Presentasi SIB
Data Analytics Zenius

Nabila Asshafa Putri (20090105)

Data Analytics

Data analytics adalah suatu ilmu untuk menganalisis data mentah menjadi informasi koheren yang bermakna dan dapat ditindaklanjuti.
Adapun data analytics adalah penggabungan teori dan praktik mengidentifikasi serta mengomunikasikan insight berbasis data yang memungkinkan stakeholder mengambil keputusan lebih baik.

Silabus Data Analytics

Mempelajari tentang :

- Kemampuan Literasi, Numerasi dan Berfikir Saintifik
- Data Analytics

1. Kemampuan Literasi, Numerasi, dan Berpikir Saintifik

Mempelajari tentang:

- Struktur Kalimat, Tanda Baca, dan Penulisan Paragraf Argumentatif
- Penarikan Kesimpulan dengan Penalaran Deduktif(benar salah) dan Menilai dan Membangun Argumentasi
- Fondasi Bermatematika dan Pemahaman Data Kuantitatif
- Metakonsep Sains, Abad Pencerahan dan Kelahiran Sains Modern, dan Big History
- Membaca dan Meninjau Jurnal Ilmiah

2. Data Analytics

Mempelajari tentang:

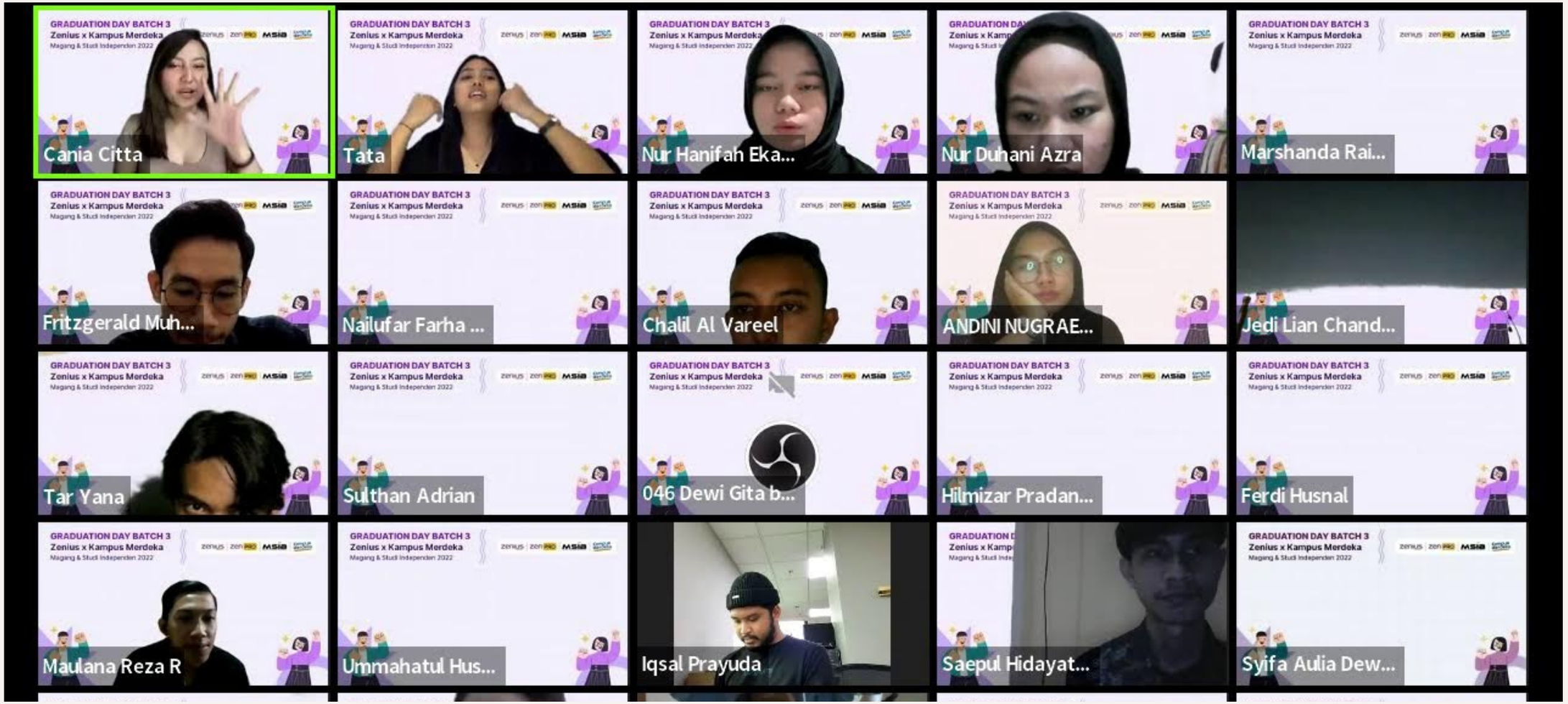
- Pengenalan Data Science
- Python untuk Data Science
- Visualisasi Data dengan Phyton
- Statistik untuk Data Science
- Exploratory Data Analysis
- Pemodelan Statistik
- Database: SQL Query
- Data Product Development (Dashboard)
- Proyek Akhir

JADWAL PEMBELAJARAN

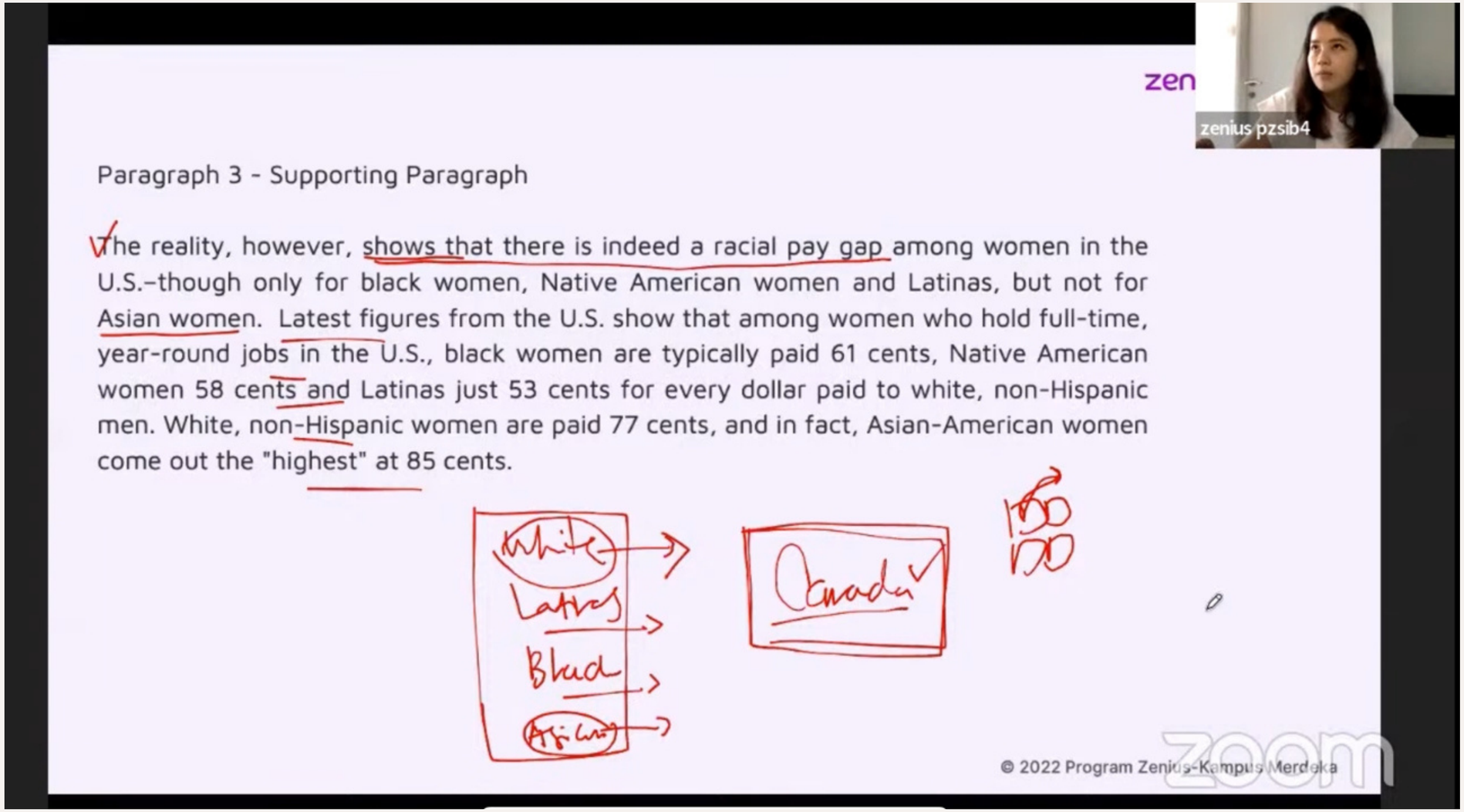
Jadwal pembelajaran mata kuliah Data Analysis di mitra zenius dilaksanakan seminggu 2 kali yaitu pada hari Selasa dan Jumat. Untuk jam pelaksanaannya dimulai pukul 18.30-21.00 WIB.

https://zenius.instructure.com/courses/161/pages/jadwal-live-class-dan-asesmen?module_item_id=4566
https://zenius.instructure.com/courses/165/pages/jadwal-kelas-dan-asesmen?module_item_id=5716

