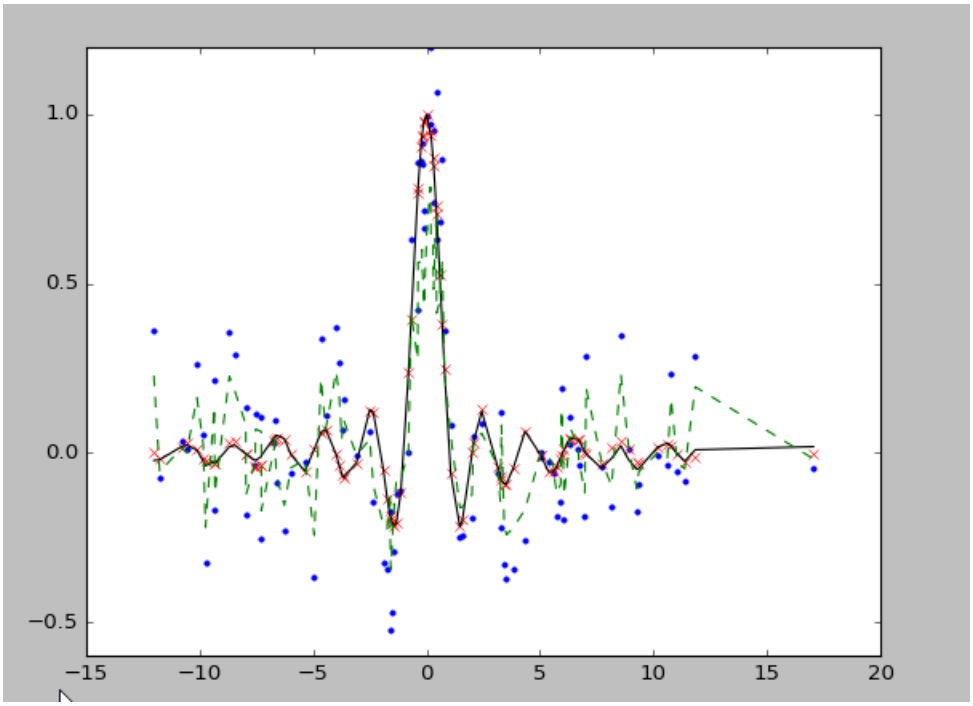
Jack Barry

AI final exam

Pablo Rivas

12/16/2016

To test my code I to ensure I had the correct modules installed to run the code I did an initial run of 100 data points, yielding the following visualization. Once the code was working I let it run for several hours to find a solution for 1000 data points.

