

Nama: Anindya Dewi Maharani

Instansi: Univeritas Sriwijaya

Dalam melakukan *machine learning* memiliki runtutan langkah agar produk yang dihasilkan merupakan versi yang seterbaiik mungkin. Langkah-langkah ini juga akan memudahkan dalam proses pembuatan menjadi lebih rapih dan terstruktur. Langkah ini disebut juga dengan istilah CRISP-DM (*Cross Industry Standard Process For Data Mining*). Flow yang dilakukan dalam *machine learning* adalah sebagai berikut:

1. Business Understanding

Pada tahap ini dibutuhkan pemahaman mengenai objek bisnis dengan tujuan utama menyelaraskan tujuan model untuk tujuan bisnis sehingga model yang dibangun tepat sasaran. *Output* dari tahap ini adalah masalah yang teridentifikasi atau *use case*. Identifikasi masalah dilakukan dengan pendekatan logis sehingga menghasilkan pertanyaan mendasar.

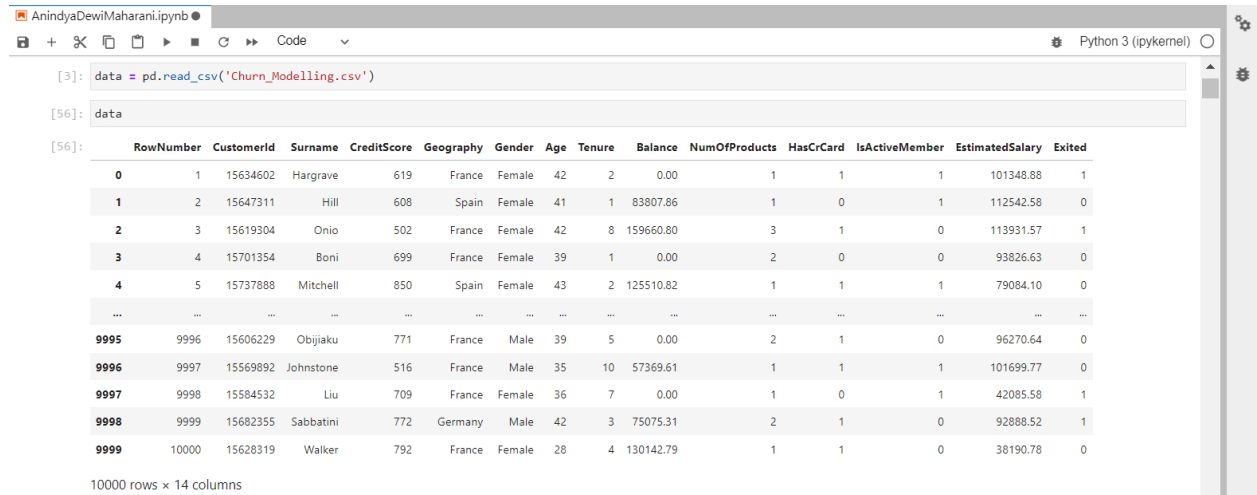
Untuk kasus ini business understanding dilakukan untuk memahami masalah yang dialami bank. Salah satu produk bank adalah pemberian pinjaman, namun sering kali ada nasabah yang mengalami tunggakan pembayaran. Maka dapat diidentifikasi masalahnya yaitu bank mengalami kesulitan dalam penentuan kelayakan penerima pinjaman untuk nasabah. Solusinya adalah dibutuhkan suatu penentuan *score* secara *objective* untuk menilai kelayakan nasabah dalam menerima pinjaman.

2. Data Understanding

Data understanding harus dilakukan untuk menemukan masalah pada data. Esensinya pada tahap ini berguna untuk memahami data yang dimiliki. Hal-hal yang mungkin ditemukan dalam tahap ini adalah distribusi data yang tidak normal, *outlier* dsb. *Output* dari tahap ini adalah data summary mulai dari atribut apa saja yang akan digunakan dan masalah-masalah mengenainya.

Untuk kasus ini *data understanding* dilakukan dengan menyiapkan data yang berisi karakteristik nasabah. Kemudian dilakukan *scanning* apakah data tersebut sudah sesuai

dengan yang dibutuhkan. Data tersebut bisa memuat atribut-atribut yang dapat berguna untuk penentuan *score*. Data didapatkan dari Kaggle sebanyak 10.000 *record*.



The screenshot shows a Jupyter Notebook interface with the following code and output:

```
[3]: data = pd.read_csv('Churn_Modelling.csv')
```

```
[56]: data
```

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Exited
0	1	15634602	Hargrave	619	France	Female	42	2	0.00	1	1	1	101348.88	1
1	2	15647311	Hill	608	Spain	Female	41	1	83807.86	1	0	1	112542.58	0
2	3	15619304	Onio	502	France	Female	42	8	159660.80	3	1	0	113931.57	1
3	4	15701354	Boni	699	France	Female	39	1	0.00	2	0	0	93826.63	0
4	5	15737888	Mitchell	850	Spain	Female	43	2	125510.82	1	1	1	79084.10	0
...
9995	9996	15606229	Obijaku	771	France	Male	39	5	0.00	2	1	0	96270.64	0
9996	9997	15569892	Johnstone	516	France	Male	35	10	57369.61	1	1	1	101699.77	0
9997	9998	15584532	Liu	709	France	Female	36	7	0.00	1	0	1	42085.58	1
9998	9999	15682355	Sabbatini	772	Germany	Male	42	3	75075.31	2	1	0	92888.52	1
9999	10000	15628319	Walker	792	France	Female	28	4	130142.79	1	1	0	38190.78	0

10000 rows x 14 columns

Atribut Data



The screenshot shows a Jupyter Notebook interface with the following code and output:

```
[5]: data.columns
```

```
[5]: Index(['RowNumber', 'CustomerId', 'Surname', 'CreditScore', 'Geography',  
        'Gender', 'Age', 'Tenure', 'Balance', 'NumOfProducts', 'HasCrCard',  
        'IsActiveMember', 'EstimatedSalary', 'Exited'],  
       dtype='object')
```

Gambar di atas menjelaskan atribut-atribut dalam data yaitu: RowNumber, CustomerID, Surname, CreditScore, Geography, Gender, Age, Tenure, Balance, NumOfProducts, HasCrCard, IsActiveMember, EstimatedSalary, Exited dari dataset *Churn_Modelling*.

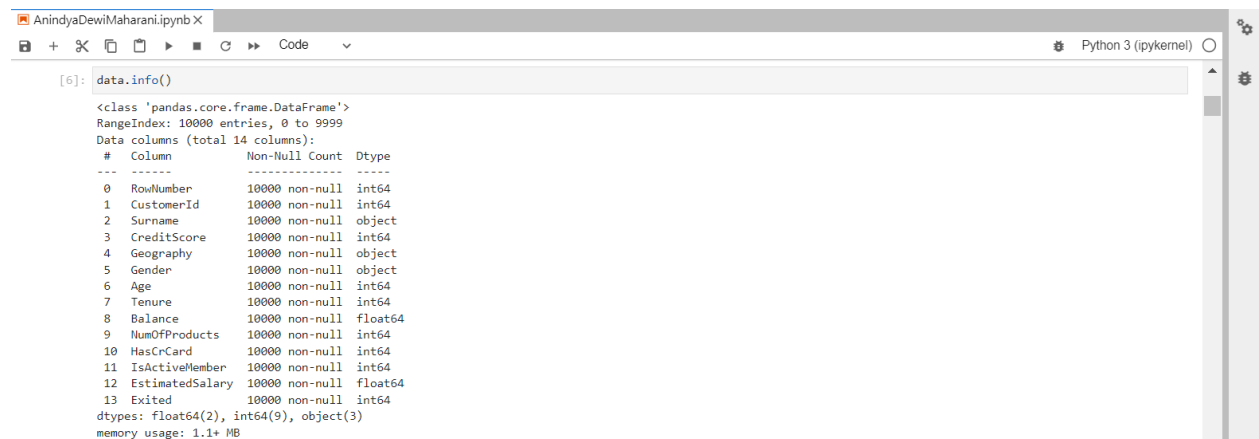
3. Data Preparation

Pada dasarnya tahap ini berguna untuk memperbaiki masalah yang sebelumnya ditemukan pada tahap *Data Understanding*. Langkah-langkah yang mungkin dilakukan dalam tahap ini seperti membuang *outlier*, mengubah tipe data, menghapus duplikasi data dsb. *Output* dari tahap ini adalah data bersih yang siap pakai untuk selanjutnya dilakukan pemodelan di tahap berikutnya. Pada tahap ini, mungkin ada atau tidak ada masalah dengan data margin Anda. Catatan yang digunakan dan duplikat. Untuk alasan ini, teknik pretreatment berikut diperlukan.

Data Cleaning

Membantu membersihkan nilai kosong, tupel yang tidak konsisten atau berpotensi kosong (nilai dan noise yang hilang).

