**Team Members:**

*Sanjay Reddy Banda, UFID: 5878-2239*

*Chandra Sekhar Deverakonda, UFID:*

*Suprith Reddy Gurudu, UFID: 9961-2134*

**Implementation:**

As mentioned in the project description, we have used three classifier models namely K-Nearest Neighbors, Multi-layer Perceptron, and Support Vector Machine on all three datasets and three regressor models includes K-Nearest Neighbors, Multi-layer perceptron, and Linear Regression on the intermediate (multi-label) dataset only. We have calculated the accuracies and normalized confusion matrices using the predicted and true labels from the models. A detailed explanation is provided below for each model.

***Multi-layer Perceptron (MLP) Classifier:***

***Multi-layer Perceptron (MLP) Regressor:***

***K-Nearest Neighbors (KNN) Classifier:***

***K-Nearest Neighbors (KNN) Regressor:***

***Support Vector Machine (SVM) Classifier:***

An SVM is a supervised machine learning model used for classification tasks by constructing a hyperplane or set of hyperplanes that separates the data into classes.

Prior to model building, we have loaded the data as features and labels, assigned the test size to 20% of original dataset, applied a function to split the data into train and test subsets, and initialized k-fold strategy to 10 to overcome the challenge of overfitting.

Once the preprocessing steps are completed, the model is built based on the hyper parameters passed into the SVM classifier function. Among the parameters, kernel is a significant one which is set to ‘linear’ (final dataset) and ‘rbf’ (intermediate datasets) to reduce the computations involved in higher dimensions and get better predictions.

Finally, the predicted labels and accuracy is calculated using *classifier.predict()* and *classifier.score()* functions along with the confusion matrix.

***Linear Regressor using normal equations:***

The objective of a linear regression function is to fit the best line/hyper-plane to a set of observations by computing the parameters that minimizes the cost function, for instance, mean squared error (MSE).

Prior to model building, we have loaded the data as features and labels, assigned the test size to 20% of original dataset, applied a function to split the data into train and test subsets, and initialized k-fold strategy to 10 to overcome the challenge of overfitting.

After preprocessing, the procedure of applying regression on the training set involves five different steps: (i) transposing the training feature matrix, (ii) multiplying the transposed matrix with the original training feature set, (iii) inverting the product from the previous step, (iv) multiplying the inverted matrix with transposed matrix, and (v) multiplying the resultant from previous step to the training label vector that produces theta which describes slope and intercept of the line/hyper-plane.

Finally, the predicted labels are calculated by multiplying theta with testing sample of features. As the predicted values are continuous, we cannot define accuracy like classification. So, we have rounded off the predicted values to the nearest integer that further used to compute the accuracy using *metrics.accuracy\_score()* function.

**Evaluation:**

***Accuracy and normalized confusion matrices:***

Table

Description automatically generated

***Best methods in classification and regression:***

On final dataset, KNN classifier and regressor works best as the accuracy reaches 100 and 93 percent, respectively. Similarly, SVM classifier is the best method applied on intermediate multilabel dataset and MLP classifier on intermediate single label dataset. MLP classifier predicts more accurate values on testing samples for intermediate single label dataset and KNN regressor for intermediate multilabel dataset.

***Investigation:***

***Scaling:***