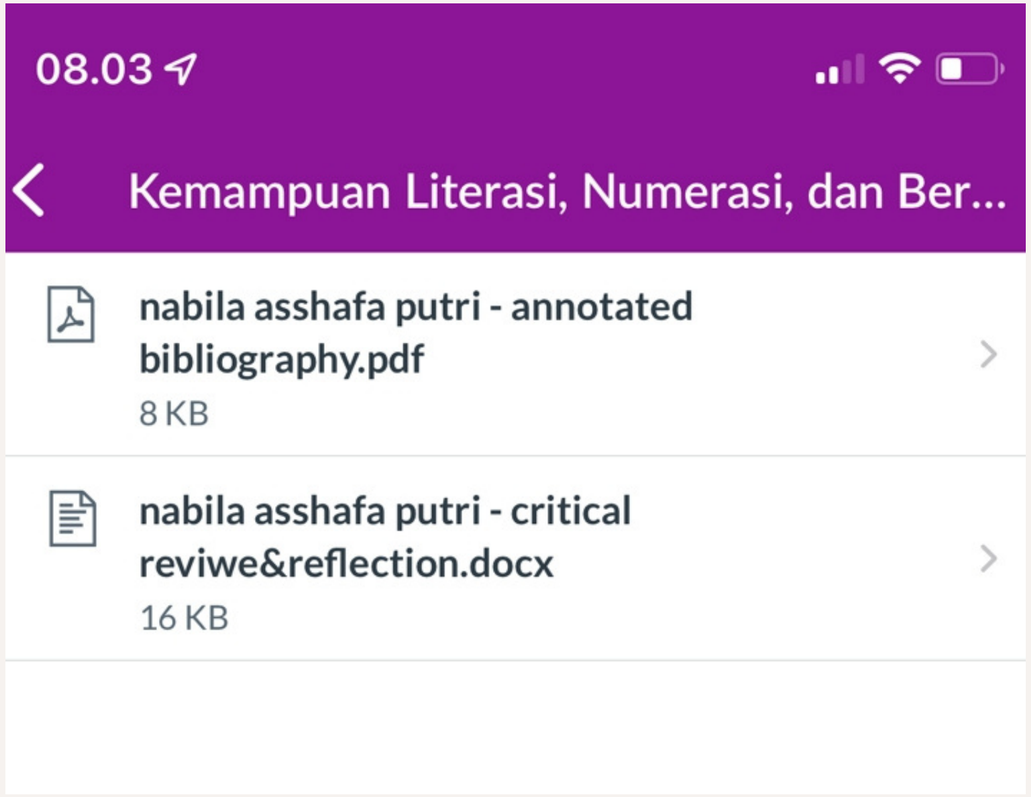
DOKUMENTASI KEGIATAN

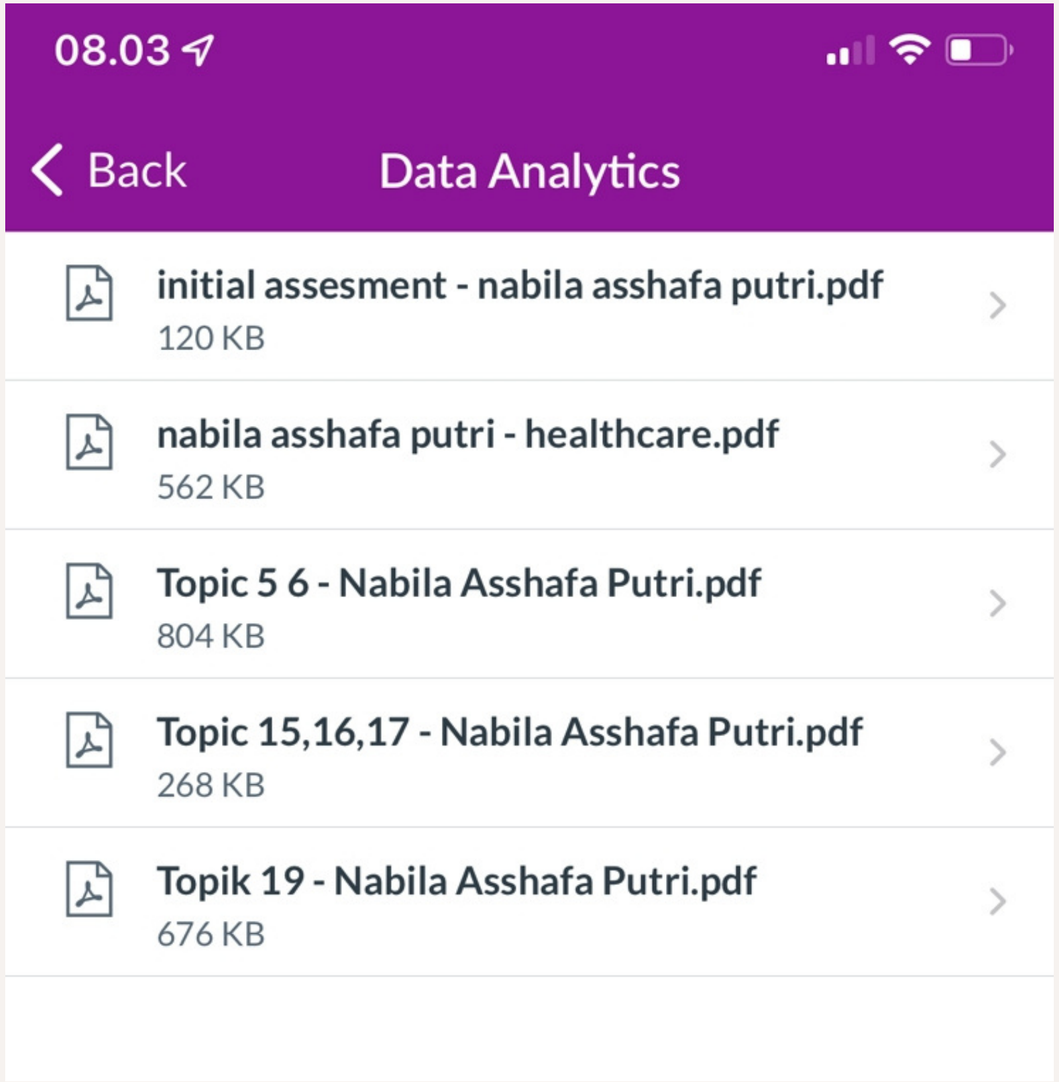


DOKUMENTASI KEGIATAN

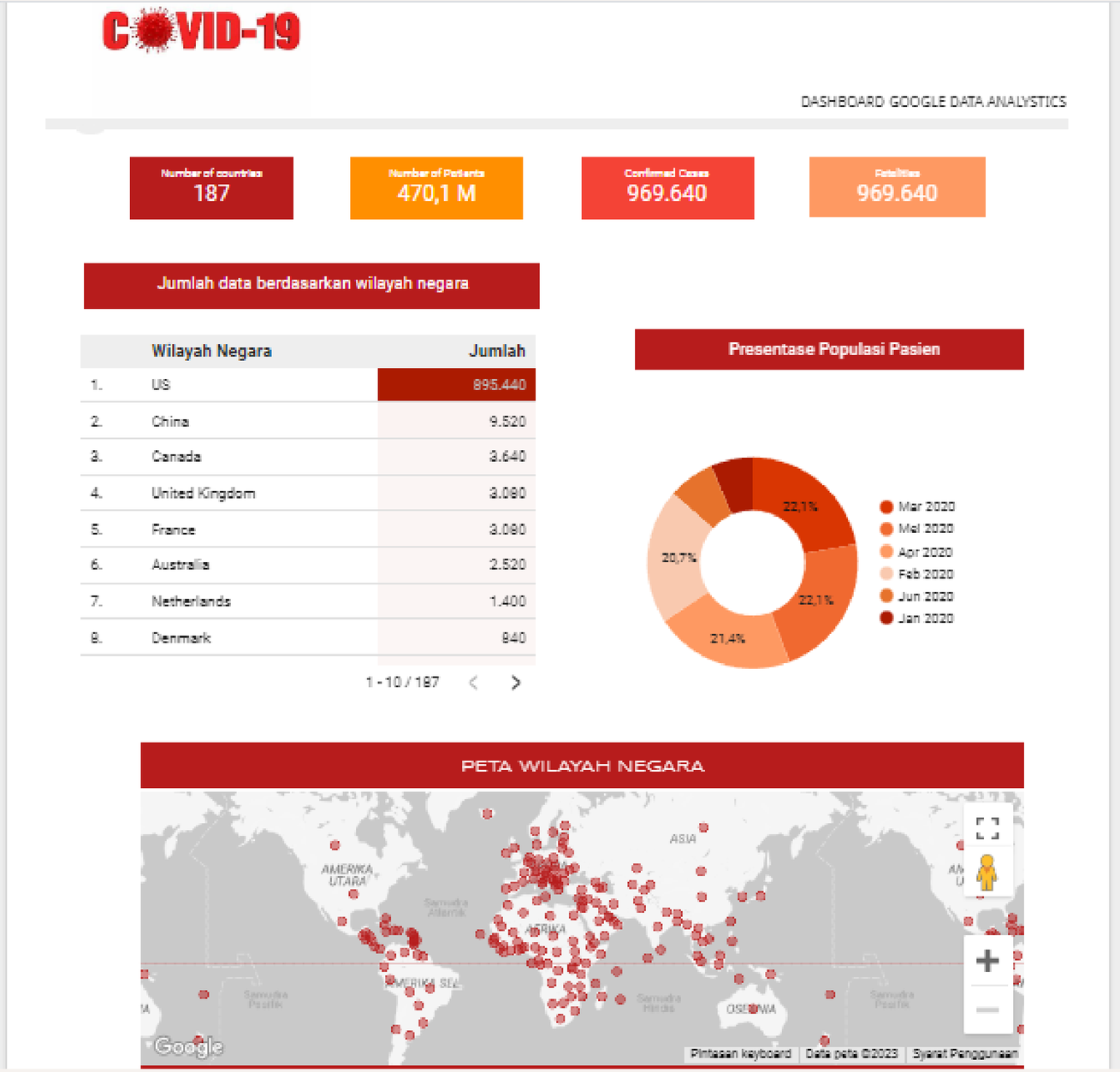


MINI PROJECT





MINI PROJECT



SERTIFIKAT PESERTA

zenius

SERTIFIKAT STUDI INDEPENDEN

No. ZSI-DA-2022-0056

Dengan bangga diberikan kepada

NABILA ASSHAFA PUTRI

Atas penyelesaian pelatihan

Studi Independen Bersertifikat
Data Analytics

Jakarta, 30 Desember 2022

Mengetahui

Sabda PS
Chief Education Officer Zenius

Diverifikasi oleh Kampus Merdeka

Kampus Merdeka
INDONESIA JAYA

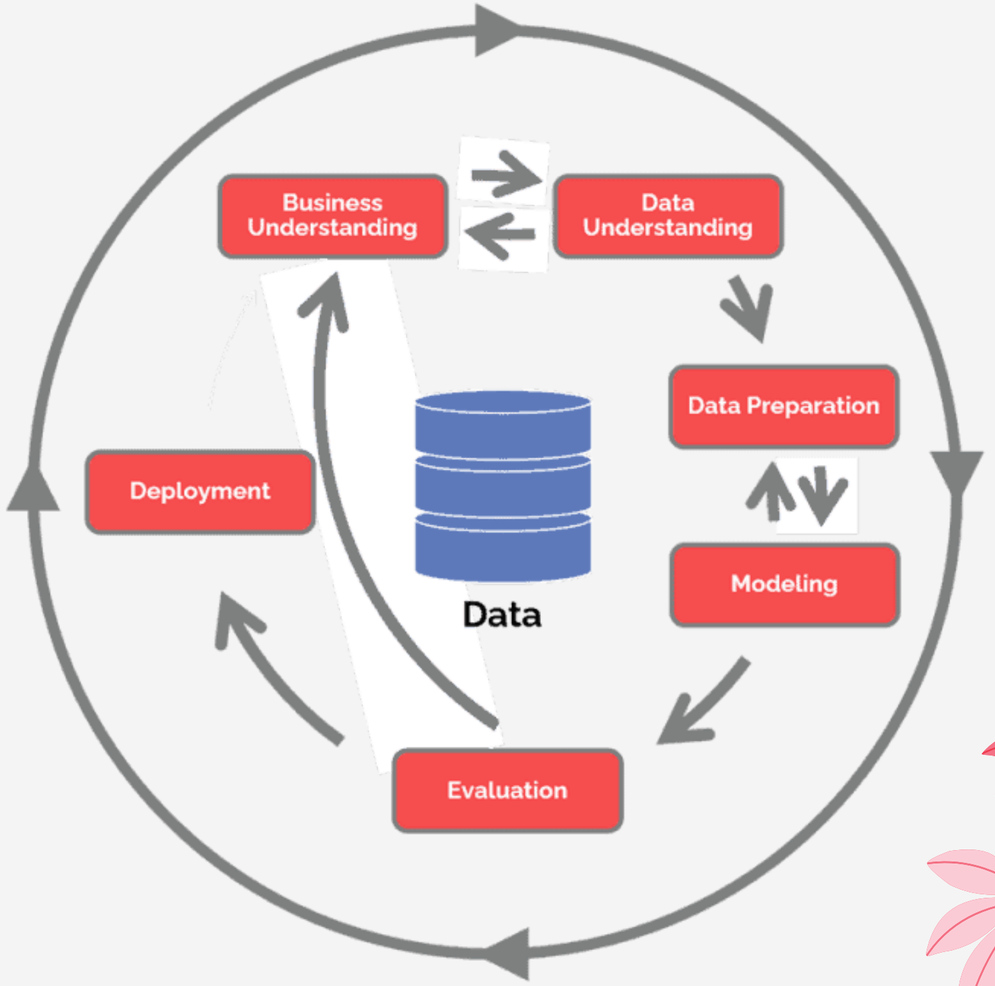
Final Project-Home Credit

Kelompok 11

- Muhammad Afifudin
- Nabila Asshafa Putri
- Ladyanna Kurniawan
- Sri Septiani Kurnia Putri
- Indri Oktaviani
- Mohamad Fakhrul Ikhrom

CRISP-DM

Cross-Industry Standard Process for Data Mining atau CRISP-DM adalah salah satu model proses datamining (datamining framework)



1) Business Understanding

Kami akan melakukan analisa dan membuat model dari data Home Credit. Home Credit adalah perusahaan internasional penyedia layanan peminjaman untuk keperluan kredit.

