M4-L2 Problem 3

In this problem, we will investigate kernel selection and regularization strength in support vector regression for a 1-D problem.

Run each cell below, then try out the interactive plot to answer the questions.

In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.svm import SVR

xs = np.array([0.094195,0.10475,0.12329,0.12767,0.1343,0.11321,0.16134,0.16622,0.15
ys = np.array([0.51123,0.50881,0.50546,0.50756,0.51653,0.50797,0.49658,0.50899,0.50

x_gt = np.array([0.0,0.010101,0.020202,0.030303,0.040404,0.050505,0.060606,0.070707
y_gt = np.array([0.46193,0.47566,0.48699,0.49609,0.50315,0.50836,0.51189,0.51393,0.
```

In [2]:
```
%matplotlib inline
from ipywidgets import interact, interactive, fixed, interact_manual, Layout, Float


def plotting_function(kernel, log_C, log_epsilon):
    C = np.power(10.,log_C)
    epsilon = np.power(10.,log_epsilon)

    model = SVR(kernel=kernel,C=C,epsilon=epsilon)
    model.fit(xs.reshape(-1,1),ys)

    xfit = np.linspace(0,1,200)
    yfit = model.predict(xfit.reshape(-1,1))

    plt.figure(figsize=(12,7))
    plt.scatter(xs,ys,s=10,c="k",label="Data")
    plt.plot(xfit,yfit,linewidth=3, label="SVR")
    plt.plot(x_gt,y_gt,"--",label="Ground Truth")
    title = f"Kernel: {kernel},  C = {C:.1e}, eps = {epsilon:.1e}"
    plt.legend(loc="lower left")
    plt.xlabel("$x_1$")
    plt.ylabel("$y$")
    plt.title(title)
    plt.show()

slider1 = FloatSlider(
    value=0,
    min=-5,
    max=5,
    step=.5,
    description='C',
    disabled=False,
    continuous_update=True,
    orientation='horizontal',
```

```
    readout=False,
    layout = Layout(width='550px')
)

slider2 = FloatSlider(
    value=-1,
    min=-7,
    max=-1,
    step=.5,
    description='epsilon',
    disabled=False,
    continuous_update=True,
    orientation='horizontal',
    readout=False,
    layout = Layout(width='550px')
)

dropdown = Dropdown(
    options=['linear', 'rbf','sigmoid'],
    value='linear',
    description='kernel',
    disabled=False,
)


interactive_plot = interactive(
    plotting_function,
    kernel = dropdown,
    log_C = slider1,
    log_epsilon = slider2
    )
output = interactive_plot.children[-1]
output.layout.height = '500px'

interactive_plot
```

Out[2]:
```
interactive(children=(Dropdown(description='kernel', options=('linear', 'rbf', 'si
gmoid'), value='linear'), Fl…
```

Questions

1. Which kernel produced the best fit overall? (Assume this kernel for subsequent questions.)
2. As 'C' increases, does model performance on in-sample data generally improve or worsen?
3. As 'C' increases, does model performance on out-of-sample data (on the intervals [0.0, 0.1] and [0.9, 1.0]) generally improve or worsen?
4. What 'C' value would you recommend for this kernel?
5. What 'epsilon' value would you recommend?

1. rbf gives the best fit
2. As C increases, the model performance for in-sample data generally improves
3. As C increases, the model performance for out-of-sample data generally worsen
4. C = 3.2e3
5. epsilon = 1e-2