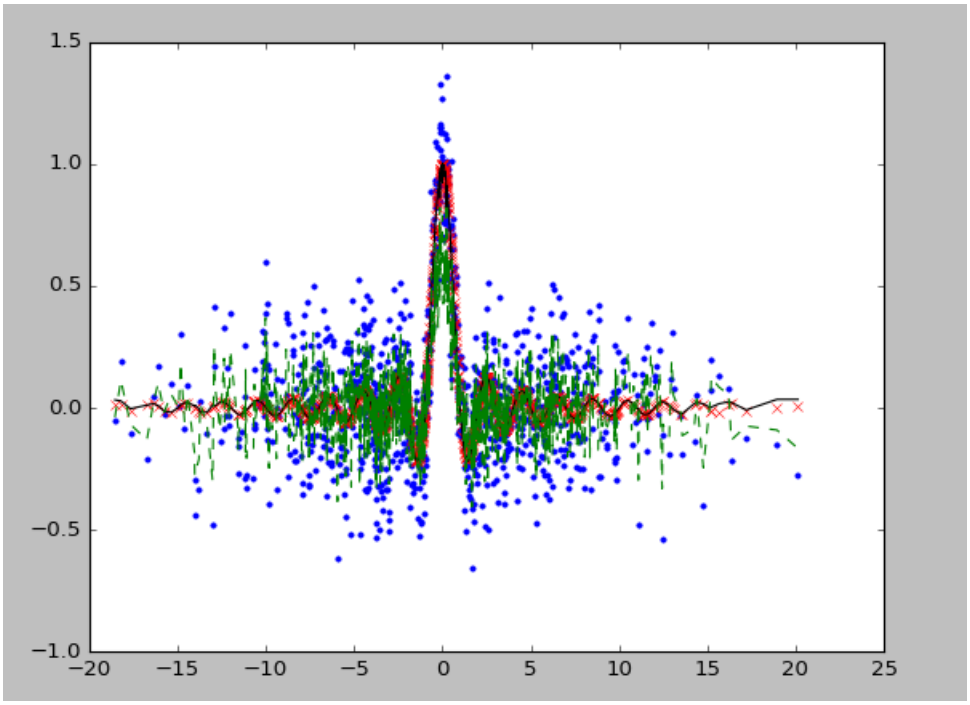


After running for several hours the SVM reached the solution below

```
C:\Users\Jack\Miniconda2\pythonw.exe C:/Users/Jack/PycharmProjects/AI_Final_Exam/SVM.py
C 0.03125, epsilon 0.0, gamma 3.0517578125e-05. Testing set CV score: -0.211768
C 0.03125, epsilon 0.0, gamma 2.0. Testing set CV score: 0.085976
C 0.125, epsilon 0.0, gamma 2.0. Testing set CV score: 0.117079
[LibSVM]*..
Warning: using -h 0 may be faster
*
optimization finished, #iter = 2654
obj = -1.095814, rho = -0.034979
nSV = 766, nBSV = 265
Training set score: 0.999656
Testing set score: 0.999192
```

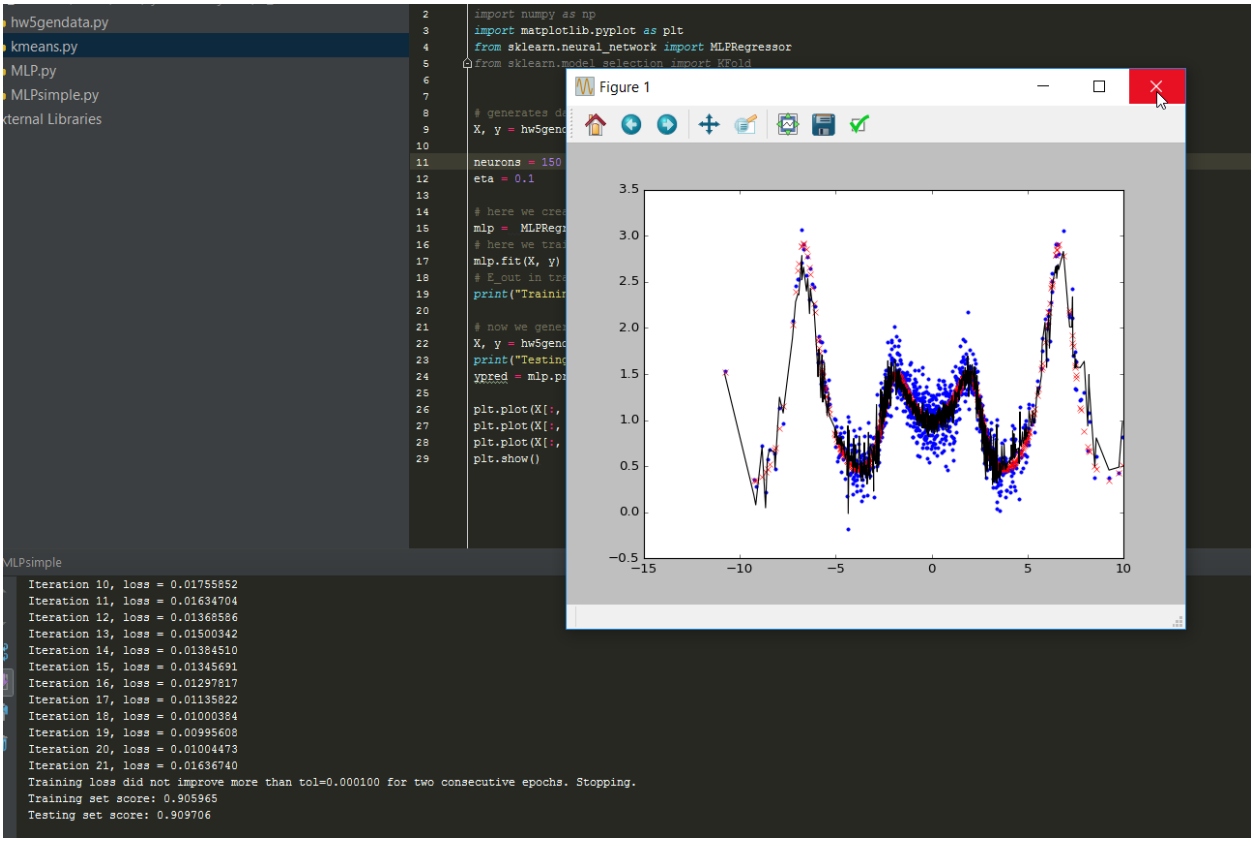
A support vector machine is a discriminative classifier formally defined by a separating hyperplane. It provides a hyperplane that best separates data of multiple dimensions. SVM is a supervised learning model. The code outputs C, epsilon, gamma, and a cross validation score. These metrics are key for

