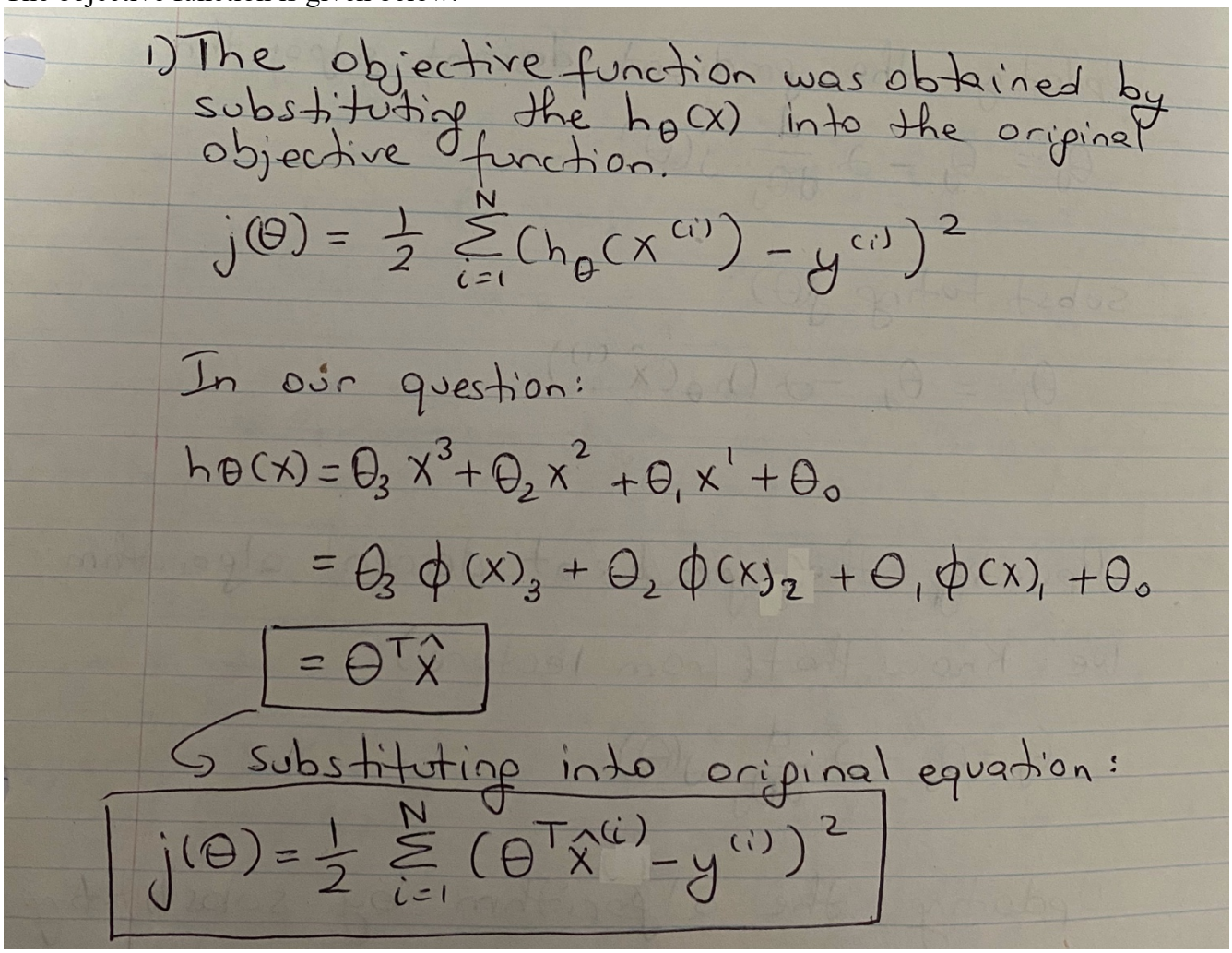
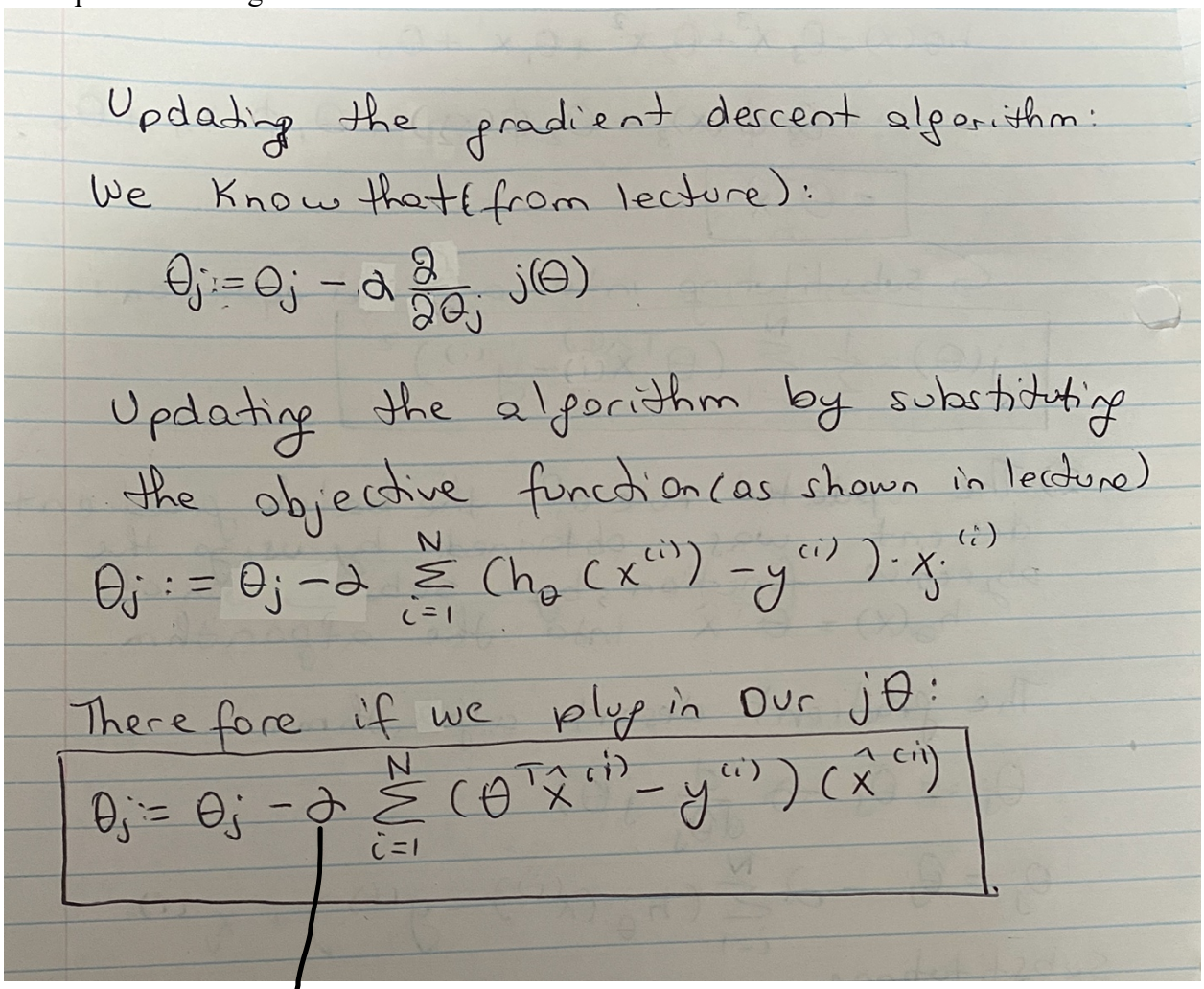
CP8318 Assignment 1
Student Name: Albina Cako

