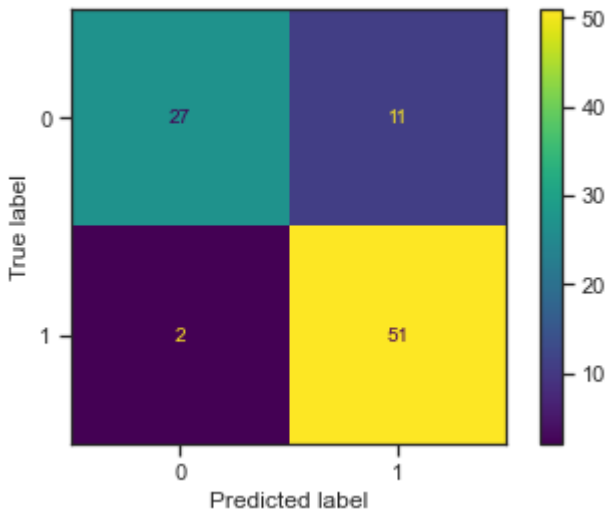
# Experiment: Classification Problem

## 1. Using Logistic Regression without regularization:

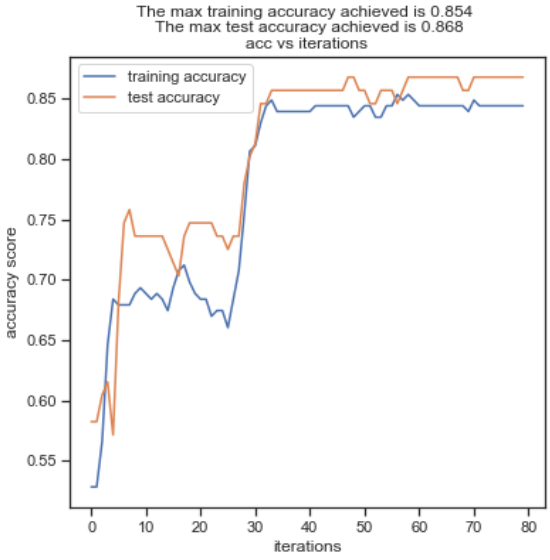
- This is an example of binary classification with two target classes: 0,1
- Dataset has 303 training examples, 13 features.
- Data is stored as a DataFrame.
- Train-Test-split is done.
- **Initially a Logistic Regression model is implemented without regularization with no.of iterations equal to 150. The observations recorded are:**
    - Training accuracy : 0.844
    - Test accuracy score is 0.857
    - f1_score is 0.887
    - Precision is 0.823
    - Recall is 0.962
    - Confusion Matrix ( y = 1 is taken as Positive case):

