Diabetes Prediction App

The Diabetes Prediction App is a web application built using Streamlit and Python. This app allows users to input various health parameters to predict whether they are likely to have diabetes. It uses a machine learning model for predictions and provides interactive visualizations to help users understand their input parameters in the context of a broader dataset.

Overview

The app is designed to:

1. **Collect User Input:** Users can input their health parameters via sliders in a sidebar.
2. **Display Input Parameters:** Shows the values entered by the user.
3. **Make Predictions:** Uses a pre-trained machine learning model to predict the likelihood of diabetes based on the user's inputs.
4. **Visualize Data:** Provides visual comparisons of the user's input parameters against a dataset to give context to the predictions.

Components

1. Importing Libraries

```
import pandas as pd
import pickle
import time
import plotly.express as px
import streamlit as st
from PIL import Image
```

- **pandas**: Used for data manipulation and analysis.
- **pickle**: Allows loading the pre-trained machine learning model.
- **time**: Creates a progress indicator while the app processes the data.
- **plotly.express**: Creates interactive visualizations to display data.
- **streamlit**: Builds the web app interface.
- **PIL (Pillow)**: Handles and displays images.

2. Sidebar for User Input

```
st.sidebar.header("User Input Parameters : ")

def user_input_features():
    Pregnancy = st.sidebar.slider('Pregnancy Period', 0, 12, 1)
    Glucose = st.sidebar.slider('Glucose',40, 220, 1)
    BloodPressure = st.sidebar.slider('BloodPressure',30, 200, 1)
    SkinThickness = st.sidebar.slider('SkinThickness',0, 120, 1)
    Insulin = st.sidebar.slider('Insulin',0, 120, 1)
    BMI = st.sidebar.slider('BMI',17, 50, 1)
    DiabetesPedigreeFunction = st.sidebar.slider('DiabetesPedigreeFunction',0.0, 3.0, 0.1)
    Age = st.sidebar.slider('Age',0.0, 100.0, 0.1)
    data={'Pregnancies': Pregnancy,
            'Glucose':Glucose,
            'BloodPressure': BloodPressure,
            'SkinThickness': SkinThickness,
            'Insulin': Insulin,
            'BMI': BMI,
            'DiabetesPedigreeFunction': DiabetesPedigreeFunction,
            'Age': Age}
    data = pd.DataFrame(data, index=[0])
    return data
```

- The sidebar provides sliders for users to input values for various health parameters, such as pregnancy period, glucose level, blood pressure, etc.

3. Displaying the App Header and Image

```
st.write("## Diabetes Prediction App")

import os
image = Image.open('Diabetes-Prediction-App/IMG.png')
image = image.resize((500,300))
st.image(image, caption='Diabetes')
```

- Displays the app title and an image related to diabetes to provide a visual context for the application.

4. Progress Bar and User Inputs

```
data = user_input_features()
st.write("---")
st.header("Your Parameters : ")
latest_iteration = st.empty()
bar = st.progress(0)
for i in range(100):
    bar.progress(i + 1)
    time.sleep(0.01)
st.write(data)
```

- Shows a progress bar to indicate that the app is processing the user's inputs and then displays the values entered by the user.

5. Data Loading and Labeling

```
df=pd.read_csv("/app/data-science/Diabetes-Prediction-App/diabetes.csv")

def add_label(row):
    if row['Outcome'] == 1:
        return "Diabetic"
    else:
        return "Non-Diabetic"
df['label'] = df.apply(add_label, axis=1)
```

- Loads a dataset containing historical diabetes data and labels each row as 'Diabetic' or 'Non-Diabetic' based on the outcome.

6. Prediction

```
st.write("---")
st.header("Prediction : ")
loaded_model = pickle.load(open("/app/data-science/Diabetes-Prediction-App/Diabetes.sav", 'rb'))
prediction = loaded_model.predict(data)

if(prediction[0] == 0):
    prediction = "Not Diabetes"
else:
    prediction = "Diabetes"
st.write(prediction)
```

- Loads the pre-trained machine learning model and makes a prediction based on the user's input. Displays whether the prediction is 'Diabetes' or 'Not Diabetes'.