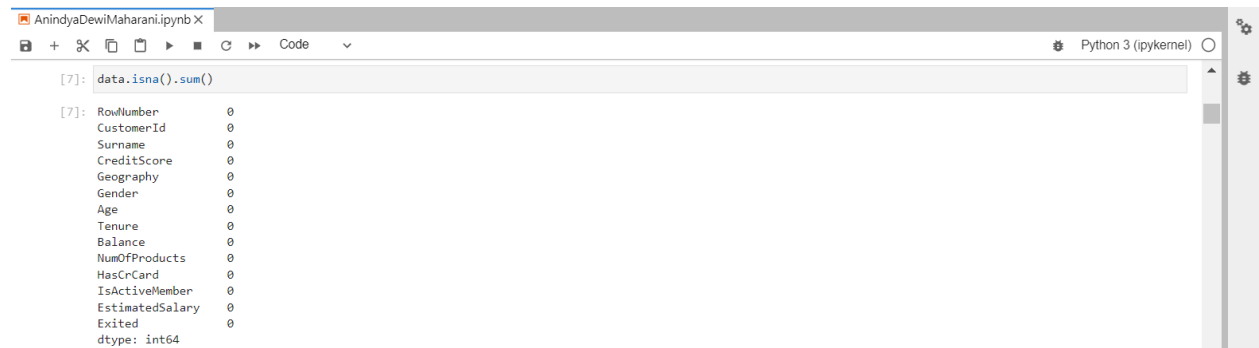
Mencari Missing Value



```
[6]: data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 14 columns):
 #   Column        Non-Null Count  Dtype  
---  -
 0   RowNumber     10000 non-null  int64  
 1   CustomerId    10000 non-null  int64  
 2   Surname       10000 non-null  object  
 3   CreditScore   10000 non-null  int64  
 4   Geography     10000 non-null  object  
 5   Gender        10000 non-null  object  
 6   Age           10000 non-null  int64  
 7   Tenure        10000 non-null  int64  
 8   Balance       10000 non-null  float64 
 9   NumOfProducts 10000 non-null  int64  
10   HasCrCard     10000 non-null  int64  
11   IsActiveMember 10000 non-null  int64  
12   EstimatedSalary 10000 non-null float64 
13   Exited        10000 non-null  int64  
dtypes: float64(2), int64(9), object(3)
memory usage: 1.1+ MB
```

Gambar di atas menampilkan bahwa semua atribut lengkap berjumlah 10.000 tanpa adanya yang bernilai Null. Adapun untuk tipe data juga dapat dijelaskan bahwa sudah benar semuanya.



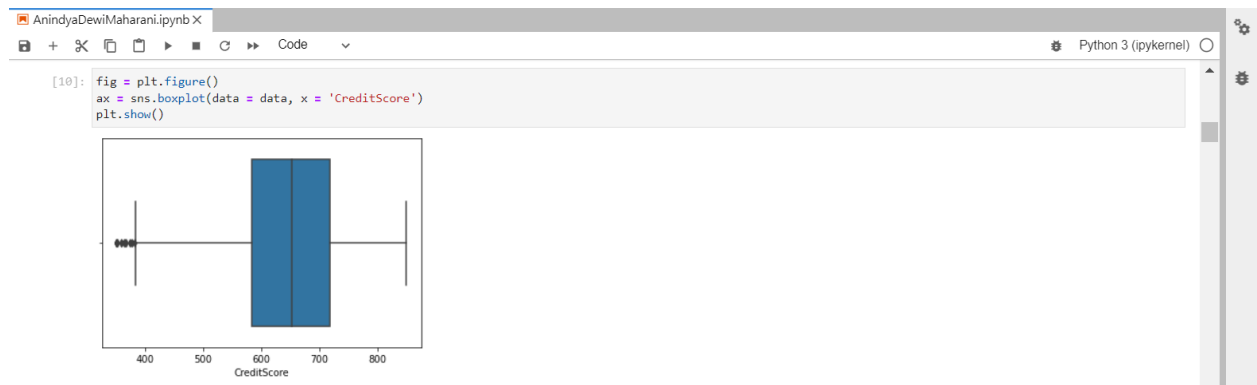
```
[7]: data.isna().sum()

[7]: RowNumber      0
     CustomerId    0
     Surname        0
     CreditScore    0
     Geography      0
     Gender         0
     Age            0
     Tenure         0
     Balance        0
     NumOfProducts  0
     HasCrCard      0
     IsActiveMember 0
     EstimatedSalary 0
     Exited         0
dtype: int64
```

Gambar di atas menampilkan bahwa tidak ada atribut yang datanya tidak ada nilai atau kosong.

