MACHINE LEARNING

Supervised learning

Bob Chrismansyah H071201087

Supervised learning

Proses pembelajaran yang dilakukan di bawah pengawasan atau adanya supervisor

- Clasification (Categorical) and Regression (Numerical)
- **Logistic Regression**
- Model Ensemble
- Time series

Import Library

- numpy
- pandas
- matloblib.pylot
- seaborn
- scipy
- sklearn

```
import  numpy as np
import pandas as pd
```

```
from scipy import stats
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn import preprocessing
from scipy.stats import zscore
from sklearn import metrics
from sklearn.metrics import accuracy_score, confusion_matrix
from sklearn.metrics import classification_report
```

Read Data
Merge Data

- Data1.csv
- Data2.csv

```
data1=pd.read_csv('Data1.csv')
data1.head()
```

	ID	Age	CustomerSince	HighestSpend	ZipCode	HiddenScore	MonthlyAverageSpend	Level
0	1	25	1	49	91107	4	1.6	1
1	2	45	19	34	90089	3	1.5	1
2	3	39	15	11	94720	1	1.0	1
3	4	35	9	100	94112	1	2.7	2
4	5	35	8	45	91330	4	1.0	2

```
data2=pd.read_csv('Data2.csv')
data2.head()
```

	ID	Mortgage	Security	FixedDepositAccount	InternetBanking	CreditCard	LoanOnCard
0	1	0	1	0	0	0	NaN
1	2	0	1	0	0	0	NaN
2	3	0	0	0	0	0	NaN
3	4	0	0	0	0	0	NaN
4	5	0	0	0	0	1	NaN

```
data=data1.merge(data2)
data.head()
```

	ID	Age	CustomerSince	HighestSpend	ZipCode	HiddenScore	MonthlyAverageSpend	Level	Mortg
0	1	25	1	49	91107	4	1.6	1	
1	2	45	19	34	90089	3	1.5	1	
2	3	39	15	11	94720	1	1.0	1	
3	4	35	9	100	94112	1	2.7	2	
4	5	35	8	45	913				

```
data.drop('ID',axis=1,inplace=True)
```

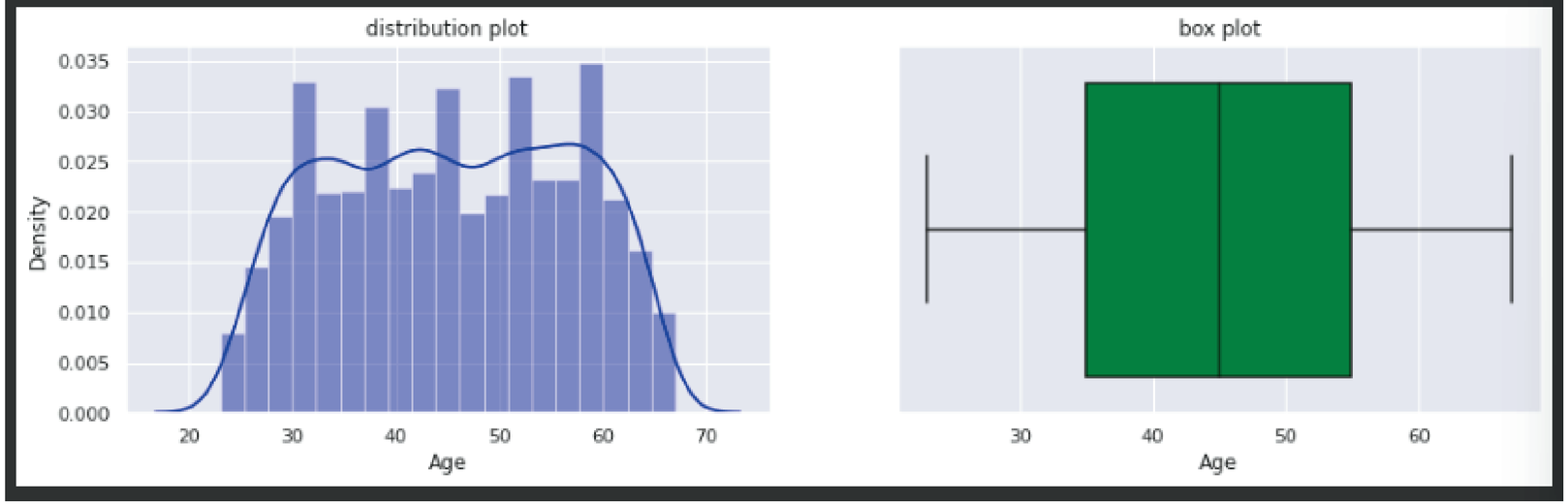
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 5000 entries, 0 to 4999
Data columns (total 13 columns):
 #   Column               Non-Null Count  Dtype  
---  ------               --------------  -----  
 0   Age                  5000 non-null   int64  
 1   CustomerSince        5000 non-null   int64  
 2   HighestSpend         5000 non-null   int64  
 3   ZipCode              5000 non-null   int64  
 4   HiddenScore          5000 non-null   int64  
 5   MonthlyAverageSpend  5000 non-null   float64
 6   Level                5000 non-null   int64  
 7   Mortgage             5000 non-null   int64  
 8   Security             5000 non-null   int64  
 9   FixedDepositAccount  5000 non-null   int64  
 10  InternetBanking      5000 non-null   int64  
 11  CreditCard           5000 non-null   int64  
 12  LoanOnCard           4980 non-null   float64
dtypes: float64(2), int64(11)
memory usage: 546.9 KB
```

Analisis

- **Univariate**
- Bivariate
- Multivariate