Tantangan yang dihadapi Home Credit yaitu berupaya untuk memperluas inklusi keuangan yang positif dan aman, oleh karena itu perlu analisa awal untuk memprediksi kemampuan pembayaran klien mereka.

Membuat credit scoring model untuk memastikan bahwa klien yang mampu membayar tidak ditolak saat pengajuan pinjaman dan mengidentifikasi potensi mangkir berdasarkan data yang diberikan oleh klien.

https://www.kaggle.com/competitions/home-credit-default-risk/overview
https://www.homecredit.net/about-us/our-company.aspx

2) Data Understanding

- **Langkah pertama yang harus dilakukan saat mengolah data yakni data understanding. Ketika memulai bermain dengan data maka harus mengetahui tentang data tersebut.**

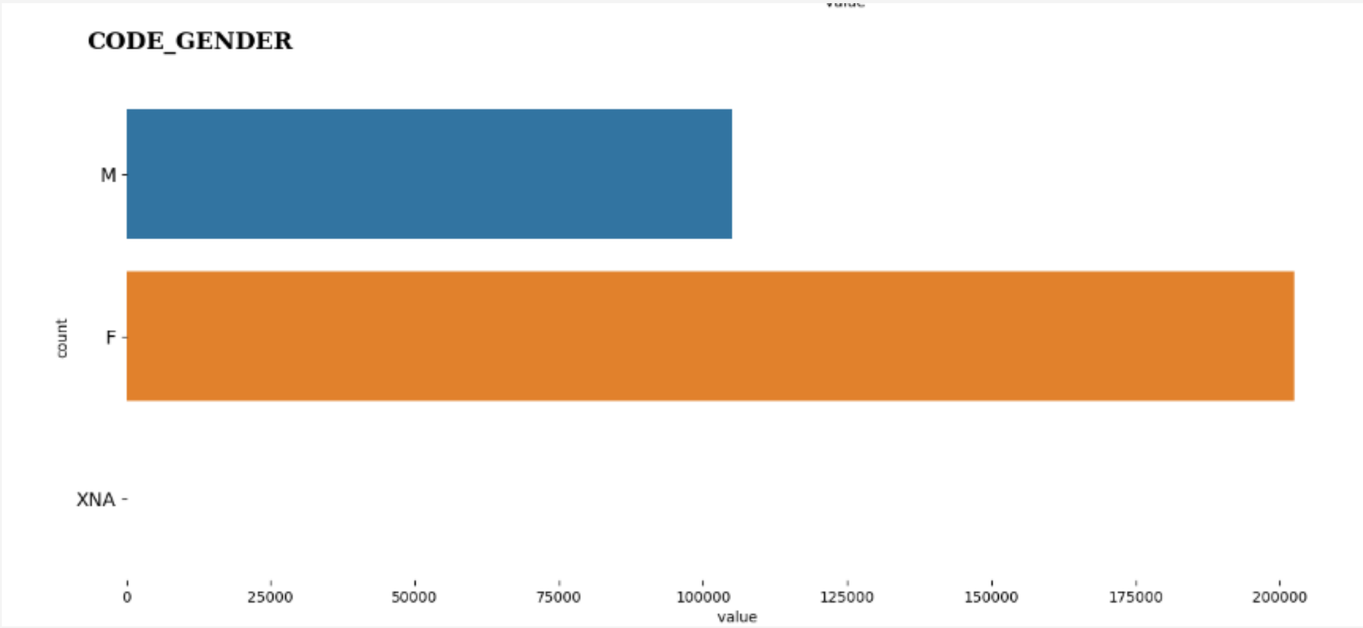
- **Dataset yang digunakan merupakan dataset public yang berjudul home credit default risk. Dataset yang digunakan berasal dari Kaggle dataset repository yang dapat diunduh di https://www.kaggle.com/competitions/home-credit-default-risk/data.**

- **Terdapat 7 sumber data dari dataset Home Credit. Namun kami hanya berfokus kepada data "application_train" untuk melatih model Machine Learning dan "application_test" untuk menguji performa model Machine Learning.**