Mencari Outliers

Atribut Credit Score



Gambar di atas menampilkan bahwa terdapat outliers pada atribut Credit Score

Atribut Credit Score Menggunakan Z Score

AnindyaDewiMaharani.ipynb X

+

🔗

📄

📄

▶

🔄

Code

▼

Python 3 (ipykernel)

[14]:

```

q1 = data["CreditScore"].quantile(0.25)
q3 = data["CreditScore"].quantile(0.75)

iqr = q3-q1 #Interquartile range
fence_low = q1-1.5*iqr
fence_high = q3+1.5*iqr

data.loc[(data["CreditScore"] < fence_low) | (data["CreditScore"] > fence_high)]

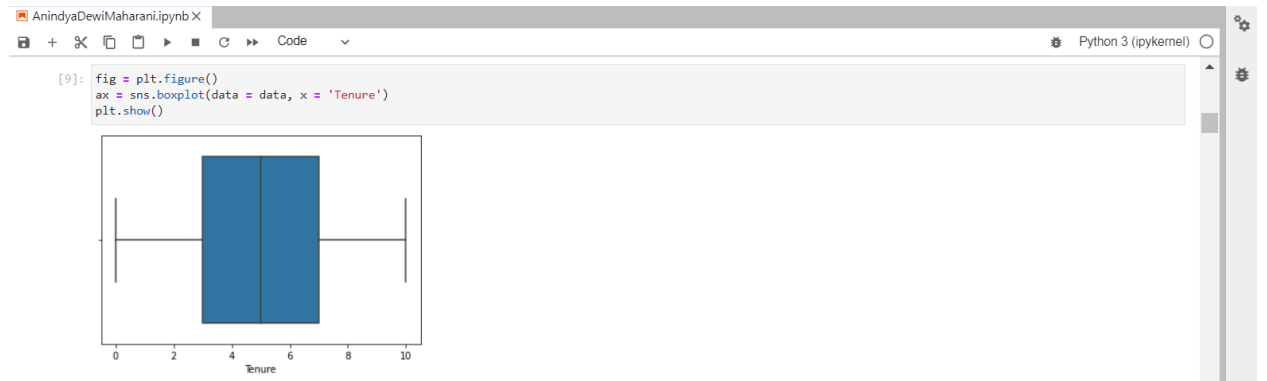
```

[14]:

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Exited	
	7	8	15656148	Obinna	376	Germany	Female	29	4	115046.74	4	1	0	119346.88	1
	942	943	15804586	Lin	376	France	Female	46	6	0.00	1	1	0	157333.69	1
	1193	1194	15779947	Thomas	363	Spain	Female	28	6	146098.43	3	1	0	100615.14	1
	1405	1406	15612494	Panicucci	359	France	Female	44	6	128747.69	1	1	0	146955.71	1
	1631	1632	15685372	Azubuike	350	Spain	Male	54	1	152677.48	1	1	1	191973.49	1
	1838	1839	15758813	Campbell	350	Germany	Male	39	0	109733.20	2	0	0	123602.11	1
	1962	1963	15692416	Aikenhead	358	Spain	Female	52	8	143542.36	3	1	0	141959.11	1
	2473	2474	15679249	Chou	351	Germany	Female	57	4	163146.46	1	1	0	169621.69	1
	2579	2580	15597896	Ozoemena	365	Germany	Male	30	0	127760.07	1	1	0	81537.85	1
	8154	8155	15791533	Ch'ien	367	Spain	Male	42	6	93608.28	1	1	0	168816.73	1
	8723	8724	15803202	Onyekachi	350	France	Male	51	10	0.00	1	1	1	125823.79	1
	8762	8763	15765173	Lin	350	France	Female	60	3	0.00	1	0	0	113796.15	1
	9210	9211	15792650	Watts	382	Spain	Male	36	0	0.00	1	1	1	179540.73	1
	9356	9357	15734711	Loggia	373	France	Male	42	7	0.00	1	1	0	77786.37	1
	9624	9625	15668309	Maslow	350	France	Female	40	0	111098.85	1	1	1	172321.21	1

Gambar di atas adalah test Z Score terhadap atribut Credit Score didapatkan hasil terdapat beberapa outliers tetapi tidak perlu dilakukan tindakan karena merupakan data yang sah sesuai dengan keadaan aslinya.