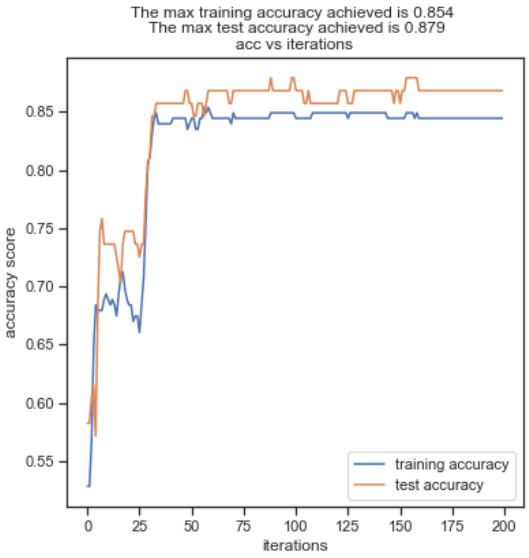



**Here** the dataset is the **Balanced-type dataset.**

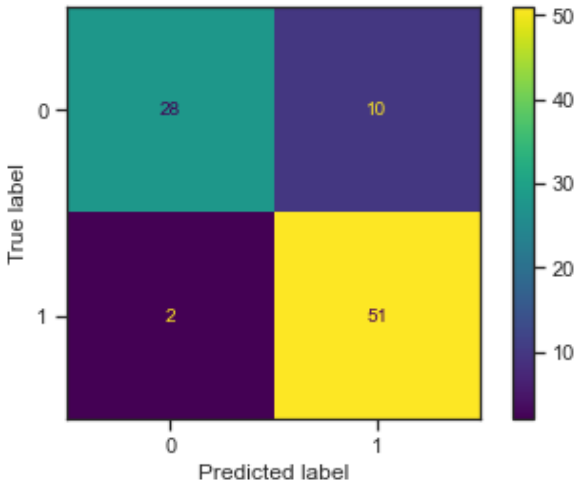
# 2. Tweaking the parameters:

- For different values of iterations ( upto 200 ), it was observed that max training and test accuracy is achieved at (approximately) 75 iterations, so optimal value of max-iter is **75**.

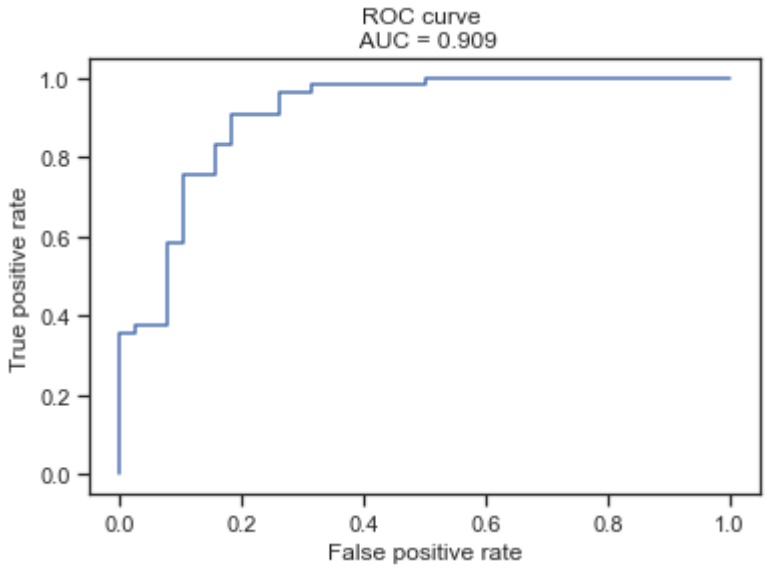


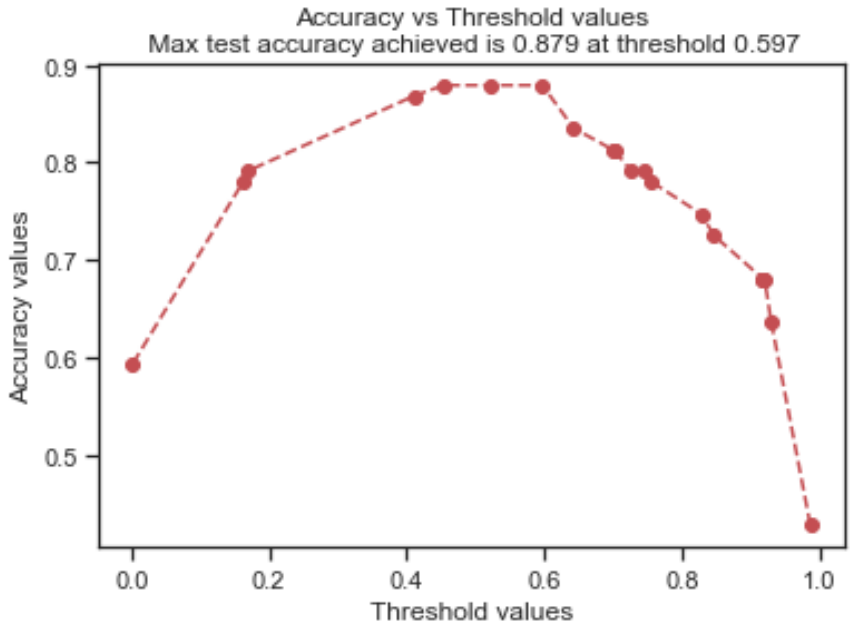

- **Therefore,** the final observations made for **iterations = 75** are**:**
  - Training accuracy : 0.844
  - Test accuracy score is 0.868
  - f1_score is 0.895
  - Precision is 0.836
  - Recall is 0.962
  - Confusion Matrix ( y = 1 is taken as Positive case):



