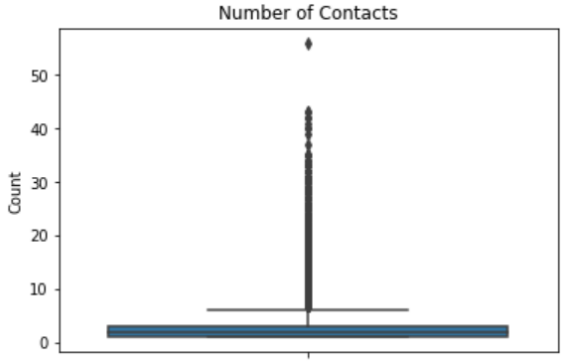
Variabel yang bertipe integer dan float divisualisasikan dalam bentuk boxplot sebagai berikut.



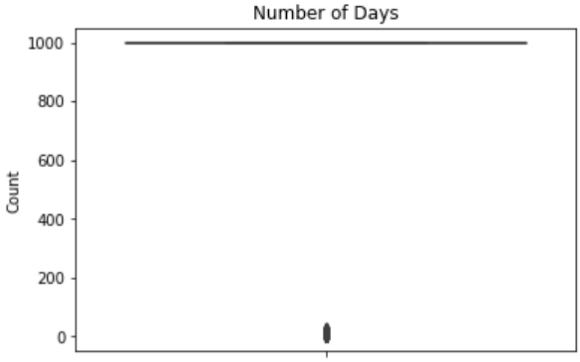
Berdasarkan boxplot Client Age di atas terlihat bahwa terdapat data outlier yaitu data yang nilainya jauh berbeda daripada data lainnya. Bentuk boxplot menjurai ke atas yang artinya nasabah bank yang berusia 40 tahun ke atas lebih banyak.



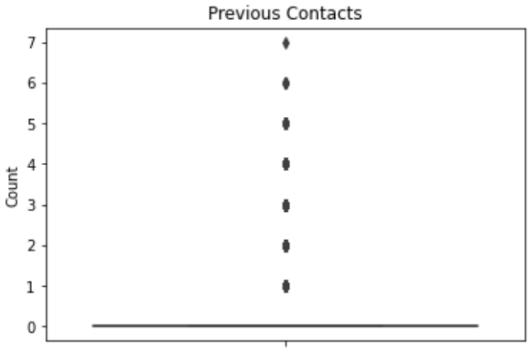
Pada boxplot Last Contact Duration di atas, terlihat bahwa terdapat data outlier. Bentuk boxplot menjurai ke atas yang artinya bank lebih sering mengontak nasabah dengan durasi yang lama. Durasi tersebut sangat mempengaruhi apakah nasabah berlangganan produk deposito berjangka nantinya.



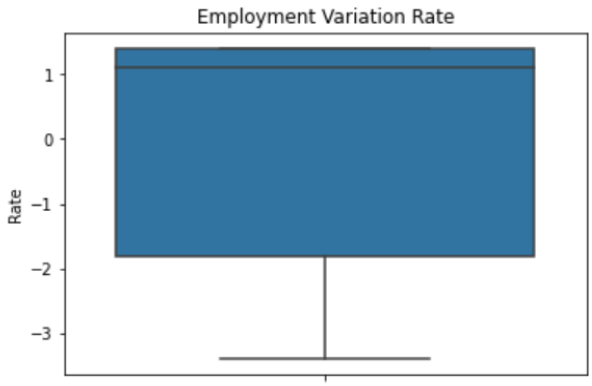
Berdasarkan boxplot Number of Contacts di atas terlihat bahwa terdapat data outlier. Bentuk boxplot menjurai ke atas yang artinya bank sering berkomunikasi dengan nasabahnya untuk melakukan kampanye.



Pada boxplot Number of Days di atas, terlihat bahwa terdapat data outlier. Bentuk boxplot tidak berpola karena nilai kuartil 1, kuartil 2, dan kuartil 3 sama yaitu 999. Angka 999 yang artinya sebagian besar nasabah sebelum kampanye tidak dihubungi.



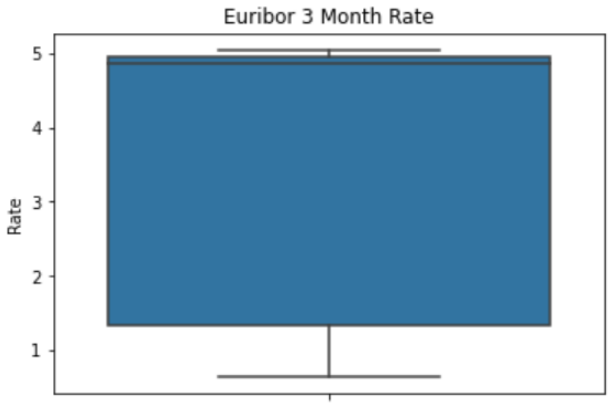
Berdasarkan boxplot Previous Contacts di atas terlihat bahwa terdapat data outlier. Bentuk boxplot tidak berpola karena nilai kuartil 1, kuartil 2, dan kuartil 3 sama yaitu 0 yang artinya sebagian besar nasabah bank tidak menerima panggilan dari bank sebelum kampanye dilakukan.



