

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
KHOA CÔNG NGHỆ THÔNG TIN 1



BÁO CÁO KẾT QUẢ
ASSIGNMENT 2

Họ và tên : Phạm Thành Long
Mã sinh viên : B22DCCN505
Lớp chính quy : D22CNPM01
Môn học : Phát triển các hệ thống thông minh
Lớp học phần : Nhóm 02
Nhóm : 01
GV hướng dẫn : PGS. TS. Trần Đình Quế

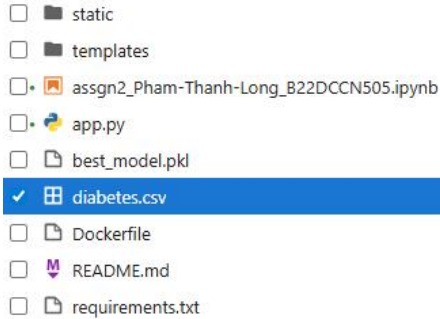
Hà Nội, tháng 8 năm 2025

MỤC LỤC

1. Load data. Cleaning data - Present steps...and corresponding code	3
2. Select 5 features (not 3) and Using 5 models in Basic Machine	4
3. Evaluate and compare with metrics: accuracy, mae, mse, rmse. Explain metrics	5
a) MAE (Mean Absolute Error) – Sai số tuyệt đối trung bình, Áp dụng cho hồi quy (regression) hoặc khi bạn quan tâm độ lệch dự đoán.	5
b) MSE (Mean Squared Error) – Sai số bình phương trung bình, Phạt nặng hơn khi dự đoán sai nhiều (do bình phương).	5
c) MSE (Mean Squared Error) – Sai số bình phương trung bình	5
d) RMSE (Root Mean Squared Error) – Căn bậc 2 của MSE	5
4. Visualize and show with 5 types of diagrams	6
5. Deploy on web via Flask and other technology	7
a) app.py	8
b) index.html	8
c) style.css	9
d) Link deploy	11

1. Load data. Cleaning data - Present steps...and corresponding code

- Bước này nhằm mục đích làm sạch dữ liệu chuẩn bị data.
- Ban đầu phải tải file “diabetes.csv”



- Import thư viện

```
# Pham Thanh Long - B22DCCN505

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import pickle

from sklearn.metrics import accuracy_score, mean_absolute_error, mean_squared_error, confusion_matrix
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn.model_selection import train_test_split
```

- Sau đó đọc và cleaning dữ liệu.

```
print("-----1-----")

# Load data
print("-----load data-----")
df = pd.read_csv('diabetes.csv')
df.info()

# Cleaning data
print("-----cleaning data-----")

# ---check for null values---
print("Nulls")
print("=====")
print(df.isnull().sum())

# ---check for 0s---
print("0s")
print("==")
print(df.eq(0).sum())

# replace the 0 values with NaN for selected columns
print("-----Replace value-----")

cols = ['Glucose', 'BloodPressure', 'SkinThickness',
        'Insulin', 'BMI', 'DiabetesPedigreeFunction', 'Age']

df[cols] = df[cols].replace(0, np.nan)

# fill NaN with the mean of each column
df.fillna(df.mean(), inplace=True)

# check again
print(df.eq(0).sum())
```

- Kết quả:

```

-----1.-----
-----load data-----
<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
-----cleaning data-----
Nulls
=====
Pregnancies            0
Glucose                0
BloodPressure          0
SkinThickness          0
Insulin                0
BMI                   0
DiabetesPedigreeFunction 0
Age                   0
Outcome                0
dtype: int64
0s
==
Pregnancies            111
Glucose                5
BloodPressure          35
SkinThickness          227
Insulin                374
BMI                   11
DiabetesPedigreeFunction 0
Age                   0
Outcome                500
dtype: int64
-----Replace value-----
Pregnancies            111
Glucose                0
BloodPressure          0

```

2. Select 5 features (not 3) and Using 5 models in Basic Machine

- Chia dữ liệu và trực quan hóa dữ liệu bằng ma trận tương quan và train mô hình bằng 5 thuật toán basic của machine learning

```

# select 5 features (not 3) and Using 5 models in Basic Machine
print("-----2.-----")

# so đo tương quan
corr = df.corr(method='pearson', numeric_only=True)
print(corr)

# select 5 features
features = ['Glucose', 'BMI', 'Age', 'BloodPressure', 'Insulin']
X = df[features]
y = df['Outcome']

# Chia dữ liệu train/test
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Train & predict với từng model ( 5 model cơ bản trong machine learning)
models = {
    "LogisticRegression": LogisticRegression(max_iter=1000),
    "DecisionTree": DecisionTreeClassifier(random_state=42),
    "RandomForest": RandomForestClassifier(n_estimators=100, random_state=42),
    "KNN": KNeighborsClassifier(n_neighbors=5),
    "SVM": SVC(kernel='linear')
}

results = {}

for name, model in models.items():
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    results[name] = y_pred

```

- Kết quả:

```

-----2-----
Pregnancies      Glucose      BloodPressure      SkinThickness  \
Pregnancies      1.000000      0.127911      0.208522      0.082989
Glucose          0.127911      1.000000      0.218367      0.192991
BloodPressure    0.208522      0.218367      1.000000      0.192816
SkinThickness    0.082989      0.192991      0.192816      1.000000
Insulin          0.056027      0.420157      0.072517      0.158139
BMI              0.021565      0.230941      0.281268      0.542398
DiabetesPedigreeFunction -0.033523      0.137060      -0.002763      0.100966
Age              0.544341      0.266534      0.324595      0.127872
Outcome          0.221898      0.492928      0.166074      0.215299

Pregnancies      Insulin      BMI      DiabetesPedigreeFunction  \
Pregnancies      0.056027      0.021565      -0.033523
Glucose          0.420157      0.230941      0.137060
BloodPressure    0.072517      0.281268      -0.002763
SkinThickness    0.158139      0.542398      0.100966
Insulin          1.000000      0.166586      0.098634
BMI              0.166586      1.000000      0.153400
DiabetesPedigreeFunction 0.098634      0.153400      1.000000
Age              0.136734      0.025519      0.033561
Outcome          0.214411      0.311924      0.173844

Pregnancies      Age      Outcome
Pregnancies      0.544341      0.221898
Glucose          0.266534      0.492928
BloodPressure    0.324595      0.166074
SkinThickness    0.127872      0.215299
Insulin          0.136734      0.214411
BMI              0.025519      0.311924
DiabetesPedigreeFunction 0.033561      0.173844
Age              1.000000      0.238356
Outcome          0.238356      1.000000

```

3. Evaluate and compare with metrics: accuracy, mae, mse, rmse. Explain metrics

```

# 3. Evaluate and compare with metrics: accuracy, mae, mse, rmse. Explain metrics
print("-----3-----")

metrics = {}
for name, y_pred in results.items():
    acc = accuracy_score(y_test, y_pred)
    mae = mean_absolute_error(y_test, y_pred)
    mse = mean_squared_error(y_test, y_pred)
    rmse = np.sqrt(mse)
    metrics[name] = {"Accuracy": acc, "MAE": mae, "MSE": mse, "RMSE": rmse}

metrics_df = pd.DataFrame(metrics).T
print(metrics_df)

```

- a) **MAE (Mean Absolute Error)** – Sai số tuyệt đối trung bình, Áp dụng cho **hồi quy (regression)** hoặc khi bạn quan tâm độ lệch dự đoán.
- b) **MSE (Mean Squared Error)** – Sai số bình phương trung bình, Phạt nặng hơn khi dự đoán sai nhiều (do bình phương).
- c) **MSE (Mean Squared Error)** – Sai số bình phương trung bình
- d) **RMSE (Root Mean Squared Error)** – Căn bậc 2 của MSE

