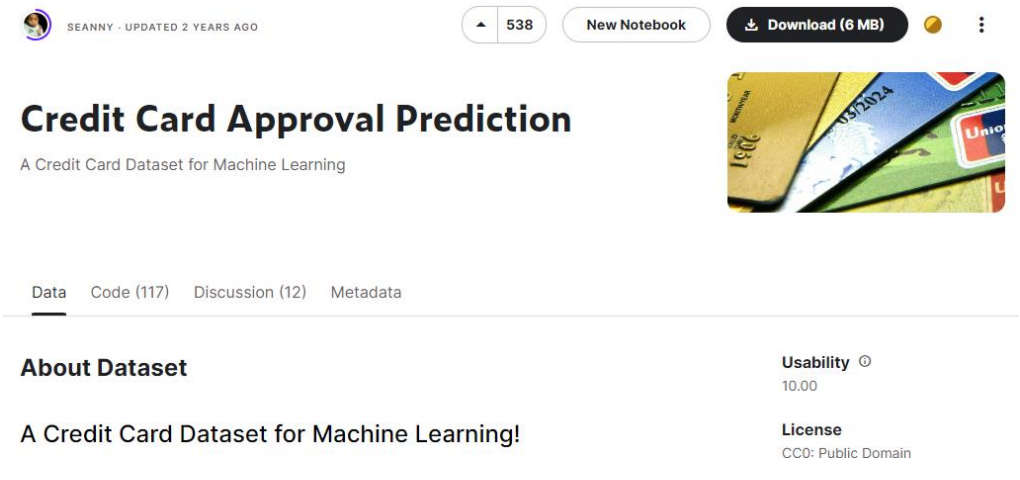
Nama : Vina
Jurusan : Ekonomi Pembangunan
Asal Kampus : Universitas Sriwijaya
Program : Accelerated Machine Learning

CREDIT SCORING IN BANKING

Credit scoring adalah sistem yang digunakan oleh pihak bank untuk menentukan apakah nasabah layak untuk diberikan pinjaman atau tidak. Dalam menentukan credit scoring diperlukan data profil nasabah. Semakin tepat data yang digunakan dan semakin banyak data yang didapatkan akan semakin akurat *credit scoring*nya. Di era serba cepat ini, proses tersebut akan membutuhkan waktu lama jika diproses secara konvensional. Di saat tingginya kompetisi di bidang finance menjadikan nasabah memiliki beragam tempat untuk meminjam. Oleh karena itu, diperlukan peran *machine learning* untuk mengolah dengan efisien dan memperoleh peminjam yang memiliki tingkat *credit scoring* yang tinggi. Melalui *machine learning* dapat mempermudah untuk mengetahui mana saja nasabah-nasabah yang layak untuk diberikan pinjaman.

Tahapan pertama yang dilakukan adalah dataset. Dataset yang akan digunakan adalah data credit card profilnya yang diperoleh dari *credit card approval prediction* dari kaggle.

Gambar 1. *Credit Card Approval Prediction*



Sumber: kaggle

Variabel yang digunakan adalah profil nasabah yang meliputi jenis kelamin, pendapatan tahunan, jumlah anak jika sudah berkeluarga, tingkat pendidikan, ataupun status pernikahan. Dari tahapan ini akan dilanjutkan dengan memprediksi apakah nasabah tersebut merupakan nasabah yang *good* atau *bad* credit.

Tahapan kedua adalah cleaning data. Ada empat tahapan dala, cleaning data yang meliputi:

1. Cek *missing value.*

Gambar 2. Kode Data Profil Nasabah

```
#>                 ID        CODE_GENDER        FLAG_OWN_CAR     FLAG_OWN_REALTY
#>                  0                  0                   0                   0
#>       CNT_CHILDREN    AMT_INCOME_TOTAL    NAME_INCOME_TYPE NAME_EDUCATION_TYPE
#>                  0                  0                   0                   0
#>  NAME_FAMILY_STATUS  NAME_HOUSING_TYPE         DAYS_BIRTH       DAYS_EMPLOYED
#>                  0                  0                   0                   0
#>         FLAG_MOBIL     FLAG_WORK_PHONE          FLAG_PHONE          FLAG_EMAIL
#>                  0                  0                   0                   0
#>    OCCUPATION_TYPE    CNT_FAM_MEMBERS
#>             134203                  0
```