Question 1
The objective function is given below:



(I notated alpha in the equation, as my handwriting might be unclear)
The updated rule is given below:



alpha

Part 2.

For this question the normal equation was used to plot the graph. The normal equation is as follows:

$$\theta = \left(X^T X\right)^{-1} . \left(X^T y\right)$$

The normal equation allows us to predict the values of the parameter (theta) without using an optimization algorithm, such as the gradient descent. For small features this equation would give the results that fit the data best. The normal equation is the most optimal way to fit the data in a hypothesis curve for small features. In this section we did used the normal equation on the training data so we can compare it with the GD and SGD optimization algorithms in the further questions.

Here is the plot obtained through the code. As we can see, the learnt hypothesis line fits smoothly to the training data.

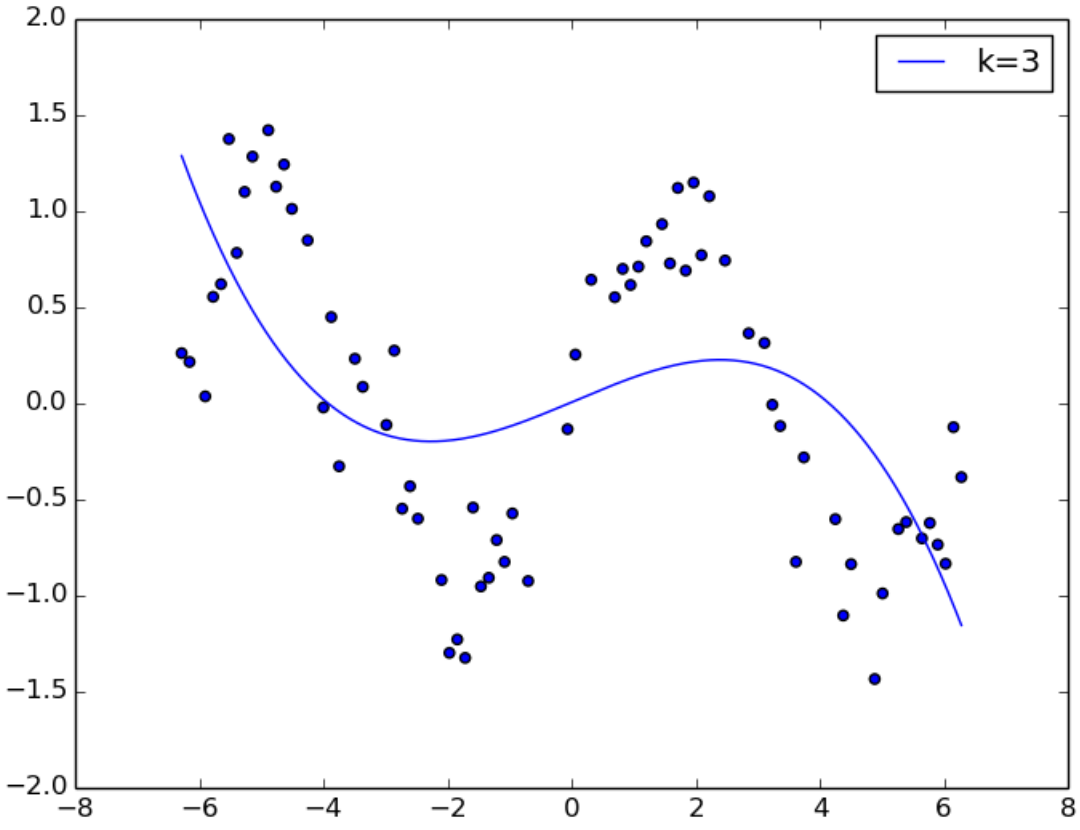


Figure 1. Scatter plot using the normal equation.

Part 3.