- The **ROC curve** for this model is:




- **observe accuracy by varying threshold :**

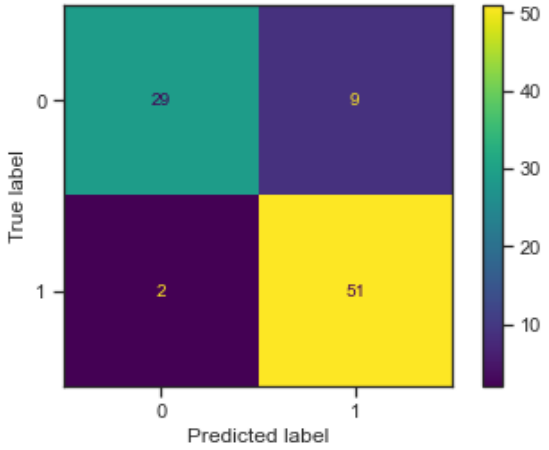



- **Its observed that around *<u>iterations = 75</u>* and *<u>threshold = 0.597</u>*, we have *<u>maximum test accuracy of 0.879</u>*.**

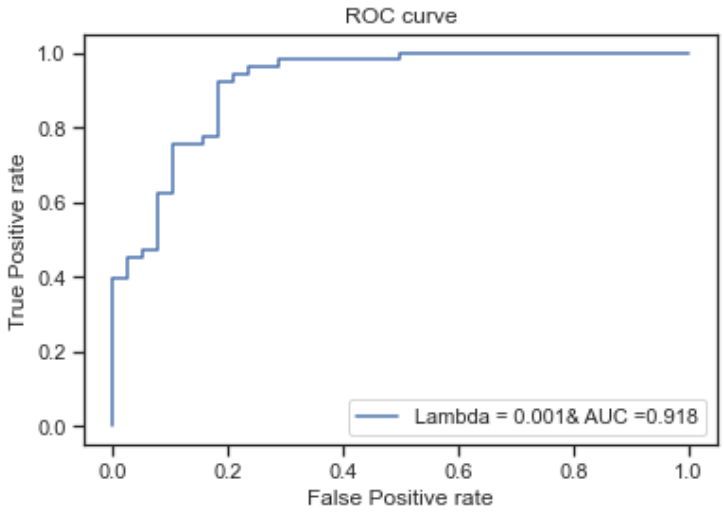
# 3. Using Logistic Regression with regularization:

- A Logistic Regression with regularization is used with regularization parameter , **Lambda** , = **0.001** and selecting **L2 regularization.** The observations are**:**