tuning an SVM model. C is a regularization parameter that controls the tradeoff between low error and having a plane that is skewed towards one dimension of the data more than the other. Our result of C = 0.125 was increased from the iteration before that which had a C of 0.03125. This is a large jump from one iteration relatively speaking but it is still a small C. Our small C shows that we have high bias and low variance. Our epsilon was 0 for every iteration. Gamma is the free parameter of the Gaussian rbf. We ended with a relatively small gamma of 2.0 which means most of our data points are being considered in how they affect the model or in other words the model has a large reach. The final scores for the models testing and training set are very good, showering a strong model for classification. Below is the visualization produced from N = 1000 of question 1.



For question 2 we see a complete change in the parameters as they go from small values to large values which shows how the model must be adapted to the data that is being worked with. The Large C means our model will have high bias and low variance, looking at the visual it makes sense because the data is fuzzy. Epsilon was 2.5. Gamma is 0.25 which is close to question one's solution. The final scores for this model were not great. The data being so closely smashed together challenges the model to reach accuracy. All in all the final exam showed how important understanding the data you are working with is to conducting and analysis.

```
C 4096.0, epsilon 2.5, gamma 0.0625. Testing set CV score: 0.305019
C 4096.0, epsilon 2.5, gamma 0.125. Testing set CV score: 0.290031
C 4096.0, epsilon 2.5, gamma 0.25. Testing set CV score: 0.279366
[LibSVM].....................................................................*.........
optimization finished, #iter = 832437
obj = -20904836.075191, rho = -88.056101
nSV = 3383, nBSV = 3372
Training set score: 0.347343
```


hw5gendata.py
kmeans.py
MLP.py
MLPsimple.py
xternal Libraries
import numpy as np
import matplotlib.pyplot as plt
from sklearn.neural_network import MLPRegressor
from sklearn.model_selection import KFold
neurons = 150
eta = 0.1
mlp.fit(X, y)
print("Trainin
print("Testing
plt.plot(X[:,
plt.plot(X[:,
plt.plot(X[:,
plt.show()
Figure 1
MLPsimple
Iteration 10, loss = 0.01755852
Iteration 11, loss = 0.01634704
Iteration 12, loss = 0.01368586
Iteration 13, loss = 0.01500342
Iteration 14, loss = 0.01384510
Iteration 15, loss = 0.01345691
Iteration 16, loss = 0.01297817
Iteration 17, loss = 0.01135822
Iteration 18, loss = 0.01000384
Iteration 19, loss = 0.00995608
Iteration 20, loss = 0.01004473
Iteration 21, loss = 0.01636740
Training loss did not improve more than tol=0.000100 for two consecutive epochs. Stopping.
Training set score: 0.905965
Testing set score: 0.909706