Name: Faris Chaudhry
Batch Code: LISUM25
Submission Date: 08/30/23

Toy Data Set

Diabetes data set (https://scikit-learn.org/stable/datasets/toy_dataset.html) (7.1.2)

442 samples; 10 features (described below) along with target value.

Features are scaled by mean and std deviation; no null entries or wrong data.

Number of Instances:	442
Number of Attributes:	First 10 columns are numeric predictive values
Target:	Column 11 is a quantitative measure of disease progression one year after baseline
Attribute Information:	• age age in years • sex • bmi body mass index • bp average blood pressure • s1 tc, total serum cholesterol • s2 ldl, low-density lipoproteins • s3 hdl, high-density lipoproteins • s4 tch, total cholesterol / HDL • s5 ltg, possibly log of serum triglycerides level • s6 glu, blood sugar level

The MEANS Procedure

Variable	N	Mean	Std Dev
age	442	48.5180995	13.1090278
sex	442	1.4683258	0.4995612
bmi	442	26.3757919	4.4181216
bp	442	94.6470136	13.8312834
s1	442	189.1402715	34.6080517
s2	442	115.4391403	30.4130810
s3	442	49.7884615	12.9342022
s4	442	4.0702489	1.2904499
s5	442	4.6414109	0.5223906
s6	442	91.2601810	11.4963347
y	442	152.1334842	77.0930045

Name: Faris Chaudhry
Batch Code: LISUM25
Submission Date: 08/30/23

Model Training

Standard training parameters used.

70% used as training data, 30% as testing data.

Model saved using pickle.

Saved as ‘model.pkl’

```
import pandas as pd
import pickle
from sklearn.datasets import load_diabetes
from sklearn.model_selection import train_test_split
from xgboost import XGBClassifier
from sklearn.preprocessing import LabelEncoder



X, y = load_diabetes(return_X_y=True, as_frame=True)
X.head()

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=0)
model = XGBClassifier(random_state=0)
le = LabelEncoder()
y_train = le.fit_transform(y_train)
model.fit(X_train, y_train)

XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
              colsample_bynode=1, colsample_bytree=1, gamma=0, gpu_id=-1,
              importance_type='gain', interaction_constraints='',
              learning_rate=0.300000012, max_delta_step=0, max_depth=6,
              min_child_weight=1, monotone_constraints='()',
              n_estimators=100, n_jobs=0, num_parallel_tree=1,
              objective='multi:softprob', random_state=0, reg_alpha=0,
              reg_lambda=1, scale_pos_weight=None, subsample=1,
              tree_method='exact', validate_parameters=1, verbosity=None)

print(X_train.head())

pickle.dump(model, open('.\model.pkl', 'wb'))
```

Name: Faris Chaudhry
Batch Code: LISUM25
Submission Date: 08/30/23

Flask Deployment

app.py (scale factors are from the MEANS approach above – hard coded here since they are constant)

```python
from flask import Flask, request, render_template
import numpy as np
import pandas as pd
import math
import pickle

scale_factor = math.sqrt(442)
app = Flask(__name__)
model = pickle.load(open('.\model.pkl', 'rb'))

# home endpoint
@app.route('/')
def home():
    return render_template('index.html')

# prediction endpoint
@app.route('/predict', methods=['POST'])
def predict():
    int_features = [float(x) for x in request.form.values()]
    final_features = [np.array(int_features)]

    # each feature has to be scaled by some specific variable found here:
    # https://www4.stat.ncsu.edu/~boos/var.select/diabetes.read.tab.out.txt
    df = pd.DataFrame({ 'age': (final_features[0][0] -  48.5180995) / ( 13.1090278 * scale_factor),
                        'sex': (final_features[0][1] -  1.4683258) / (0.4995612 * scale_factor),
                        'bmi': (final_features[0][2] - 26.3757919) / (4.4181216 * scale_factor),
                        'bp': (final_features[0][3] - 94.6470136) / (13.8312834 * scale_factor),
                        's1': (final_features[0][4] - 189.1402715) / (34.6080517 * scale_factor),
                        's2': (final_features[0][5] - 115.4391403) / (30.4130810 * scale_factor),
                        's3': (final_features[0][6] - 49.7884615) / (12.9342022 * scale_factor),
                        's4': (final_features[0][7] - 4.0702489) / (1.2904499 * scale_factor),
                        's5': (final_features[0][8] - 4.6414109) / (0.5223906 * scale_factor),
                        's6': (final_features[0][9] - 152.1334842) / (77.093004 * scale_factor)},
                       index=[0])

    prediction = model.predict(df)
    return render_template('index.html', prediction_text='Regression value is {}'.format(output))

if __name__ == '__main__':
    app.run(port=5000, debug=True)
```

Index.html and style.css modified from https://www.w3docs.com/learn-html/html-form-templates.html (free to copy and use)

CSS: https://github.com/farischaudhry/heroku-demo/blob/master/static/css/style.css

HTML: https://github.com/farischaudhry/heroku-demo/blob/master/templates/index.html

Name: Faris Chaudhry
Batch Code: LISUM25
Submission Date: 08/30/23

Final Product

2 decimal places allowed in entries.

Units specified for blood test results.

Sex has dropdown list for male and female.

All entries are set to required.

Regression value is showed in bottom-right corner (higher value means disease likely to be further progressed one year from now).