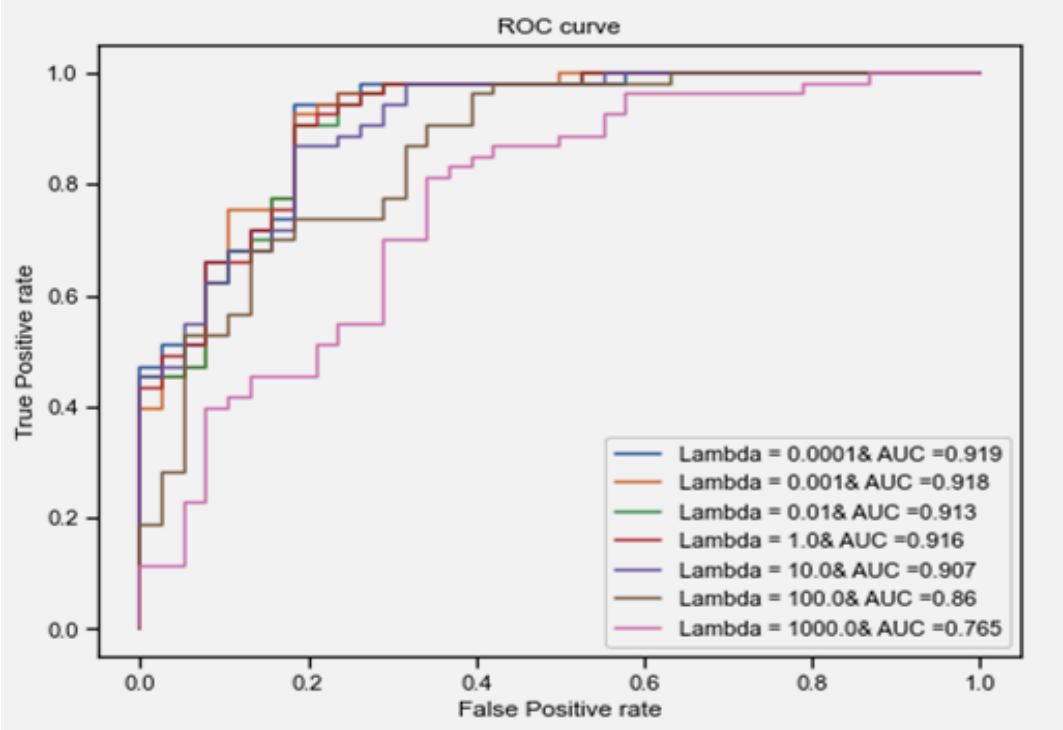
  - training accuracy :0.854
  - test accuracy : 0.879
  - f1_score is 0.903
  - Precision is 0.85
  - Recall is 0.962
  - Confusion Matrix :




  - ROC curve :



# 4. Logistic Regression with different lambda values:



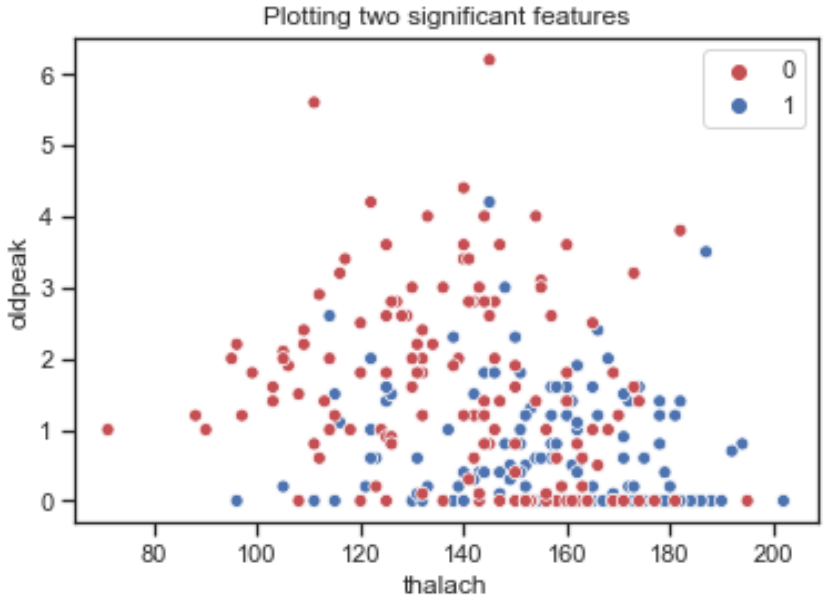

- As the lambda values are increased, from above graph, it is observed that the accuracy decreases which is due to the fact that by increasing lambda, the model becomes more simpler and underfits the data, thus cannot learn the complex features in dataset and therefore the accuracy decreases.

