Machine Learning Assignment 2
Logistic Regression

Group Members

Ritesh Kumar Singh	2019H1030154H
Simran Batra	2019H1030024H
Parth Bhope	2019H1030023H

Dataset description and Train-Test Split

The dataset has information about bank notes authentication, there are 1372 labelled examples with the labels being either 0 or 1. There are 4 input features all have real valued numeric data.
From the entire data 80% examples were randomly selected to make the training set, and the remaining 20 % was used to test the model.

Feature Scaling

Before building the model the features were scaled using mean normalisation.

X = (X-mu) / sigma
Each training example X is transformed using the above formula where mu is the mean vector and sigma is the standard deviation vector of the training dataset.
Each testing example is also scaled using the mean and standard deviation of the training data

Approach

Our approach was to build the three models

1. Applying no regularization
2. With L1 (Lasso) regularization
3. With L2 (Ridge) regularisation

We experimented with 3 different learning rates 0.25, 0.05, 0.1 in the gradient descent step.

For the regularisation constant lambda different values were chosen for L1 and L2 regularisation based on trial and error basis.

The initial weight vector was done by the following ways

1. Randomly selecting the weights in the range [-10 10]
2. Randomly selection weights in the range [-0.5, 0.5]
3. Initializing as a unit vector ie all weights as 1
4. Initializing the weights using a normal distribution with mean as 2 and std as 0.5

The training data was used to obtain the best possible vector for classification using stochastic gradient descent to reach as close as possible to the minima.
The lambda for lasso was 0.001 and for ridge 0.005

Results

Case 1: Randomly selecting the weights in the range [-10 10]
First value is the accuracy and second value is the F Score

	L rate = 0.25	L rate = 0.05	L rate = 0.1
No regularization	0.9781 , 0.9804	0.8577, 0.8669	0.9635 ,0.9673
Ridge	0.7774 ,0.7798	0.6606, 0.6265	0.708 ,0.697
Lasso	0.9781, 0.9804	0.8467, 0.8562	0.9599, 0.9642

Case 2 : Randomly selection weights in the range [-0.5, 0.5]

	L rate = 0.25	L rate = 0.05	L rate = 0.1
No regularization	0.9781 0.9804	0.9599 0.9642	0.9781 0.9804
Ridge	0.9781 0.9804	0.9526 0.9592	0.9562 0.9618
Lasso	0.9781 0.9804	0.9599 0.9642	0.9781 0.9804

Case 3 : Initializing as a unit vector ie all weights as 1

	L rate = 0.25	L rate = 0.05	L rate = 0.1
No regularization	0.9781 0.9804	0.9526 0.9579	0.9708 0.974
Ridge	0.9526 0.9579	0.7518 0.7639	0.9051 0.9156
Lasso	0.9781 0.9804	0.9526 0.9579	0.9745 0.9772

Case 4 : Initializing the weights using a normal distribution with mean as 2 and std 0.5