Sumber: Algoritma Show

Gambar 3. Missing Value

```
data_clean <- credit %>%
              left_join(application) %>%
              select(-c(OCCUPATION_TYPE, DAYS_BIRTH, DAYS_EMPLOYED)) %>%
              na.omit() %>%
              select(-ID) %>%
              filter(STATUS != "X") %>%
              mutate(STATUS = as.factor(ifelse(STATUS == "C", "good credit", "bad credit"))) %>%
              mutate_at(.vars = c("FLAG_MOBIL", "FLAG_WORK_PHONE",
                                  "FLAG_PHONE", "FLAG_EMAIL"), as.factor) %>%
              mutate_if(is.character, as.factor) %>%
              data.frame()
```

Sumber: Algoritma Show

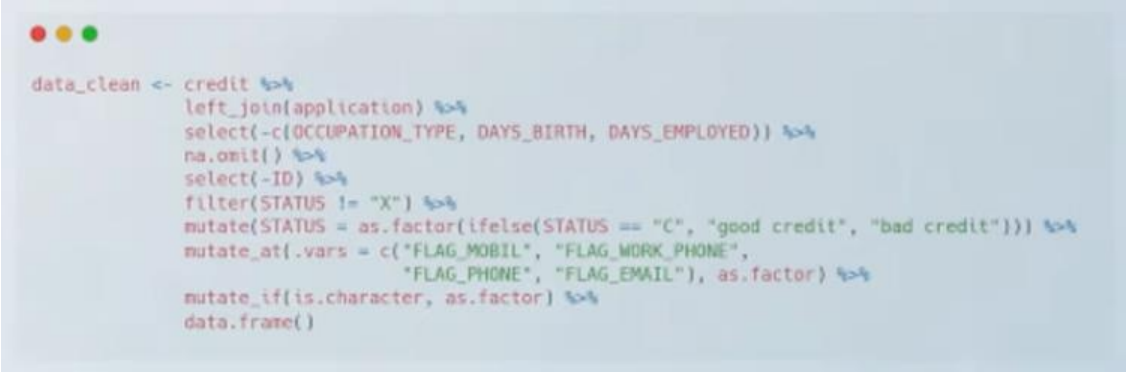
Cek missing value digunakan untuk mengetahui apakah data sudah lengkap atau ada data yang hilang. Pada tahapan ini, ada beberapa cara untuk mengatasi missing value. Ada dua cara dalam mengatasi *missing value*, yaitu (1) *take out variabel* atau membuang variabel-variabel yang memiliki jumlah *missing value* yang sangat besar (lebih dari 50%) dari total

observasi dan (2) *complete case* adalah membuang baris-baris yang memiliki *missing value* karena jumlah observasi tidak terlalu banyak.

2. Menyesuaikan tipe data.

   Untuk data-data yang kategorik yang sebelumnya memiliki tipe data karakter akan diubah menjadi tipe faktor.

Gambar 4. Menyesuaikan Tipe Data

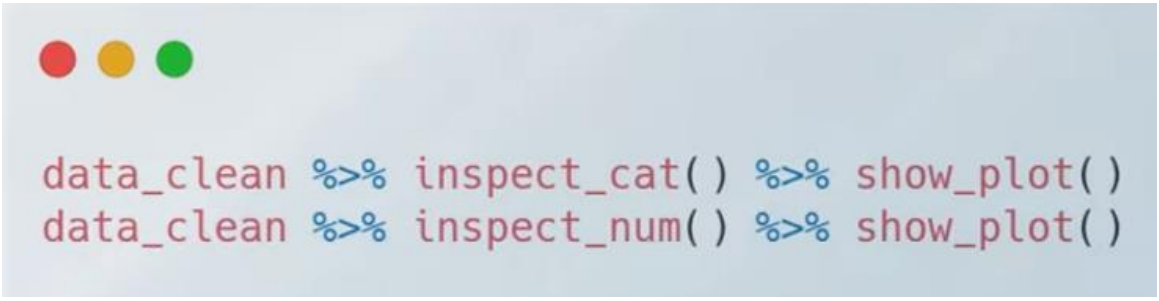


```
data_clean <- credit %>%
              left_join(application) %>%
              select(-c(OCCUPATION_TYPE, DAYS_BIRTH, DAYS_EMPLOYED)) %>%
              na.omit() %>%
              select(-ID) %>%
              filter(STATUS != "X") %>%
              mutate(STATUS = as.factor(ifelse(STATUS == "C", "good credit", "bad credit"))) %>%
              mutate_at(.vars = c("FLAG_MOBIL", "FLAG_WORK_PHONE",
                                  "FLAG_PHONE", "FLAG_EMAIL"), as.factor) %>%
              mutate_if(is.character, as.factor) %>%
              data.frame()
```