GD Graph:

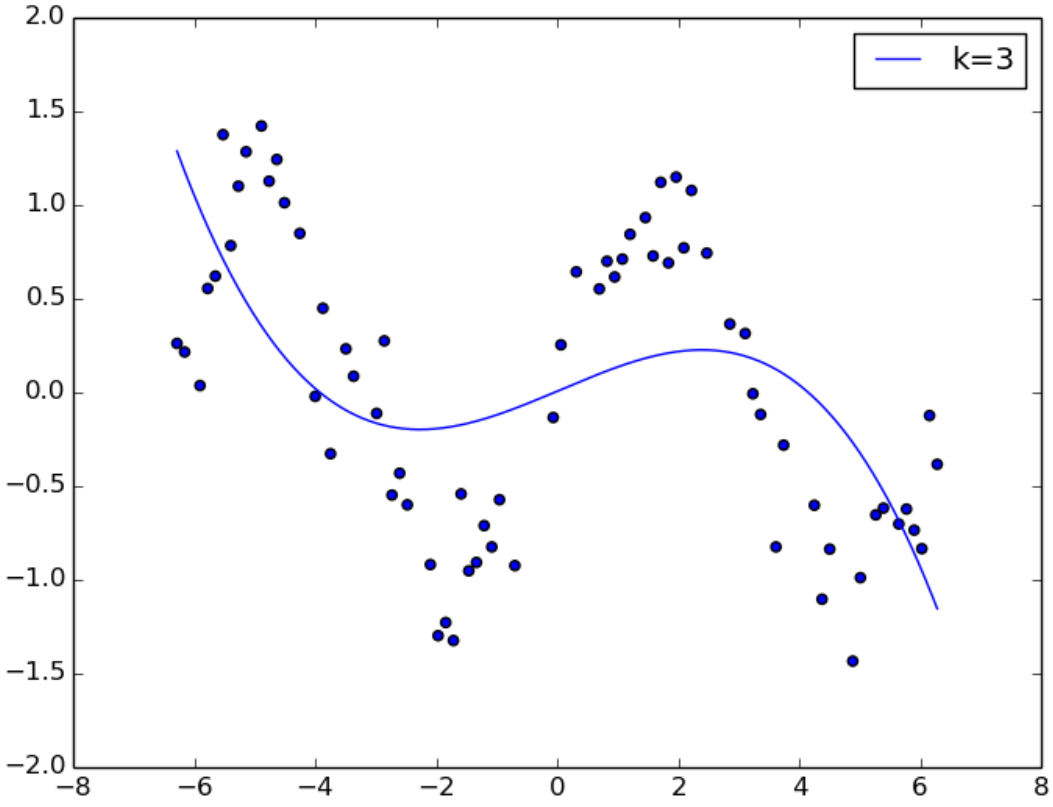


Figure 2. Scatter plot using the GD algorithm to predict the data

SGD Graph:
The graph for gradient descent is shown below:

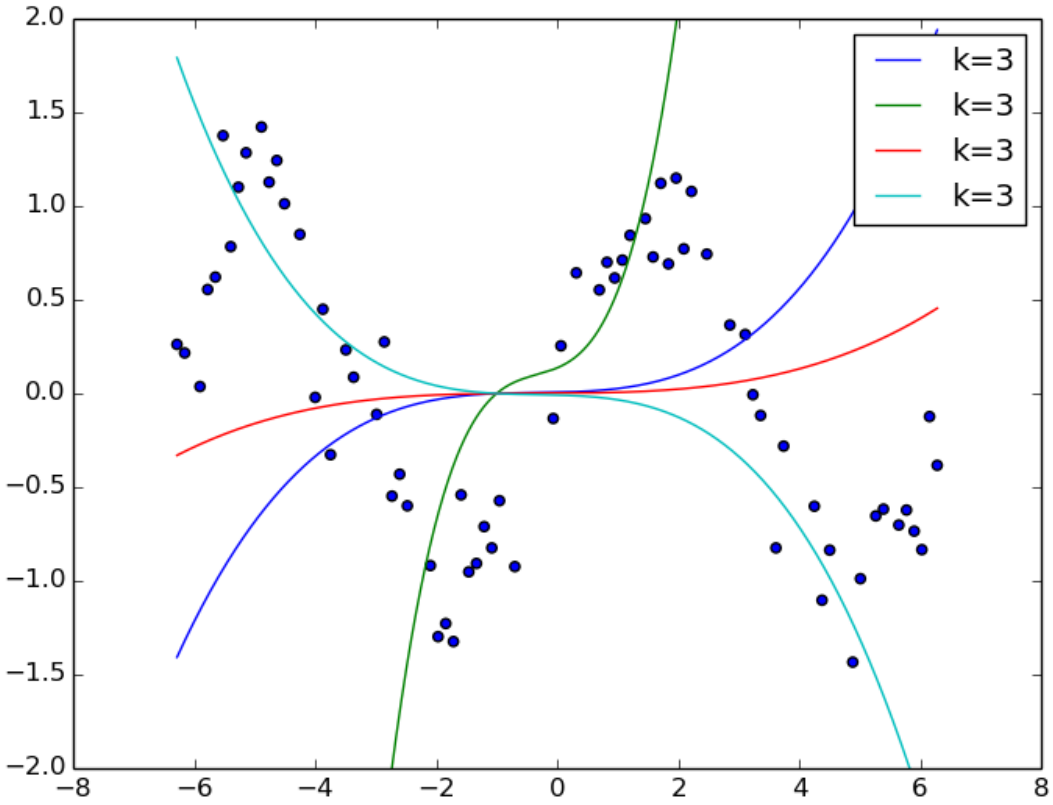


Figure 3. Figure 1. Scatter plot using the SGD algorithm to predict the model.
(light blue line was the line that fit the data best)

As we see on part 3, the Gradient Descent algorithm is does a better job at reducing the cost function (error), rather than the SGD algorithm. This is because the GD algorithm runs through all the samples in the training set in order to do a single parameter update in the iteration, while the SGD uses only one sample or a subset of the sample values to update a parameter. Therefore, the SGD would run faster, however it would not minimize the cost (error) function and fit the graph as well as the GD algorithm. As we see, the SGD does not fit the data well.

I also observed that the higher the number of iterations, the closer the GD and SGD graphs would resemble the graph from the normal equations. This is because as the number of iterations increases, the better the optimization of the algorithm. The GD algorithm did a better job at minimizing the cost (error) function and it's graph to be closest (identical) to the normal equation graph, then the SGD graph. Therefore, the 10000 iterations would lead to the closest to normal function for both GD and SGD in this example.