```
plt.figure(figsize=(15,4))
plt.subplot(1,2,1)
plt.title('distribution plot')
sns.distplot(data['Age'],color='blue')
plt.subplot(1,2,2)
plt.title('box plot')
sns.boxplot(data['Age'],orient='h',color='green')
plt.show()
```

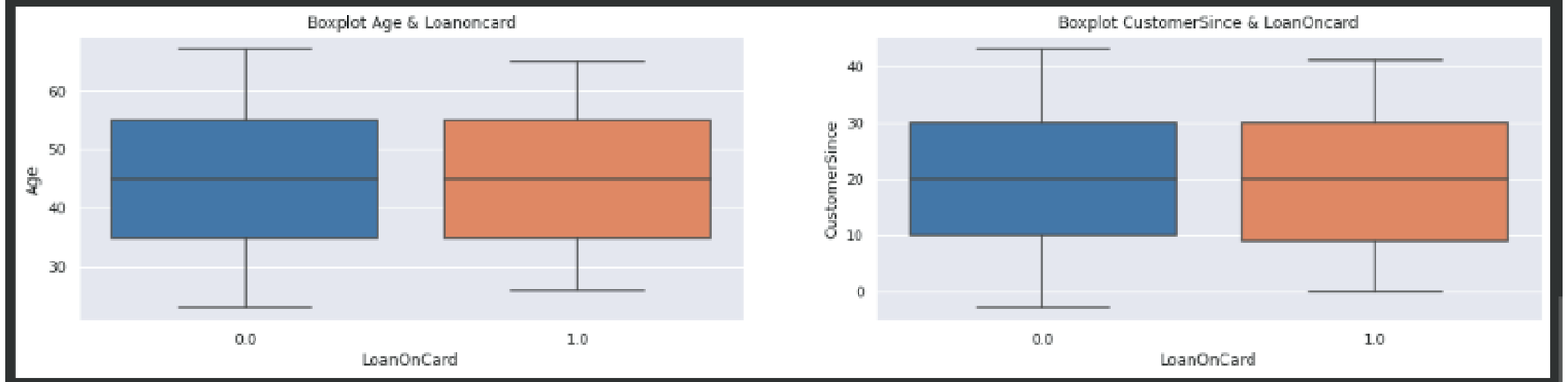


Univariate merupakan analisa data feature tunggal

Analisis

- Univariate
- **Bivariate**
- Multivariate