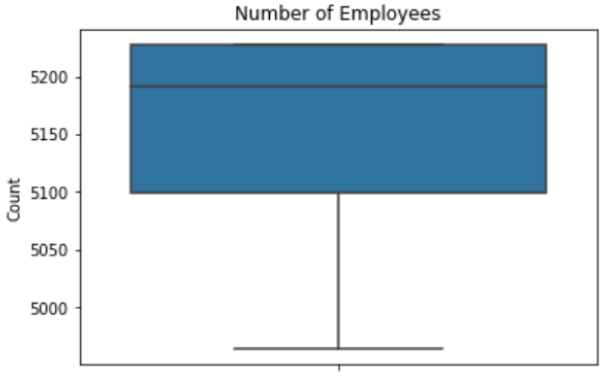
Pada boxplot Employment Variation Rate di atas terlihat bahwa tidak terdapat data outlier. Bentuk boxplot menjurai ke bawah yang artinya sebagian besar pekerja mempunyai tingkat variasi pekerjaan yang rendah.

Berdasarkan boxplot Consumer Price Index di atas terlihat bahwa tidak terdapat data outlier. Bentuk menjurai ke bawah yang artinya indeks harga konsumen yang rendah lebih menyebar.

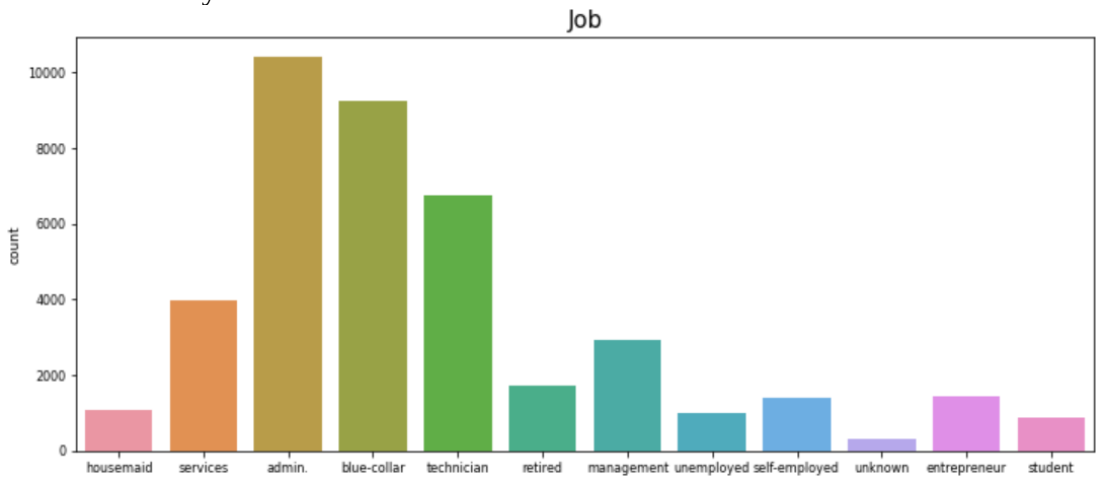
Pada boxplot Consumer Confidence Index di atas terlihat bahwa terdapat data outlier. Bentuk boxplot menjurai ke atas yang artinya indeks kepercayaan konsumen yang tinggi lebih menyebar.



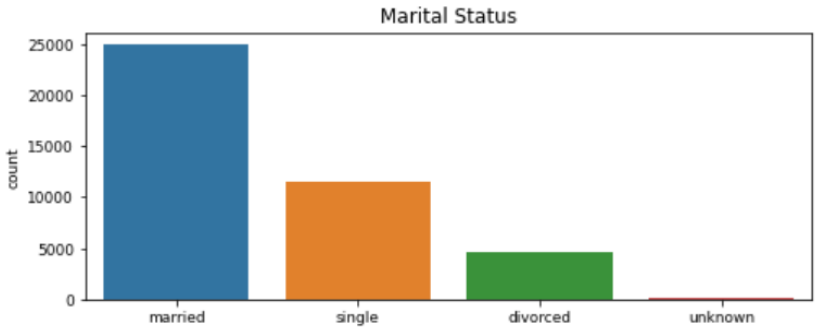
Berdasarkan boxplot Euribor 3 Month Rate di atas terlihat bahwa tidak terdapat data outlier. Bentuk menjurai ke bawah yang artinya tarif *euribor* 3 bulan yang rendah lebih menyebar.



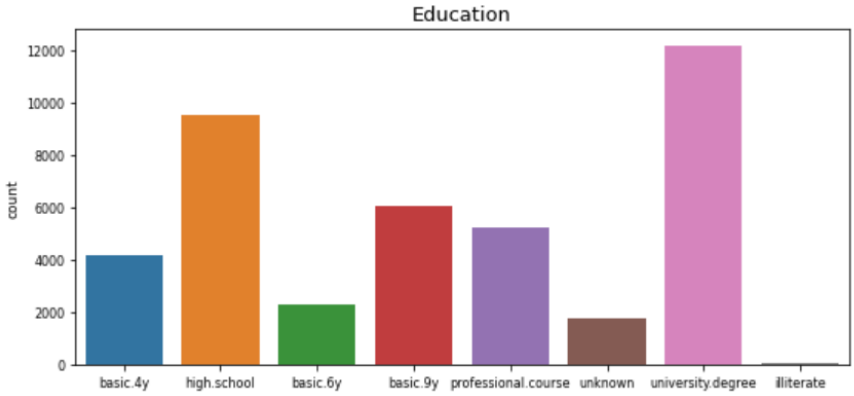
Pada boxplot Number of Employees di atas terlihat bahwa tidak terdapat data outlier. Bentuk boxplot menjurai ke bawah yang artinya jumlah karyawan yang sedikit dengan indikator triwulanan lebih menyebar.