Atribut Tenure



Gambar di atas menampilkan bahwa tidak terdapat outliers pada atribut Tenure.

Atribut Tenure Menggunakan Z Score

```
AnindyaDewiMaharani.ipynb X
Python 3 (ipykernel)

[13]: q1 = data["Tenure"].quantile(0.25)
      q3 = data["Tenure"].quantile(0.75)

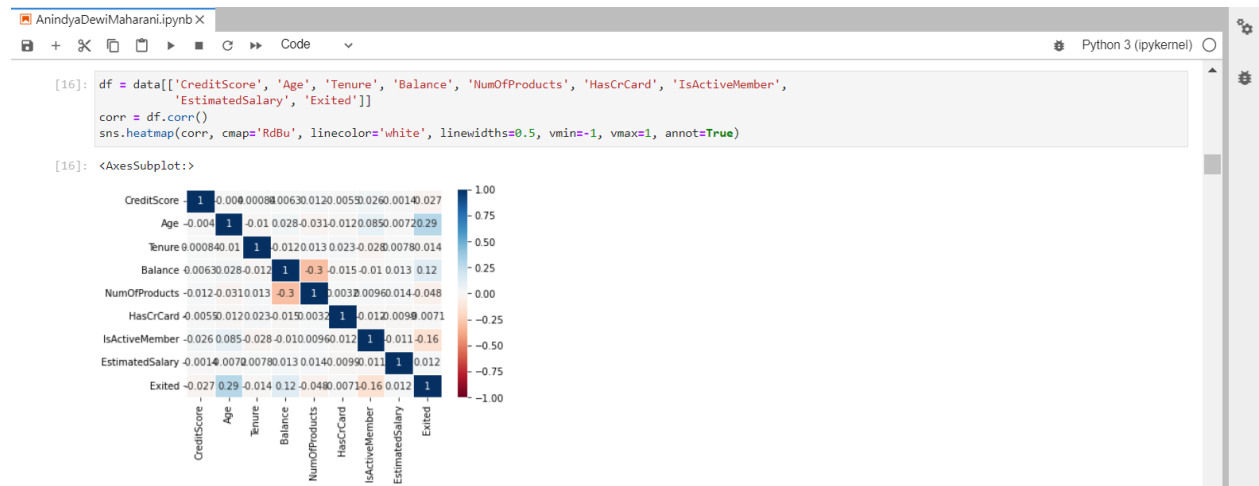
      iqr = q3-q1 #Interquartile range
      fence_low = q1-1.5*iqr
      fence_high = q3+1.5*iqr

      data.loc[(data["Tenure"] < fence_low) | (data["Tenure"] > fence_high)]

[13]: RowNumber CustomerId Surname CreditScore Geography Gender Age Tenure Balance NumOfProducts HasCrCard IsActiveMember EstimatedSalary Exited
```

Gambar di atas menampilkan bahwa tidak terdapat outliers pada atribut Tenure saat dicoba menggunakan Z Score.

Melakukan Uji Korelasi



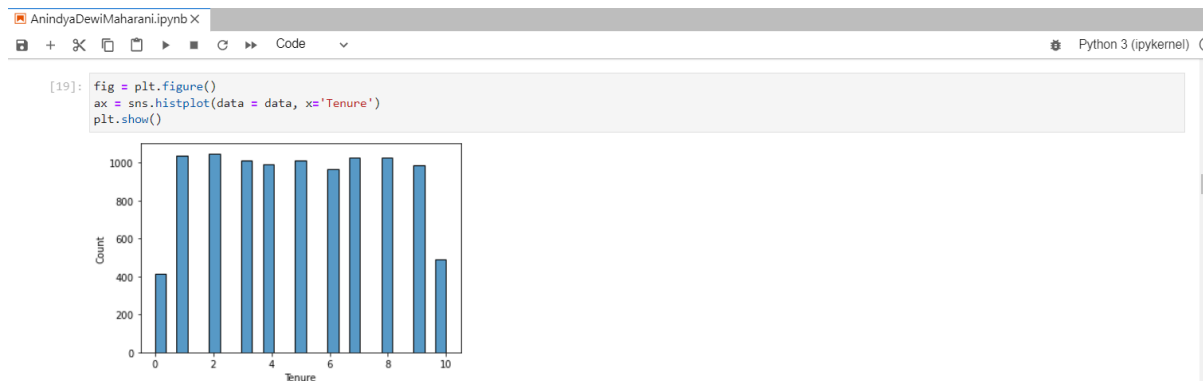
Gambar di atas menjelaskan angka setiap korelasi antar atribut tidak ada yang mendekati satu yang berarti korelasi antar atribut rendah. Nilai tertinggi yaitu 0.3 yang mana berarti korelasinya rendah.