Alpha was kept at default 0.000002 for the write up of this assignment. It is important to understand alpha in order to use these algorithms. If alpha is too high, we will end up overshooting. If alpha is too low, it will take way too many iterations to achieve the desired result. I kept the alpha 0.000002 as it worked well on the algorithm. Increasing alpha was not necessary for the GD as it fit the data well. For SGD alpha could have been increased slightly to achieve better results (for example alpha = 0.000006).

Part 4.

For this section the normal equation was used to plot the graph. The graph is shown below:

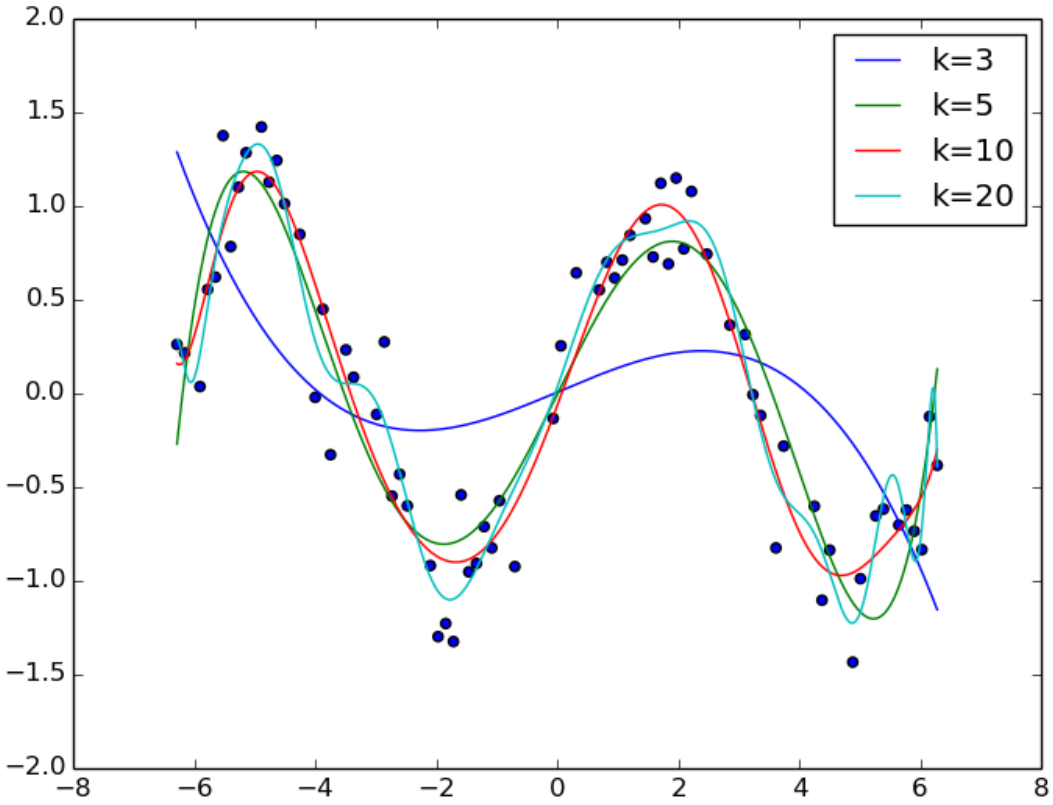


Figure 4: Scatter plot using k degree polynomials using normal equation.

We notice as the k increases, the fitting of the training dataset improves. This means that higher degree polynomials will fit the data better. However, if we do this process using the GD and SGD, we might get overflow errors, since for example, at k=20, the dataset is very large and the gradient will have to go through a large set of data for each iteration. This could lead to overflow due to calculation errors, as polynomials with a very high degree can be numerically unstable.

Part 5.

The data was plotted using the sine function. Below is the graph:

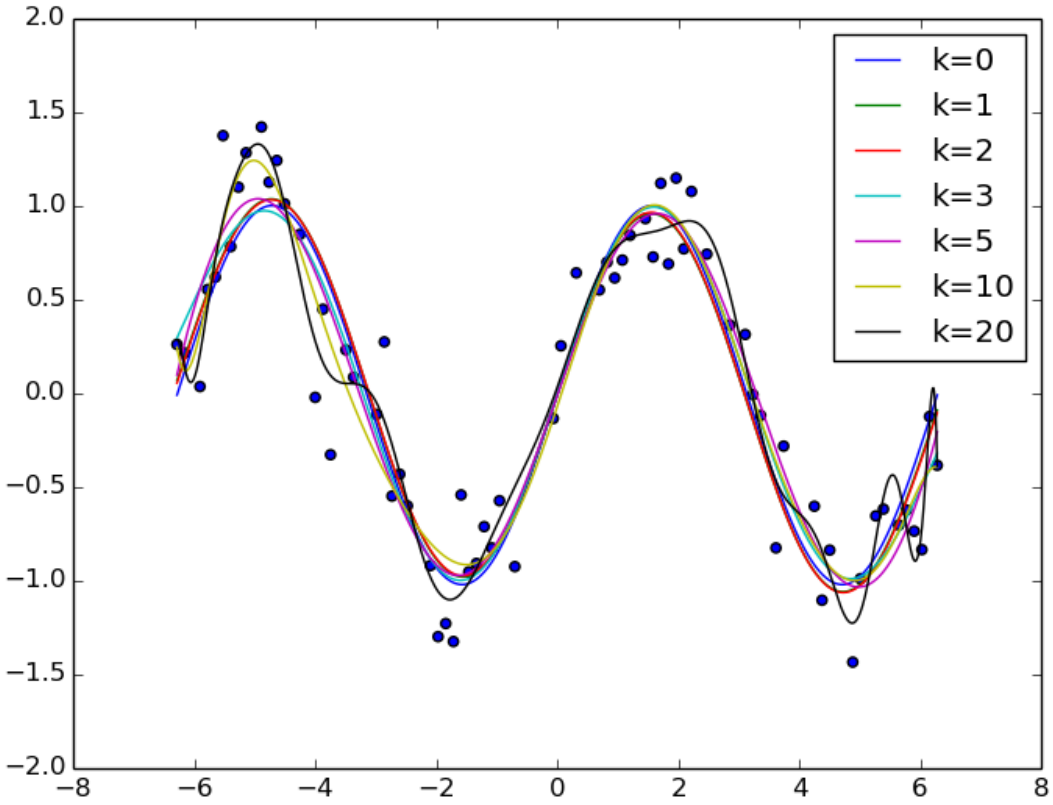


Figure 5: Scatter plot using k degree polynomials using sine function.