```
plt.figure(figsize=(20,4))
plt.subplot(1,2,1)
plt.title('Boxplot Age & Loanoncard')
sns.boxplot(x='LoanOnCard', y='Age', data = data );
plt.subplot(1,2,2)
plt.title('Boxplot CustomerSince & LoanOncard')
sns.boxplot(x='LoanOnCard',y='CustomerSince', data=data );
```



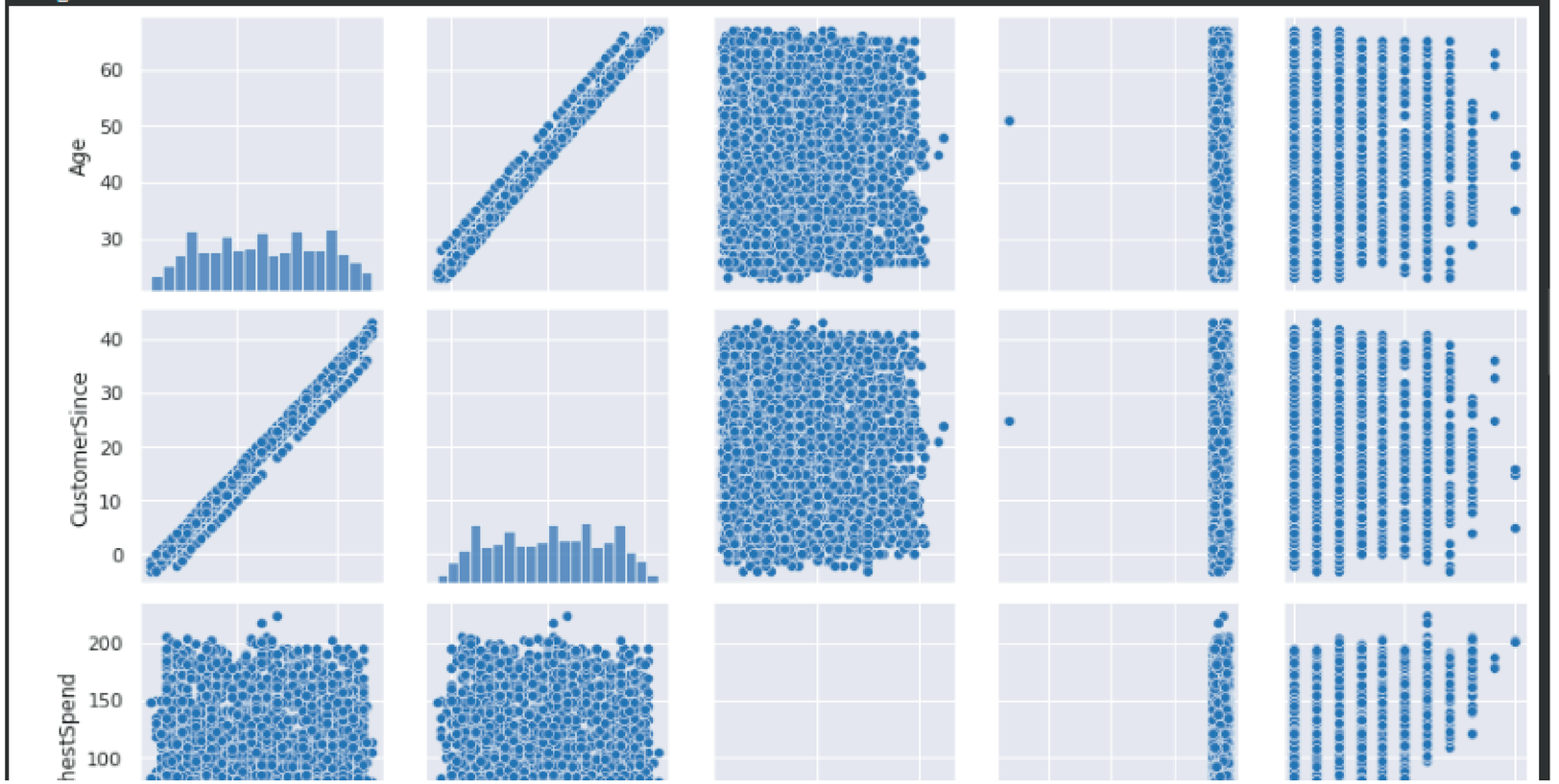
Bivariate analysis digunakan untuk untuk menganalisa 2 variables dan menemukan sebuah relasi. Analisis bivariate juga merupakan salah satu cara untuk menggunakan koefisien korelasi dalam rangka menemukan apakah dua variabel memiliki relasi atau tidak.

Analisis

- Univariate
- Bivariate
- **Multivariate**

```
plt.figure(figsize=(20,8));
sns.pairplot(data[['Age', 'CustomerSince', 'HighestSpend', 'ZipCode' ,'MonthlyAverageSpend']])
```

```
<seaborn.axisgrid.PairGrid at 0x7ff289d61190>
<Figure size 1440x576 with 0 Axes>
```



Multivariate analysis digunakan untuk menganalisa lebih dari 2 variabel di waktu yang sama, trends yang dihasilkan bisa menjadi multidimensi secara alami

Membersihkan data

Data yang tidak terlalu mempengaruhi variabel dihapus untuk mengurangi noise dalam data saat pengujian

