ml_regressions

Experiments in linear, logistic, polynomial regression and regularization

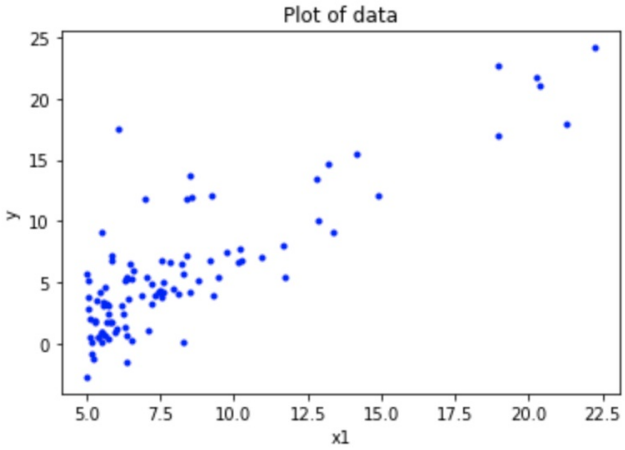
To execute the project, simply

```
git clone
jupyter notebook Linear\ regression.ipynb
```

Documentation is provided in code comments

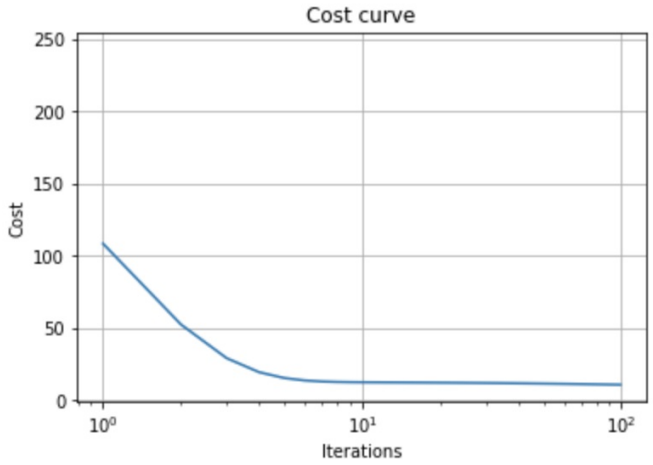
Dataset 1.

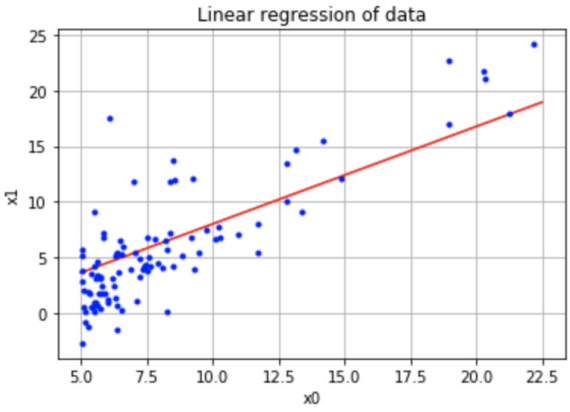
A visialization of the data provided:




```
iterations = 100
learn_rate = 0.01
```

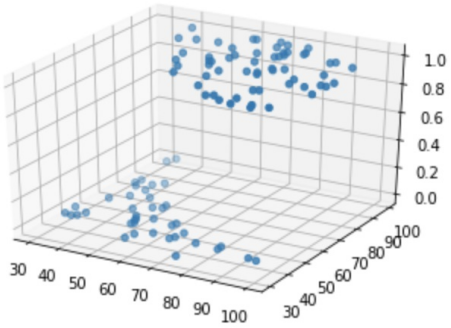
With the above parameters and using gradient descent of the MeanSquareError cost function. We obtained the following (log) cost curve that found the minima quickly. Resulting in the following output line

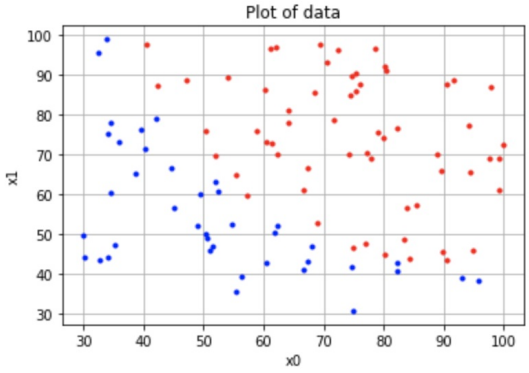




Dataset 2.

