**Here's a sample README file for this project:**

**Dataset**: Online News Popularity Data

**Classifier:** Support Vector Machine

**Improvement**: Additional kernel function - Quadratic kernel

**1) Data preprocessing**

After data preprocessing and feature removal, the number of selected features was 39 out of 61.

Training data = 70% and Testing data = 30%

**2) SVM Dual Implementation**

Implemented Support Vector Machine (SVM) using dual optimization technique using Python. The implementation includes the following features:

- Linear, quadratic, and Gaussian kernel functions
- Automatic tuning of kernel hyperparameters

**3) Usage**

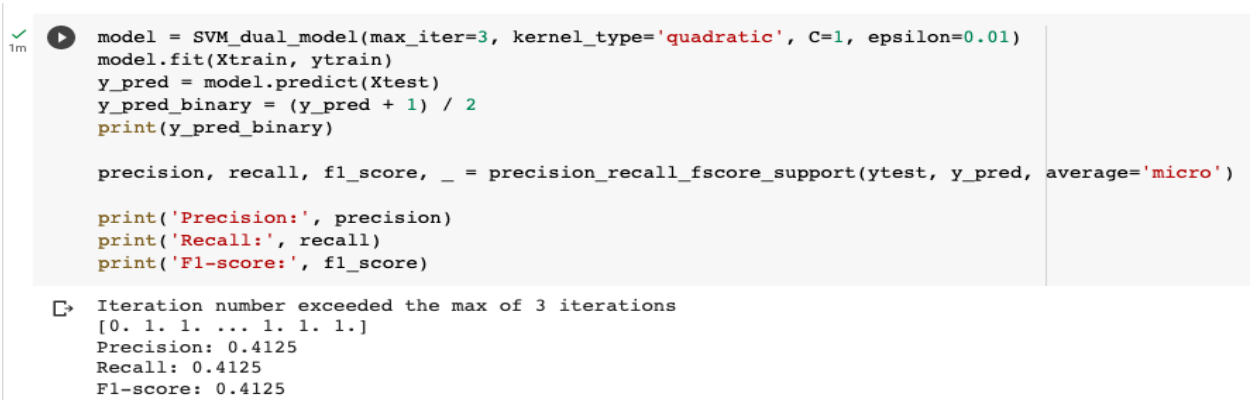
- Same base code for traditional and improvised SVM models. An additional kernel function is added to the base model as part of improvisation.
- I already added the values for C and step size in a list, so you could test just by running the code and it will automatically tune the hyperparameters and print the results. Also, it is separately implemented for each kernel function in the code. So, no need to change the kernel function as well.
- Here's an example screenshot for testing

```
C_values = [1, 0.1, 0.001, 10, 100]
step_sizes = [0.001, 0.01, 0.1, 1, 10]

for c in C_values:
    for step_size in step_sizes:
        model = SVM_dual_model(max_iter=3, kernel_type='linear', C=c, epsilon=step_size)
        model.fit(Xtrain, ytrain)
        Training_Accuracy = model.evaluate(Xtrain,ytrain)
        Test_Accuracy = model.evaluate(Xtest,ytest)
        print("C={}, Step Size={}, Training Accuracy={}, Test Accuracy={}".format(c, step_size, Training_Accuracy, Test_Accuracy))
```

```
Iteration number exceeded the max of 3 iterations
C=1, Step Size=0.001, Training Accuracy=0.6694847863390951, Test Accuracy=0.6689453125
Iteration number exceeded the max of 3 iterations
C=1, Step Size=0.01, Training Accuracy=0.6508182312811284, Test Accuracy=0.6548828125
Iteration number exceeded the max of 3 iterations
C=1, Step Size=0.1, Training Accuracy=0.668856987402168, Test Accuracy=0.668359375
Iteration number exceeded the max of 3 iterations
C=1, Step Size=1, Training Accuracy=0.7237266144895995, Test Accuracy=0.71591796875
```

- Precision, Recall, and F1 score are also evaluated for each kernel function, I gave the same values for hyperparameters for all kernels to maintain consistency, you could just run the code, with no need to change the kernel names. I implemented it separately. Here's an example screenshot for testing.



```
model = SVM_dual_model(max_iter=3, kernel_type='quadratic', C=1, epsilon=0.01)
model.fit(Xtrain, ytrain)
y_pred = model.predict(Xtest)
y_pred_binary = (y_pred + 1) / 2
print(y_pred_binary)

precision, recall, f1_score, _ = precision_recall_fscore_support(ytest, y_pred, average='micro')

print('Precision:', precision)
print('Recall:', recall)
print('F1-score:', f1_score)
```

```
Iteration number exceeded the max of 3 iterations
[0. 1. 1. ... 1. 1. 1.]
Precision: 0.4125
Recall: 0.4125
F1-score: 0.4125
```

4) Also compared the results for implemented SVM Lagrange optimized classifier and SVM without Lagrange multipliers.
SVM without Lagrange multipliers code reference from online.

5) Compared results with another classifier Logistic Regression(from the sklearn library) and Quadratic SVM optimized classifier that was implemented.