Experiment dan Feature Extraction

Nama : Linggar Maretva Cendani
NIM : 24060117120031

Data Preparation

Data Preparation

Sebelum data kita gunakan untuk melakukan pemodelan dan klasifikasi, data perlu dipersiapkan agar bisa diolah dan digunakan dalam proses training dan klasifikasi.

Data yang digunakan adalah data **Pima Indians Diabetes Database**, yang berisi 8 fitur numerik dan 1 target ('Outcome') yang berisi 2 kelas (1 untuk pengidap diabetes, dan 0 untuk bukan pengidap diabetes).

Data Preparation yang dilakukan adalah membaca file csv, dan handle missing value (jika ada), standarisasi fitur (agar memiliki skala yang sama), serta memisahkan data untuk training dan testing.

Data Preparation (lanjutan.)

Berikut adalah informasi contoh data pada dataset Pima Indians Diabetes dan infonya. Karena tidak ada *missing value*, maka kita tidak perlu melakukan handle *missing value data*.

Terlihat data memiliki skala yang berbeda - beda, sehingga perlu dilakukan standarisasi nilai fitur.

```
[14] df_diabetes.head()
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1

```
df_diabetes.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
 #   Column                    Non-Null Count  Dtype  
---  ------                    --------------  -----  
 0   Pregnancies               768 non-null    int64  
 1   Glucose                   768 non-null    int64  
 2   BloodPressure             768 non-null    int64  
 3   SkinThickness             768 non-null    int64  
 4   Insulin                   768 non-null    int64  
 5   BMI                       768 non-null    float64
 6   DiabetesPedigreeFunction  768 non-null    float64
 7   Age                       768 non-null    int64  
 8   Outcome                   768 non-null    int64  
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
```

Data Preparation (lanjutan.)

```
df_diabetes.describe()
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
count	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000
mean	3.845052	120.894531	69.105469	20.536458	79.799479	31.992578	0.471876	33.240885	0.348958
std	3.369578	31.972618	19.355807	15.952218	115.244002	7.884160	0.331329	11.760232	0.476951
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.078000	21.000000	0.000000
25%	1.000000	99.000000	62.000000	0.000000	0.000000	27.300000	0.243750	24.000000	0.000000
50%	3.000000	117.000000	72.000000	23.000000	30.500000	32.000000	0.372500	29.000000	0.000000
75%	6.000000	140.250000	80.000000	32.000000	127.250000	36.600000	0.626250	41.000000	1.000000
max	17.000000	199.000000	122.000000	99.000000	846.000000	67.100000	2.420000	81.000000	1.000000

Data Preparation (lanjutan.)

Dataset yang sudah dipisah kolom untuk fitur dan kelas target, kemudian dilakukan standarisasi untuk tiap nilai pada kolom tiap fitur agar memiliki skala yang sama dan dapat dilakukan komputasi dengan baik.

```
x_df_transformed
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age
0	0.639947	0.848324	0.149641	0.907270	-0.692891	0.204013	0.468492	1.425995
1	-0.844885	-1.123396	-0.160546	0.530902	-0.692891	-0.684422	-0.365061	-0.190672
2	1.233880	1.943724	-0.263941	-1.288212	-0.692891	-1.103255	0.604397	-0.105584
3	-0.844885	-0.998208	-0.160546	0.154533	0.123302	-0.494043	-0.920763	-1.041549
4	-1.141852	0.504055	-1.504687	0.907270	0.765836	1.409746	5.484909	-0.020496

Data Preparation (lanjutan.)

Selanjutnya tahap terakhir adalah melakukan *splitting* atau membagi data untuk training dan testing.

Disini, pembagian untuk testing ada adalah 25% dari keseluruhan data, sehingga training datanya adalah 75%.

```
# memisahkan data untuk training dan testing
from sklearn.model_selection import train_test_split

x_train, x_test, y_train, y_test = train_test_split(x_transformed, y, test_size=0.25, random_state=0)
```

Eksperimen dengan Model Random Forest

Random Forest

Eksperimen yang pertama dilakukan adalah dengan menggunakan algoritma Random Forest sebagai model yang diujicobakan. Digunakan nilai parameter sebagai berikut :

```
[28] parameters = {'bootstrap': True,
                   'min_samples_leaf': 3,
                   'n_estimators': 50,
                   'min_samples_split': 10,
                   'max_features': 'sqrt',
                   'max_depth': 6,
                   'max_leaf_nodes': None}
```

Dan didapatkan **akurasi sebesar 78%**.