4. Visualize and show with 5 types of diagrams

```
# (1) Heatmap correlation
plt.figure(figsize=(8,6))
sns.heatmap(df[features + ['Outcome']].corr(), annot=True, cmap="coolwarm")
plt.title("Correlation Heatmap (5 features + Outcome)")
plt.show()

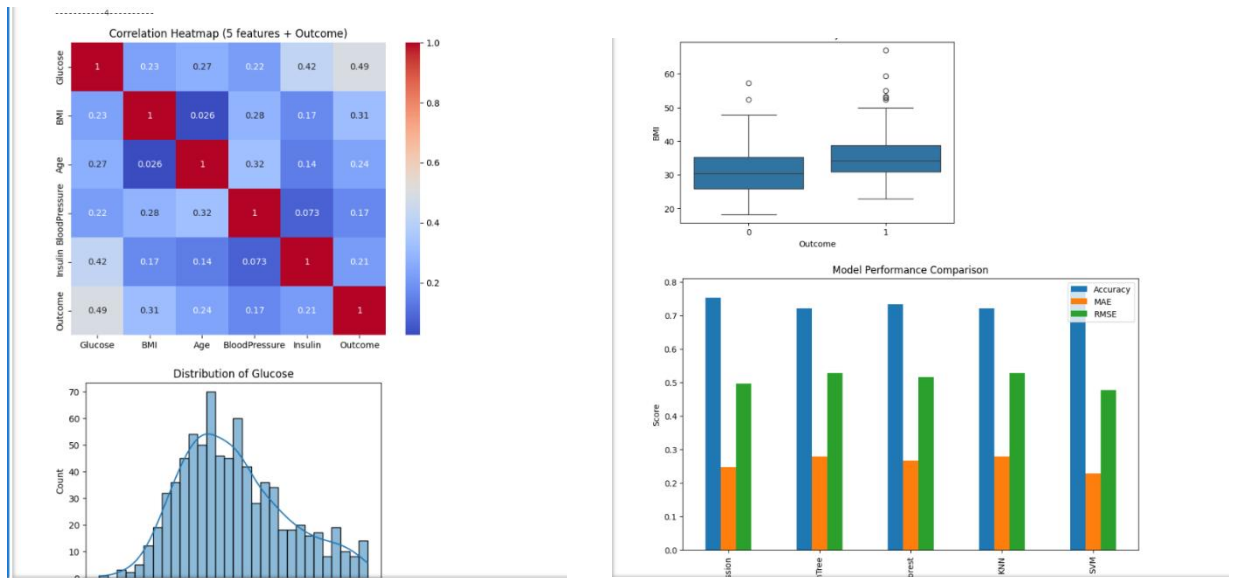
# (2) Histogram of Glucose
plt.figure(figsize=(6,4))
sns.histplot(df['Glucose'], bins=30, kde=True)
plt.title("Distribution of Glucose")
plt.show()

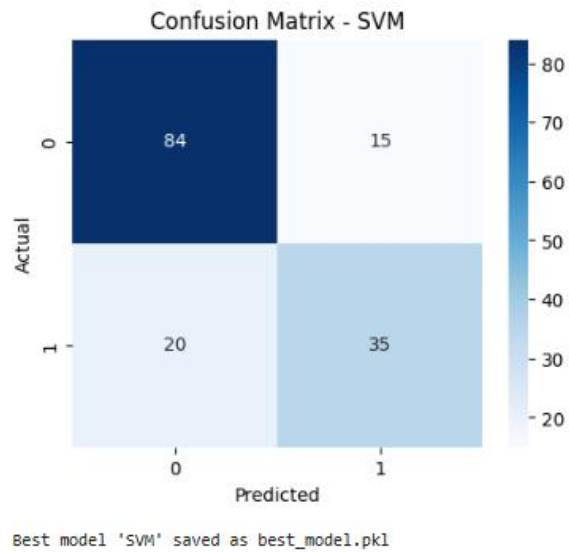
# (3) Boxplot BMI vs Outcome
plt.figure(figsize=(6,4))
sns.boxplot(x='Outcome', y='BMI', data=df)
plt.title("BMI Distribution by Outcome")
plt.show()

# (4) Bar chart of metrics
metrics_df[['Accuracy','MAE','RMSE']].plot(kind='bar', figsize=(10,6))
plt.title("Model Performance Comparison")
plt.ylabel("Score")
plt.show()

# (5) Confusion Matrix heatmap for RandomForest (v\ du)
best_model_name = metrics_df['Accuracy'].idxmax()
cm = confusion_matrix(y_test, results[best_model_name])
plt.figure(figsize=(5,4))
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues", xticklabels=[0,1], yticklabels=[0,1])
plt.title(f"Confusion Matrix - {best_model_name}")
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.show()
```

- Kết quả:





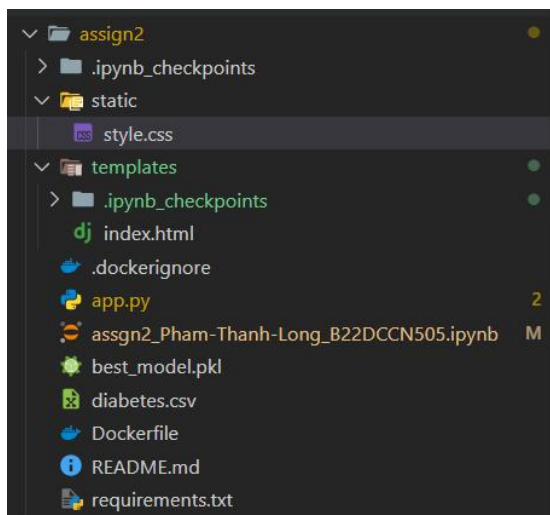
- Select the best one for deploying on web
- Sau khi tìm hiểu và xem xét thì mô hình cho độ chính xác cao nhất là RandomTree chọn mô hình đó và lưu vào best_model.pkl

```
# =====
# 4. Save Best Model
# =====
best_model = models[best_model_name]
with open("best_model.pkl", "wb") as f:
    pickle.dump(best_model, f)

print(f"Best model '{best_model_name}' saved as best_model.pkl")
```

