HOMEWORK 6

YangGao
9083410275

Solution 1

Solution 1.1

Assuming each $z_1 ... z_m$ is unique, the matrix K has ones on the diagonal ($z_i = z_i$), and zeroes elsewhere ($z_i \neq z_j$).
$v^T K v = \sum_{i=1}^{m} \sum_{j=1}^{m} v_i v_j K_{ij}$
when $i \neq j$, $v_i v_j K_{ij}$ = $v_i v_j * 0$ = 0
when $i = j$, $v_i v_j K_{ij} = v_i^2 * 1$ = positive

therefore $v^T K v$ is adding 0's and/or positive values together, and $v^T K v >= 0$

Solution 1.2

each α_i corresponds to a sample, and all other $\alpha_i' s$ are multiplying with 0, therefore each sample can be assigned a label determined by the α_i associated with that sample

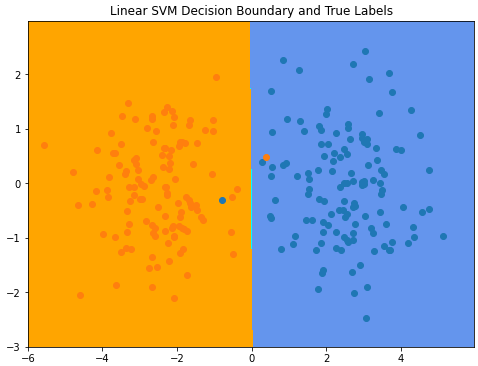
Solution 1.3

For a z that is not in the training set (i.e z does not equal any previous z in the training set), the model will predict whatever the bias term is. If bias is positive, it will predict positive, if bias is negative, it will predict negative class.

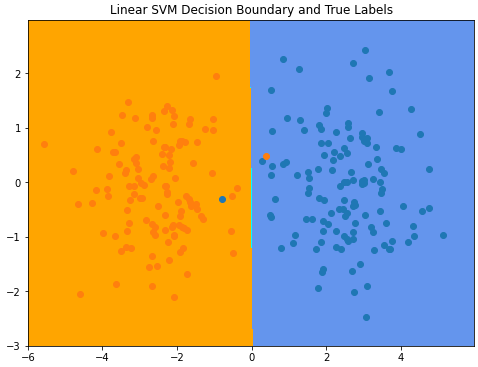
Solution 2

Solution 2.2.1

Decision Boundaries with linear SVM and logistic regression classifiers

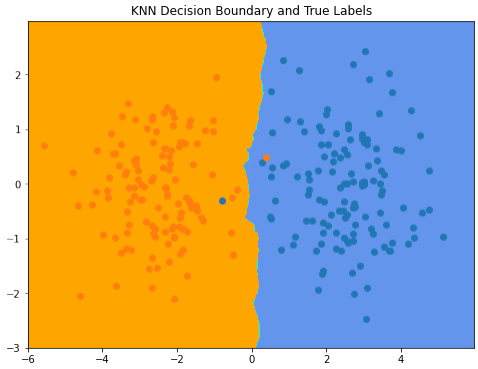


Linear SVM Decision Boundary, test accuracy = 0.992

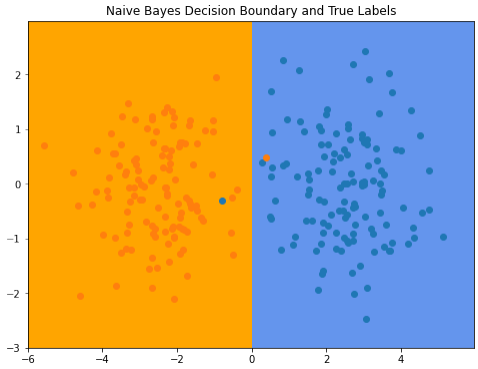


Logistic Regression Decision Boundary, test accuracy = 0.992

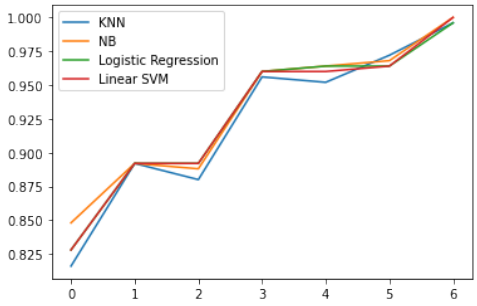
Decision Boundaries with KNN and NB Classifier



KNN Decision Boundary, test accuracy = 0.992



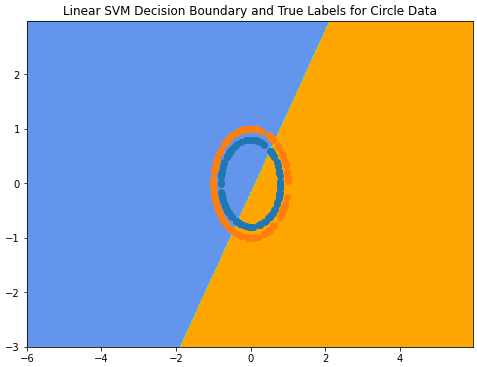
NB Decision Boundary, test accuracy = 0.992