Memvisualisasikan beberapa kolom supaya mempermudah membaca data

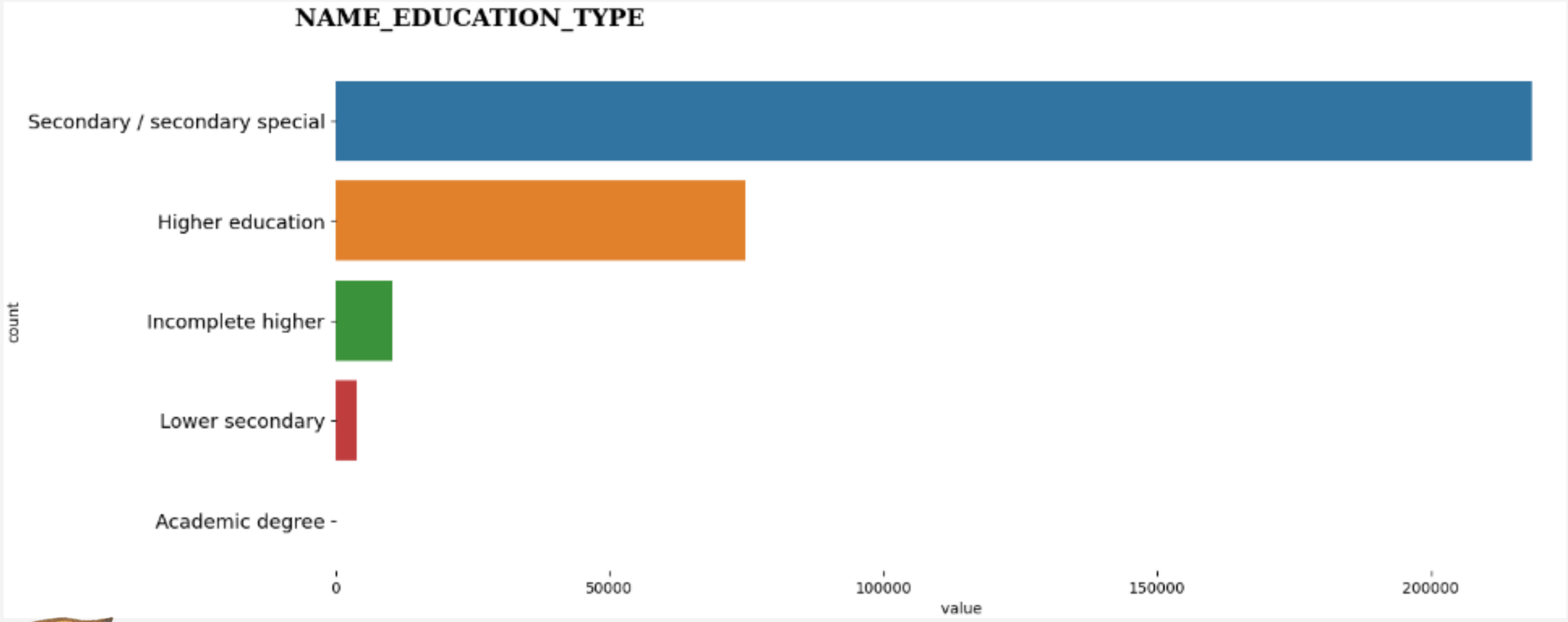
Distribusi Jenis Kelamin

Mayoritas nasabah adalah perempuan dibandingkan dengan nasabah laki-laki



Distribusi Pendidikan

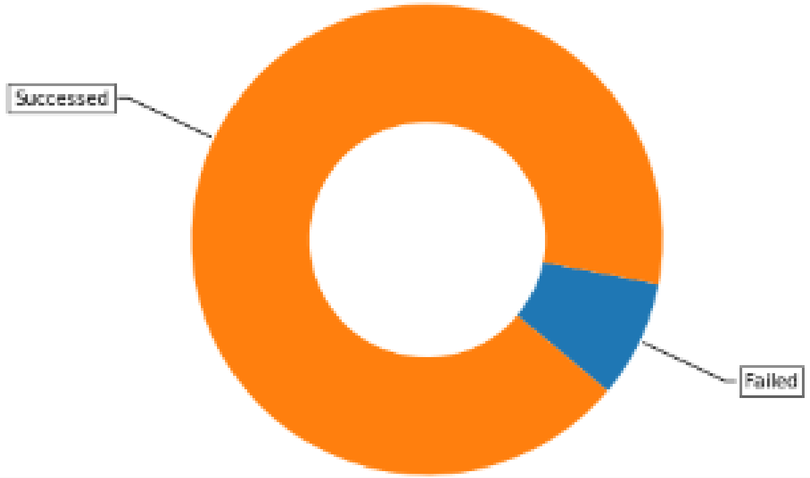
Mayoritas pendidikan nasabah yaitu secondary/secondary special



Status Kredit

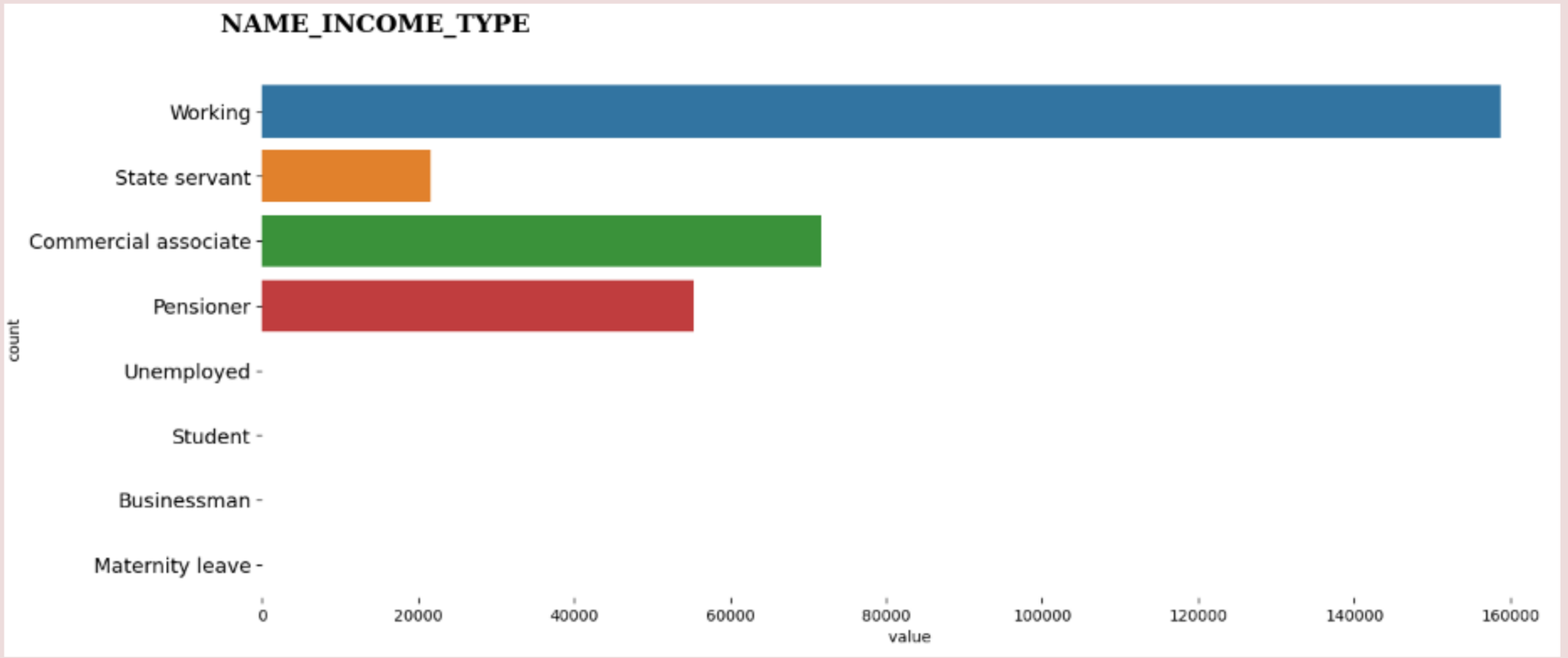
Sebanyak 91,93% atau sekitar 282.686 nasabah lancar mengembalikan pinjaman dan sekitar 8,07% nasabah tidak lancar mengembalikan pinjaman.

```
Number of customers who will not repay the loan on time:  24825 , ( 8.072881945686495 %)
Number of customers who will repay the loan on time:  282686 , ( 91.92711805431351 %)
```