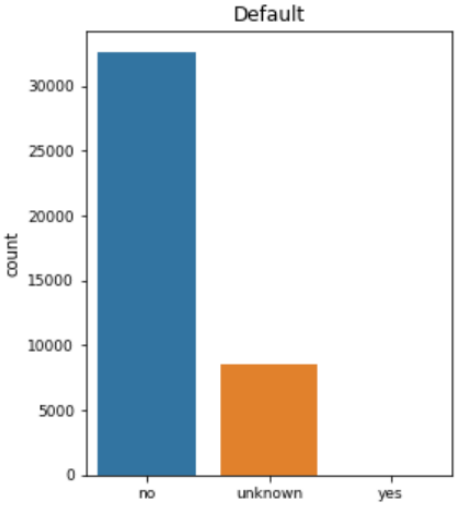
Berdasarkan diagram batang Job di atas terlihat bahwa pekerjaan nasabah bank yang paling dominan pertama yaitu admin, dominan kedua yaitu karyawan, dan dominan ketiga yaitu teknisi.



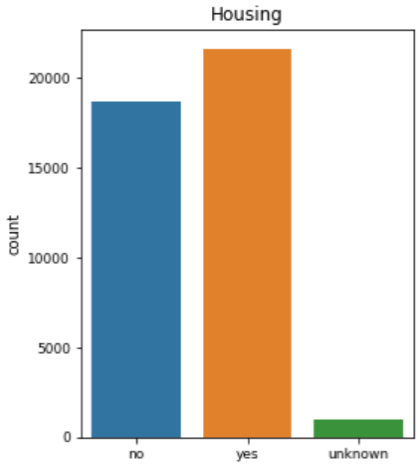
Berdasarkan diagram batang Marital Status di atas terlihat bahwa sebagian besar nasabah bank merupakan orang yang sudah menikah.



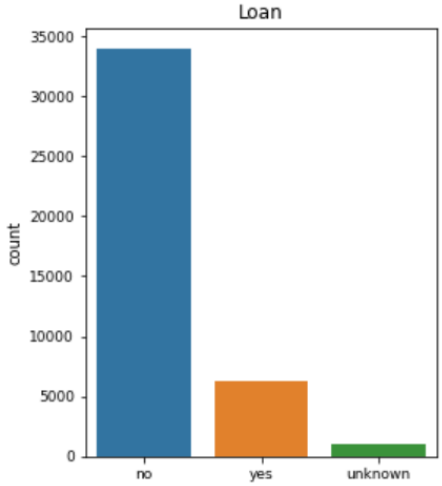
Berdasarkan diagram batang Education di atas terlihat bahwa sebagian nasabah bank merupakan mahasiswa dan siswa SMA.



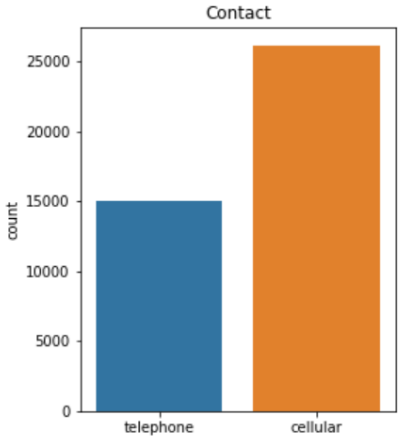
Berdasarkan diagram batang Default di atas terlihat bahwa sebagian besar nasabah bank tidak memiliki kredit terhadap bank.



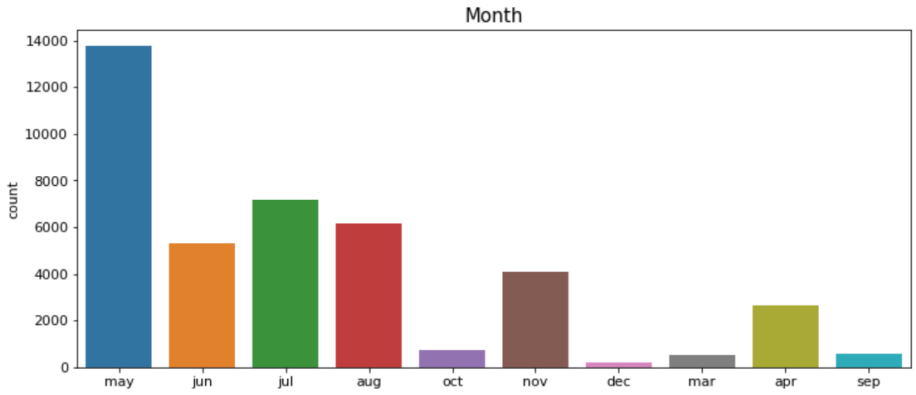
Berdasarkan diagram batang Housing di atas terlihat bahwa jumlah nasabah bank yang memiliki pinjaman perumahan lebih banyak daripada yang tidak memiliki pinjaman perumahan.