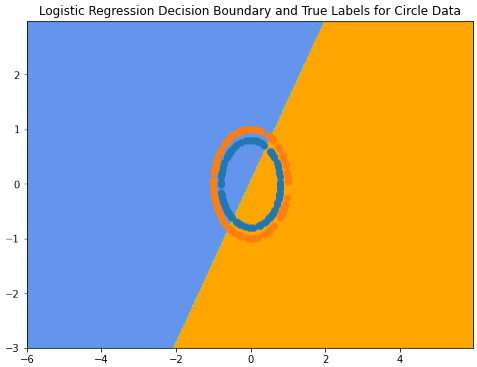
Accuracies with varying mu

My conclusions: - As μ increases, the accuracy also increases for all classifiers. This makes sense as with increasing μ, our sampled cluster centers move farther apart while covariance stays the same. - For NB, linear SVM and Logistic regression, we have straight lines as the decision boundary. This is because these are linear models, and they separate samples with a linear boundary. For KNN, the decision boundary is not straight as it is made by seeing what the closest K neighbours are.

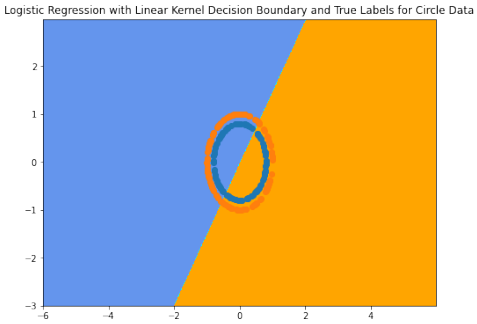
Solution 2.2.2



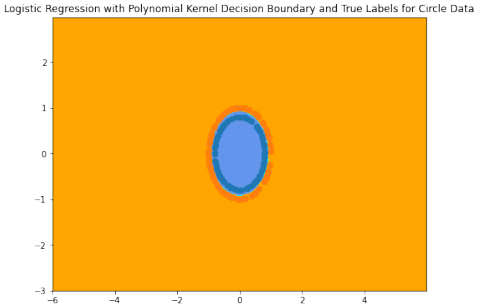
Linear SVM Decision Boundary, accuracy = 0.436



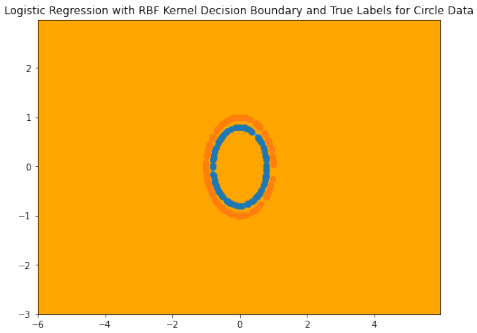
Logistic Regression Decision Boundary, accuracy = 0.488



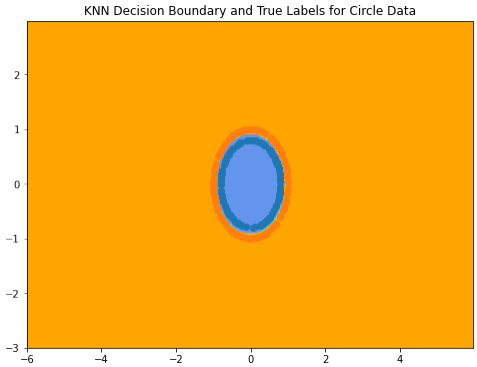
LR linear kernel Decision Boundary, accuracy = 0.48



LR poly Decision Boundary (d=4), accuracy = 1.0



LR rbf Decision Boundary, accuracy = 1



KNN Decision Boundary, accuracy = 1

My conclusions: linear classifiers can't be used directly to achieve high performance on highly non-linear data such as circles. However, using non-linear kernels, we can achieve very high performance.

Solution 2.3

model	accuracy
Linear SVM	0.9482758620689655
SVM with linear kernel	0.9655172413793104
SVM with polynomial kernel (d = 3)	0.9137931034482759
SVM with rbf kernel	0.6896551724137931
Logistic Regression	0.9482758620689655
Logistic Regression with linear kernel	0.9827586206896551
Logistic Regression with polynomial kernel (d=3)	1.0
Logistic Regression with rbf kernel	0.6896551724137931
KNN with K = 15	0.9827586206896551

I would use C parameter regularization that pushes W to be small
From linear SVM, I found the largest weights in W, which correspond to these features: [’mean concavity’, ’mean concave points’, ’radius error’, ’worst texture’, ’worst concavity’, ’worst symmetry’]