As we see, the sine function fits the data better than the normal equation. This makes sense, as the data was sampled using a sine function. In this question, it is also noticed that as the polynomial degree increases (k), the sine function fits it better. The same result was noticed on 1.4. However, we have to be cautious with very high degree polynomials, as they are unstable.

Part 6 . The solution graph to this part is shown below:

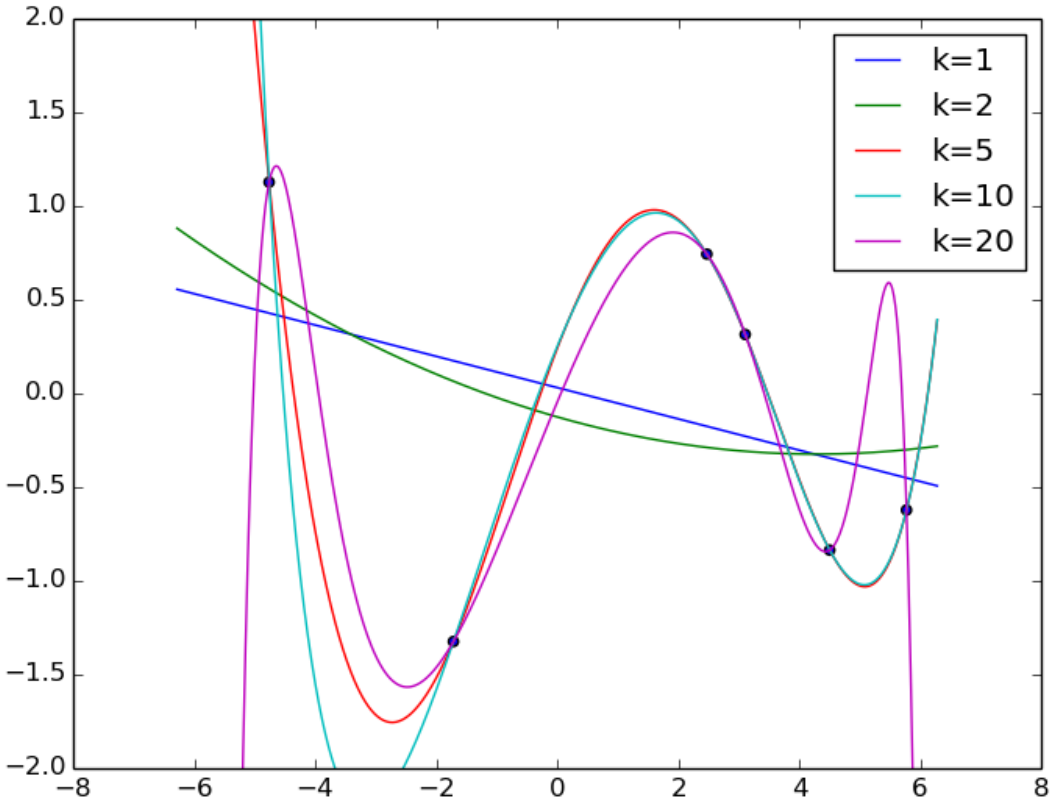


Figure 6: Scatter plot using k degree polynomials using normal equation in small dataset.

By using the small dataset, we notice that as k (the degree of the polynomial) increases, the model overfits the data. As stated in the remark section, overfitting is when the model fits the dataset very well then goes “wild”. This can be seen in our graph. This is because when you have a small data to represent the model, the model will overfit the training data. However, when you use this model on a new dataset, the model’s performance on the new dataset will not be optimal. This type of error, is called overfitting.

Question 2.

Part 1: This graph shows the logistic regression run at 1000 iterations.

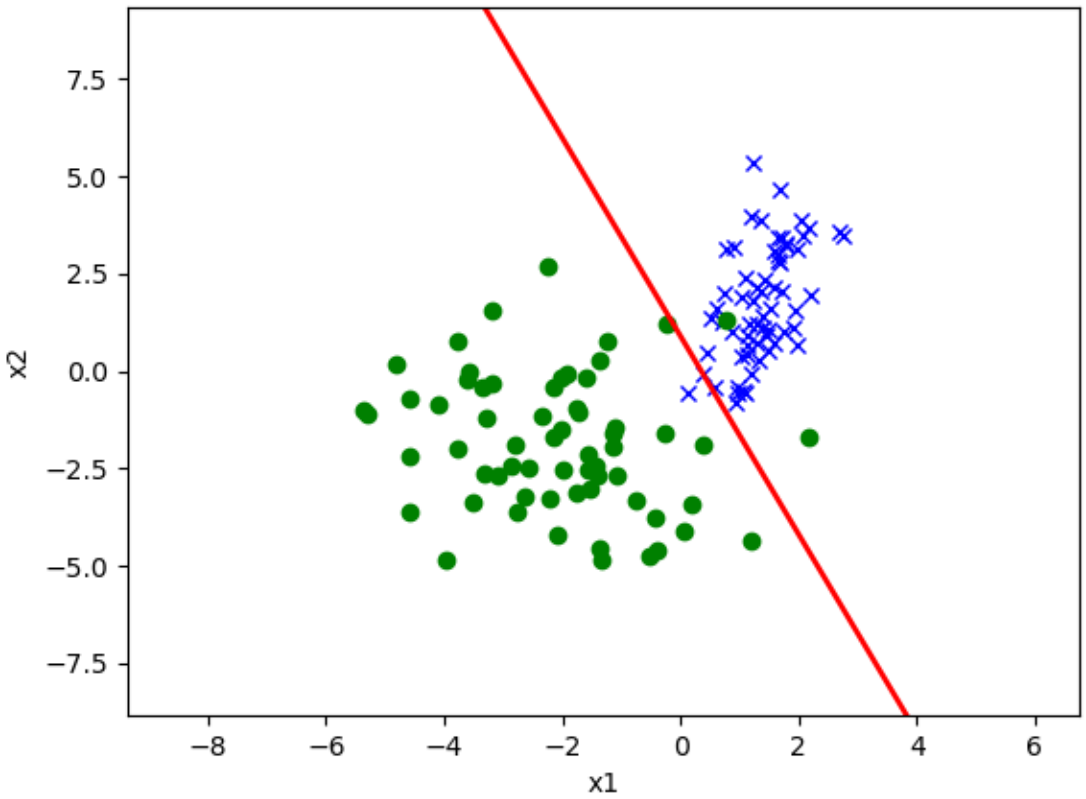


Figure 7. Test set decision boundary using the t-labels for training.

In this image we see the decision boundary that separates the two groups of the test sets, where the blue x shaped represent the test samples of true label $t^{(i)} = 0$ and the green circles represent the samples of the true label $t^{(i)} = 1$. By looking at the graph, the model predicts the probability well, for both $t^{(i)} = 1$ given x and $t^{(i)} = 0$ given x. This is because the decision boundary separates the two samples well, as seen in the graph. However it does not do a perfect job as we seen in the graph, as there are some circles in the true label $t^{(i)} = 0$ sample and some x's in the true label $t^{(i)} = 1$ sample. The graph was obtained at 1000 iterations to save time, because increase of iterations did not make a difference in the graph, even though theoretically it would have given a better decision boundary.

Part 2: The graph is shown below.

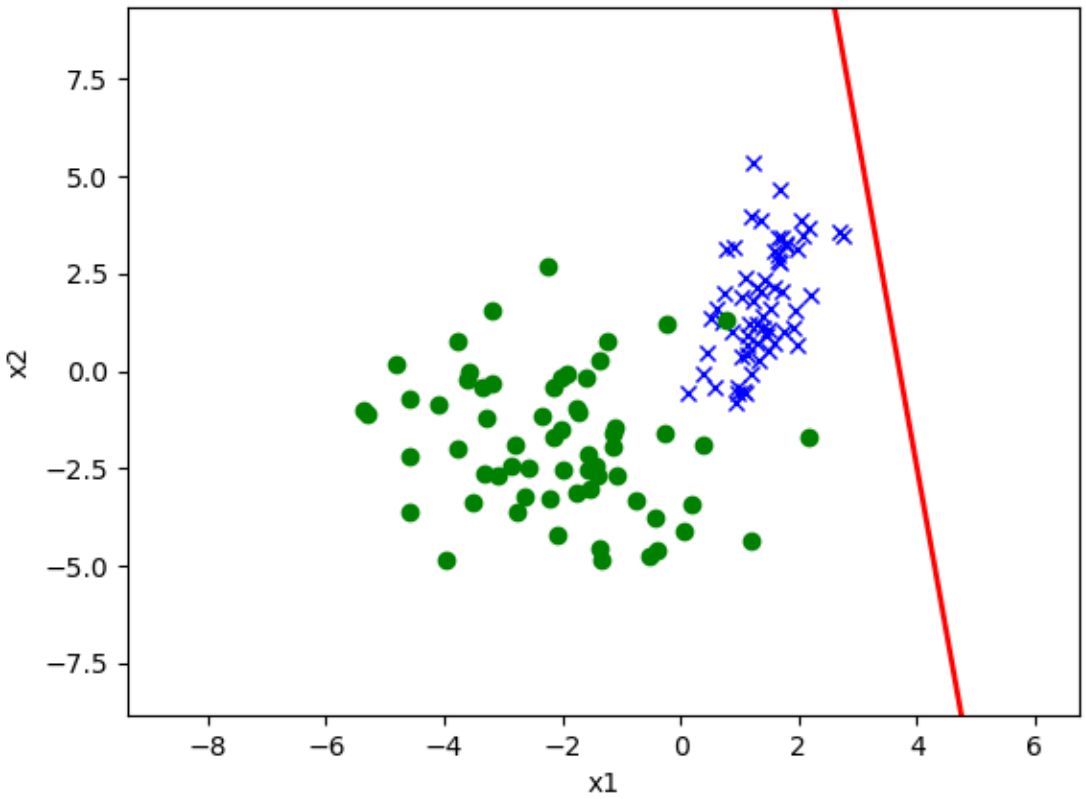


Figure 8. Test set decision boundary using the y-labels for training .

In this image we see the decision boundary that separates the two group of the test sets, where the blue x shaped represent the test samples of true label $t^{(i)} = 0$ and the green circles represent the samples of the true label $t^{(i)} = 1$. However, in this section we trained the test set by using the y-labels and used the true labels for plotting. By looking at the graph, the model does not predict well the probability of $t^{(i)} = 1$ given x and $t^{(i)} = 0$ given x. This can be seen as the decision boundary is outside of the samples. This shows us that it is better to do the training with the true labels, then the y-labels. At this graph 1000 iterations are shown to save time, as increasing the iterations did not make a difference on the graph, even though theoretically it would have given a better decision boundary.

Part 3:

2.3

For this question I used Baye's Theorem and the law of total probability. Then, I substituted values given in the question.

Bayes rule:

$$P(A|B) = \frac{P(B|A)\,P(A)}{P(B)}$$

Law of total probability:

$$P(A) = P(A|B)\,P(B) + P(A|B)\,P(B)$$

Using Bayes rule $\Rightarrow P(t^{(i)}=1|y^{(i)}=1, x^{(i)}) = \dfrac{P(y^{(i)}=1|t^{(i)}=1, x^{(i)})\,P(t^{(i)}=1|x^{(i)})}{P(y^{(i)}=1|x^{(i)})}$

$$= \frac{P(y^{(i)}=1|t^{(i)}=1, x^{(i)})\,P(t^{(i)}=1|x^{(i)})}{\left(\begin{array}{l} P(y^{(i)}=1|t=0, x^{(i)})\cdot P(t^{(i)}=0|x^{(i)}) \\ + P(y^{(i)}=1|t^{(i)}=1, x^{(i)})\,P(t^{(i)}=1, x^{(i)}) \end{array}\right)}$$

expanding by law of total probability

$$= \frac{P(y^{(i)}=1|t^{(i)}=1, x^{(i)})\,P(t^{(i)}=1|x^{(i)})}{\left(\begin{array}{l} 0\cdot P(t^{(i)}=0|x^{(i)}) \\ + P(y^{(i)}=1|t^{(i)}=1, x^{(i)})\,P(t^{(i)}=1|x^{(i)}) \end{array}\right)}$$

substituted with values given in the question

$$= \frac{P(y^{(i)}=1|t^{(i)}=1, x^{(i)})\,P(t^{(i)}=1|x^{(i)})}{P(y^{(i)}=1|t^{(i)}=1, x^{(i)})\,P(t^{(i)}=1|x^{(i)})}$$

$$= 1$$

Hilary

Part 4:

2.4.

To solve this question I used the law of total probability. Then, I substituted values given in the question.

Using law of total probability:

$$p(y^{(i)}=1|x^{(i)}) = p(y^{(i)}=1|t^{(i)}=1, x^{(i)}) \cdot p(t^{(i)}=1, x^{(i)}) + p(y^{(i)}=1|t^{(i)}=0, x^{(i)})\, p(t^{(i)}=0|x^{(i)})$$

given:

$$p(y^{(i)}=1|t=1, x^{(i)}) = \alpha$$

$$p(y^{(i)}=1|t^{(i)}=0, x^{(i)}) = 0.$$

Substitute back in equation:

$$p(y^{(i)}=1|x^{(i)}) = \alpha \cdot p(t^{(i)}=1, x^{(i)}) + 0 \cdot p(t^{(i)}=0|x^{(i)})$$

$$= \alpha\, p(t^{(i)}=1, x^{(i)})$$

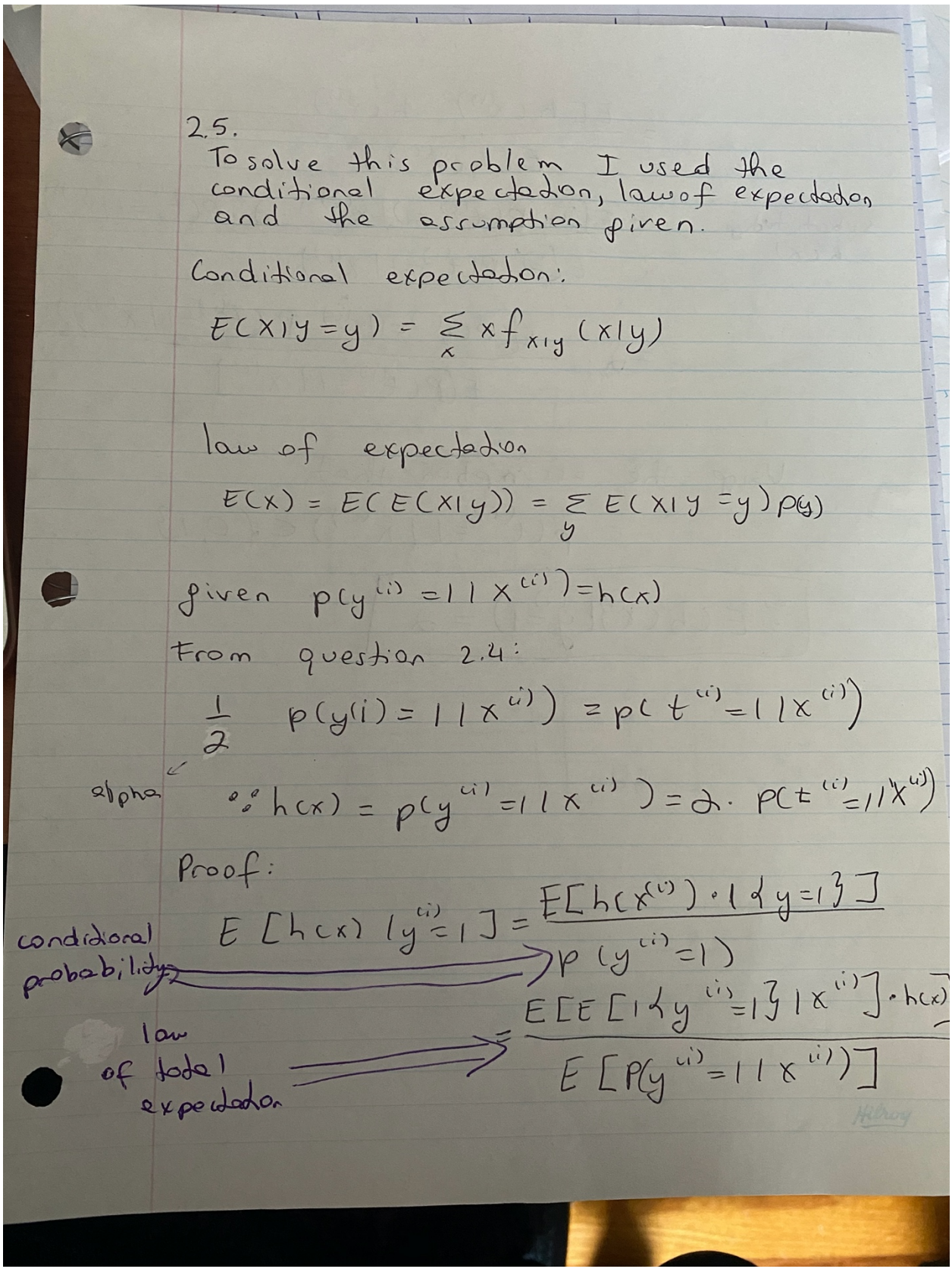
If we divide both sides by α:

$$\frac{p(y^{(i)}=1|x^{(i)})}{\alpha} = p(t^{(i)}=1, x^{(i)})$$

Rewriting it:

$$\boxed{p(t^{(i)}=1, x^{(i)} = \frac{1}{\alpha} \cdot p(y^{(i)}=1|x^{(i)})}$$

Part 5:



