Ryan Young

CSE 4095

3/24/2020

Homework 2

Exercise 1

1.) *Recall that naïve Bayes is a generative model for classification, in contrast to logistic regression, which is a discriminative model.*

a.) *In logistic regression, we try to estimate P(y|x; θ) directly. In naïve Bayes, what do we instead first try to estimate?*
Instead of trying to estimate P(y|x; θ) directly in naïve Bayes we first try to estimate the prior based off our training data, or we could even assume all priors are equal if the training data is bad.

b.) *Once the model has been learned, how do we compute P(y|x;*θ) to make a prediction?
Once the model has been learned, we compute P(y|x; θ) to make a prediction by calculating the prior(the probability that a randomly chosen observation or data point, will belong to one of the classes), then for every feature/class combination for each data point/observation we calculate a scaler. Then the likelihood of each class is the product of that class's priors and scalers.

c.) *What is the fundamental assumption of naïve Bayes that makes it an attractive method for high*-dimensional problems? Try to state this in words and as an equivalent mathematical expression.
The fundamental assumption that naïve Bayes makes is that all values of a feature are independent of the value of any other feature, this makes it attractive for use in high-dimensionality problems. Expressed in math:

$$P(X|Y) = \prod_{j=1}^{n} P(X_j|Y)$$

d.) *Suppose that x ∈ R^n, and all n features are modeled as Bernoulli random variables. Suppose also that the prior distribution for the class label y ∈ {0,1} is modeled as a Bernoulli random variable. How many total model parameters do we need to estimate for naïve Bayes?*
The total model parameters to estimate for naïve Bayes can be attained using maximum joint likelihood and is 2n+1.

2.) *Now let's study a possible failure mode of the naïve Bayes model. In naïve Bayes, due to the assumption you described above, we can estimate the model parameters for each feature separately. Suppose that I am classifying emails such as spam or not spam, and my features are the presence/absence of words from a dictionary in a given email. Thus, I model both my features x and class label y as Bernoulli random variables.*

a.) *Suppose the word "zebra", which is in our dictionary, does not occur in any email in our training set. What is our estimate for $P(X_{zebra} = 1| y = spam)$? What is our estimate for $P(X_{zebra} = 1| y = not spam)$?*
The $P(X_{zebra}=1 \mid y = spam)$ and $P(X_{zebra}=1 \mid y = not spam)$ are both 0 because "zebra" is not in the emails!

b.) *Now suppose we need to make a prediction on a new piece of email, which does contain the word "zebra". What happens? Is this reasonable? The above scenario shows that if one feature evaluates to probability zero, we lose information from all other features. In practice, a method called Laplace smoothing is often used to prevent this, which is typically implemented by default in standard software packages for naïve Bayes.*
If the email contains the word "zebra" then the model will be unable to make a prediction on it, this is because any time there is an instance of test data that has a word not in the training data the model will be unable to make a prediction.

Exercise 2

Recall that for ordinary least squares, our hypothesis class (set of functions mapping features to labels) is the set of linear models $y = h_\vartheta(x) = \theta_0 + \theta_1 x_1 + \dots + \theta_n x_n = \theta^T x$. Suppose we have m datapoints.

1.) *What is the cost function J(θ) that we are trying to minimize in OLS? You may express this as a sum over m datapoints.*
The loss/cost function we are trying to minimize in OLS is:

$$(\theta) = \frac{1}{2}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$$

2.) *What is the design matrix X? Show what are the rows of X.*
The design matrix X is the matrix that contains all the feature vectors for the data points. Each row of the design matrix is the feature vector and each column is the value for the feature vector.

3.) *Now express the cost function J(θ) in terms of the design matrix X, the parameter vector θ, and the column vector of labels $y = [y^{(1)}, \dots, y^{(m)}]^T$.*
The cost function J(θ) expressed:

$$J(\theta) = \frac{1}{2}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 \rightarrow \frac{1}{2}\sum_{i=1}^{m}(h_i - y_i)^2 \rightarrow \frac{1}{2}(h-y)^T(h-y)$$

4.) *Expand out the cost function to show that J(θ) is a quadratic function of θ.*
Since we know:

$$\begin{bmatrix} y^{(1)} \\ \dots \\ y^{(m)} \end{bmatrix} \begin{matrix} <=> \\ <=> \\ <=> \end{matrix} \begin{bmatrix} \theta + \theta_1 x_1^{(1)} + \theta_2 x_2^{(1)} \\ \dots \\ \theta + \theta_1 x_1^{(m)} + \theta_2 x_2^{(m)} \end{bmatrix} = \begin{bmatrix} 1 & x_1^{(1)} & x_2^{(1)} \\ & \dots & \\ 1 & x_1^{(m)} & x_2^{(m)} \end{bmatrix} \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \end{bmatrix}$$

