SVM Homework Overview2

Continuing from where we left off, here's the rest of the **HW3-3** summary, including final adjustments and setting the **Spiral** dataset as the default option.

---

4. **1D Visualization Specific Adjustments**:
   - Created a specialized **1D plot** for the Simple 1D dataset, plotting decision boundary as a line and showing `Class 0` and `Class 1` points separately.

   Code Modification:

```
python


  elif plot_type == "1D":
      fig, ax = plt.subplots(figsize=(10, 6))
      ax.scatter(X[y == 0], y[y == 0], color='blue', label='Class 0', edgecolor='k')
      ax.scatter(X[y == 1], y[y == 1], color='orange', label='Class 1', edgecolor='k')
      x_range = np.linspace(X.min() - 1, X.max() + 1, 500).reshape(-1, 1)
      decision_boundary = svm_model.decision_function(x_range)
      ax.plot(x_range, decision_boundary, color='red', linestyle='--', label='Decision
  Boundary')
      ax.axhline(0, color='black', linestyle=':', linewidth=1)
      ax.set_xlabel("Feature (x)")
      ax.set_ylabel("Decision Function Value")
      ax.legend()
      st.pyplot(fig)
```

5. **Consistent Class Labeling Across All Visualizations**:
   - Standardized the legend entries for **Class 0** and **Class 1** in both 2D and 3D visualizations, ensuring clarity and consistency.

   Code Modification:

```
python


  ax.scatter(X[y == 0][:, 0], X[y == 0][:, 1], color='blue', label='Class 0', edgecolor
  ='k')
  ax.scatter(X[y == 1][:, 0], X[y == 1][:, 1], color='orange', label='Class 1', edgecol
  or='k')
  ax.legend()
```

6. **Final Adjustments**:
   - Set the **Spiral** dataset as the default selection in the dataset type selector by placing `"Spiral"` as the first option in the list.

   Code Modification:

```
python


  dataset_type = st.sidebar.selectbox("Choose Dataset Type", ["Spiral", "Circular (2
  D)", "Simple 1D", "Moons", "Elliptical Blobs"])
```

This summary reflects the full progression of development across HW3-1, HW3-2, and HW3-3. Each step includes a summary of the prompt, code implementation, and relevant code modifications, showing how the Streamlit SVM tool evolved into a versatile, interactive application. Let me know if you need further details!