2.5.
To solve this problem I used the conditional expectation, law of expectation and the assumption given.

Conditional expectation:

$$E(X|y=y) = \sum_x x f_{x|y}(x|y)$$

law of expectation

$$E(X) = E(E(X|y)) = \sum_y E(X|y=y)P(y)$$

given $p(y^{(i)}=1|x^{(i)}) = h(x)$

From question 2.4:

$$\frac{1}{2}\ p(y(i)=1|x^{(i)}) = p(t^{(i)}=1|x^{(i)})$$

alpha

$$\therefore h(x) = p(y^{(i)}=1|x^{(i)}) = 2 \cdot p(t^{(i)}=1|x^{(i)})$$

Proof:

conditional probability

$$E[h(x)|y^{(i)}=1] = \frac{E[h(x^{(i)}) \cdot 1\{y=1\}]}{p(y^{(i)}=1)}$$

law of total expectation

$$= \frac{E[E[1\{y^{(i)}=1\}|x^{(i)}] \cdot h(x)]}{E[P(y^{(i)}=1|x^{(i)})]}$$

$$= \frac{E[h(x^{(i)}) \cdot h(x^{(i)})}{E[h(x^{(i)})]}$$

Substituting $h(x)$

$$= \frac{E[2 \cdot (p(t^{(i)}=1 \mid x^{(i)}) \cdot 2 \cdot p(t^{(i)}=1 \mid x^{(i)})]}{E[2 \cdot p(t^{(i)}=1 \mid x^{(i)})]}$$

$$= 2 \cdot \frac{E[(p(t^{(i)}=1 \mid x^{(i)}) \cdot p(t^{(i)}=1 \mid x^{(i)})]}{E[p(t^{(i)}=1 \mid x^{(i)}]}$$

Using the assumption that

$$p(t^{(i)}=1 \mid x^{(i)}) \in \{0, 1\}.$$

$$\therefore E(h(x^{(i)}) \mid y^{(i)}=1) = 2$$

Part 6:

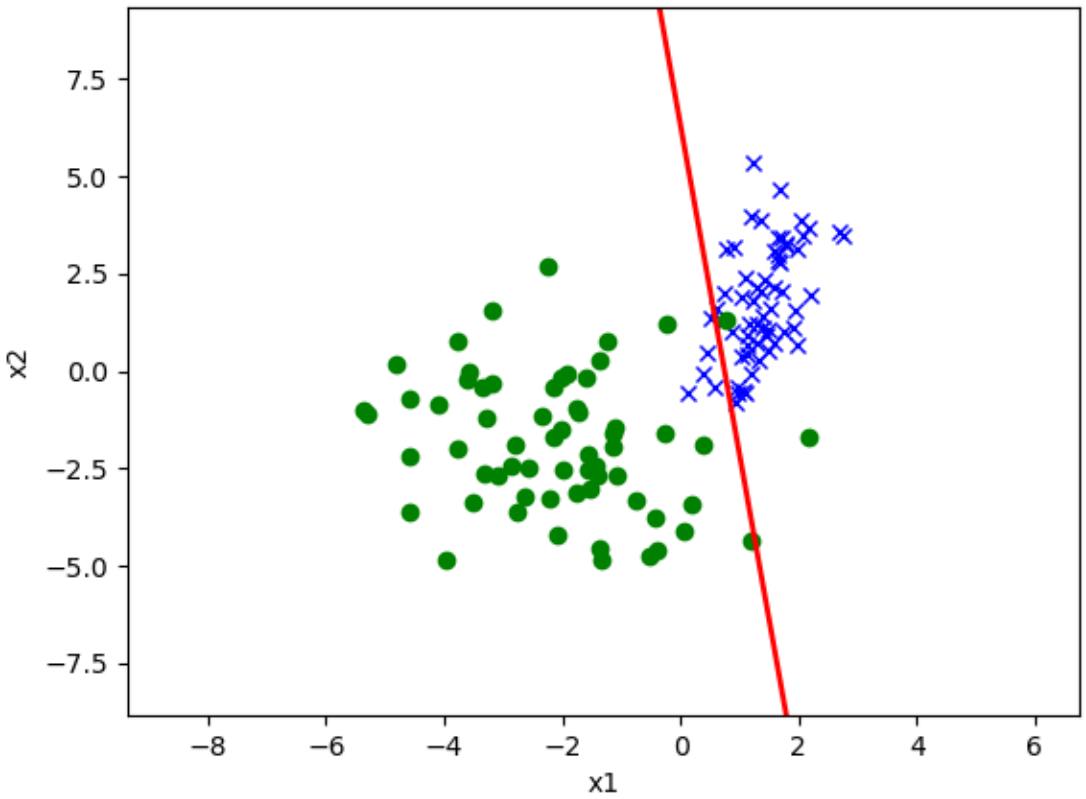


Figure 8. Test set decision boundary using the y-labels for training and alpha to plot decision boundary

In this image we see the decision boundary that separates the two group of the test sets, where the blue x shaped represent the test samples of true label $t^{(i)} = 0$ and the green circles represent the samples of the true label $t^{(i)} = 1$. In this section we see that the decision boundary did overall a good job in separating the test samples. Comparing this with the graph in 2.1, this model is not as accurate. In this section, we used y-labels to train the test set like we did in 2.2. Then we used the model for the prediction of positive examples of the validation set. Then, we estimated alpha by using the prediction of the validation set and plotted the decision boundary based on alpha. This improved our model compared to the results in 2.2. As said in the remark section, since the true probability of t given x is estimated by a constant factor away from probability of y given x, then if we sort the data in raking order (which data one is more to be involved in the training of the model), then we do not need to calculate the alpha. This graph is shown at 1000 iterations. Increasing the iterations gives a slightly more accurate graph. The difference was not very visible so I used 1000 iterations to save time when I obtained this graph.