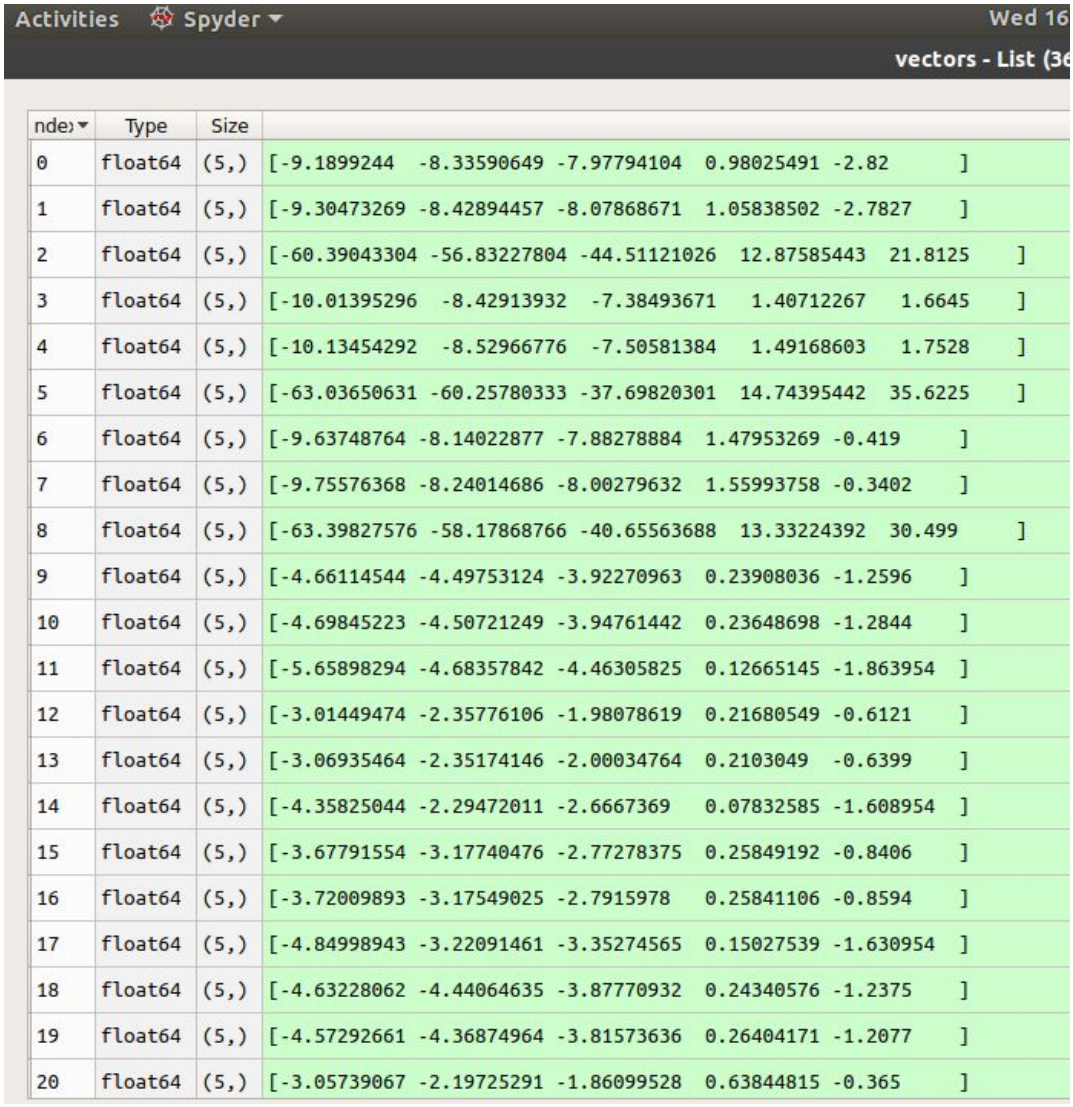
	L rate = 0.25	L rate = 0.05	L rate = 0.1
No regularization	0.9781 0.9804	0.9453 0.9521	0.9672 0.9707
Ridge	0.9124 0.9231	0.4745 0.4627	0.646 0.6667
Lasso	0.9781 0.9804	0.9416 0.9487	0.9672 0.9707

Inference

With learning rate as low as 0.05 we got low accuracy and with 0.1 accuracy improved a little bit , maximum accuracy was obtained when learning rate was 0.25, similar trend was observed in F Scores. With lasso regression the accuracy decreased slightly, ridge did not perform as good as lasso.

The most important feature

On observing the weights of the classification vectors the most important feature is the one which has the highest absolute value.
We have obtained 36 weight vectors in our experiments with 3 different learning rates and 4 different initialisation methods and 3 different regularisations
We can claim that the 0th feature i.e feature w in the original dataset is the most important feature as it has the highest weight among all other weights.
Please refer to the figure below