Sumber: Algoritma Show

3. *Exploratory data.*

Gambar 5. Exploratory Data



```
data_clean %>% inspect_cat() %>% show_plot()
data_clean %>% inspect_num() %>% show_plot()
```

Sumber: Algoritma Show

Pada tahap ini akan dilakukan visualisasi data kategorik maupun data numerik.

Gambar 6. Visualisasi Data

Sumber: Algoritma Show

4. *Cross-validation.*

Gambar 7. Cross-Validation

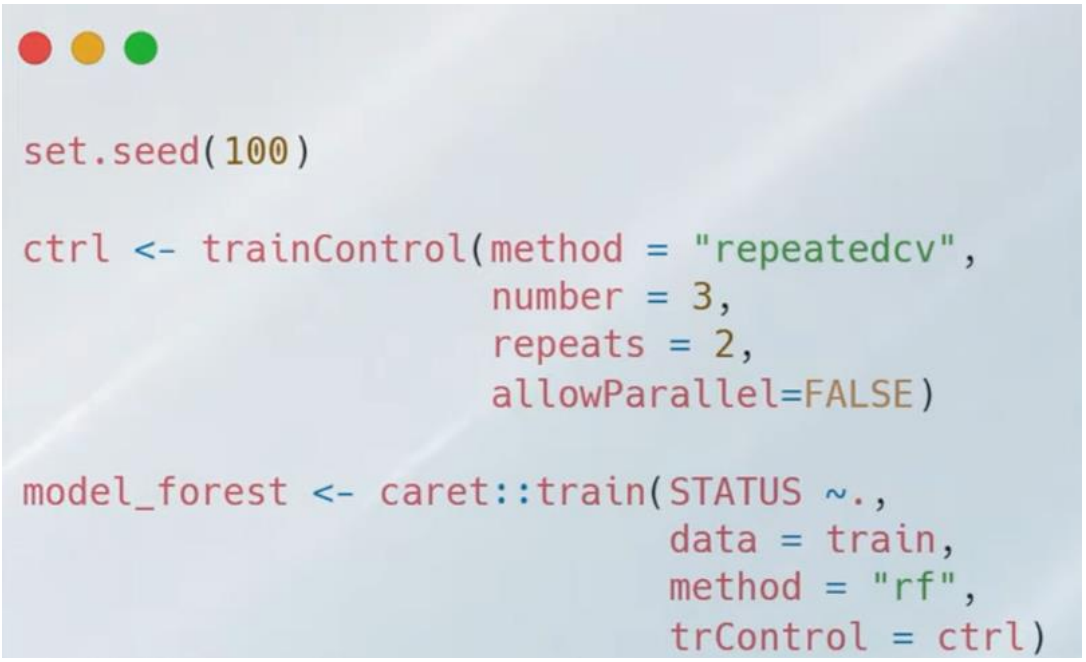
```
set.seed(100)
index <- initial_split(data = data_clean, prop = 0.8, strata = "STATUS")
train <- training(index)
test <- testing(index)
```

Sumber: Algoritma Show

Pada tahap ini data akan dibagi menjadi dua, yaitu data *train* dan data teks. 80% data akan digunakan data train untuk modelling dan 20% data akan kita jadikan data tes sebagai evaluasi.

Tahapan ketiga adalah modelling. Saat melakukan tahap modelling dapat membandingkan beberapa model yang bisa digunakan pada data tersebut seperti model random forest, XGBoost, dan lainnya. Pada esai ini difokuskan model random forest dan XGBoost.