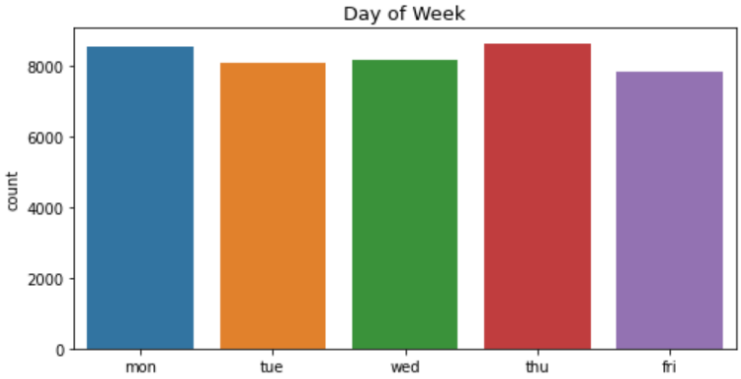
Berdasarkan diagram batang Loan di atas terlihat bahwa sebagian besar nasabah bank memiliki pinjaman pribadi.



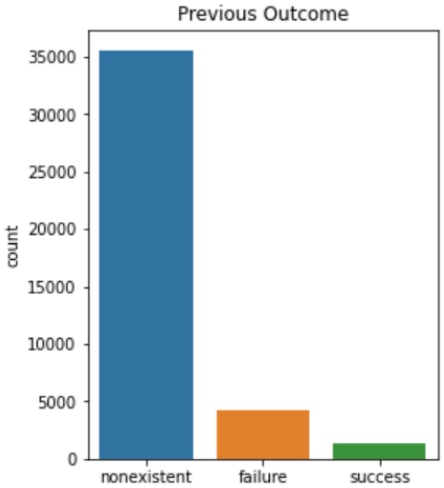
Berdasarkan diagram batang Contact di atas terlihat bahwa nasabah yang dihubungi bank melalui seluler lebih banyak daripada telepon.



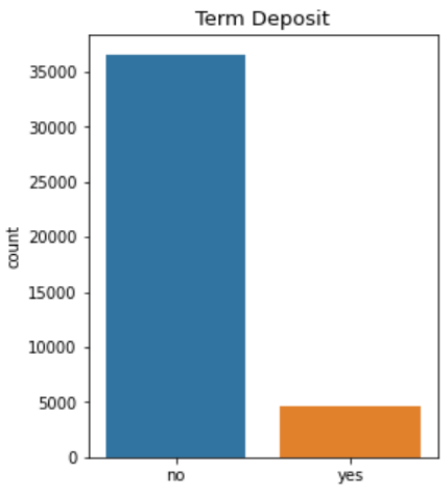
Berdasarkan diagram batang Month di atas terlihat bahwa bulan dilakukannya komunikasi terakhir dengan nasabah sebagian besar dilakukan pada bulan Mei, Juli, dan Agustus.



Berdasarkan diagram batang Day of Week di atas terlihat bahwa hari dilakukannya komunikasi terakhir dengan nasabah sebagian besar dilakukan pada hari Kamis.



Berdasarkan diagram batang Previous Outcome di atas terlihat bahwa hasil dari kampanye *marketing* sebelumnya sebagian besar tidak ada.



Berdasarkan diagram batang Term Deposit di atas terlihat bahwa banyak nasabah yang tidak berlangganan produk deposito berjangka.

Dilakukan pembuangan variabel *contact, month,* dan *day of week* dikarenakan variabel tersebut dirasa kurang berpengaruh pada keputusan apakah nasabah akan berlangganan produk deposito berjangka yang dipasarkan atau tidak.

```
bank = bank.drop(['day_of_week', 'contact', 'month'], axis=1)
```

Kemudian, dilakukan pembentukan variabel dummy dengan fungsi *pd.get_dummies* pada semua variabel yang memiliki tipe *object.* Fungsi tersebut menghasilkan variabel dummy termasuk untuk *reference category*, sehingga dilakukan pembuangan variabel dummy yang merupakan *reference category* untuk mencegah terjadinya *multikolinearitas* sempurna

```
bank_dmy = pd.get_dummies(bank)
bank_dmy = bank_dmy.drop(['job_admin.', 'marital_married', 'education_university.degree', 'default_no', 'housing_no', 'loan_no', 'poutcome_nonexistent', 'y_no'],
bank.dmy = bank_dmy.rename(columns={'y_yes':'y'}, inplace = True) #yes = 1, no = 0
bank_dmy.head()
```

Setelah prapemrosesan data selesai, dilakukan penyimpanan data hasil prapemrosesan tersebut ke dalam bentuk file .csv yang baru. Hal ini dilakukan untuk mempermudah proses pemodelan dan analisis.

```
bank_dmy.to_csv('Data Preprocessed.csv', index=False)
```

```
bank_dmy = pd.read_csv('/content/drive/MyDrive/Data Preprocessed.csv')
```

Setelah itu, kita tentukan variabel independen (X) dan variabel dependennya (Y).

```
X = bank_dmy.drop(['y'], axis = 1)
Y = bank_dmy['y']
```

