ML Project no2

Supervisor: Dr. Sajedi

Yazdan Zandiye Vakili

610397193

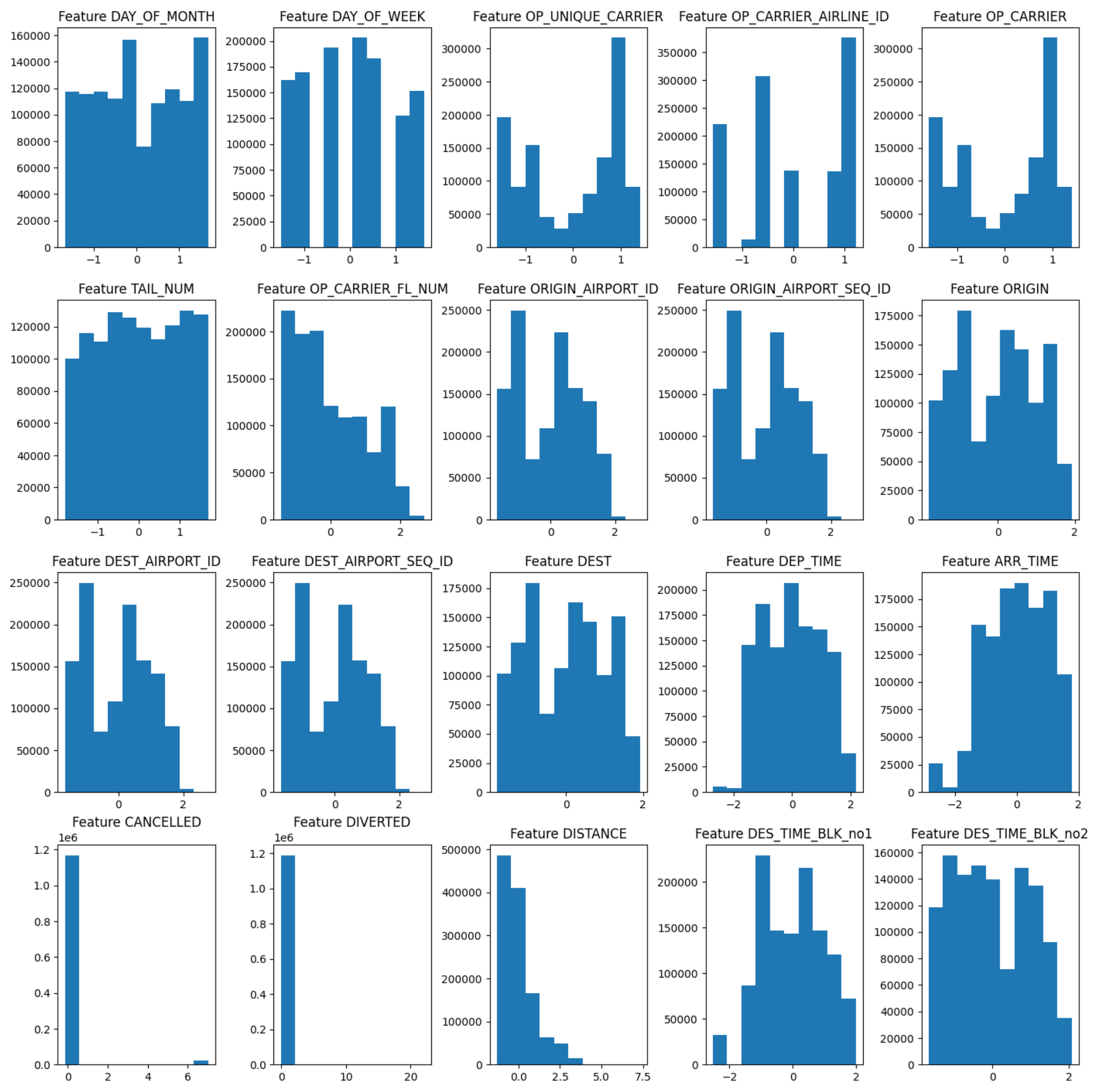
Introduction

The aim of this project is to compare some of the Deep Learning methods on a data set related to the departure and arrival time of whole 2019 & 2022 flights in whole USA. Comparing methods are MLP (Multiple Layers Perceptron), ELM (Extreme Learning Models) and a classic ML method that I chose to compare with these two methods, Decision Tree. For scaling down the data as the project overview mentioned, we need to implement an Auto Encoder for that but I implemented PCA and LDA too to compare these two to implemented Auto Encoder.