And:

$$\frac{1}{2}(h-y)^T(h-y)$$

We can simply take this second equation and expand it out:

$$\frac{1}{2}(h-y)^T(h-y) =$$
$$\frac{1}{2}[h^Th - h^Ty - y^Th + y^Ty] =$$
$$\frac{1}{2}h^Th - \frac{1}{2}(h^Ty + y^Th) + \frac{1}{2}y^Ty =$$
$$\frac{1}{2}h^Th - y^Th + \frac{1}{2}y^Ty =$$
$$\frac{1}{2}h^Th - y^Th + \frac{1}{2}y^Ty =$$
$$\frac{1}{2}\theta^Tx^Tx\theta - y^Tx\theta + \frac{1}{2}y^Ty$$

5.) *What is the gradient of the cost function with respect to θ?*
The gradient of the cost function with respect to θ is:
$\nabla_\theta J = (x^Tx)\theta - x^T$

6.) *We can solve for θ directly by setting the gradient to zero. Why is this sufficient?*
We can solve for θ directly by setting the gradient to zero because the function has a positive and definite hessian. This means that the function is convex and that the critical point we would find (Because the hessian is positive and definite) is a minimum.

Exercise 3

Recall that the optimization problem for soft margin support vector classification is given by:

$$\min_{w,b,\xi} \frac{1}{2}w^Tw + C\sum_{i=1}^{m}\xi_i$$
$$\text{s.t. } y^{(i)}\left(w^Tx^{(i)} + b\right) \geq 1 - \xi_i \text{ , } \forall i \in \{1, ..., m\}$$
$$\xi_i \geq 0 \text{ , } \forall i \in \{1, ..., m\}$$

In this problem, we will explore the effect of the hyperparameter C on the decision boundary.

1.) *For which datapoints $x^{(i)}$ will the optimal* $\xi_i^* = 0$?
The datapoints $x^{(i)}$ that are on the correct side of the soft margin will have the optimal $\xi_i^* = 0$.

2.) *For which datapoints $x^{(i)}$ will the optimal* $\xi_i^* > 1$?
The datapoints $x^{(i)}$ that are completely on the wrong side of the soft margin will have the optimal $\xi_i^* > 1$.

3.) *Download the data files xvals.dat and yvals.dat included in the homework (these are the same as in Homework 1, and contain the features and labels, respectively, for 99 datapoints). Also download the Jupyter Notebook file hw2ex3-34.ipynn. Review the Python script to make sure you understand the components. The scikit-learn documentation may be helpful:* https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html.

a.) *Recall that w is the vector orthogonal to the decision boundary hyperplane, and thus could technically have arbitrary length. In the optimization problem for SVMs, we choose to solve for the unique w such that the margin is a specific function of w. What is this function?*
This function is:
$\min_{w,b} \frac{1}{2}\|w\|^2$, such that $y^{(i)}(w^Tx^{(i)} + b) \geq 1$, $\forall i$

b.) *The decision boundary can also be defined in terms of w and b. Write down the equation that describes all x on the decision boundary.*

The equation in the code is described as (-w[0]/w[1])*xx1 – b/w[1] but in terms of math is described as: $d = (\frac{-w^T}{||w||})x - \frac{b}{||w||}$.

c.) *With respect to the decision boundary and margin, which datapoints $x^{(i)}$ are the support vectors (in general, not specifically for this data set)?*

With respect to the decision boundary and margin, the datapoints that are the support vectors are the closes points to the decision line, from each class.

4.) *Now we will look at the effects of varying the hyperparameter C. Note where C is called in the Python script, as an argument to SVC(). In the script, we use C = 1.*

a.) *What is the margin in this case?*

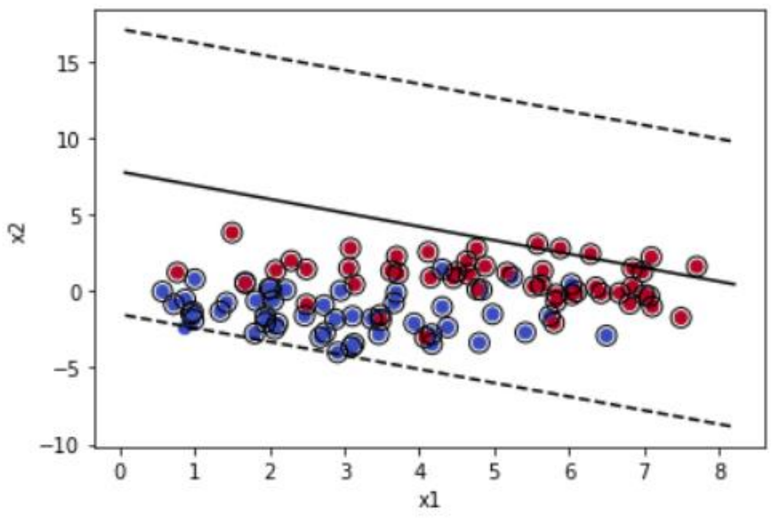
The margin is 1.2217 and this comes from 1 / ||w||.