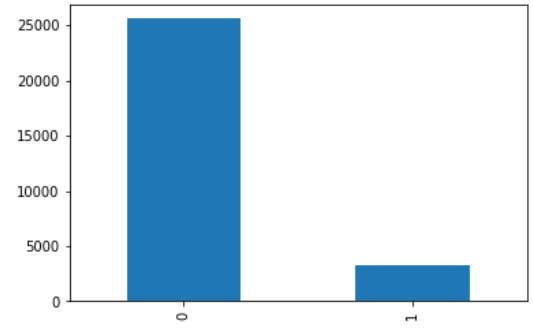
Lalu membagi data menjadi data latih dan data uji dengan ukuran data data ujinya 0.3.

```
#Untuk Data Klasifikasi
X_trainc, X_testc, Y_trainc, Y_testc = train_test_split(X, Y, test_size = 0.3)
```

Kemudian tinjau distribusi kelas pada data latih dan data uji. Akan dilihat label pada variabel dependen untuk data klasifikasi.

```
Y_trainc.value_counts().plot(kind = 'bar')
```

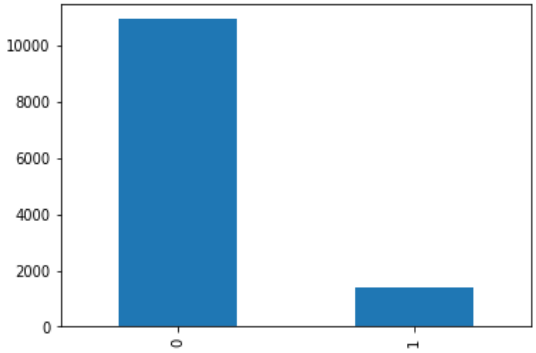
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fee48b3b350>
```



Berdasarkan *output* plot untuk variabel dependen data latih di atas, dapat dilihat bahwa terdapat perbedaan proporsi antara kedua klasifikasi. Sehingga dapat dikatakan bahwa jumlah label tidak seimbang.

```
Y_testc.value_counts().plot(kind = 'bar')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fee48a81f10>
```



Dari *output* plot untuk variabel dependen data uji di atas, dapat dilihat bahwa terdapat perbedaan proporsi antara kedua klasifikasi. Sehingga dapat dikatakan bahwa jumlah label tidak seimbang. Pada kedua output kelas mayoritas pada data latih dan data uji merupakan kelas yes (0), dan perbedaan dari kedua kelas cukup signifikan, sehingga performa model diperhatikan dari nilai F1 (F1-score) yang dihasilkan.

```
print(X_trainc.shape)
print(X_testc.shape)
print(Y_trainc.shape)
print(Y_testc.shape)
```

```
(28831, 39)
(12357, 39)
(28831,)
(12357,)
```

Untuk data klasifikasi, terdapat 28831 observasi dengan 39 variabel independen pada data latih dan 12357 observasi dengan 39 variabel independen untuk data uji.

```
np.random.seed(123)
```

Lalu gunakan *random seed* untuk menghasilkan data yang sama saat *syntax* dijalankan.

Proses Pemodelan

Decision Tree

Untuk membuat model decision tree di Python, kita dapat gunakan library sklearn lalu mengimport DecisionTreeClassifier dan plot_tree.

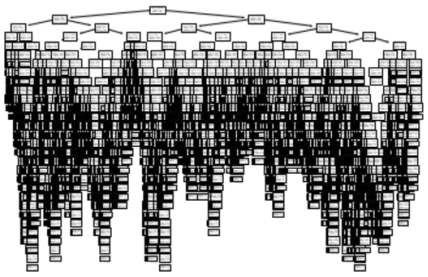
```
from sklearn.tree import DecisionTreeClassifier, plot_tree
```

Pertama dibentuk model *decision tree* dengan ukuran seleksi variabel *Information Gain* dengan sintaks sebagai berikut

```
clf1 = DecisionTreeClassifier(criterion = 'entropy') #Information Gain
clf1.fit(X_trainc,Y_trainc)
```

```
DecisionTreeClassifier(criterion='entropy')
```

Diperoleh plot tree dari klasifikasi yang telah dilakukan sebagai berikut.



Setelah itu dilakukan prediksi pada data uji. Lalu dibentuk *confusion matrix*.

```
Y_pred = clf1.predict(X_testc)
print(classification_report(Y_testc, Y_pred))
```

```
              precision    recall  f1-score   support

           0       0.94      0.94      0.94     10940
           1       0.54      0.54      0.54      1417

    accuracy                           0.89     12357
   macro avg       0.74      0.74      0.74     12357
weighted avg       0.89      0.89      0.89     12357
```

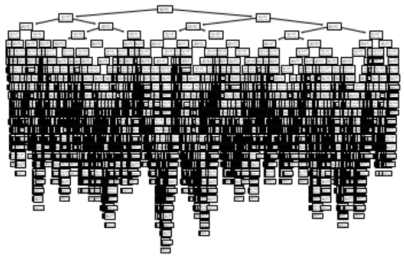
Berdasarkan *confusion matrix* tersebut, dapat dilihat bahwa f1-score yang dihasilkan adalah sebesar 0.94 (mengingat bahwa data tidak seimbang dan kelas mayoritas dari data adalah kelas 0), maka dapat dikatakan bahwa performa model decision tree ini sudah cukup baik.

Sebagai perbandingan, dibentuk model decision tree dengan ukuran seleksi variabel *Gini Index* sebagai berikut.