5. Deploy on web via Flask and other technology

- Cấu trúc thư mục:



a) app.py

```
from flask import Flask, render_template, request
import pickle
import numpy as np

# Load model
with open("best_model.pkl", "rb") as f:
    model = pickle.load(f)

app = Flask(__name__)

@app.route("/", methods=["GET", "POST"])
def index():
    result = None
    if request.method == "POST":
        # Lấy dữ liệu từ form
        glucose = float(request.form["glucose"])
        bmi = float(request.form["bmi"])
        age = float(request.form["age"])
        bp = float(request.form["bp"])
        insulin = float(request.form["insulin"])

        # Chuyển thành numpy array
        features = np.array([[glucose, bmi, age, bp, insulin]])

        # Dự đoán
        prediction = model.predict(features)[0]
        result = "Có khả năng bị tiểu đường" if prediction == 1 else "Không bị tiểu đường"

    return render_template("index.html", result=result)

if __name__ == "__main__":
    app.run(debug=True)
```

b) index.html

```
<!DOCTYPE html>
<html>
<head>
    <title>Dự đoán Tiểu đường</title>
    <link rel="stylesheet" type="text/css" href="{{ url_for('static', filename='style.css') }}">
</head>
<body>
    <h2>Dự đoán Tiểu đường</h2>
    <form method="POST">
        <label>Glucose:</label><input type="number" step="any" name="glucose" required><br>
        <label>BMI:</label><input type="number" step="any" name="bmi" required><br>
        <label>Age:</label><input type="number" step="any" name="age" required><br>
        <label>Blood Pressure:</label><input type="number" step="any" name="bp" required><br>
        <label>Insulin:</label><input type="number" step="any" name="insulin" required><br>
        <button type="submit">Dự đoán</button>
    </form>

    {% if result %}
        <h3>Kết quả: {{ result }}</h3>
    {% endif %}
</body>
</html>
```


c) style.css

```
1  body {
2      font-family: Arial, sans-serif;
3      margin: 50px;
4      text-align: center;
5      background-color: #f9f9f9;
6  }
7
8  form {
9      display: inline-block;
10     padding: 20px;
11     background: white;
12     border-radius: 8px;
13     box-shadow: 0px 0px 10px #aaa;
14 }
15
16 input, button {
17     margin: 10px;
18     padding: 8px;
19     border-radius: 5px;
20     border: 1px solid #ddd;
21 }
22 button {
23     background: #28a745;
24     color: white;
25     cursor: pointer;
26 }
27 button:hover {
28     background: #218838;
29 }
```

- Kết quả:
- Giao diện

Dự đoán Tiểu đường

Glucose:

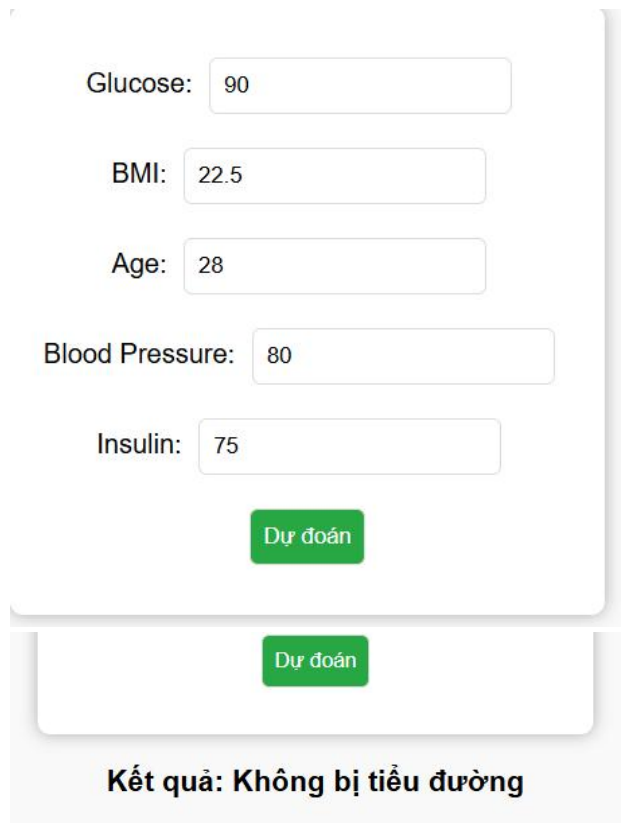
BMI:

Age:

Blood Pressure:

Insulin:

- Với bộ test không bị tiểu đường:



Glucose:

BMI:

Age:

Blood Pressure:

Insulin:

Kết quả: Không bị tiểu đường

- Với các test khác:



Kết quả: Có khả năng bị tiểu đường

Dự đoán Tiểu đường

Glucose:

BMI:

Age:

Blood Pressure:

Insulin:

Kết quả: Không bị tiểu đường

d) Link deploy

[Dự đoán Tiểu đường](#)