Mencari Korelasi Menggunakan Heat Map



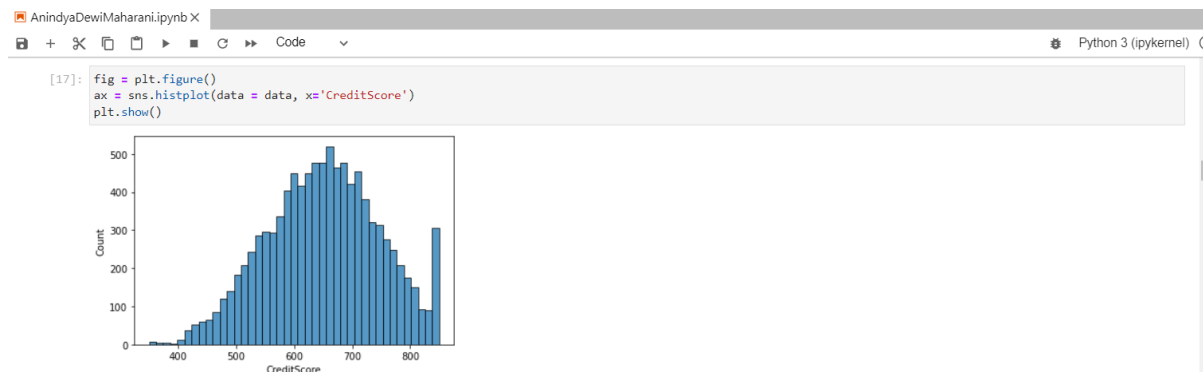
Mencari Korelasi Menggunakan Visualiasasi

- **Atribut Tenure**



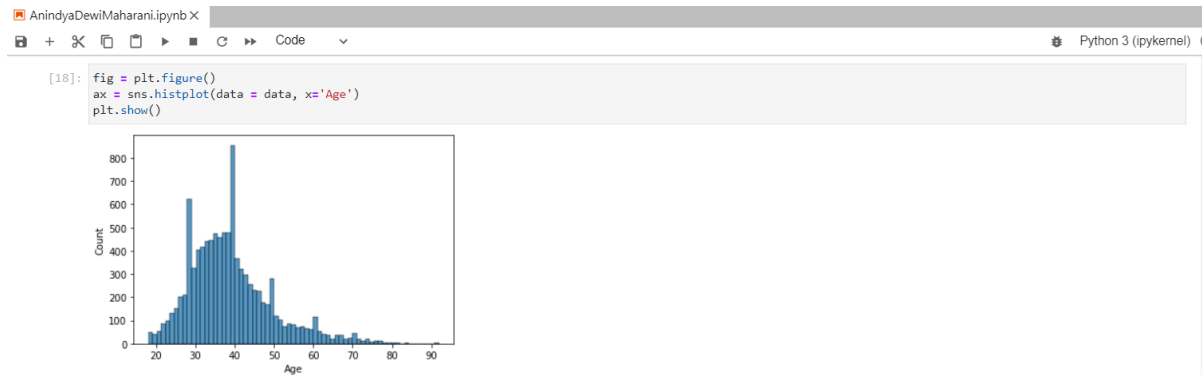
Gambar di atas menampilkan bahwa pelanggan dengan tenure lama dan singkat sedikit berdasarkan visualisasi diatas.

- **Atribut Credit Score**



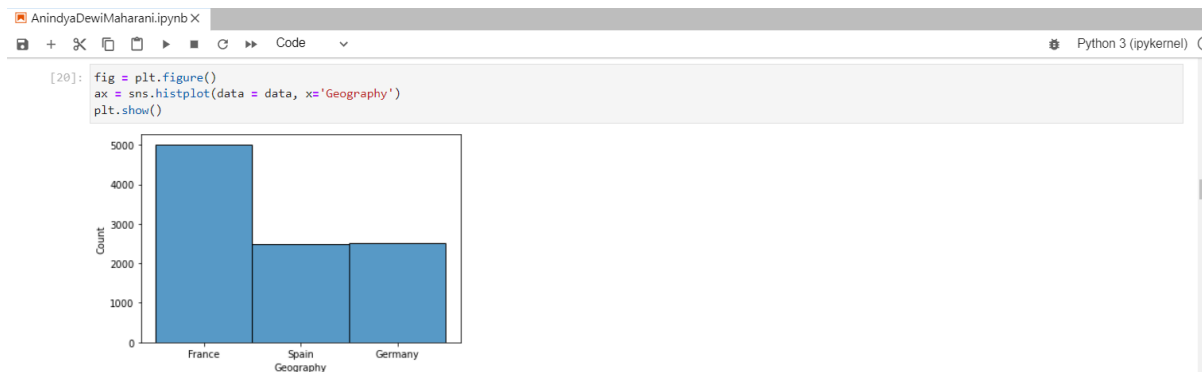
Gambar di atas menampilkan bahwa banyak pelanggan yang memiliki credit score tinggi.