b.) *How many support vectors are there?*

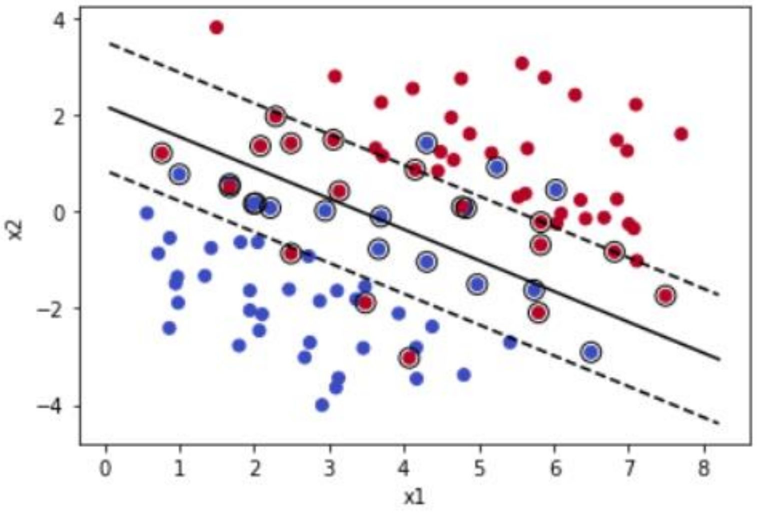
There are 33 support vectors.

c.) *Re-fi the Support Vector Classifier for different values of C, and observer how this changes the decision boundary, margin, and set of support vectors. Submit the plots for* $C \in [10^{-3}, 1, 10^6, 10^{10}]$.

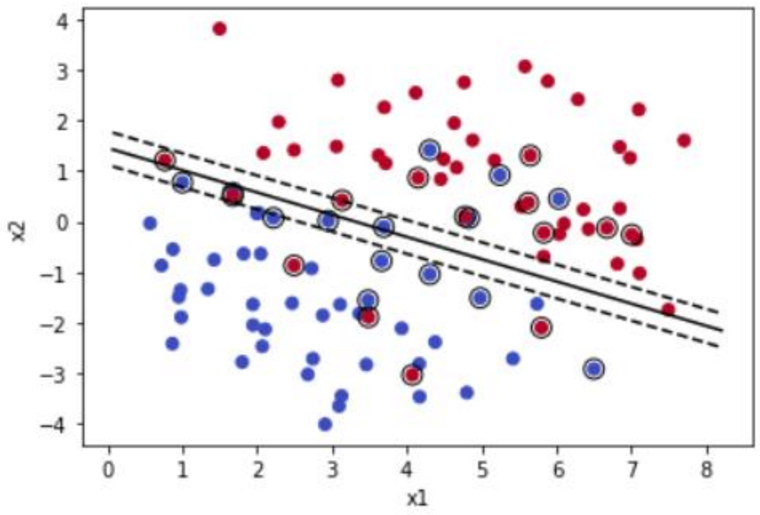
C: 10^{-3}



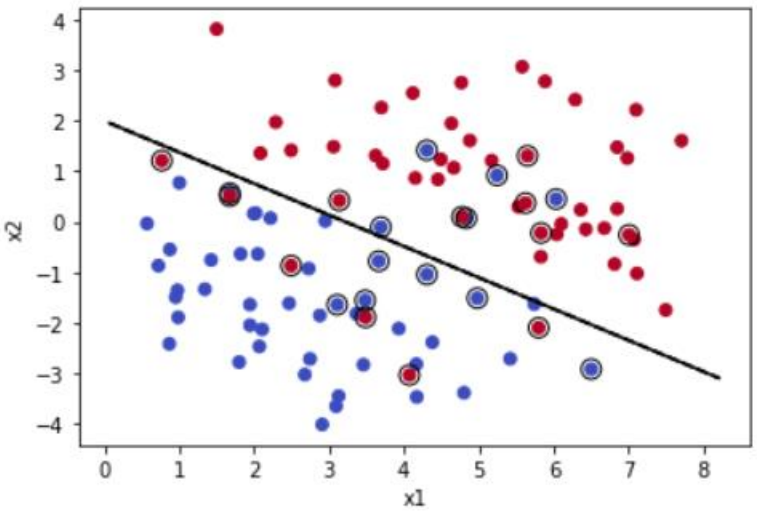
C: 1



C: 10^6



C: 10^{10}



d.) *As C increases, how does the margin change qualitatively?*
As C increases it makes the margin decrease in size(move closer to 0).

