Visualizations

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.metrics import confusion_matrix, roc_curve, auc, precision_recall_curve
from sklearn.ensemble import RandomForestClassifier
from wordcloud import WordCloud
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.model_selection import train_test_split
from plotly import graph_objects as go
data = pd.read_csv('healthcare_dataset.csv')
data['binary_label'] = data['Test Results'].apply(lambda x: 1 if x == 'Normal' else 0)
data['text'] = data['Medical Condition'] + ' ' + data['Medication'] + ' ' + data['Test Results']


X = data['text']
y = data['binary_label']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
vectorizer = CountVectorizer()
X_train_vect = vectorizer.fit_transform(X_train)
X_test_vect = vectorizer.transform(X_test)
model = RandomForestClassifier()
model.fit(X_train_vect, y_train)
y_pred = model.predict(X_test_vect)
y_proba = model.predict_proba(X_test_vect)[:, 1]
cm = confusion_matrix(y_test, y_pred)
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues")
plt.title('Confusion Matrix')
plt.xlabel('Predicted')
plt.ylabel('True')
plt.show()
fpr, tpr, _ = roc_curve(y_test, y_proba)
roc_auc = auc(fpr, tpr)
plt.plot(fpr, tpr, color='darkorange', lw=2, label='ROC curve (area = %0.2f)' % roc_auc)
plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')

plt.xlabel('False Positive Rate')

plt.ylabel('True Positive Rate')

plt.title('Receiver Operating Characteristic')

plt.legend(loc="lower right")

plt.show()
precision, recall, _ = precision_recall_curve(y_test, y_proba)

plt.plot(recall, precision, lw=2, color='b', label='Precision-Recall curve')

plt.xlabel('Recall')

plt.ylabel('Precision')

plt.title('Precision-Recall Curve')

plt.legend(loc="lower left")

plt.show()
all_text = ' '.join(X)

wordcloud = WordCloud(width=800, height=400, background_color='white').generate(all_text)

plt.imshow(wordcloud, interpolation='bilinear')

plt.axis('off')

plt.title('Word Cloud')

plt.show()
sns.scatterplot(data=data, x='Age', y='Billing Amount', hue='Medical Condition')

plt.title('Scatter Plot of Age vs Billing Amount')

plt.show()
data['Age'].plot(kind='hist', bins=10, color='skyblue')

plt.title('Age Distribution Histogram')

plt.xlabel('Age')

plt.ylabel('Frequency')

plt.show()
data['Medical Condition'].value_counts().plot(kind='bar', color='lightgreen')

plt.title('Medical Condition Count Bar Chart')

plt.xlabel('Medical Condition')

plt.ylabel('Count')

plt.show()
sankey_data = go.Sankey(
    node=dict(
        pad=15,
        thickness=20,
        line=dict(color="black", width=0.5),
        label=["Cancer", "Obesity", "Diabetes", "Asthma", "Hypertension", "Arthritis", "Normal", "Inconclusive", "Abnormal"]
    ),
    link=dict(
        source=[0, 0, 1, 1, 2, 2, 3, 3, 4, 5, 5],  # Medical Condition indices
        target=[6, 7, 6, 7, 8, 6, 8, 6, 7, 6, 8],  # Test Results indices
        value=[4, 1, 1, 3, 1, 1, 2, 3, 1, 1, 1]  # Sample value counts
    )
)
fig = go.Figure(sankey_data)
fig.update_layout(title_text="Sankey Diagram of Medical Condition to Test Results", font_size=10)
fig.show()
```