nde›	Type	Size	
0	float64	(5,)	[-9.1899244 -8.33590649 -7.97794104 0.98025491 -2.82]
1	float64	(5,)	[-9.30473269 -8.42894457 -8.07868671 1.05838502 -2.7827]
2	float64	(5,)	[-60.39043304 -56.83227804 -44.51121026 12.87585443 21.8125]
3	float64	(5,)	[-10.01395296 -8.42913932 -7.38493671 1.40712267 1.6645]
4	float64	(5,)	[-10.13454292 -8.52966776 -7.50581384 1.49168603 1.7528]
5	float64	(5,)	[-63.03650631 -60.25780333 -37.69820301 14.74395442 35.6225]
6	float64	(5,)	[-9.63748764 -8.14022877 -7.88278884 1.47953269 -0.419]
7	float64	(5,)	[-9.75576368 -8.24014686 -8.00279632 1.55993758 -0.3402]
8	float64	(5,)	[-63.39827576 -58.17868766 -40.65563688 13.33224392 30.499]
9	float64	(5,)	[-4.66114544 -4.49753124 -3.92270963 0.23908036 -1.2596]
10	float64	(5,)	[-4.69845223 -4.50721249 -3.94761442 0.23648698 -1.2844]
11	float64	(5,)	[-5.65898294 -4.68357842 -4.46305825 0.12665145 -1.863954]
12	float64	(5,)	[-3.01449474 -2.35776106 -1.98078619 0.21680549 -0.6121]
13	float64	(5,)	[-3.06935464 -2.35174146 -2.00034764 0.2103049 -0.6399]
14	float64	(5,)	[-4.35825044 -2.29472011 -2.6667369 0.07832585 -1.608954]
15	float64	(5,)	[-3.67791554 -3.17740476 -2.77278375 0.25849192 -0.8406]
16	float64	(5,)	[-3.72009893 -3.17549025 -2.7915978 0.25841106 -0.8594]
17	float64	(5,)	[-4.84998943 -3.22091461 -3.35274565 0.15027539 -1.630954]
18	float64	(5,)	[-4.63228062 -4.44064635 -3.87770932 0.24340576 -1.2375]
19	float64	(5,)	[-4.57292661 -4.36874964 -3.81573636 0.26404171 -1.2077]
20	float64	(5,)	[-3.05739067 -2.19725291 -1.86099528 0.63844815 -0.365]

Activities Spyder

vectors -

nde	Type	Size	
15	float64	(5,)	[-3.67791554 -3.17740476 -2.77278375 0.25849192 -0.8406]
16	float64	(5,)	[-3.72009893 -3.17549025 -2.7915978 0.25841106 -0.8594]
17	float64	(5,)	[-4.84998943 -3.22091461 -3.35274565 0.15027539 -1.630954]
18	float64	(5,)	[-4.63228062 -4.44064635 -3.87770932 0.24340576 -1.2375]
19	float64	(5,)	[-4.57292661 -4.36874964 -3.81573636 0.26404171 -1.2077]
20	float64	(5,)	[-3.05739067 -2.19725291 -1.86099528 0.63844815 -0.365]
21	float64	(5,)	[-2.94587606 -2.07040435 -1.74533737 0.2810794 -0.5775]
22	float64	(5,)	[-2.90255998 -1.99599654 -1.68301373 0.31013741 -0.5447]
23	float64	(5,)	[-1.92429357 1.13469565 1.14484604 1.6366916 0.1865]
24	float64	(5,)	[-3.63862643 -3.02994278 -2.6541998 0.28784629 -0.806]
25	float64	(5,)	[-3.59554191 -2.9627596 -2.59608171 0.31100818 -0.7792]
26	float64	(5,)	[-2.33711441 -0.59691951 -0.4853458 1.05659218 -0.116]
27	float64	(5,)	[-4.61282138 -4.4035067 -3.8500109 0.24885507 -1.2219]
28	float64	(5,)	[-4.56453644 -4.33643662 -3.79123224 0.26508511 -1.1921]
29	float64	(5,)	[-2.40862508 -1.50443902 -1.15909994 0.72442594 -0.330156]
30	float64	(5,)	[-2.92113586 -1.91093134 -1.61530502 0.32419461 -0.5634]
31	float64	(5,)	[-2.87720263 -1.83651782 -1.55624312 0.35773012 -0.5366]
32	float64	(5,)	[-1.59526038 3.62457435 3.53007283 2.81250129 0.078844]
33	float64	(5,)	[-3.61830954 -2.95092979 -2.5941803 0.30109666 -0.7914]
34	float64	(5,)	[-3.58019582 -2.88493181 -2.53525896 0.3218308 -0.7626]
35	float64	(5,)	[-1.8455325 0.55260317 0.66381017 1.56518063 -0.216656]