A visialization of the data provided:



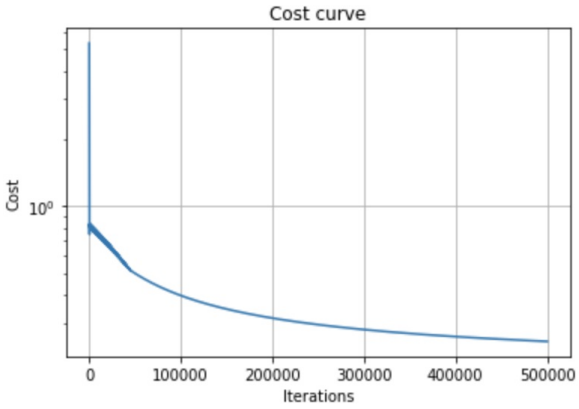


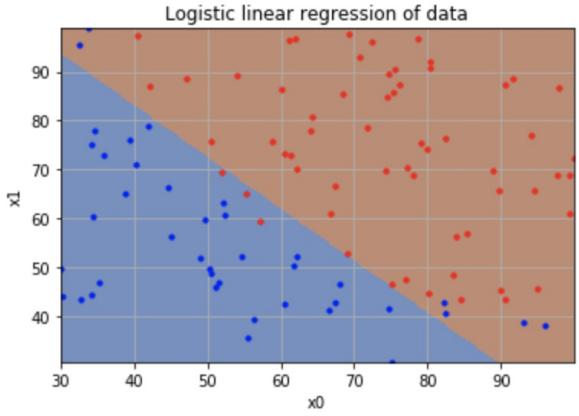

```
iterations = 500000
learn_rate = 0.0011
```

A logistic linear model was selected because the predictions to be generated should be discreet.

$$h_\theta(x) = g(\theta^T x)\,,\; g(z) = \frac{1}{1+e^{-z}}$$

I used the above hyper-parameters and obtained the following cost curve and decision boundary

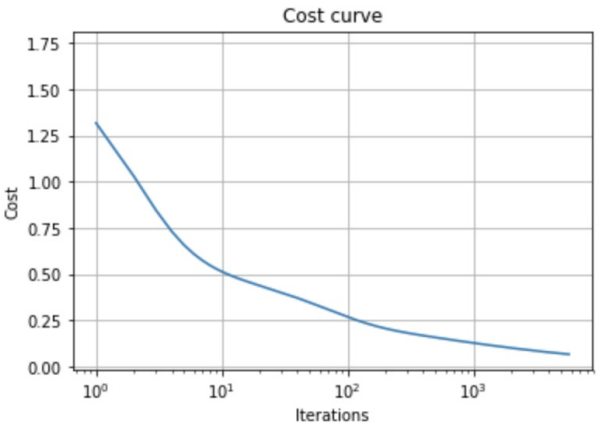


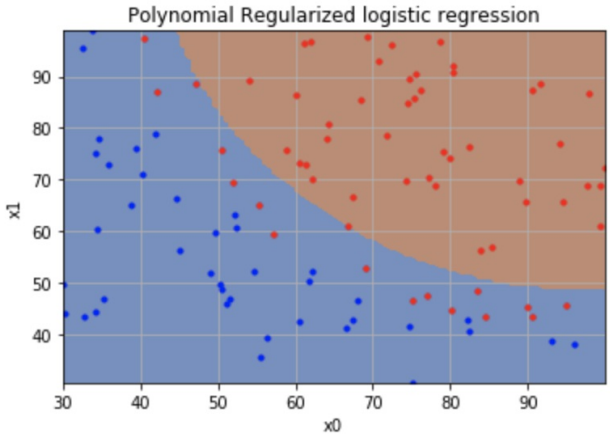


However, as the data polynomial, I expanded the input feature-set to include 5 degrees. I also normalzied the input space to the standard deviation of the input. This, enabled us to have lower iterations and a higher training rate:

```
iterations = 50000
learn_rate = 0.1
```

