TPFP

October 27, 2024

[1]:

```
import numpy as np
import ipywidgets as widgets
from ipywidgets import interact

import matplotlib.pyplot as plt

# Function to calculate sensitivity, specificity, PPV, and NPV
def calculate_metrics(tp, fp, fn, tn):
    sensitivity = tp / (tp + fn) if (tp + fn) > 0 else 0
    specificity = tn / (tn + fp) if (tn + fp) > 0 else 0
    ppv = tp / (tp + fp) if (tp + fp) > 0 else 0
    npv = tn / (tn + fn) if (tn + fn) > 0 else 0
    return sensitivity, specificity, ppv, npv

# Function to update the plot
def update_plot(population_size, true_positive_rate, false_positive_rate):
    # Simulate data
    tp = int(population_size * true_positive_rate)
    fp = int(population_size * false_positive_rate)
    fn = population_size - tp
    tn = population_size - fp

    # Calculate metrics
    sensitivity, specificity, ppv, npv = calculate_metrics(tp, fp, fn, tn)

    # Plot metrics
    metrics = ['Sensitivity', 'Specificity', 'PPV', 'NPV']
    values = [sensitivity, specificity, ppv, npv]

    plt.figure(figsize=(10, 6))
    plt.bar(metrics, values, color=['blue', 'green', 'orange', 'red'])
    plt.ylim(0, 1)
    plt.ylabel('Value')
    plt.title('Metrics Visualization')
    plt.show()

# Create sliders
```

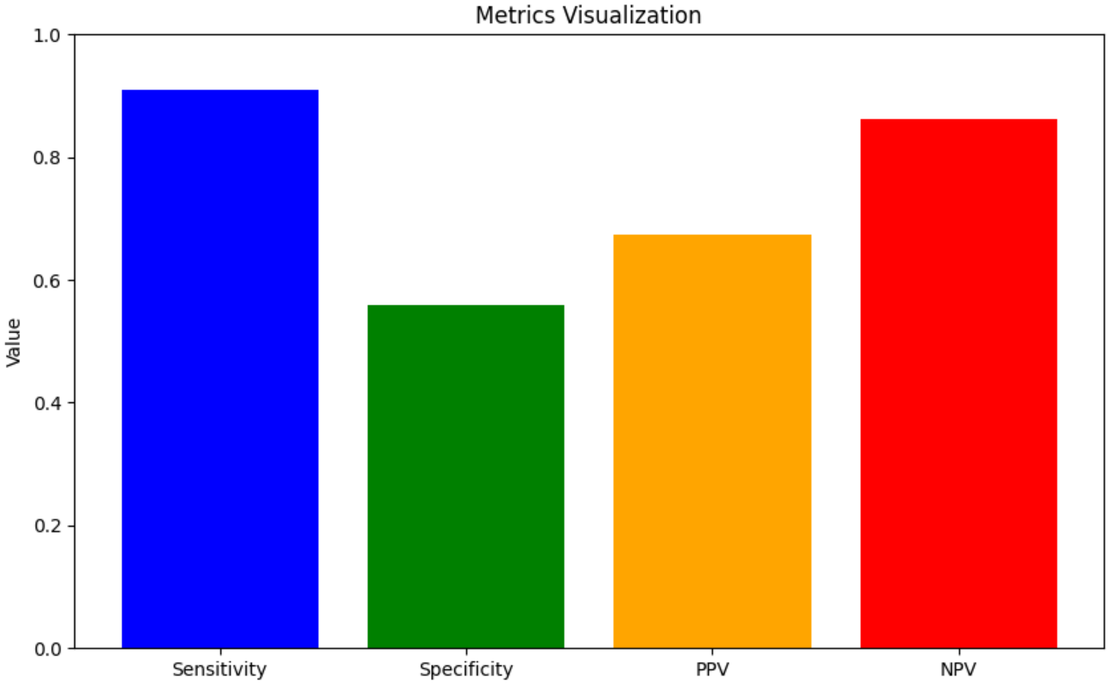
```
population_size_slider = widgets.IntSlider(min=100, max=10000, step=100,␣
 ↪value=1000, description='Population Size')
true_positive_rate_slider = widgets.FloatSlider(min=0, max=1, step=0.01,␣
 ↪value=0.1, description='True Positive Rate')
false_positive_rate_slider = widgets.FloatSlider(min=0, max=1, step=0.01,␣
 ↪value=0.1, description='False Positive Rate')

# Use interact to update the plot with sliders
interact(update_plot, population_size=population_size_slider,␣
 ↪true_positive_rate=true_positive_rate_slider,␣
 ↪false_positive_rate=false_positive_rate_slider)
```

```
interactive(children=(IntSlider(value=1000, description='Population Size',␣
 ↪max=10000, min=100, step=100), Floa…
```

[1]:
```
<function __main__.update_plot(population_size, true_positive_rate,
false_positive_rate)>
```

[3]:
```
# Directly call the update_plot function with the current slider values
update_plot(population_size_slider.value, true_positive_rate_slider.value,␣
 ↪false_positive_rate_slider.value)
```



[ ]: