Diabetes Predictor using Random Forest

Consider below dataset

Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
6	148	72	35	0	33.6	0.627	50	1
1	85	66	29	0	26.6	0.351	31	0
8	183	64	0	0	23.3	0.672	32	1
1	89	66	23	94	28.1	0.167	21	0
0	137	40	35	168	43.1	2.288	33	1
5	116	74	0	0	25.6	0.201	30	0
3	78	50	32	88	31	0.248	26	1
10	115	0	0	0	35.3	0.134	29	0
2	197	70	45	543	30.5	0.158	53	1
8	125	96	0	0	0	0.232	54	1

Diabetes predictor application using Random Forest algorithm

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from warnings import simplefilter

simplefilter(action='ignore', category=FutureWarning)

print("---- Marvellous Infosystems by Piyush Khairnar-----")

print("---- Diabetes predictor using Random Forest -----")

diabetes = pd.read_csv('diabetes.csv')

print("Columns of Dataset")
print(diabetes.columns)

print("First 5 records of dataset")
print(diabetes.head())

print("Dimension of diabetes data: {}".format(diabetes.shape))
```

```
X_train, X_test, y_train, y_test = train_test_split(diabetes.loc[:, diabetes.columns
!= 'Outcome'], diabetes['Outcome'], stratify=diabetes['Outcome'],
random_state=66)

rf = RandomForestClassifier(n_estimators=100, random_state=0)
rf.fit(X_train, y_train)
print("Accuracy on training set: {:.3f}".format(rf.score(X_train, y_train)))
print("Accuracy on test set: {:.3f}".format(rf.score(X_test, y_test)))

rf1 = RandomForestClassifier(max_depth=3, n_estimators=100,
random_state=0)
rf1.fit(X_train, y_train)
print("Accuracy on training set: {:.3f}".format(rf1.score(X_train, y_train)))
print("Accuracy on test set: {:.3f}".format(rf1.score(X_test, y_test)))

def plot_feature_importances_diabetes(model):
    plt.figure(figsize=(8,6))
    n_features = 8
    plt.barh(range(n_features), model.feature_importances_, align='center')
    diabetes_features = [x for i,x in enumerate(diabetes.columns) if i!=8]
    plt.yticks(np.arange(n_features), diabetes_features)
    plt.xlabel("Feature importance")
    plt.ylabel("Feature")
    plt.ylim(-1, n_features)
    plt.show()

plot_feature_importances_diabetes(rf)
```

Output of above application

```
---- Marvellous Infosystems by Piyush Khairnar——

---- Diabetes predictor using Random Forest -----

Columns of Dataset
Index(['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
       'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome'],
      dtype='object')

First 5 records of dataset
   Pregnancies  Glucose  BloodPressure  ...  DiabetesPedigreeFunction  Age
Outcome
0            6      148             72  ...                     0.627   50
1
1            1       85             66  ...                     0.351   31
0
2            8      183             64  ...                     0.672   32
1
3            1       89             66  ...                     0.167   21
0
4            0      137             40  ...                     2.288   33
1
```



```
[5 rows x 9 columns]

Dimension of diabetes data: (768, 9)

Accuracy on training set: 1.000

Accuracy on test set: 0.786

Accuracy on training set: 0.800

Accuracy on test set: 0.755
```