```
[33] score = accuracy_score(y_test ,RF_predictions)
     print('Accuracy Random Forest Model:',score)

     Accuracy Random Forest Model: 0.78125
```

Eksperimen dengan Model Neural Network

Neural Network

Selanjutnya dilakukan eksperimen dengan Neural network dengan arsitektur sebagai berikut :

```
# Build a neural network :
NN_model = Sequential()

NN_model.add(Dense(256, input_dim = x_transformed.shape[1], activation='relu'))
NN_model.add(Dense(256, activation='relu'))
NN_model.add(Dense(256, activation='relu'))
NN_model.add(Dense(256, activation='relu'))
NN_model.add(Dense(1, activation='sigmoid'))
NN_model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
```

Digunakan binary_crossentropy sebagai loss function dan adam sebagai optimizer-nya, dan didapatkan **akurasi sebesar 77%** (dengan 150 epoch).

```
[42] score = accuracy_score(y_test ,predictions)
     print('Test Accuracy:',score)

     Test Accuracy: 0.7708333333333334
```

Other Classifier (Stacking Ensemble Learning)

Ensemble Learning

Terakhir kita lakukan Ensemble Learning dengan cara stacking menggunakan 13 buah algoritma yang akan digunakan sebagai model klasifikasi. 13 model yang akan digunakan adalah :

- Logistic Regression
- SVC
- Gradient Boosting Classifier
- Extra Trees Classifier
- Bagging Classifier
- AdaBoost Classifier
- Gaussian NB
- MLP Classifier
- XGB Classifier
- LGBM Classisfier
- K Nearest Neighbour Classifier
- Decison Tree Classifier
- Random Forest Classifier

Ensemble Learning (lanjutan.)

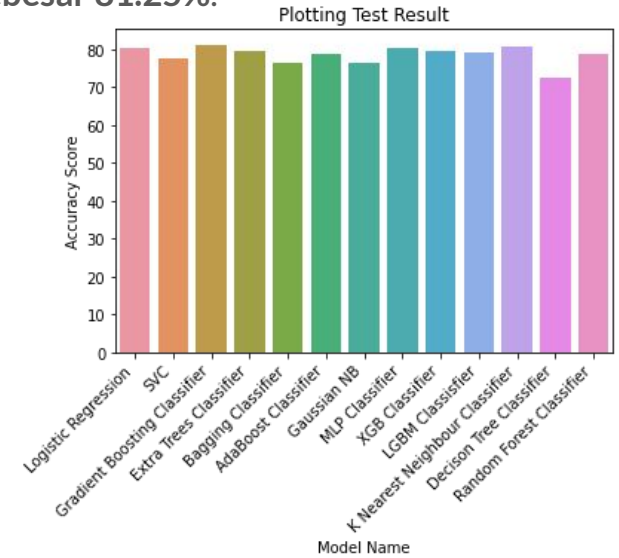
Dari 13 Model yang digunakan, dihasilkan akurasinya masing - masing sebagai berikut :

```
Validation result for Logistic Regression
accuracy : 80.21%
train and test time: 0.01s
----------------------------------------------------
Validation result for SVC
accuracy : 77.60%
train and test time: 0.01s
----------------------------------------------------
Validation result for Gradient Boosting Classifier
accuracy : 81.25%
train and test time: 0.16s
----------------------------------------------------
Validation result for Extra Trees Classifier
accuracy : 79.69%
train and test time: 0.16s
----------------------------------------------------
Validation result for Bagging Classifier
accuracy : 76.56%
train and test time: 0.03s
----------------------------------------------------
Validation result for AdaBoost Classifier
accuracy : 78.65%
train and test time: 0.10s
----------------------------------------------------
```