```
clf2 = DecisionTreeClassifier(criterion = 'gini') #Gini Index
clf2.fit(X_trainc,Y_trainc)
```

```
DecisionTreeClassifier()
```

Setelah itu dilihat plot tree dari klasifikasi.



Dilakukan prediksi pada data uji. Lalu dibentuk *confusion matrix*.

```
Y_pred = clf2.predict(X_testc)
print(classification_report(Y_testc, Y_pred))
```

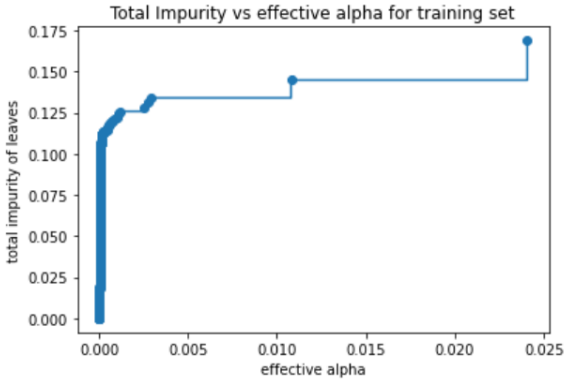
```
              precision    recall  f1-score   support

           0       0.94      0.94      0.94     10940
           1       0.53      0.52      0.52      1417

    accuracy                           0.89     12357
   macro avg       0.73      0.73      0.73     12357
weighted avg       0.89      0.89      0.89     12357
```

Dapat dilihat bahwa f1-score yang dihasilkan adalah sebesar 0.94. Didapatkan hasil yang sama dengan ukuran seleksi variabel *information gain*, sehingga dapat dikatakan bahwa model decision tree dengan ukuran seleksi variabel Gini Index menghasilkan performa yang sama baiknya.

Dari *plot tree* sebelumnya, terlihat bahwa “pohon” yang dihasilkan terlalu rimbun. Maka, salah satu solusinya adalah dengan melakukan prunning. Pertama dibentuk model decision tree seperti biasa, kemudian dihitung nilai ccp alpha lawan impurity yang dihasilkan pada data latih sebagai berikut.

```
#Post Pruning
clf = DecisionTreeClassifier()
path = clf.cost_complexity_pruning_path(X_trainc, Y_trainc)
ccp_alphas, impurities = path.ccp_alphas, path.impurities
```

```
fig, ax = plt.subplots()
ax.plot(ccp_alphas[:-1], impurities[:-1], marker = "o", drawstyle = "steps-post")
ax.set_xlabel("effective alpha")
ax.set_ylabel("total impurity of leaves")
ax.set_title("Total Impurity vs effective alpha for training set")
```

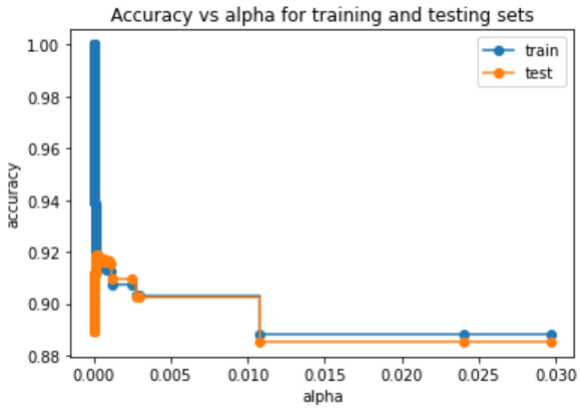


Kemudian, dibentuk model *decision tree* berdasarkan nilai ccp alpha yang diberikan, dan dibentuk plot nilai ccp alpha terhadap akurasi yang dihasilkan pada data latih dan data uji sebagai berikut.

```
clfs = []
for ccp_alpha in ccp_alphas:
    clf = DecisionTreeClassifier(ccp_alpha = ccp_alpha)
    clf.fit(X_trainc, Y_trainc)
    clfs.append(clf)
print(
    "Number of nodes in the last tree is: {} with ccp_alpha: {}".format(
        clfs[-1].tree_.node_count, ccp_alphas[-1]
    )
)
```

```
Number of nodes in the last tree is: 1 with ccp_alpha: 0.029695839650283717
```

```
train_scores = [clf.score(X_trainc, Y_trainc) for clf in clfs]
test_scores = [clf.score(X_testc, Y_testc) for clf in clfs]

fig, ax = plt.subplots()
ax.set_xlabel("alpha")
ax.set_ylabel("accuracy")
ax.set_title("Accuracy vs alpha for training and testing sets")
ax.plot(ccp_alphas, train_scores, marker = "o", label = "train", drawstyle = "steps-post")
ax.plot(ccp_alphas, test_scores, marker = "o", label = "test", drawstyle = "steps-post")
ax.legend()
plt.show()
```



Diperoleh nilai akurasi tertinggi pada data uji diperoleh pada nilai ccp alpha disekitar 0.0005. Oleh karena itu, dibentuk model *decision tree* dengan ukuran seleksi variabel *Gini Index* dengan nilai ccp alpha sebesar 0.0005 sebagai berikut.