- **Atribut Age**



Gambar di atas menampilkan bahwa pelanggan dengan umur 29-39 tahun banyak mendominasi pada data.

- **Atribut Geography**



Gambar di atas menampilkan bahwa pelanggan yang berlokasi di negara France mendominasi data.

4. Modeling

Setelah data dibersihkan maka data siap untuk diolah dengan menggunakan algoritma yang cocok dengan identifikasi masalah yang telah dirumuskan. Pada tahap ini juga pihak bank

harus menentukan Teknik Data Mining yang ingin dilakukan. Secara garis besar terdapat empat Teknik yaitu klasifikasi, klusteriasi, forecasting dan estimasi. Setiap Teknik tersebut juga memiliki algoritma yang bisa digunakan untuk melakukan pemodelan. Algoritma dalam *machine learning* secara garis besar terbagi dalam 3 kategori yaitu *supervised learning*, *unsupervised learning*, dan *reinforcement learning*. Selanjutnya dilakukan proses pemodelan data sesuai dengan algoritma yang telah dipilih. Output dari tahap ini adalah berupa model data yang memuat pengetahuan-pengetahuan baru untuk selanjutnya akan diuji pada tahap *evaluation*.

Tahap Modeling



```
[41]: # Random Forest
from sklearn.ensemble import RandomForestClassifier
classifier_rf = RandomForestClassifier(random_state=5, n_jobs=-1, max_depth=5,
                                     n_estimators=100, oob_score=True)
classifier_rf.fit(x_train, y_train)

[42]: RandomForestClassifier(max_depth=5, n_jobs=-1, oob_score=True, random_state=5)

[43]: # Hyperparameter Tuning for Random Forest
rf = RandomForestClassifier(random_state=5, n_jobs=-1)

[44]: # Create one variable (params) to deposit whatever we will try to do with the model
params = {
    'max_depth': [2, 3, 5, 10, 20],
    'min_samples_leaf': [5, 10, 20, 50, 100, 200],
    'n_estimators': [10, 25, 50, 100, 200]
}

[45]: from sklearn.model_selection import GridSearchCV
# Instantiate the grid search model
grid_search = GridSearchCV(estimator=rf,
                           param_grid=params,
                           cv=5,
                           n_jobs=-1, verbose=1, scoring='accuracy')
grid_search.fit(x_train, y_train)

Fitting 5 folds for each of 180 candidates, totalling 900 fits

[46]: GridSearchCV(cv=5, estimator=RandomForestClassifier(n_jobs=-1, random_state=5),
                  param_grid={'max_depth': [2, 3, 5, 10, 20],
                              'min_samples_leaf': [5, 10, 20, 50, 100, 200],
                              'n_estimators': [10, 25, 50, 100, 200]},
                  scoring='accuracy', verbose=1)

[47]: # To see best score
grid_search.best_score_

[47]: 0.8617499999999999

[48]: rf_best = grid_search.best_estimator_
rf_best.fit(x_train, y_train)

[49]: RandomForestClassifier(max_depth=10, min_samples_leaf=5, n_estimators=25,
                           n_jobs=-1, random_state=5)
```

5. Evaluation

Setelah didapatkan hasil dari pemodelan pada fase sebelumnya selanjutnya harus melakukan evaluasi dari hasil pemodelan tersebut apakah sudah sesuai dengan apa yang diinginkan pada tahap awal atau tidak. Apabila sudah sesuai hasilnya maka akan didapatkan sebuah keputusan untuk diambil. Evaluasi bisa menggunakan *Confussion Matrix*, *Classification Report*, *AUC*. Berikut hasil evaluasi dari ketiga model matik :