```
Validation result for Gaussian NB
accuracy : 76.56%
train and test time: 0.00s
----------------------------------------------------
Validation result for MLP Classifier
accuracy : 80.21%
train and test time: 0.70s
----------------------------------------------------
Validation result for XGB Classifier
accuracy : 79.69%
train and test time: 0.09s
----------------------------------------------------
Validation result for LGBM Classisfier
accuracy : 79.17%
train and test time: 0.07s
----------------------------------------------------
Validation result for K Nearest Neighbour Classifier
accuracy : 80.73%
train and test time: 0.01s
----------------------------------------------------
Validation result for Decison Tree Classifier
accuracy : 72.40%
train and test time: 0.00s
----------------------------------------------------
Validation result for Random Forest Classifier
accuracy : 78.65%
train and test time: 0.20s
----------------------------------------------------
```

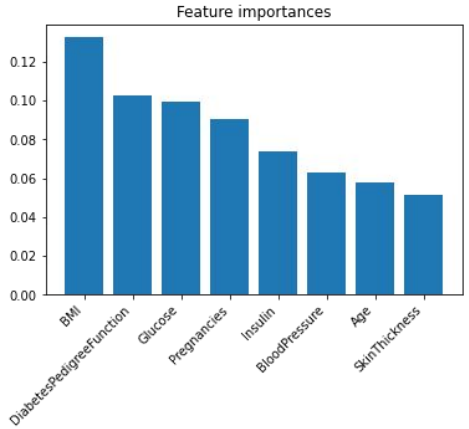
Ensemble Learning (plotting)

Selanjutnya dihasilkan plotting sebagai berikut, dari hasil akurasi, didapatkan yang tertinggi adalah model **Gradient Boosting Classifier sebesar 81.25%.**



Ensemble Learning (feature importance)

Dari 8 fitur yang ada pada data, kita bisa mencari fitur - fitur apa saja yang memilki pengaruh paling besar untuk menentukan hasil klasifikasi, sehingga nantinya bisa dilakukan feature reduction, atau menggunakan beberapa fitur terpenting saja untuk mempercepat komputasi tanpa mengurangi akurasi secara signifikan. Berikut adalah urutan fitur mulai dari yang memiliki pengaruh paling besar (paling penting).



Ensemble Learning (stacking)

Kemudian dilakukan stacking ensemble learning, dan didapatkan hasil sebagai berikut :

```
Number of model: 13
task:         [regression]
metric:       [accuracy_score]
mode:         [oof_pred_bag]
n_models:     [13]

model  0:     [LogisticRegression]
    ----
    MEAN:     [0.76909722] + [0.02009694]
    FULL:     [0.76909722]

model  1:     [SVC]
    ----
    MEAN:     [0.75347222] + [0.02009694]
    FULL:     [0.75347222]

model  2:     [GradientBoostingClassifier]
    ----
    MEAN:     [0.74652778] + [0.01299187]
    FULL:     [0.74652778]

model  3:     [ExtraTreesClassifier]
    ----
    MEAN:     [0.73437500] + [0.02586747]
    FULL:     [0.73437500]

model  4:     [BaggingClassifier]
    ----
    MEAN:     [0.73958333] + [0.01948780]
    FULL:     [0.73958333]

model  5:     [AdaBoostClassifier]
    ----
    MEAN:     [0.72743056] + [0.02734031]
    FULL:     [0.72743056]

model  6:     [GaussianNB]
    ----
    MEAN:     [0.75520833] + [0.02655739]
    FULL:     [0.75520833]

model  7:     [MLPClassifier]
    ----
    MEAN:     [0.74479167] + [0.03897560]
    FULL:     [0.74479167]

model  8:     [XGBClassifier]
    ----
    MEAN:     [0.74826389] + [0.01718662]
    FULL:     [0.74826389]

model  9:     [LGBMClassifier]
    ----
    MEAN:     [0.74479167] + [0.01533292]
    FULL:     [0.74479167]

model 10:     [KNeighborsClassifier]
    ----
    MEAN:     [0.70486111] + [0.02182258]
    FULL:     [0.70486111]

model 11:     [DecisionTreeClassifier]
    ----
    MEAN:     [0.68576389] + [0.03835195]
    FULL:     [0.68576389]

model 12:     [RandomForestClassifier]
    ----
    MEAN:     [0.75520833] + [0.01533292]
    FULL:     [0.75520833]
```

Ensemble Learning (stacking)

Dan didapatkan **akurasi stackingnya adalah 79.69%**.

```
## Print vecstack result
accuracy = accuracy_score(y_test, y_test_pred)*100
print("Stacking accuracy : {0:.2f}%".format(accuracy))
print("train and test time: {0:.2f}s".format(train_test_time))
```

```
Stacking accuracy : 79.69%
train and test time: 3.76s
```

Terima Kasih