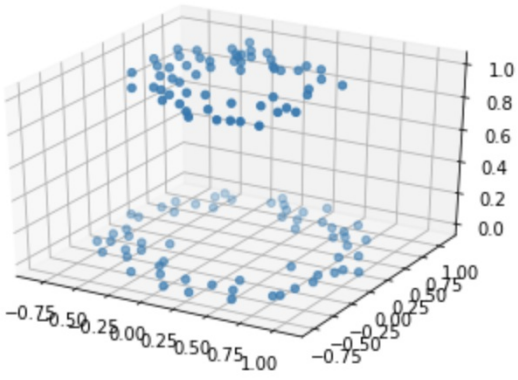
Todo: Fix the shifting of prediction.

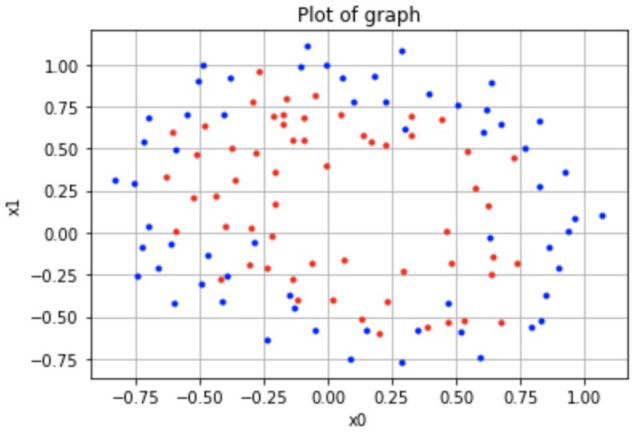




Dataset 3.

A visialization of the data provided:



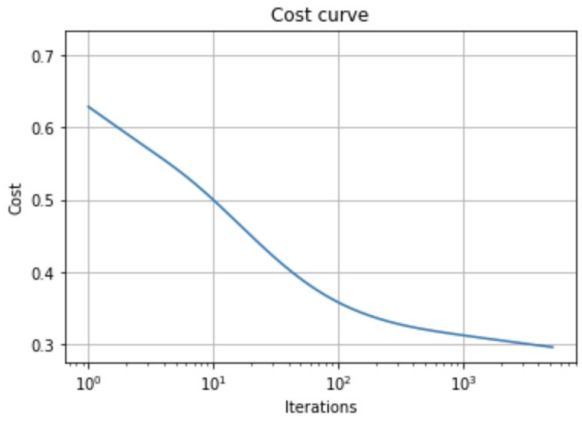


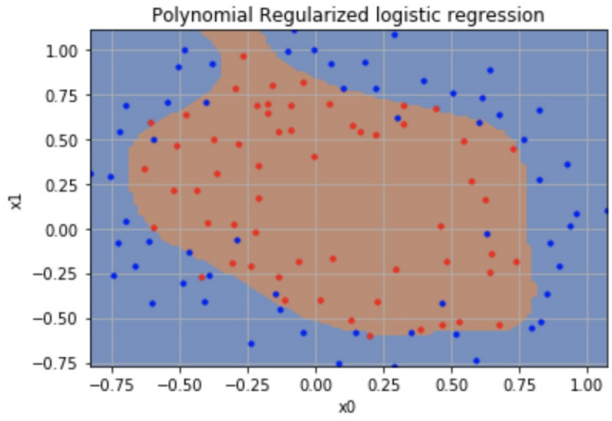
A polynomial logistic linear regression model was created due to the polynomial and discreet nature of the data.

```
poly_count = 10
iterations = 100000
learn_rate = 5
reg_lambda = 0
```

Note: The above hyper-parameters are higher than comparitive models on the same dataset because normalization was not applied in this case. By not normalizing, we used higher parameters to have similar results.