# 5. Plotting the decision boundary for different lambda values:

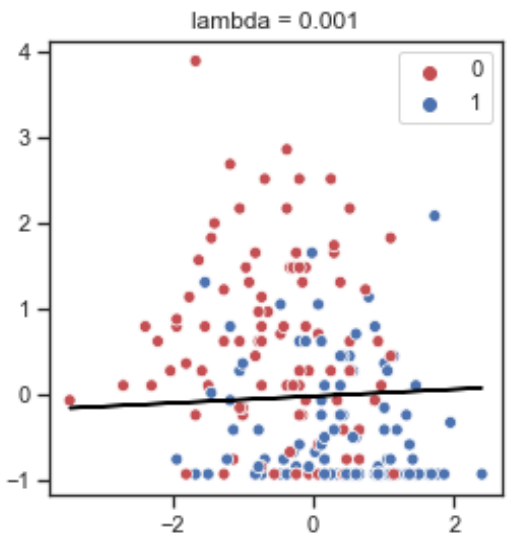
- To draw a decision, we have to select two features which are strongly correlated to the target.

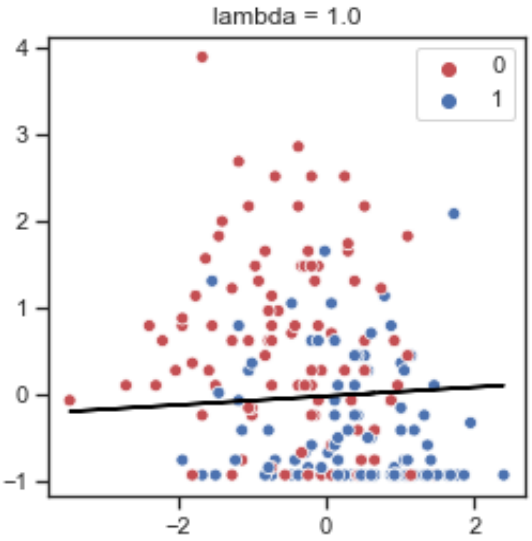
| | age | sex | cp | trestbps | chol | fbs | restecg | thalach | exang | oldpeak | slope | ca | thal | target |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **age** | 1.000000 | -0.098447 | -0.068653 | 0.279351 | 0.213678 | 0.121308 | -0.116211 | -0.398522 | 0.096801 | 0.210013 | -0.168814 | 0.276326 | 0.068001 | -0.225439 |
| **sex** | -0.098447 | 1.000000 | -0.049353 | -0.056769 | -0.197912 | 0.045032 | -0.058196 | -0.044020 | 0.141664 | 0.096093 | -0.030711 | 0.118261 | 0.210041 | -0.280937 |
| **cp** | -0.068653 | -0.049353 | 1.000000 | 0.047608 | -0.076904 | 0.094444 | 0.044421 | 0.295762 | -0.394280 | -0.149230 | 0.119717 | -0.181053 | -0.161736 | 0.433798 |
| **trestbps** | 0.279351 | -0.056769 | 0.047608 | 1.000000 | 0.123174 | 0.177531 | -0.114103 | -0.046698 | 0.067616 | 0.193216 | -0.121475 | 0.101389 | 0.062210 | -0.144931 |
| **chol** | 0.213678 | -0.197912 | -0.076904 | 0.123174 | 1.000000 | 0.013294 | -0.151040 | -0.009940 | 0.067023 | 0.053952 | -0.004038 | 0.070511 | 0.098803 | -0.085239 |
| **fbs** | 0.121308 | 0.045032 | 0.094444 | 0.177531 | 0.013294 | 1.000000 | -0.084189 | -0.008567 | 0.025665 | 0.005747 | -0.059894 | 0.137979 | -0.032019 | -0.028046 |
| **restecg** | -0.116211 | -0.058196 | 0.044421 | -0.114103 | -0.151040 | -0.084189 | 1.000000 | 0.044123 | -0.070733 | -0.058770 | 0.093045 | -0.072042 | -0.011981 | 0.137230 |
| **thalach** | -0.398522 | -0.044020 | 0.295762 | -0.046698 | -0.009940 | -0.008567 | 0.044123 | 1.000000 | -0.378812 | -0.344187 | 0.386784 | -0.213177 | -0.096439 | 0.421741 |
| **exang** | 0.096801 | 0.141664 | -0.394280 | 0.067616 | 0.067023 | 0.025665 | -0.070733 | -0.378812 | 1.000000 | 0.288223 | -0.257748 | 0.115739 | 0.206754 | -0.436757 |
| **oldpeak** | 0.210013 | 0.096093 | -0.149230 | 0.193216 | 0.053952 | 0.005747 | -0.058770 | -0.344187 | 0.288223 | 1.000000 | -0.577537 | 0.222682 | 0.210244 | -0.430696 |
| **slope** | -0.168814 | -0.030711 | 0.119717 | -0.121475 | -0.004038 | -0.059894 | 0.093045 | 0.386784 | -0.257748 | -0.577537 | 1.000000 | -0.080155 | -0.104764 | 0.345877 |
| **ca** | 0.276326 | 0.118261 | -0.181053 | 0.101389 | 0.070511 | 0.137979 | -0.072042 | -0.213177 | 0.115739 | 0.222682 | -0.080155 | 1.000000 | 0.151832 | -0.391724 |
| **thal** | 0.068001 | 0.210041 | -0.161736 | 0.062210 | 0.098803 | -0.032019 | -0.011981 | -0.096439 | 0.206754 | 0.210244 | -0.104764 | 0.151832 | 1.000000 | -0.344029 |
| **target** | -0.225439 | -0.280937 | 0.433798 | -0.144931 | -0.085239 | -0.028046 | 0.137230 | 0.421741 | -0.436757 | -0.430696 | 0.345877 | -0.391724 | -0.344029 | 1.000000 |