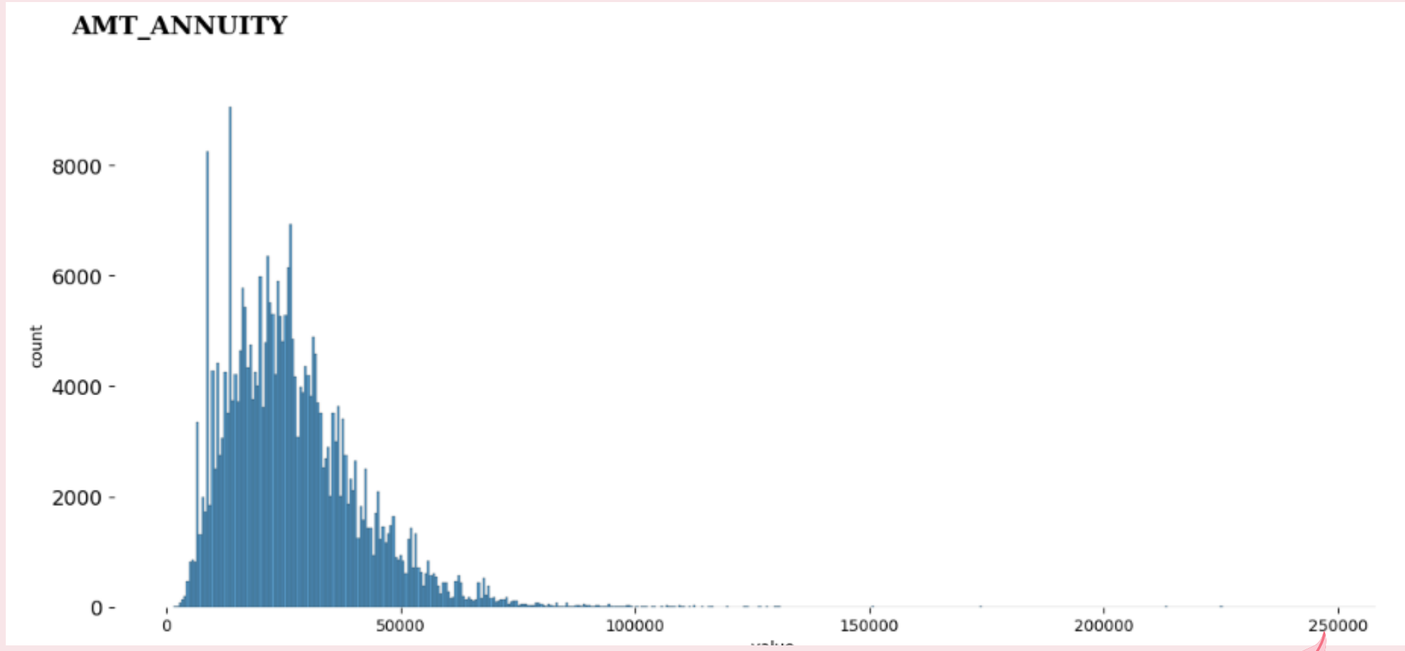
Distribusi Pekerjaan

Mayoritas pendapatan klien berasal dari bekerja



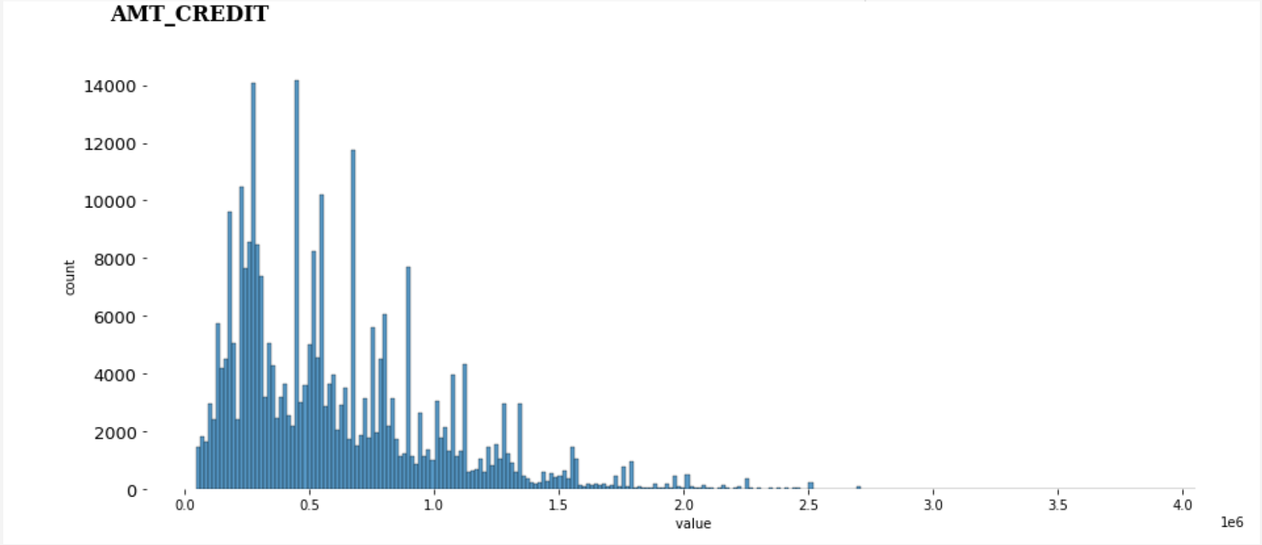
Distribusi Pembayaran Pertahun

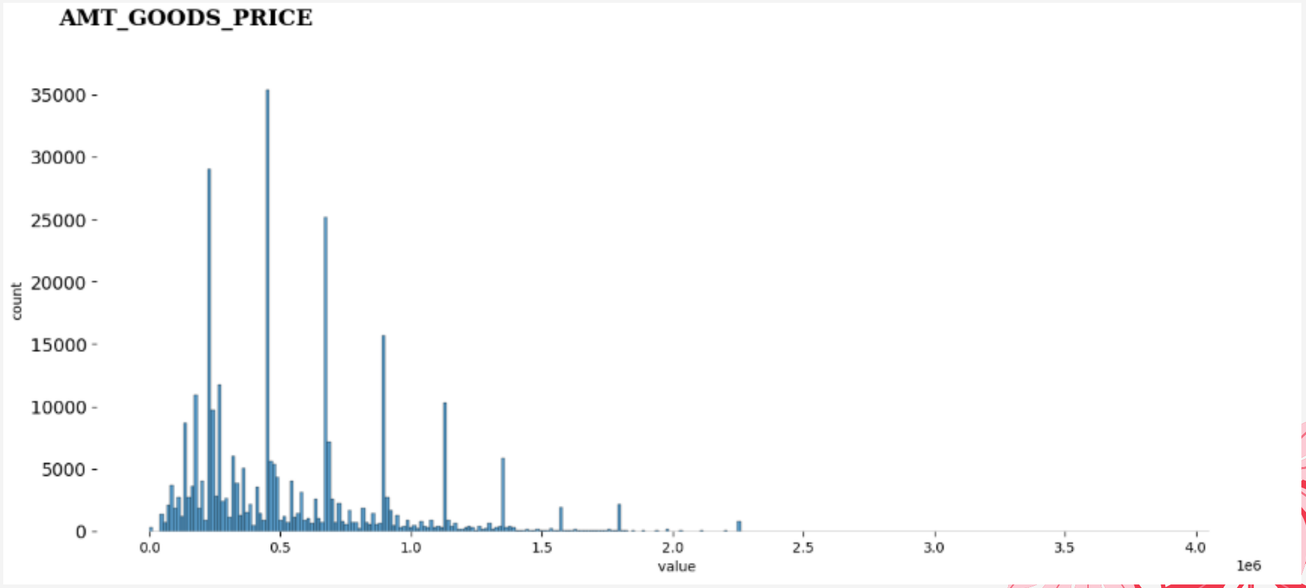
Distribusi right skewed.



Distribusi Credit & Good Price

Distribusi right skewed.





3) Data Preparation

Beberapa langkah yang dilakukan di tahap ini antara lain:

- Handling missing value
- Mengecek data yang duplikat
- Mengubah kolom kategori string menjadi numerik – label encoding
- Mengubah kolom kategori string menjadi numerik dan menambahkan kolom baru untuk menunjukkan adanya variabel kategori - one hot encoding/dummy
- split data

Handling Missing Values

- **Handling missing values dilakukan satu per satu per data subset. Pengecekkan dilakukan satu per satu karena tiap kolom memiliki penanganan missing values yang berbeda**

- **Men-drop kolom apabila missing values > 30%**

- **Selain melakukan drop kolom juga mengisi missing values dengan 0, modus, median, atau mean sesuai dengan kondisi kolom**

Checking Missing Values

```
[10] print("Melihat jumlah missing value")
     print(df_train.isnull().sum())
     print("-"*50)
     print("TOTAL MISSING VALUES:",df_train.isnull().sum().sum())
```

```
Melihat jumlah missing value
SK_ID_CURR                        0
TARGET                            0
NAME_CONTRACT_TYPE                0
CODE_GENDER                       0
FLAG_OWN_CAR                      0
                                 ...
AMT_REQ_CREDIT_BUREAU_DAY     41519
AMT_REQ_CREDIT_BUREAU_WEEK    41519
AMT_REQ_CREDIT_BUREAU_MON     41519
AMT_REQ_CREDIT_BUREAU_QRT     41519
AMT_REQ_CREDIT_BUREAU_YEAR    41519
Length: 122, dtype: int64
--------------------------------------------------
TOTAL MISSING VALUES: 9152465
```

Drop kolom apabila >30%

```
subset_3.isna().sum()/len(subset_3)
```

```
YEARS_BEGINEXPLUATATION_MODE    0.487810
YEARS_BUILD_MODE                0.664978
COMMONAREA_MODE                 0.698723
ELEVATORS_MODE                  0.532960
ENTRANCES_MODE                  0.503488
FLOORSMAX_MODE                  0.497608
FLOORSMIN_MODE                  0.678486
LANDAREA_MODE                   0.593767
LIVINGAPARTMENTS_MODE           0.683550
LIVINGAREA_MODE                 0.501933
NONLIVINGAPARTMENTS_MODE        0.694330
NONLIVINGAREA_MODE              0.551792
APARTMENTS_MEDI                 0.507497
BASEMENTAREA_MEDI               0.585160
YEARS_BEGINEXPLUATATION_MEDI    0.487810
YEARS_BUILD_MEDI                0.664978
COMMONAREA_MEDI                 0.698723
ELEVATORS_MEDI                  0.532960
ENTRANCES_MEDI                  0.503488
FLOORSMAX_MEDI                  0.497608
FLOORSMIN_MEDI                  0.678486
LANDAREA_MEDI                   0.593767
LIVINGAPARTMENTS_MEDI           0.683550
LIVINGAREA_MEDI                 0.501933
NONLIVINGAPARTMENTS_MEDI        0.694330
NONLIVINGAREA_MEDI              0.551792
FONDKAPREMONT_MODE              0.683862
HOUSETYPE_MODE                  0.501761
TOTALAREA_MODE                  0.482685
WALLSMATERIAL_MODE              0.508408
dtype: float64
```

- Hampir semua kolom terdapat missing value dengan persentase 30%, langkah yang diambil disini yaitu menghapus semua kolom pada subset 3 yaitu kolom 60:90.

```
[ ]  df_train = df_train.drop(column_list[60:90], axis=1)
```

Mengganti missing value dengan nilai 0, mean, median, modus

- mengganti missing value dengan nilai 0

```
[ ]  df_train['DAYS_LAST_PHONE_CHANGE'] = df_train['DAYS_LAST_PHONE_CHANGE'].fillna(0)
```

- mengganti missing value dengan nilai mean dari kolom

```
[ ]  df_train['EXT_SOURCE_3'] = df_train['EXT_SOURCE_3'].fillna(df_train['EXT_SOURCE_3'].mean())
```

- Mengganti missing value dengan nilai median dari data kolom

```
[ ]  df_train['AMT_ANNUITY'] = df_train['AMT_ANNUITY'].fillna((df_train['AMT_ANNUITY'].median()))
```

- mengganti missing value dengan nilai modus dari data kolom

```
[ ]  df_train['NAME_TYPE_SUITE'] = df_train['NAME_TYPE_SUITE'].fillna(st.mode(subset_1['NAME_TYPE_SUITE']))
```

Encode Categorical Data

- **Berfungsi untuk mengubah data kategorikal menjadi data numerik, karena algoritma atau model yang digunakan hanya membaca data numerik**

```
[ ]  encoder = LabelEncoder()
```

```
[ ]  for data in df_train.describe(include='object').columns:
         df_train[data]=encoder.fit_transform(df_train[data])

     df_train[categorical]
```

Split Data to Train and Test

- **Berfungsi untuk memisahkan dataset menjadi subset yang meminimalkan potensi bias dalam proses evaluasi dan validasi**

split menjadi 80% training set and 20%

```
]  x_train, x_test, y_train, y_test = train_test_split(x, y, train_size = 0.8, random_state = 42)
```