```
clf2_1 = DecisionTreeClassifier(criterion = 'gini', ccp_alpha = 0.0005) #Gini Index
clf2_1.fit(X_trainc, Y_trainc)
```

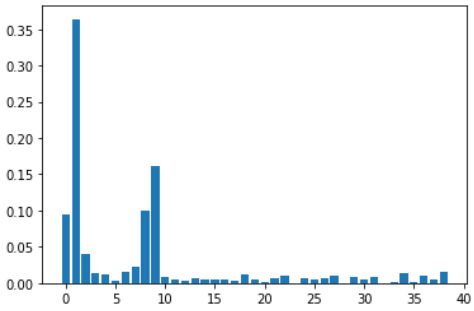
```
DecisionTreeClassifier(ccp_alpha=0.0005)
```

```
Y_pred = clf2_1.predict(X_testc)
print(classification_report(Y_testc, Y_pred))
```

```
              precision    recall  f1-score   support

           0       0.95      0.95      0.95     10940
           1       0.64      0.64      0.64      1417

    accuracy                           0.92     12357
   macro avg       0.80      0.79      0.80     12357
weighted avg       0.92      0.92      0.92     12357
```

Dapat dilihat bahwa f1-score yang dihasilkan adalah sebesar 0.95, sehingga dapat dikatakan bahwa pemangkasan pohon meningkatkan performa model *decision tree.*

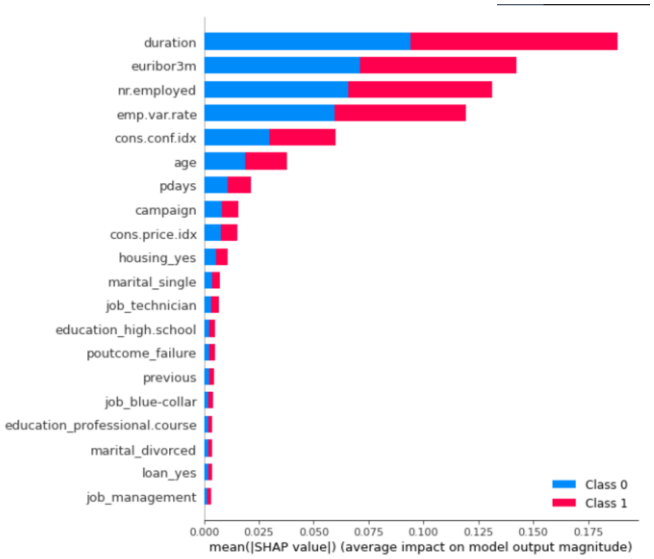
```
# get importance
importance = clf2.feature_importances_
# summarize feature importance
for i, v in enumerate(importance):
  print('Feature: %0d, Score: %.5f' % (i, v))
# plot feature importance
plt.bar([x for x in range(len(importance))], importance)
plt.show()
```



Pada output, diperoleh bahwa variabel pada kolom nomor 1 dan 9 memiliki nilai *variable importance* yang tertinggi, artinya variabel tersebut memiliki pengaruh yang signifikan terhadap model.

```
#SHAP value - Variable Importance
shap_values = shap.TreeExplainer(clf2).shap_values(X_trainc)
shap.summary_plot(shap_values, X_trainc, plot_type = "bar")
```

Plot shap di bawah ini dapat memplot nilai rata-rata shap untuk setiap kelas dengan daftar nilai shap seperti di bawah ini:



Perhatikan bahwa shap_values untuk dua kelas adalah invers aditif untuk masalah klasifikasi biner seperti yang tergambarkan pada output di atas.

Random Forest

Untuk membentuk model random forest di Python, dapat dilakukan dengan menggunakan fungsi RandomForestClassifier (untuk klasifikasi) yang di-import dari scikit-learn sebagai berikut.

```
from sklearn.ensemble import RandomForestClassifier
```

```
rf1 = RandomForestClassifier()
rf1.fit(X_trainc, Y_trainc)
```

```
RandomForestClassifier()
```

```
Y_pred = rf1.predict(X_testc)
print(classification_report(Y_testc, Y_pred))
```

```
              precision    recall  f1-score   support

           0       0.94      0.97      0.95     10940
           1       0.67      0.49      0.57      1417

    accuracy                           0.91     12357
   macro avg       0.81      0.73      0.76     12357
weighted avg       0.91      0.91      0.91     12357
```

Berdasarkan confusion matrix tersebut, dapat dilihat bahwa f1-score yang dihasilkan adalah sebesar 0.95, maka dapat dikatakan bahwa performa model *random forest* ini sudah cukup baik.

Dalam melakukan pemodelan dengan *machine learning*, salah satu tantangan yang dihadapi oleh peneliti adalah ketika data uji yang digunakan tidak memiliki nilai pada variabel dependen nya. Dengan kata lain, data yang akan diuji merupakan data yang belum pernah diobservasi sebelumnya (unseen data). Oleh sebab itu, digunakan metode stratified K-fold cross-validation sebagai solusi permasalahan ini karena data yang digunakan tidak seimbang.