Gambar 8. Model Random Forest



```
set.seed(100)

ctrl <- trainControl(method = "repeatedcv",
                     number = 3,
                     repeats = 2,
                     allowParallel=FALSE)

model_forest <- caret::train(STATUS ~.,
                             data = train,
                             method = "rf",
                             trControl = ctrl)
```

Sumber: Algoritma Show

Gambar 8. Model XGBoost

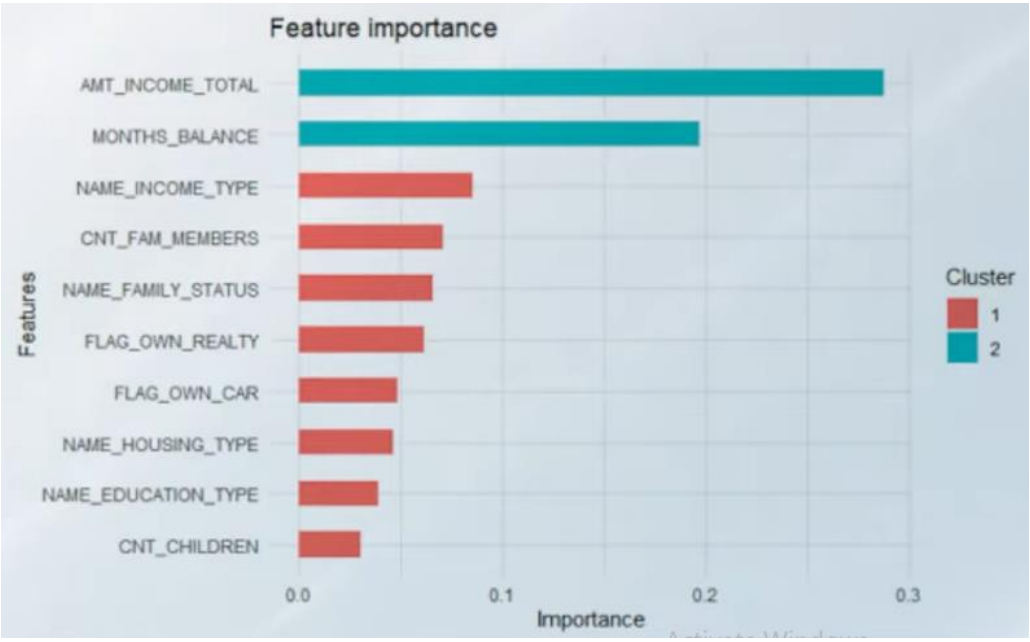
```
params <- list(booster = "gbtree",
               objective = "binary:logistic",
               eta=0.7,
               gamma=10,
               max_depth=10,
               min_child_weight=3,
               subsample=1,
               colsample_bytree=0.5)

xgbcv <- xgb.cv( params = params,
                 data = dtrain,
                 nrounds = 1000,
                 showsd = T,
                 nfold = 10,
                 stratified = T,
                 print_every_n = 50,
                 early_stopping_rounds = 20,
                 maximize = F)
```

Sumber: Algoritma Show

Setelah diimplementasikan, tahap selanjutnya adalah compare hasil dari kedua model tersebut. Model yang mana yang memiliki performance yang paling tinggi. Selanjutnya adalah tahap evaluasi model. Tahap evaluasi model adalah membandingkan matriks evaluasi dari model XGBoost dan random forest. Matriks evaluation yang difokuskan di sini adalah recall. Recall ini berguna untuk meminimalisir hasil prediksi yang data sebenarnya nasabah bad credit, tetapi diprediksi good credit. Hal tersebut yang ingin dihindari. Oleh karena itu, dicari nilai recall yang terbesar.

Dari kedua model tersebut, model XGBoost memiliki nilai rekor yang lebih besar dibandingkan model random forest. Hal ini menunjukkan bahwa model yang digunakan adalah XGBoost. Dari hasil model XGBoost dapat memperoleh informasi mengenai mana saja variabel-variabel yang paling berpengaruh dan paling penting di model tersebut.

Gambar 9. Variabel-Variabel Penting Dalam Credit Scoring



Sumber: Algoritma Show

Pada gambar di atas terdapat 10 variabel yang paling penting apakah nasabah tersebut layak atau tidak layak. Variabel annual income atau total income nasabah menjadi variabel yang paling tinggi.

Ini artinya variabel pendapatan menjadi paling penting untuk diprediksi, apakah nasabah memiliki good atau bad credit. Kemudian pada posisi kedua terdapat variabel month balance. Harapannya pihak bank dapat menggunakan model tersebut untuk mengetahui berapa probabilitas nasabah tersebut layak atau tidak diberikan pinjaman.

Gambar 10. Probabilitas Kelayakan Nasabah

Sumber: Algoritma Show

Daftar Pustaka

Ajeng. Credit Scoring. Youtube, diunggah oleh Algoritma Show 12 Juli 2021, https://youtu.be/L7564DMRcdY.