4) Data Modeling

Model yang digunakan ada 5 antara lain yaitu :

- Logistic Regression
- Random Forest
- Gaussian Naive Bayes
- DecisionTree
- Multi-LayerPerceptron

Logistic Regression

```
[ ]  logreg = LogisticRegression(max_iter=3000)
     logreg.fit(x_train, y_train)

     y_predic_logreg = logreg.predict(x_test)

     logreg_train_score = logreg.score(x_train, y_train)
     logreg_test_score = logreg.score(x_test, y_test)

     logreg_acc = accuracy_score(y_test, y_predic_logreg

     logreg_predict = logreg.predict(data_test)

     print(logreg_predict)
     print("===LOGISTIC REGRESSION===")
     print('Model train score: ', logreg_train_score)
     print('Model test score: ', logreg_test_score)
     print('Model accuracy: ', logreg_acc)
```

```
===LOGISTIC REGRESSION===
Model train score:  0.9191855549413027
Model test score:  0.9194836024258979
Model accuracy:  0.9194836024258979
```

Random Forest

```
[ ]  randfor = RandomForestClassifier(n_estimators = 100, random_state=42, verbose=1, n_jobs=-1)
     randfor.fit(x_train, y_train)

     y_predic_randfor = randfor.predict(x_test)

     randfor_train_score = randfor.score(x_train, y_train)
     randfor_test_score = randfor.score(x_test, y_test)

     randfor_acc = accuracy_score(y_predic_randfor, y_test)

     randfor_predict = randfor.predict(data_test)

     print(randfor_predict)
     print("===Random Forest===")
     print('Model train score: ', randfor_train_score)
     print('Model test score: ', randfor_test_score)
     print('Model accuracy: ', randfor_acc)
```

```
===Random Forest===
Model train score:  0.9999674807323339
Model test score:  0.9196461961205145
Model accuracy:  0.9196461961205145
```

Gaussian Naive Bayes

```
[ ]  gnb = GaussianNB()
     gnb.fit(x_train, y_train)

     y_predic_gnb = gnb.predict(x_test)

     gnb_train_score = gnb.score(x_train, y_train)
     gnb_test_score = gnb.score(x_test, y_test)

     gnb_acc = accuracy_score(y_predic_gnb, y_test)

     gnb_predict = randfor.predict(data_test)

     print("===Predicted label===")
     print(gnb_predict)
     print("===Gaussian Naive Bayes===")
     print('Model train score: ', gnb_train_score)
     print('Model test score: ', gnb_test_score)
     print('Model accuracy: ', gnb_acc)
```

```
===Gaussian Naive Bayes===
Model train score:  0.919205879483594
Model test score:  0.9195323805342829
Model accuracy:  0.9195323805342829
```

Decision Tree

```
[ ]  dectree = DecisionTreeClassifier()
     dectree.fit(x_train, y_train)

     y_predic_dectree = dectree.predict(x_test)

     dectree_train_score = dectree.score(x_train, y_train)
     dectree_test_score = dectree.score(x_test, y_test)

     dectree_acc = accuracy_score(y_predic_dectree, y_test)

     dectree_predict = dectree.predict(data_test)

     print(dectree_predict)
     print("===Decision Tree===")
     print('Model train score: ', dectree_train_score)
     print('Model test score: ', dectree_test_score)
     print('Model accuracy: ', dectree_acc)
```

```
===Decision Tree===
Model train score:  1.0
Model test score:  0.8513731037510366
Model accuracy:  0.8513731037510366
```

Multi-Layer Perceptron

```
[ ]  mlpc = MLPClassifier(solver='lbfgs', alpha=1e-5, hidden_layer_sizes=(5, 2), random_state=1)
     mlpc.fit(x_train, y_train)

     y_predic_mlpc = mlpc.predict(x_test)

     mlpc_train_score = mlpc.score(x_train, y_train)
     mlpc_test_score = mlpc.score(x_test, y_test)

     mlpc_acc = accuracy_score(y_predic_dectree, y_test)

     mlpc_predict = mlpc.predict(data_test)

     print(mlpc_predict)
     print("===Multi-Layer Perceptron===")
     print('Model train score: ', mlpc_train_score)
     print('Model test score: ', mlpc_test_score)
     print('Model accuracy: ', mlpc_acc)
```

```
===Multi-Layer Perceptron===
Model train score:  0.816928717765276
Model test score:  0.8182039900492659
Model accuracy:  0.851373103751036 6
```

5) Evaluasi

Model Evaluasi yang digunakan yaitu :

- Accuracy Score
- Mean Square Error (MSE)
- Root Mean Square Error (RMSE)
- Receiver Operating Characteristic (ROC)
- Precision Score

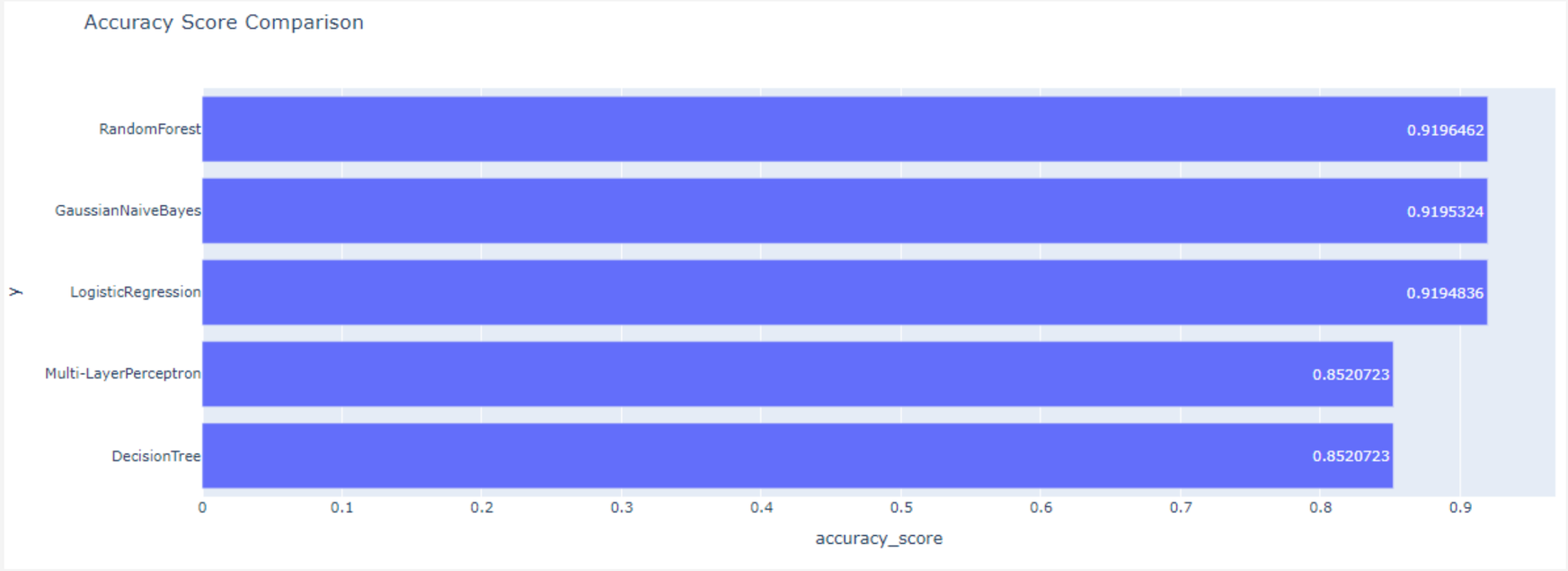
Perbandingan Nilai Model Algoritma

```
models = ['LogisticRegression', 'RandomForest', 'GaussianNaiveBayes', 'DecisionTree', 'Multi-LayerPerceptron']
data = {
    'accuracy_score': [logreg_acc, randfor_acc, gnb_acc, dectree_acc, mlpc_acc],
    'MSE': [logreg_mse_score, randfor_mse_score, gnb_mse_score, dectree_mse_score, mlpc_mse_score],
    'RMSE': [logreg_rmse_score, randfor_rmse_score, gnb_rmse_score, dectree_rmse_score, mlpc_rmse_score],
    'ROC': [logreg_roc_score, randfor_roc_score, gnb_roc_score, dectree_roc_score, mlpc_roc_score],
    'precision_score': [logreg_prec_score, randfor_prec_score, gnb_prec_score, dectree_prec_score, mlpc_prec_score]
}
comparison_table = pd.DataFrame(data, index=models)
comparison_table
```

	accuracy_score	MSE	RMSE	ROC	precision_score
LogisticRegression	0.919484	0.080516	0.283754	0.499973	0.000000
RandomForest	0.919646	0.080354	0.283467	0.501537	0.640000
GaussianNaiveBayes	0.919532	0.080468	0.283668	0.500000	0.000000
DecisionTree	0.852121	0.147879	0.384550	0.541982	0.145762
Multi-LayerPerceptron	0.852121	0.181796	0.426375	0.521327	0.105070

Hasil Evaluasi Score Model

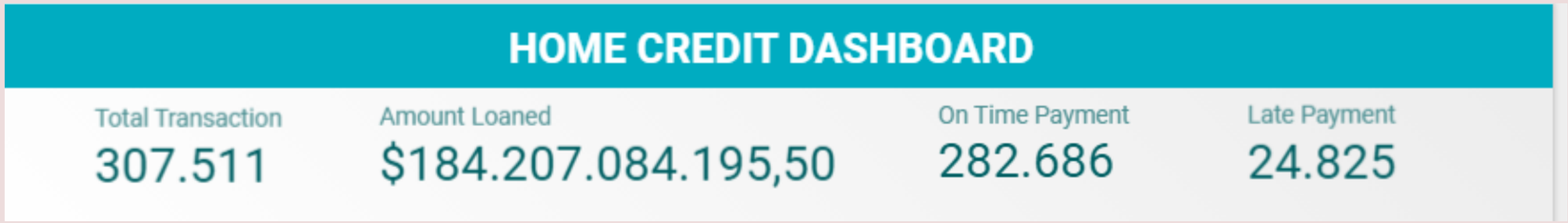
Berikut visualisasi data hasil evaluasi score model:



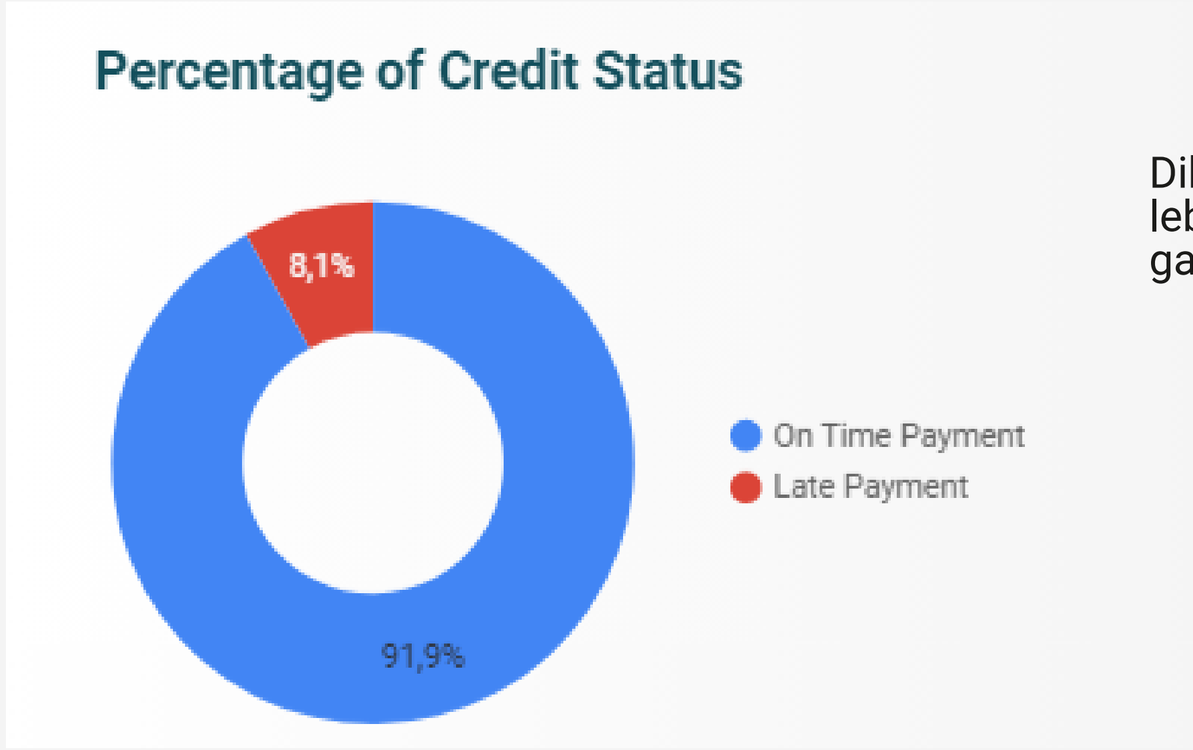
Dari semua model yang digunakan. Setelah melalui beberapa evaluasi, model Random Forest menunjukkan hasil evaluasi tertinggi daripada model lainnya. Oleh karena itu, model Random Forest dipilih sebagai model untuk prediksi.

6) Dashboard

Menampilkan hubungan beberapa matriks penting dengan fitur target

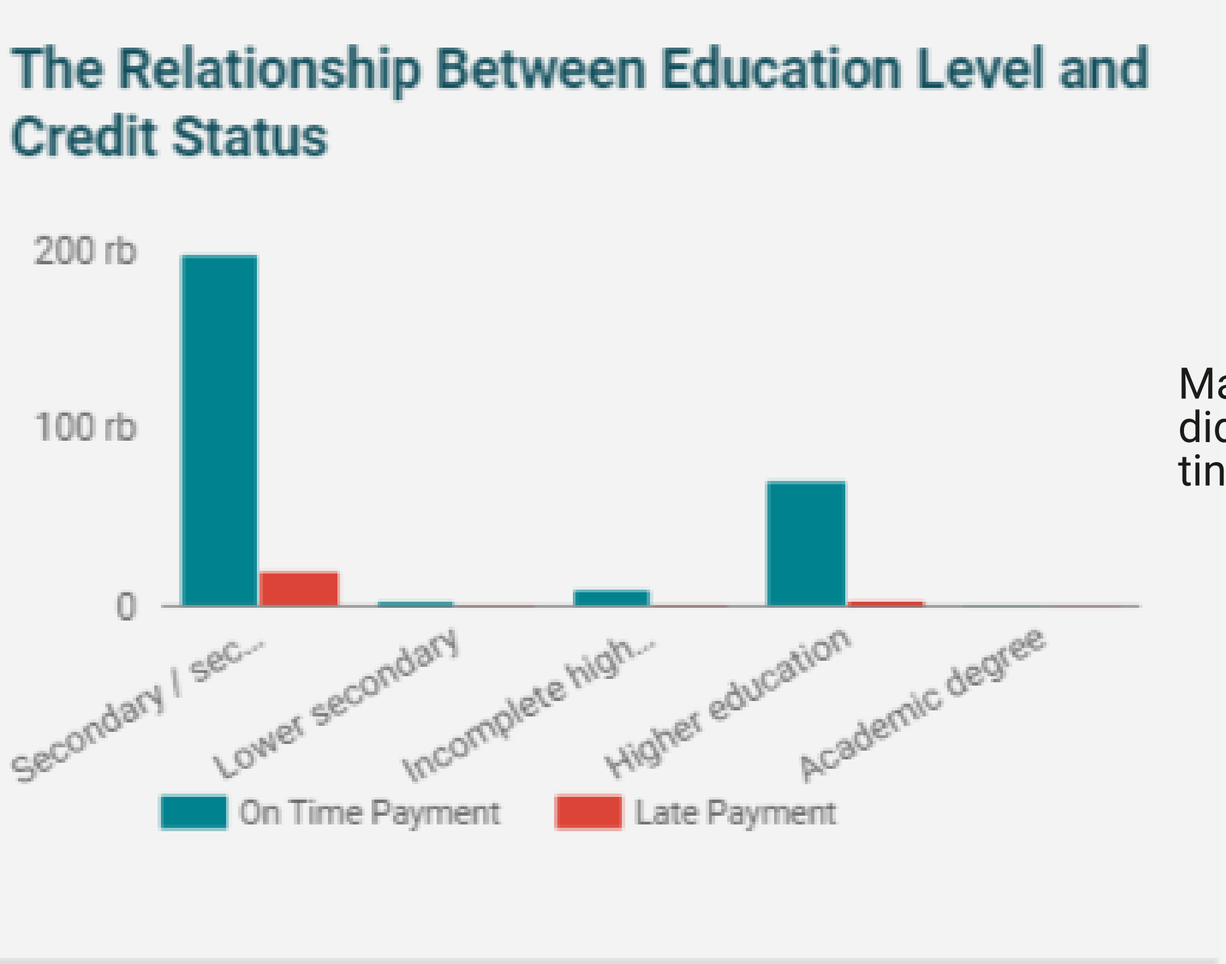


Home Credit Dashboard



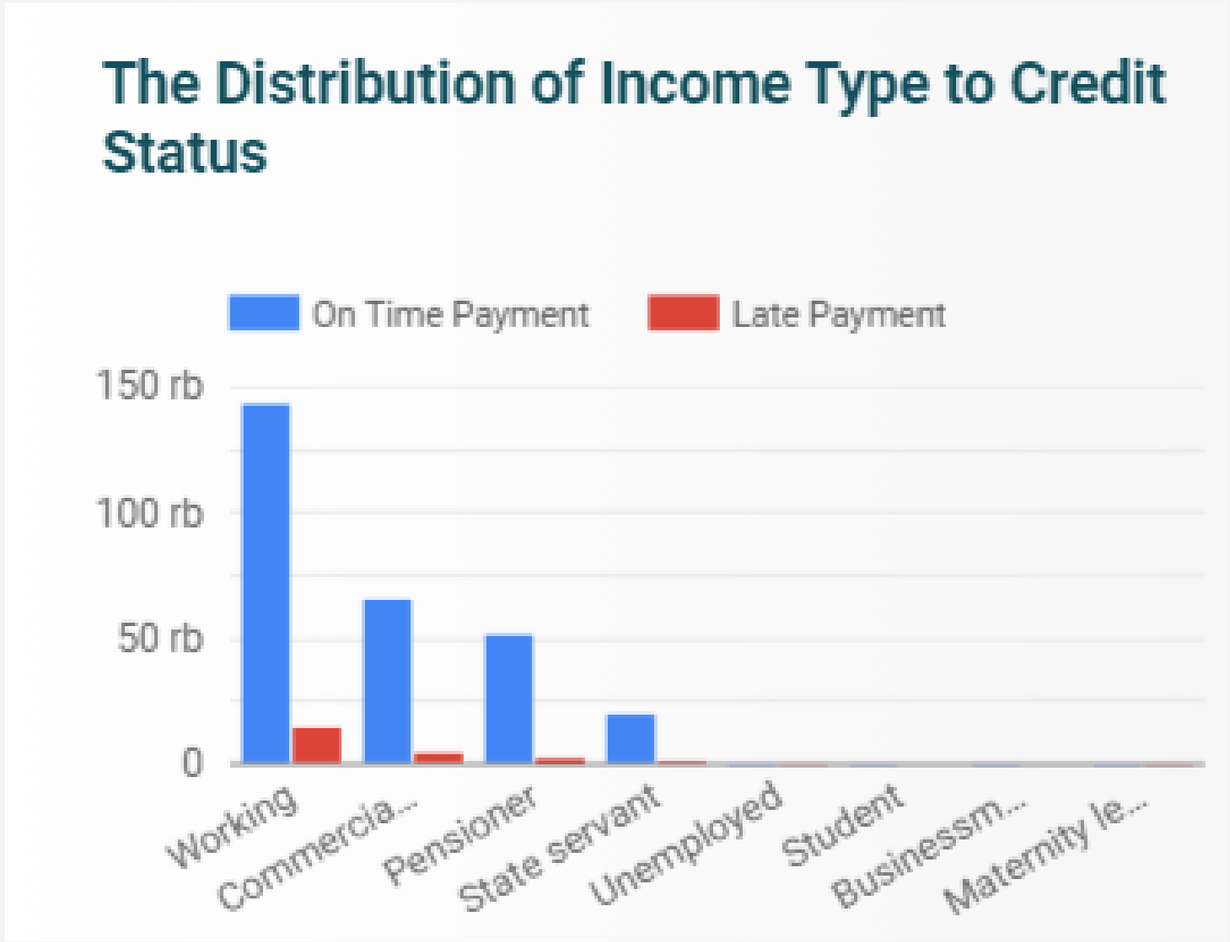
Diketahui bahwa presentase kelancaran pembayaran lebih tinggi yaitu 91,9% dibandingkan presentase tingkat gagal bayar