Load Data

At first with Pandas, loaded the two csv files related to 2019 and 2020 and concatenate them together to build-up our data set. Splitting the data set into train and test data done after applying preprocessing on whole data. Our labels are the combination from DEP_DEL15 and ARR_DEL15 columns that creates labels from 0 to 3 (4 labels).

Visualization of data features



Preprocessing

In this part multiple steps done like handling categorical features, handle missing values, normalization and elimination of high correlated features.

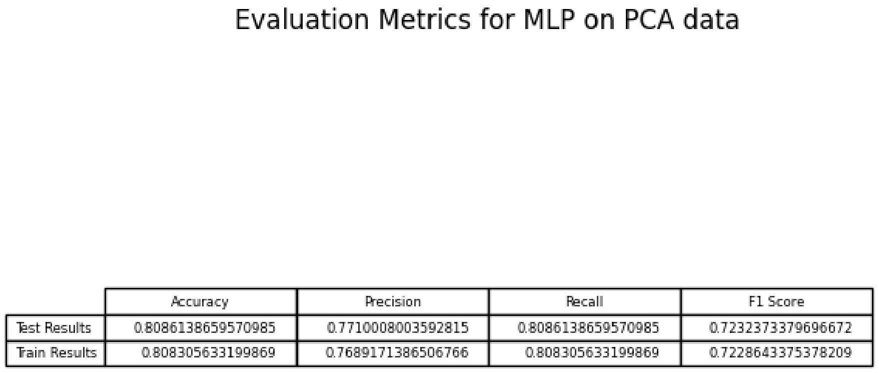
1. Handle Categorical Features
   For all the columns I used LabelEncoder in Scikit-Learn package but for column named DES_TIME_BLK separated into two different columns named DES_TIME_BLK_no1 & 2.
2. Handle Missing Values
   For this step at first as you can see in the code, I tried to apply KNN for each record that contains missing values but because of the large size of the data set, I changed the strategy to replacing missing values with mean of that feature.
3. Normalization
   For this part as always, I picked standard scalar to normalize the data, but before that I separated labels from training data.
4. Elimination of High correlated features
   In this step by plotting the heatmap of correlation matrix of data, found 4 groups of features that were highly correlated and once with PCA and another time with using LDA, scaled down each group to a new feature and replaced new features with previous groups of features.
5. Auto Encoder
   As you can see in the code, I trained 40 models before the final version of my auto encoder and ended up with 5 hidden layers, 8 as the size of my bottle neck layer, Nadam as optimizer and MSE as loss function. I provided all the tries on the code.

Note: Another way of down scaling according to project overview, is using Auto Encoder but in this method, I did not eliminate high correlated features and assigned this task to auto encoder itself. So, at the end, we ended up with 3 ways of scaled down data and then applied our classifiers on each of them.

Fitting Section

1. **MLP**
   in this section I applied MLP with 5 hidden layers (50, 100, 200, 100, 50), adaptive learning rate, Adam optimizer, Relu activation function, 0.001 as Alpha and max iteration equal to 10. As I mentioned in the code, fitting time of MLP for each model took about 20 minutes and because of that I couldn't use Grid Search or Cross Validation and manually tested each hyper parameter and after a few times ended up with these hyper parameters.