The results and cost curve:

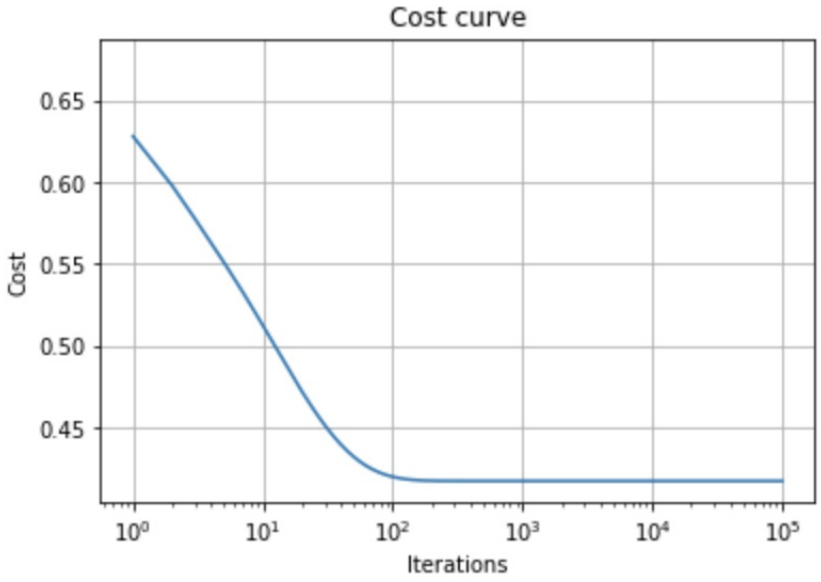


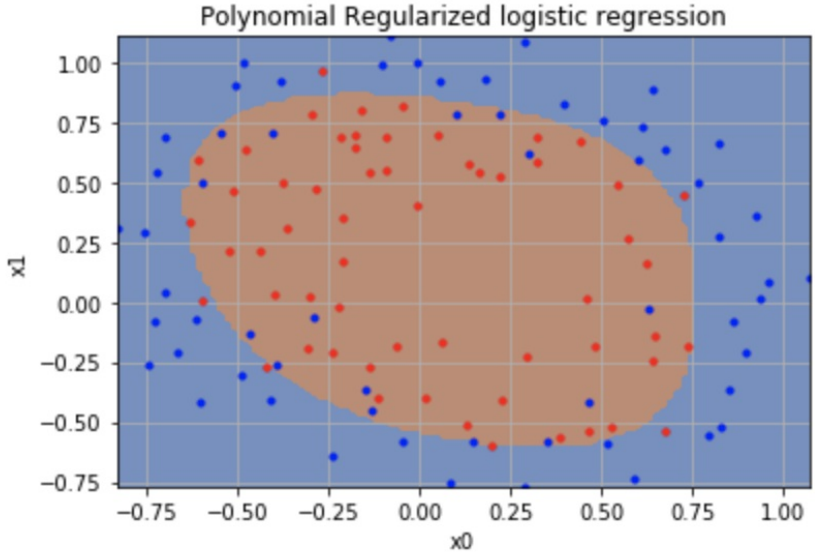


We observed that there was a lot of overfitting. So we introduced L2 regularization to our model. The following cross entropy cost equation was used.

$$J(\theta) = \frac{1}{m}\Sigma_{i=1}^{m}[-y^i log(h_\theta(x^i)) - (1-y^i)log(1-h_\theta(x^i))] + \frac{\lambda}{2\mu}\Sigma_{j=1}^{n}\theta_j^2$$

```
poly_count = 10
iterations = 100000
learn_rate = 5
reg_lambda = 0.5
```


Cost curve
Cost
Iterations
0.65
0.60
0.55
0.50
0.45
10^0
10^1
10^2
10^3
10^4
10^5


Polynomial Regularized logistic regression
x1
x0
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00