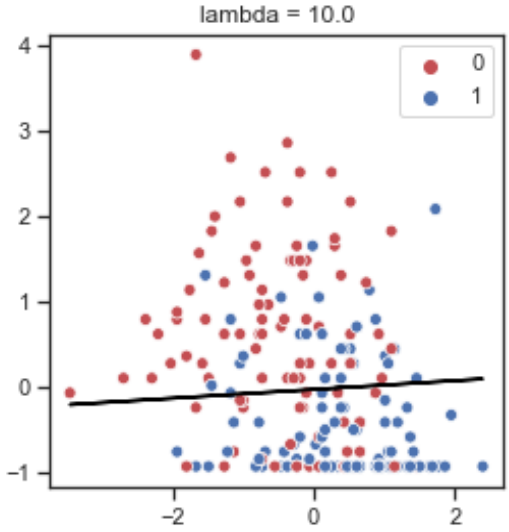
- From above table, Two features selected are ' *thalach* ' and ' *oldpeak* '

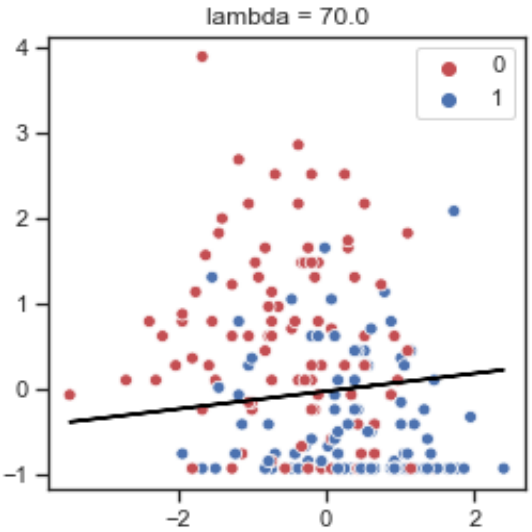


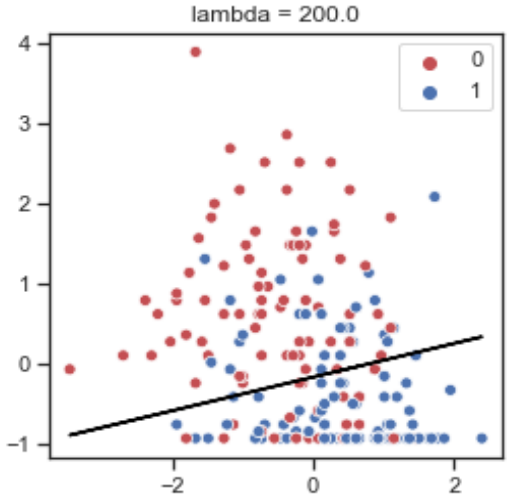

- As our objective is to obtain the optimal weights for the whole dataset and therefore plotting the decision boundary by using only two features