7. Visualization of Input Parameters

```
st.write("---")
st.header("General Discussion ")
st.write("Your Input Parameters positions are shown below:")

for i in df.columns[0:-2]:
    fig = px.histogram(df, x=i, color='label')
    fig.add_trace(go.Scatter(
        x=[data[i][0]],
        y=[limit[i]],
        text=["Input Parameter value Position"],
        mode="text",
    ))
    fig.add_shape(type="line",
        x0=data[i][0], y0=0, x1=data[i][0], y1=limit[i],
        line=dict(color="White", width=3)
    )
    st.plotly_chart(fig, use_container_width=True)
```

- For each parameter, the app generates a histogram showing the distribution of values in the dataset. It also adds a marker and line to show where the user's input falls in relation to the dataset.

File Paths

- **Model File:** `/app/data-science/Diabetes-Prediction-App/Diabetes.sav` - The trained machine learning model.
- **Dataset File:** `/app/data-science/Diabetes-Prediction-App/diabetes.csv` - The dataset used for training the model and visualizing data.
- **Image File:** `Diabetes-Prediction-App/IMG.png` - An image used in the app interface.

Conclusion

The Diabetes Prediction App provides a user-friendly interface for predicting diabetes based on individual health parameters. It offers interactive visualizations that help users understand their input parameters in the context of a larger dataset, enhancing the overall user experience.

```
This description provides a comprehensive overview of the app's functionality, making it easier for others to understand ho

<style type="text/css">@media print {
    *, :after, :before {background: 0 0 !important;color: #000 !important;box-shadow: none !important;text-shadow: none !im
    a, a:visited {text-decoration: underline}
    a[href]:after {content: " (" attr(href) ")"}
    abbr[title]:after {content: " (" attr(title) ")"}
    a[href^="#"]:after, a[href^="javascript:"]:after {content: ""}
    blockquote, pre {border: 1px solid #999;page-break-inside: avoid}
    thead {display: table-header-group}
    img, tr {page-break-inside: avoid}
    img {max-width: 100% !important}
    h2, h3, p {orphans: 3;widows: 3}
    h2, h3 {page-break-after: avoid}
}
html {font-size: 12px}
@media screen and (min-width: 32rem) and (max-width: 48rem) {
    html {font-size: 15px}
}
@media screen and (min-width: 48rem) {
    html {font-size: 16px}
}
body {line-height: 1.85}
.air-p, p {font-size: 1rem;margin-bottom: 1.3rem}
```

```
.air-h1, .air-h2, .air-h3, .air-h4, h1, h2, h3, h4 {margin: 1.414rem 0 .5rem;font-weight: inherit;line-height: 1.42}
.air-h1, h1 {margin-top: 0;font-size: 3.998rem}
.air-h2, h2 {font-size: 2.827rem}
.air-h3, h3 {font-size: 1.999rem}
.air-h4, h4 {font-size: 1.414rem}
.air-h5, h5 {font-size: 1.121rem}
.air-h6, h6 {font-size: .88rem}
.air-small, small {font-size: .707em}
canvas, iframe, img, select, svg, textarea, video {max-width: 100%}
body {color: #444;font-family: 'Open Sans', Helvetica, sans-serif;font-weight: 300;margin: 0;text-align: center}
img {border-radius: 50%;height: 200px;margin: 0 auto;width: 200px}
a, a:visited {color: #3498db}
a:active, a:focus, a:hover {color: #2980b9}
pre {background-color: #fafafa;padding: 1rem;text-align: left}
blockquote {margin: 0;border-left: 5px solid #7a7a7a;font-style: italic;padding: 1.33em;text-align: left}
li, ol, ul {text-align: left}
p {color: #777}</style>
```