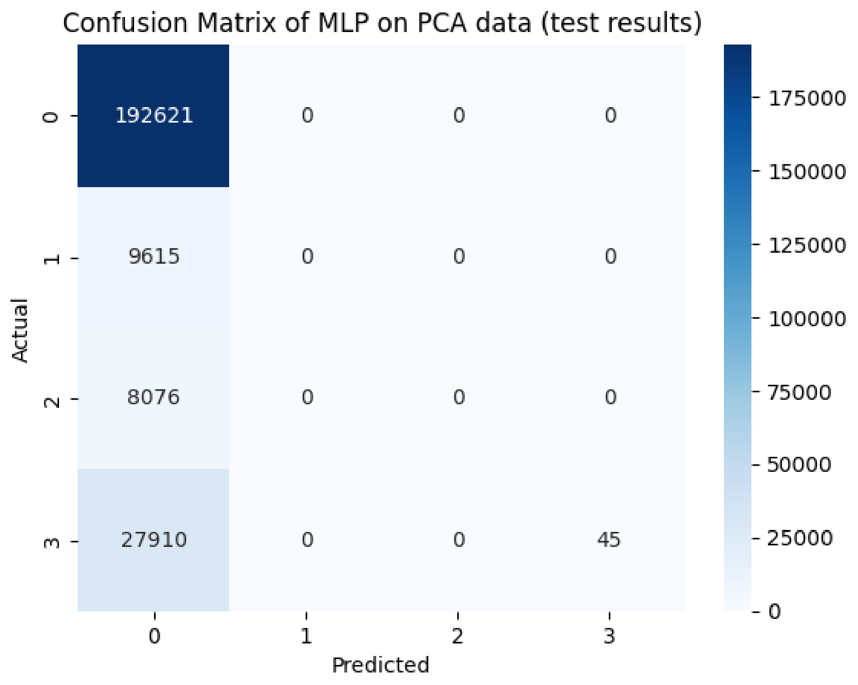
   1.1. Evaluation Metrics on PCA



Evaluation Metrics for MLP on PCA data

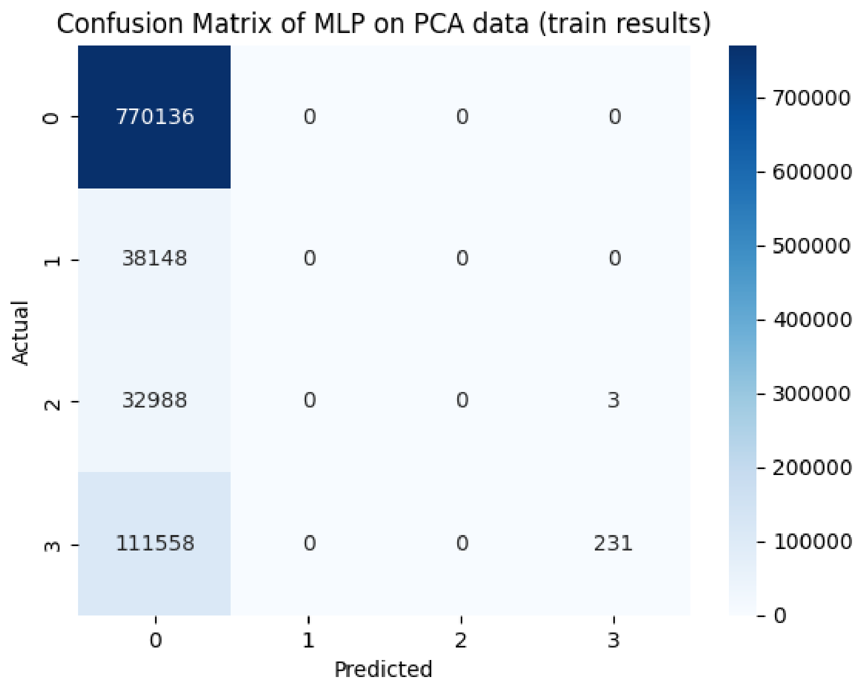
	Accuracy	Precision	Recall	F1 Score
Test Results	0.8086138659570985	0.7710008003592815	0.8086138659570985	0.7232373379696672
Train Results	0.808305633199869	0.7689171386506766	0.808305633199869	0.7228643375378209

Confusion Matrix on test data



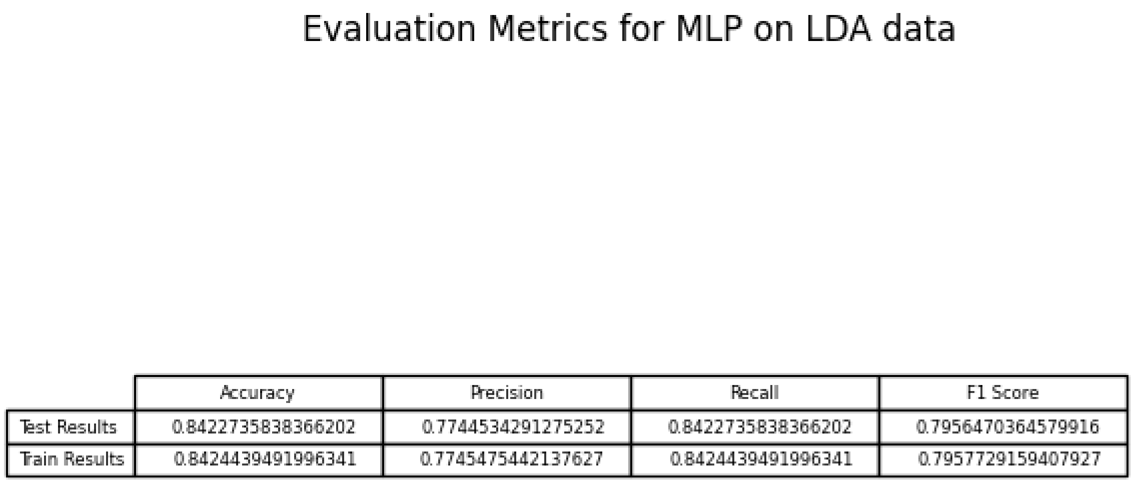
Actual \ Predicted	0	1	2	3
0	192621	0	0	0
1	9615	0	0	0
2	8076	0	0	0
3	27910	0	0	45