5.) *Download the Jupyter Notebook file hw2ex3-5.ipynb. This Python script implements k-fold cross-validation on the same problem as above. Note that the script may take from 20 seconds to a few minutes to run, depending on your processor.*

a.) *For k = 10 and* $C \in [10^{-4}, 10^{-2}, ..., 10^{6}]$*, plot the average training error and validation error(where the error is defined as 1-accuracy) over k folds as a function of C(this has been implemented for you in the script; note that C is plotted on a log scale for clarity). What value(s) of C should we use to train our linear SVC classifier?*
We should use a C value of 10^4 because it provides the lowest 1-accuracy (error) for both training and validation.

b.) *Should there exist a C for which the training error is 0 in this case? Why or why not?*
No there should not exist a C for which the training error is 0 in this case because that would be that the model is perfectly trained on the data and that there would not be any data points on the wrong side of the line.

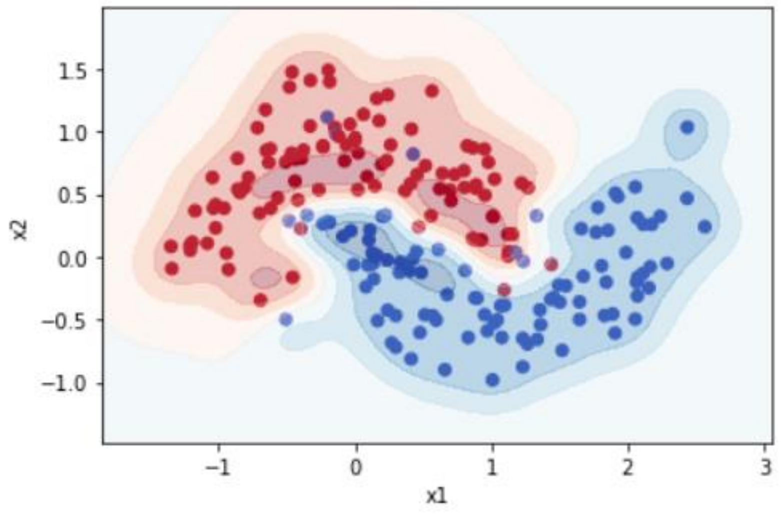
Exercise 4

In this exercise, we will explore using the rbf (Gaussian) kernel to solve a nonlinear classification problem. We will use the classic toy dataset of two interleaving half circles (moons) with noise.

1. *Download the Jupyter Notebook file hw2ex4.ipynband plot the moons dataset (this has already been implemented in the script). In datasets.make_moons(), adjust the noise parameter to noise = 0 to see the underlying shape from which the data is generated. Return the noise parameter to noise=0.25and the random seed to random_state = 2 which are the settings we will use for the remainder of this problem.*
2. *How many hyperparameters are there for rbf kernel SVM, and which ones are they? Adjust the hyperparameter values (varying over different orders of magnitude for best results) and describe qualitatively how increasing each hyperparameter changes the decision function (distance from the decision boundary, scaled by* $||w||_2$*). The script plots the decision function using colors that get darker over points further away from the decision boundary.*

There are three hyperparameters for a rbf kernel SVM and C, kernel and Gamma. When we increase C, the data points seem to decouple more from the underlying colors that represent the decision boundary. Decreasing C seems to have a minimal change. Increasing the gamma seems to increase how tightly the decision boundary background colors encapsulate the data points and decreasing seems to loosen this.

3. *For simplicity, we will first set C = 1. Perform 10-fold CV and plot the average training and validation error for γ from 10^{-3} to 10^{3} as given in this script. What is the optimal value of γ?*
   The optimal value for gamma seems to be 10^{1} because there the training and validation 1-accuracies appear to be at their lowest.
4. *Should there exist a* γ for which the training error is 0 in this case? Why or why not?
   Yes there could exist a gamma for which the training error is 0 in this case because we could have a kernel that draws a very complex decision boundary that makes sure that no datapoints are left on the wrong side of the line.
5. *Now let's optimize over all hyperparameters using grid search and 10-fold CV. Run the code for this implemented in the script. What are the optimal hyperparameter values? Submit the plot for the decision function for the model trained on all of the data using the hyperparameter values. Does it make sense?*
   The optimal hyperparameter values are C= 1^{-8} and Gamma = 10. Yes, it does make sense and the plot below is that with the script is run using a C = 1^{-8} and Gamma = 10.



1.5
1.0
0.5
0.0
−0.5
−1.0
x2
−1
0
1
2
3
x1