- *Confussion Matrix*

```
AnindyaDewiMaharani.ipynb X
Python 3 (ipykernel)

[50]: ## confusion_matrix
from sklearn.metrics import confusion_matrix
print("Logistic Regression : \n", confusion_matrix(y_test, y_lr))
print("Decision Tree : \n", confusion_matrix(y_test, y_dtree))
print("Random Forest : \n", confusion_matrix(y_test, y_rf_before))
print("Random Forest dengan Hyperparameter Tuning: \n", confusion_matrix(y_test, y_rf_after))

Logistic Regression :
[[1565  30]
 [ 388 17]]
Decision Tree :
[[1372 223]
 [ 192 213]]
Random Forest :
[[1570  25]
 [ 284 121]]
Random Forest dengan Hyperparameter Tuning:
[[1546  49]
 [ 243 162]]
```

Gambar di atas menjelaskan perbedaan algoritma yang dipilih dapat menghasilkan nilai confusion matrix yang berbeda pula. Confusion matrix menunjukkan setidaknya 4 keadaan data True Positive (TP), False Negative (FN), False Positive (FP), dan True Negative (TN).

- *Classification Report*

```
AnindyaDewiMaharani.ipynb X
Python 3 (ipykernel)

[51]: ## classification report
from sklearn.metrics import classification_report
print("Logistic Regression : \n\n", classification_report(y_test, y_lr))
print("Decision Tree : \n\n", classification_report(y_test, y_dtree))
print("Random Forest : \n\n", classification_report(y_test, y_rf_before))
print("Random Forest dengan Hyperparameter Tuning: \n\n", classification_report(y_test, y_rf_after))

Logistic Regression :
      precision    recall  f1-score   support

0       0.88      0.98      0.88      1595
1       0.36      0.04      0.08       405

accuracy      0.79      2000
macro avg     0.58      0.51      0.48      2000
weighted avg   0.71      0.79      0.72      2000

Decision Tree :
      precision    recall  f1-score   support

0       0.88      0.86      0.87      1595
1       0.49      0.53      0.51       405

accuracy      0.79      2000
macro avg     0.68      0.69      0.69      2000
weighted avg   0.88      0.79      0.88      2000

Random Forest :
      precision    recall  f1-score   support

0       0.85      0.98      0.91      1595
1       0.83      0.30      0.44       405

accuracy      0.84      2000
macro avg     0.64      0.67      0.67      2000
weighted avg   0.84      0.85      0.81      2000

Random Forest dengan Hyperparameter Tuning:
      precision    recall  f1-score   support

0       0.86      0.97      0.91      1595
1       0.77      0.40      0.53       405

accuracy      0.85      2000
macro avg     0.82      0.68      0.72      2000
weighted avg   0.84      0.85      0.84      2000
```

Gambar di atas menjelaskan nilai *precision*, *recall*, *F1-Score* dan *support*. Dimana algoritma yang berbeda menghasilkan nilai *precision*, *recall*, *F1-Score* dan *support* yang berbeda juga.

- *AUC*

```
AnindyaDewiMaharani.ipynb X Python 3 (ipykernel)

[52]: ## AUC
from sklearn.metrics import roc_curve, auc
fpr, tpr, thresholds = roc_curve(y_test, y_lr, pos_label=1) # pos_label: positive Label
print("Logistic Regression :", auc(fpr, tpr))
fpr, tpr, thresholds = roc_curve(y_test, y_dtree, pos_label=1) # pos_label: positive Label
print("Decision Tree :", auc(fpr, tpr))
fpr, tpr, thresholds = roc_curve(y_test, y_rf_before, pos_label=1) # pos_label: positive Label
print("Random Forest :", auc(fpr, tpr))
fpr, tpr, thresholds = roc_curve(y_test, y_rf_after, pos_label=1) # pos_label: positive Label
print("Random Forest dengan Hyperparameter Tuning:", auc(fpr, tpr))

Logistic Regression : 0.5115832656062542
Decision Tree : 0.6930570068501103
Random Forest : 0.6415457254537714
Random Forest dengan Hyperparameter Tuning: 0.6846394984326019
```

Gambar di atas menjelaskan besar nilai Logistic Regerssion adalah 0.5115832656062542, Decision Tree adalah 0.6908820000774024, Random Forest adalah 0.6415457254537714 dan Random Forest dengan Hyperparameter Tuning adalah 0.6846394984326019.

6. Deployment

Setelah evaluasi dilakukan dan keputusan telah dibuat maka selanjutnya informasi mengenai pengambilan keputusan tersebut harus disampaikan melalui sistem kepada department yang menangani peminjaman uang. Contoh penyebaran informasi tersebut bisa melalui sistem atau dashboard pada website.