```
#Cross Validation
rf1a = RandomForestClassifier()
cv = StratifiedKFold(n_splits = 10, shuffle = True, random_state = 1)
n_scores = cross_val_score(rf1a, X, Y, scoring = 'f1', cv = cv, n_jobs = -1, error_score = 'raise')
```

```
n_scores
```

```
array([0.60294118, 0.57462687, 0.53101737, 0.54233129, 0.56869773,
       0.54700855, 0.53015075, 0.56      , 0.59020311, 0.54846336])
```

```
print('F1-score: %.3f (%.3f)' % (np.mean(n_scores), np.std(n_scores)))
```

```
F1-score: 0.560 (0.023)
```

Berdasarkan output, dapat dilihat bahwa rata-rata nilai f1-score yang dihasilkan dengan metode stratified K-fold cross-validation adalah sebesar 0.560. Hasil ini dapat memberikan gambaran bahwa dengan mengasumsikan data uji yang baru memiliki karakteristik yang sama dengan data latih, maka performa yang dihasilkan oleh model masih belum cukup baik.

BAB V

KESIMPULAN

Berdasarkan hasil analisis menggunakan *decision tree* dan *random forest* untuk data *Bank Marketing* didapatkan setelah dilakukan pruning dibentuk model *decision tree* dengan ukuran seleksi variabel *Gini Index* dan nilai ccp alpha sebesar 0.0005 menghasilkan f1-score sebesar 0.95 yang artinya pemangkasan pohon meningkatkan performa model *decision tree*. Diperoleh pula, variabel pada kolom nomor variabel pada kolom nomor 1 dan 9 memiliki nilai *variable importance* yang tertinggi, artinya variabel tersebut memiliki pengaruh yang signifikan terhadap model. Sedangkan, untuk model *random forest* diperoleh f1-score sebesar 0.95 berdasarkan *confusion matrix*, namun dikarenakan data yang digunakan tidak seimbang maka untuk mengatasi permasalahan tersebut digunakan metode *K-fold cross-validation* dan menghasilkan f1-score sebesar 0.560 yang berarti performa yang dihasilkan model masih belum cukup baik. Sehingga, melihat dari hasil akhir f1-score dapat disimpulkan bahwa performa model *decision tree* menggunakan ukuran seleksi variabel *Gini Index* lebih baik dibandingkan performa model *Random Tree* metode *stratified K-fold cross validation*.

DAFTAR PUSTAKA

A. Parmar, R. Katariya and V. Patel, "A Review on Random Forest: An Ensemble Classifier," in International Conference on Intelligent Data Communication Technologies and Internet of Things, Springer, Cham, 2018.

A. S. More and D. P. Rana, "Review of Random Forest Classification Techniques to Resolve Data Imbalance," in International Conference on Intelligent Systems and Information Management, Aurangabad, India, 2017.

Chauhan, A. (2021). *Random Forest Classifier and its Hyperparameters*. Diakses pada 19 Mei 2022, dari https://medium.com/analytics-vidhya/random-forest-classifier-and-its-hyperparameters-8467bec755f6

David, D. (2020). *Random Forest Classifier Tutorial: How to Use Tree-Based Algorithms for Machine Learning*. Diakses pada 19 Mei 2022, dari https://www.freecodecamp.org/news/ how-to-use-the-tree-based-algorithm-for-machine-learning/

IBM Cloud Education. (2020). *Supervised Learning*. Diakses pada 18 Mei 2022, dari https://www.ibm.com/cloud/ learn/supervised-learning

IYKRA. (2018). *Mengenal Decision Tree dan Manfaatnya*. Diakses pada 18 Mei 2022, dari https://medium.com/iykra/mengenal-decision-tree-dan-manfaatnya-b98cf3cf6a8d

Jonathan (2021). *Implementasi Algoritma Random Forest untuk Klasifikasi Kategori Berita*. (Tesis Sarjana, Universitas Multimedia Nusantara, 2021) Diakses dari https://kc.umn.ac.id /16610/

Kasih, P. (2019). *Pemodelan Data Mining Decision Tree Dengan Classification Error Untuk Seleksi Calon Anggota Tim Paduan Suara*. 1(2), 63–69.

Khalimi, A. M. (2020). *Cara Menghitung Gini Index Algoritma C4.5 dengan Excel*. Diakses pada 18 Mei 2022, dari https://www.pengalaman-edukasi.com/2020/11/cara-menghitung-gini -index-algoritma.html

Pamela, S. (2019). *Pengolahan Data Traffic Pada Perangkat Internet Of Things Dengan Menggunakan Algoritma Random Forest*. (Tesis Sarjana, Universitas Siliwangi, 2019) Diakses dari http://repositori.unsil.ac.id/246/

Plaosan, S. V. (2019). *Random Forest*. Diakses pada 19 Mei 2022, dari http://learningbox.coffeecup.com/05_2_randomforest.html

Rasyid, A. (2014). *Sistem Prediksi Prestasi Akademik Mahasiswa Menggunakan Metode Decision Tree C4.5*. (Tesis Sarjana, Universitas Muhammadiyah Gresik, 2019) Diakses dari http://eprints.umg.ac.id/1533/

S. Moro, P. Cortez and P. Rita. A Data-Driven Approach to Predict the Success of Bank Telemarketing. Decision Support Systems (2014), doi:10.1016/j.dss.2014.03.001.

Sumathi, S. 2006. Introduction to Data Mining and Its Applications. Germany: Springer Verlag berlin Heidelberg.

Widodo, P. P., Handayanto, R. T. dan Herlawati. 2013. Penerapan Data Mining dengan Matlab. Bandung: Rekayasa Sains.

www.ojk.go.id. (n.d.). Perbankan. [online] Available at: https://www.ojk.go.id/id/kanal/perbankan/Pages/Bank-Umum.aspx.

Yamahata, Henrique. (2017). Bank Marketing, Kaggle. diakses pada 1 Mei 2022, https://www.kaggle.com/datasets/henriqueyamahata/bank-marketing