```
df=data.drop(columns=['Age','ZipCode','CustomerSince'])
```

Train

```
x=df.drop(columns='LoanOnCard')
y=data['LoanOnCard']
```

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3,random_state=1)
```

Keakuratan training model

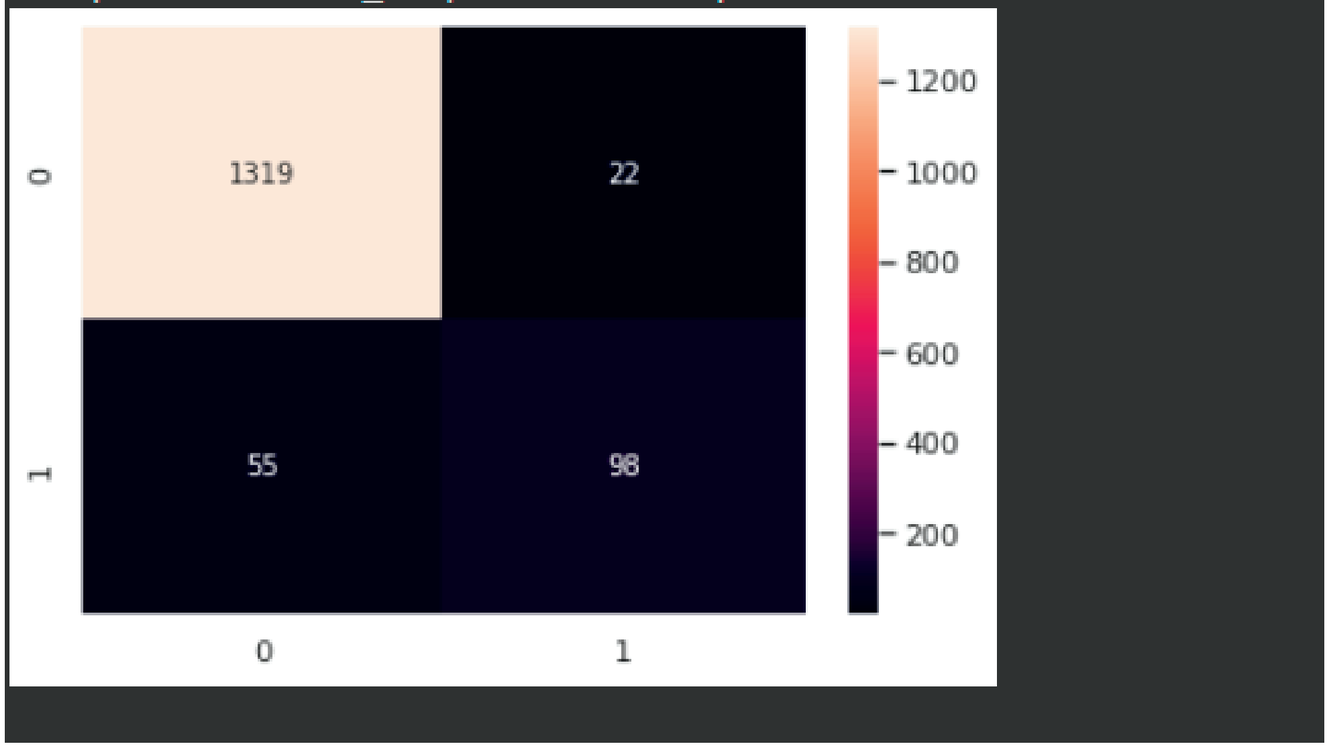
```
pred=logit.predict(x_test)
print('Accuracy on Training data:',logit.score(x_train, y_train) )
print('Accuracy on Test data:',logit.score(x_test, y_test) )
```

```
Accuracy on Training data: 0.9515203671830178
Accuracy on Test data: 0.9484605087014726
```

Confusion Matrix

```
sns.heatmap(confusion_matrix(y_test,pred),annot=True,fmt='g')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7ff289f87b20>
```



- True Positif
- False Positif
- True Negatif
- False Negatif

Classification Matrix

```
print("classification  Matrix:\n",classification_report(y_test,pred));
```

```
classification  Matrix:
              precision    recall  f1-score   support

         0.0       0.96      0.98      0.97      1341
         1.0       0.82      0.64      0.72       153

    accuracy                           0.95      1494
   macro avg       0.89      0.81      0.84      1494
weighted avg       0.95      0.95      0.95      1494
```