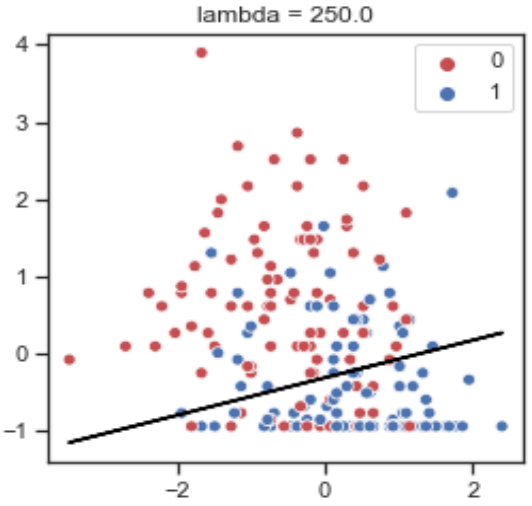

lambda = 0.001
0
1
4
3
2
1
0
−1
−2
0
2

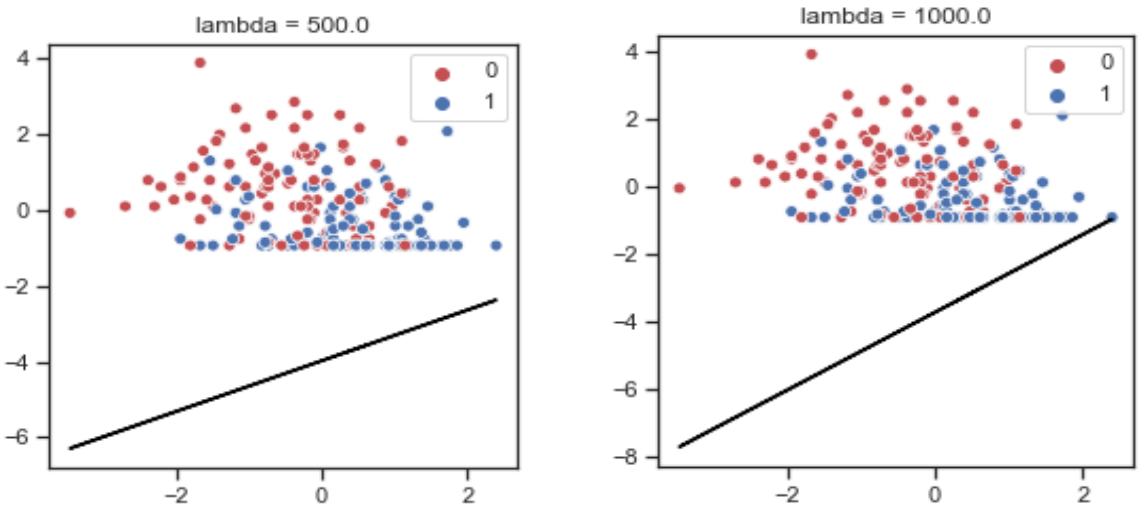

lambda = 1.0
0
1
4
3
2
1
0
−1
−2
0
2


lambda = 10.0
0
1
4
3
2
1
0
−1
−2
0
2


lambda = 70.0
0
1
4
3
2
1
0
−1
−2
0
2


lambda = 200.0
0
1
4
3
2
1
0
−1
−2
0
2


lambda = 250.0
0
1
4
3
2
1
0
−1
−2
0
2




- The data points above decision boundary are classified as class 0 whereas those below decision boundary are classified as class 1.
- From the observations, its clear that for small lambda values the model works fine as it doesnot overfits the data, so the accuracy for unregularised model and regularised model with small lamda vales have approximately equal accuracy values and therefore, the decision boundary for the these models remain approximately the same
- For higher lamda values, its clear that model **underfits** the data and thus it classifies very poorly and simply classifies all the examples to some particular target class. For example, for lambda = [500, 1000] , model simply classifies each and every example as class 0.