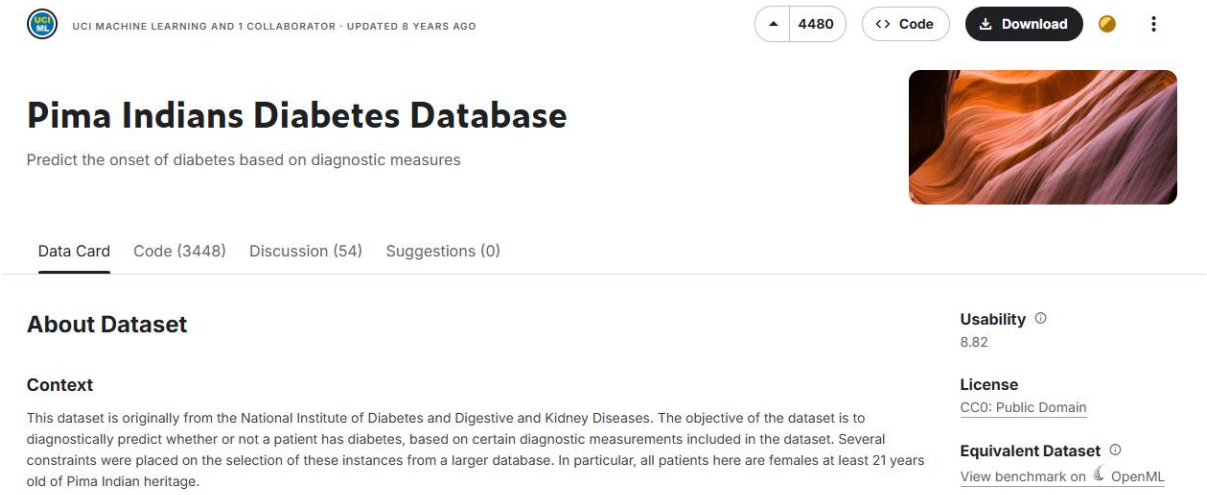
Nama Dosen : Teguh Iman Hermanto, M.Kom
Mata Kuliah : Machine Learning 2
Pembahasan : ANN Diabetes
Pokok Pemb : Membuat model klasifikasi penyakit diabetes menggunakan ANN




```
!pip install kaggle
```

```
!kaggle datasets download -d uciml/pima-indians-diabetes-database
```

```
!unzip pima-indians-diabetes-database.zip -d content
```

```
import pandas as pd
import numpy as np

import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from tensorflow.keras.utils import plot_model

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
```

PREPROCESSING DATA

```
data = pd.read_csv('content/diabetes.csv')
```

```
data.head()
```

```
# Pisahkan fitur (X) dan target (y)
X = data.drop("Outcome", axis=1)
y = data["Outcome"]
```

```
# Split dataset menjadi training dan testing set
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
# Normalisasi data
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

MODELING

```python
# Membangun model ANN
model = Sequential()
model.add(Dense(16, activation='relu', input_shape=(X_train.shape[1],)))  # Hidden layer
model.add(Dense(8, activation='relu'))  # Hidden layer kedua
model.add(Dense(1, activation='sigmoid'))  # Output layer untuk klasifikasi biner
```

```python
# Compile model
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
```

```python
# Plot arsitektur model
plot_model(model, show_shapes=True, show_layer_names=True)
```

```python
# Melatih model
history = model.fit(X_train, y_train, epochs=50, batch_size=16, validation_split=0.2, verbose=1)
```

TESTING MODEL

```python
# Prediksi pada data test
y_pred_prob = model.predict(X_test)
y_pred = (y_pred_prob > 0.5).astype(int)  # Konversi probabilitas ke kelas biner
```

```python
# Menghitung akurasi
accuracy = accuracy_score(y_test, y_pred)
print(f'Akurasi: {accuracy * 100:.2f}%')
```

```python
# Menampilkan confusion matrix dan classification report
print("\nConfusion Matrix:")
print(confusion_matrix(y_test, y_pred))
```

```python
print("\nClassification Report:")
print(classification_report(y_test, y_pred))
```

SIMULASI MODEL

```python
# Contoh input data baru (8 fitur)
new_data = np.array([[6, 148, 72, 35, 0, 33.6, 0.627, 50]])  # Contoh data pasien

# Normalisasi input data baru
new_data_scaled = scaler.transform(new_data)

# Prediksi
prediction_prob = model.predict(new_data_scaled)
prediction = (prediction_prob > 0.5).astype(int)

# Hasil prediksi
if prediction[0] == 1:
    print("Pasien diprediksi terkena diabetes.")
else:
    print("Pasien diprediksi tidak terkena diabetes.")
```

```python
model.save("diabetes_model.h5")
```

OPTIMASI MODEL

```
from tensorflow.keras.layers import Dropout
from tensorflow.keras.regularizers import l2
```

```
# Membangun model ANN Optimasi
model = Sequential()
model.add(Dense(64, activation='relu', input_shape=(X_train.shape[1],), kernel_regularizer=l2(0.01)))  # Hidden layer
Dropout(0.3)
model.add(Dense(32, activation='relu', kernel_regularizer=l2(0.01)))  # Hidden layer kedua
Dropout(0.3)
model.add(Dense(16, activation='relu', kernel_regularizer=l2(0.01)))
model.add(Dense(1, activation='sigmoid'))  # Output layer untuk klasifikasi biner
```

```
# Compile model
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
```

```
# Plot arsitektur model
plot_model(model, show_shapes=True, show_layer_names=True)
```

```
# Melatih model
from tensorflow.keras.callbacks import EarlyStopping

# Callback untuk early stopping
early_stopping = EarlyStopping(monitor='val_loss', patience=10, restore_best_weights=True)

# Melatih model
history = model.fit(
    X_train, y_train,
    epochs=100,  # Lebih banyak epoch
    batch_size=16,
    validation_split=0.2,
    callbacks=[early_stopping],
    verbose=1
)
```