2. Model.ipnyb

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import confusion_matrix, accuracy_score,
precision_recall_fscore_support
import matplotlib.pyplot as plt
import numpy as np
from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline

df_klasifikasi_train = pd.read_csv("ObesityDataSet_Kaggle.csv")
df_klasifikasi_test = pd.read_csv("Testing.csv")

X_train = df_klasifikasi_train[['Gender', 'Age', 'Height',
'Weight']]
y_train = df_klasifikasi_train['Label']
X_test_new = df_klasifikasi_test[['Gender', 'Age', 'Height',
'Weight']]
y_test_new = df_klasifikasi_test['Label']

numerical_features = ['Age', 'Height', 'Weight']
categorical_features = ['Gender']
preprocessor = ColumnTransformer(
    transformers=[
        ('num', StandardScaler(), numerical_features),
        ('cat', OneHotEncoder(), categorical_features)])

# KNN
knn_pipeline = Pipeline(steps=[('preprocessor', preprocessor),
                               ('classifier',
KNeighborsClassifier(n_neighbors=5))])
knn_model = knn_pipeline.fit(X_train, y_train)
y_pred_new = knn_pipeline.predict(X_test_new)
cm_new = confusion_matrix(y_test_new, y_pred_new)

plt.figure(figsize=(16, 6))
plt.imshow(cm_new, interpolation='nearest', cmap=plt.cm.Blues)
plt.title('Confusion Matrix')
plt.colorbar()
classes = y_test_new.unique()
tick_marks = range(len(classes))
plt.xticks(tick_marks, classes)
plt.yticks(tick_marks, classes)
for i in range(cm_new.shape[0]):
    for j in range(cm_new.shape[1]):
        plt.text(j, i, format(cm_new[i, j], 'd'),
                 horizontalalignment="center",
```

```
                color="white" if cm_new[i, j] > cm_new.max() / 2
else "black")
plt.ylabel('True label')
plt.xlabel('Predicted label')
plt.tight_layout()
plt.show()

accuracy_new = accuracy_score(y_test_new, y_pred_new)
print('Akurasi Testing KNN :', accuracy_new)
P_new = precision_recall_fscore_support(y_test_new, y_pred_new,
average='macro')
print("Precision : ", P_new[0])
print("Recall : ", P_new[1])
print("F Score : ", P_new[2])


# SVM
svm_pipeline = Pipeline(steps=[('preprocessor', preprocessor),
                               ('classifier', SVC(kernel='linear',
random_state=42))])
svm_model = svm_pipeline.fit(X_train, y_train)
y_pred_new = svm_pipeline.predict(X_test_new)
cm_new = confusion_matrix(y_test_new, y_pred_new)

plt.figure(figsize=(16, 6))
plt.imshow(cm_new, interpolation='nearest', cmap=plt.cm.Blues)
plt.title('Confusion Matrix')
plt.colorbar()
classes = y_test_new.unique()
tick_marks = range(len(classes))
plt.xticks(tick_marks, classes)
plt.yticks(tick_marks, classes)

for i in range(cm_new.shape[0]):
    for j in range(cm_new.shape[1]):
        plt.text(j, i, format(cm_new[i, j], 'd'),
                 horizontalalignment="center",
                 color="white" if cm_new[i, j] > cm_new.max() / 2
else "black")
plt.ylabel('True label')
plt.xlabel('Predicted label')
plt.tight_layout()
plt.show()

accuracy_new = accuracy_score(y_test_new, y_pred_new)
print('Akurasi Testing SVM :', accuracy_new)

P_new = precision_recall_fscore_support(y_test_new, y_pred_new,
average='macro')
print("Precision : ", P_new[0])
```

```
print("Recall : ", P_new[1])
print("F Score : ", P_new[2])

import pickle
# Latih model SVM pada dataset latih{
svm_model = svm_pipeline.fit(X_train, y_train)
# Simpan model ke file menggunakan pickle
with open('svm_model.pkl', 'wb') as model_file:
    pickle.dump(svm_model, model_file)
```