Home Credit Dashboard



Mayoritas dari masing-masing tingkatan pendidikan didominasi lancar dalam pembayaran kredit serta tingkat pendidikan yang paling banyak yaitu secondary

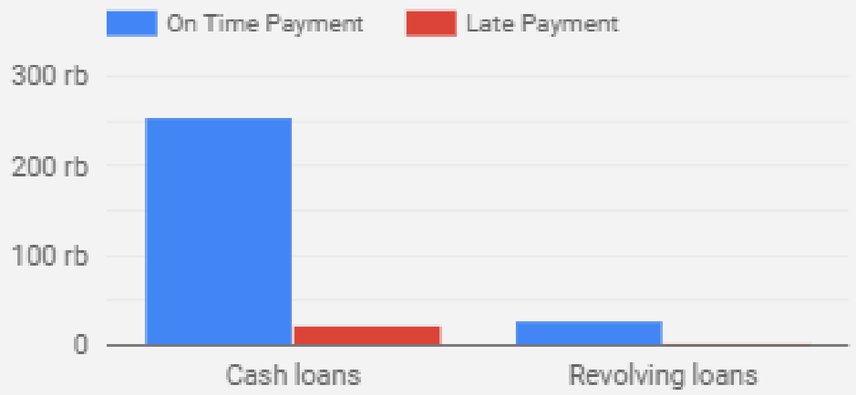
Home Credit Dashboard



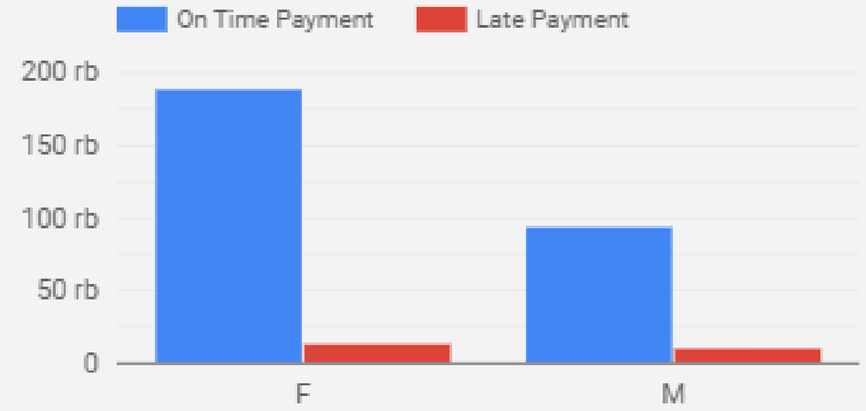
Sumber pemasukan mayoritas berasal dari bekerja dan tingkat kelancaran pembayaran kredit yang baik.

Home Credit Dashboard

Client by Contract Types & Gender



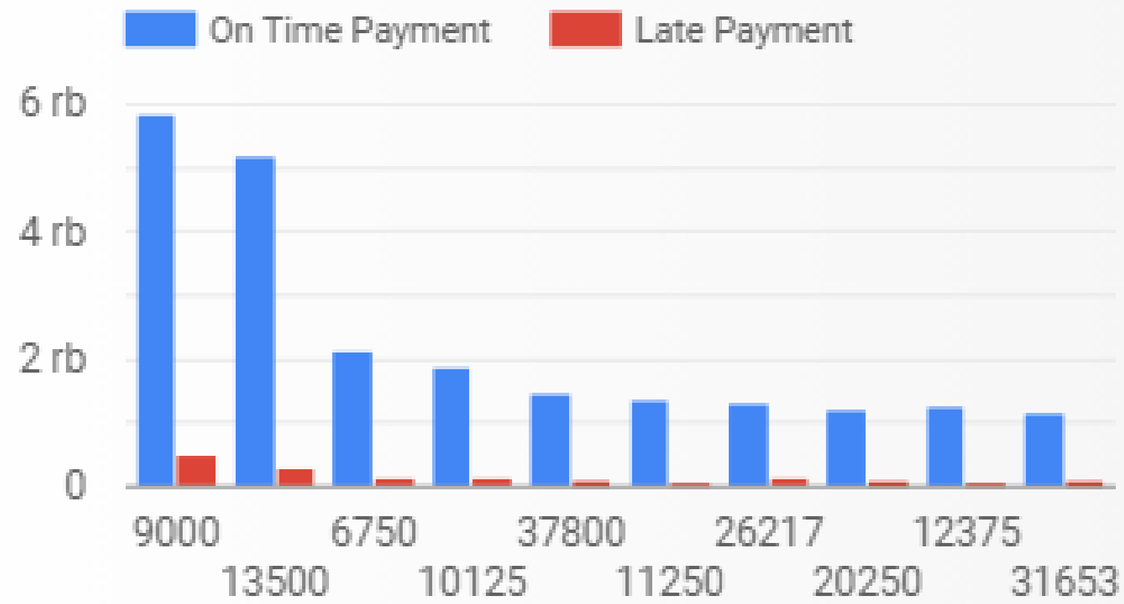
Contract types paling banyak didominasi oleh cash loan dengan tingkat kelancaran pembayaran kredit yang baik.



Client dengan gender female lebih banyak dibandingkan male serta keduanya mempunyai tingkat kelancaran pembayaran kredit yang tinggi.

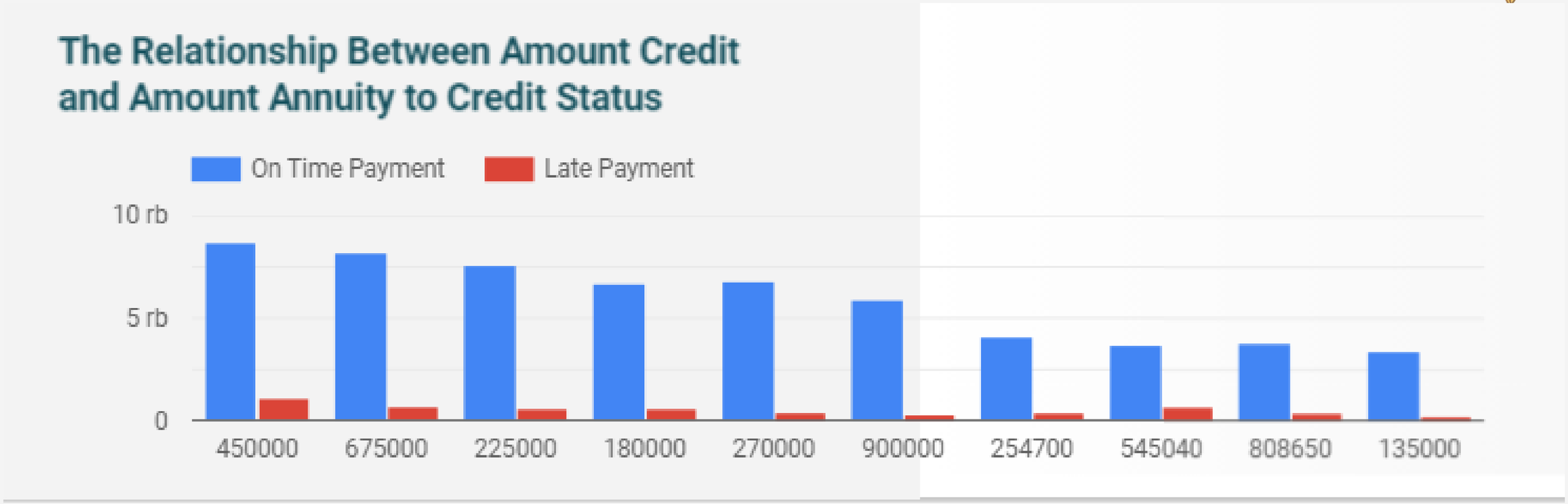
Home Credit Dashboard

The Distribution of Amount Annuity to Credit Status



Distribusi annuity atau pembayaran per tahun rata-rata client lancar dalam melakukan pembayaran kredit.

Home Credit Dashboard



Total annuity atau pembayaran pertahun dan jumlah credit rata-rata client lancar dalam melakukan pembayaran kredit.

7) Business Solution

- Algoritma yang direkomendasikan dalam pembuatan model adalah algoritma Random forest classification.
- Memfokuskan campaign ke nasabah dengan jenis kelamin perempuan dan tipe kontrak revolving loans
- Memfokuskan menerima nasabah pekerja dan memiliki pendidikan secondary/higher education
- Melakukan seleksi yang lebih ketat terhadap client yang mengajukan pinjaman dengan jenis cash loan

List of Links:

- Google Colab: https://colab.research.google.com/drive/1gDh1vL5v1NRNtSUal-1PvG9yHjs-lt_a?usp=sharing

- Google Data Studio: https://datastudio.google.com/s/r8wENk3oftc