CSE 574 Intro to Machine Learning, PA1
Group No. 36
Name: Sreya Dhar
Vaishnavi Vukku
Sai Lakshmi Navya Maddu

Report 1. Ordinary Least-square Estimation has been performed on the diabetes dataset. The dataset has been predefined into train and test sets. RMSE without intercept on the training data is 138.20. RMSE with an intercept on train data is 46.77. RMSE without intercept on test data is 326.76. And RMSE with an intercept on test data is 60.89. RMSE with intercept/bias term is providing better results on both train and test sets.

Report 2. Gradient Descent algorithm has been applied on Diabetes dataset to minimize the loss function. The weights calculated using this gradient-based learning are then used for making predictions on the test data and the Root Mean square error(RMSE) is calculated. Gradient Descent Linear Regression RMSE on train data is 48.11. Gradient Descent Linear Regression RMSE on test data is 54.70.

By comparing the results with RMSE obtained from direct minimization, we can say that better predictions are obtained from the Gradient descent algorithm since the RMSE on the test data is low (~54.70) compared to direct minimization(~60.89).

Report 3. The perceptron model has been trained by calling the scipy.optimize.minimize method, and the evaluateLinearModel function has been used to calculate the accuracy for the training and test data. Using gradient descent, the perceptron model has an accuracy of 0.84 on both train and test data.

Report 4. We are trying to minimize the objective function "logisticObjVal" that takes (X, y, w) as inputs and returns the error. Following that, LogisticGradient and Logistic Hessian functions are returning gradient as a vector and hessian as a matrix. The accuracy of logistics regression for training data is 0.86 and for test data is 0.86. The logistic regression performed better than the perceptron model and ordinary least square estimation algorithm.

Report 5. Sample pickle data is trained using the SVM model by calling the trainSGDSVM method for 200 iterations by setting the linear rate parameter eta to 0.01. The weight vector is updated at each iteration based on the data points obtained from random sampling. By using evaluateLinearModel with the obtained weights, we get SVM Accuracy of 0.85 and 0.87 on train and test data, respectively.

Report 6.1 There is no significant difference in the Perceptron, Logistic, and SVM models' performance as the data is linearly separable. Logistic regression and SVM have consistently performed better than the Perceptron Learning. There is a random sampling of datapoints in the SVM model, which has caused the model to fall behind the Logistic regression at times. In most cases, *SVM outperforms logistic regression and perceptron models.*

Report 6.2 The weight vectors from all three models have been given in Table 1. Classification boundaries from these classifiers on train data have been given in Fig. 1. There is very marginal difference among the boundaries of the classifiers and the accuracy metrics are also very comparable for these classifiers. Having said that, SVM tries to find the best margins to separate the classes; the misclassification error is comparatively less in the SVM model. Perceptron and Logistic regression perform in a similar manner and

have classification accuracy of 0.84 and 0.86, respectively, on both train and test sets. The classification boundaries for these two models are likewise. The intercept of the classification boundary from the perceptron model is negative (-0.09), and for logistic regression and SVM, the intercept is positive (0.2 and 0.1).

Table 1 Weights obtained from three models	
Models	**Weight vectors**
Perceptron	[-0.0881355,0.28506787,-0.44939862]
Logistic	[0.1966793,0.56303967,-1.15204657]
SVM (sample weights)	[0.1, 0.32817734,-0.82734603]

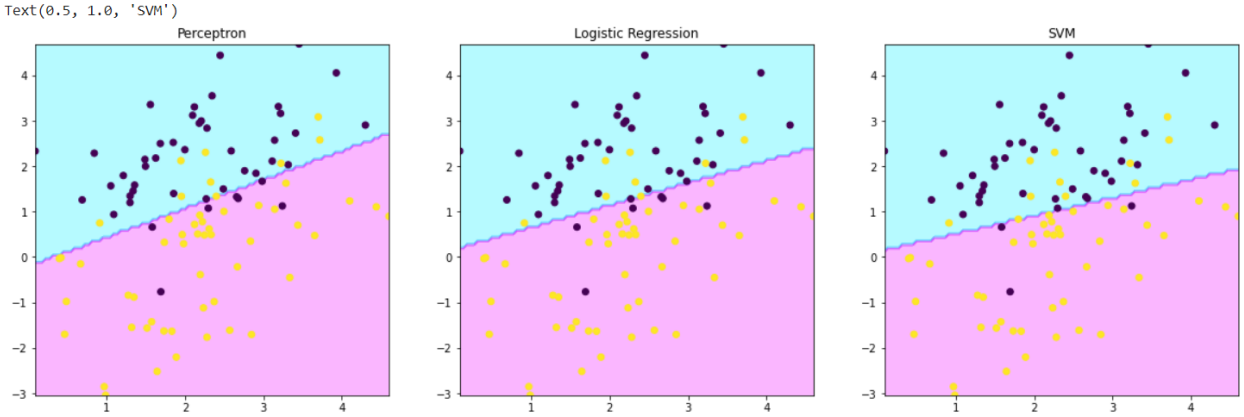

Fig. 1 Comparison of three different classifiers on the train set