Confusion Matrix on train data



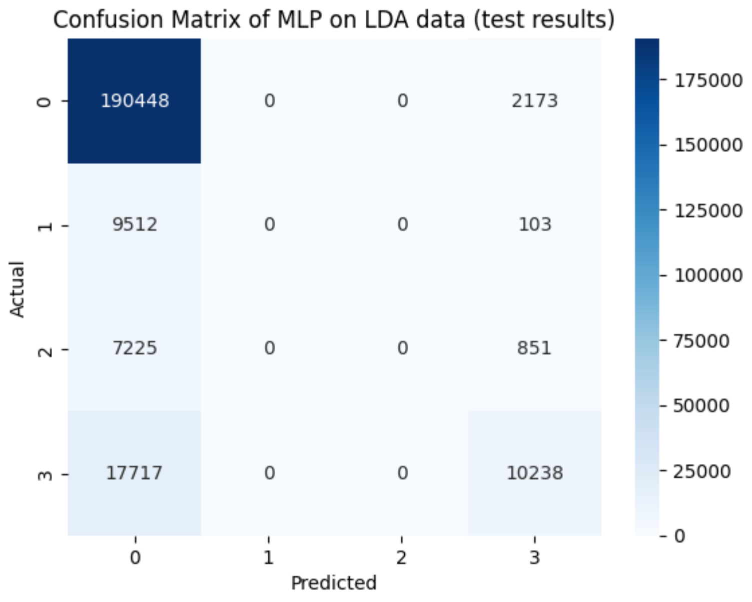
Actual \ Predicted	0	1	2	3
0	770136	0	0	0
1	38148	0	0	0
2	32988	0	0	3
3	111558	0	0	231

1.2. Evaluation Metrics on LDA

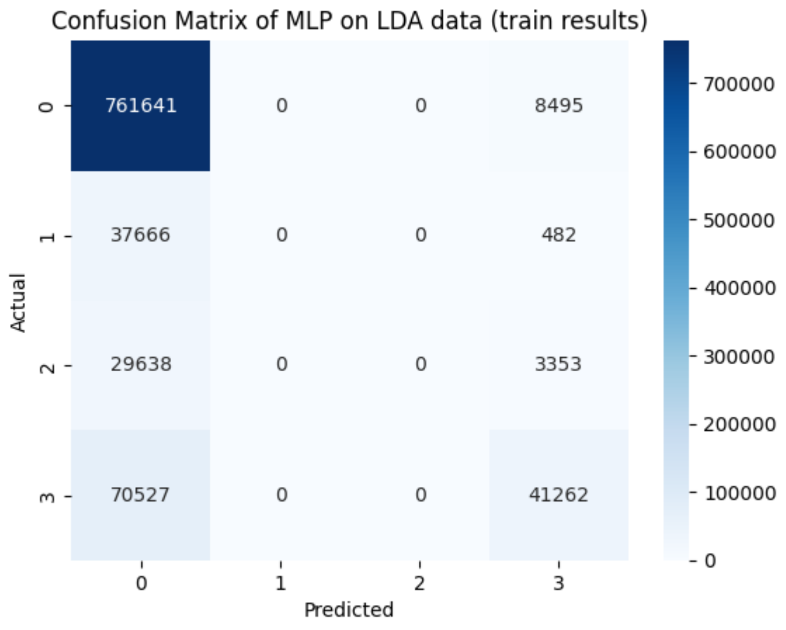

Evaluation Metrics for MLP on LDA data

	Accuracy	Precision	Recall	F1 Score
Test Results	0.8422735838366202	0.7744534291275252	0.8422735838366202	0.7956470364579916
Train Results	0.8424439491996341	0.7745475442137627	0.8424439491996341	0.7957729159407927

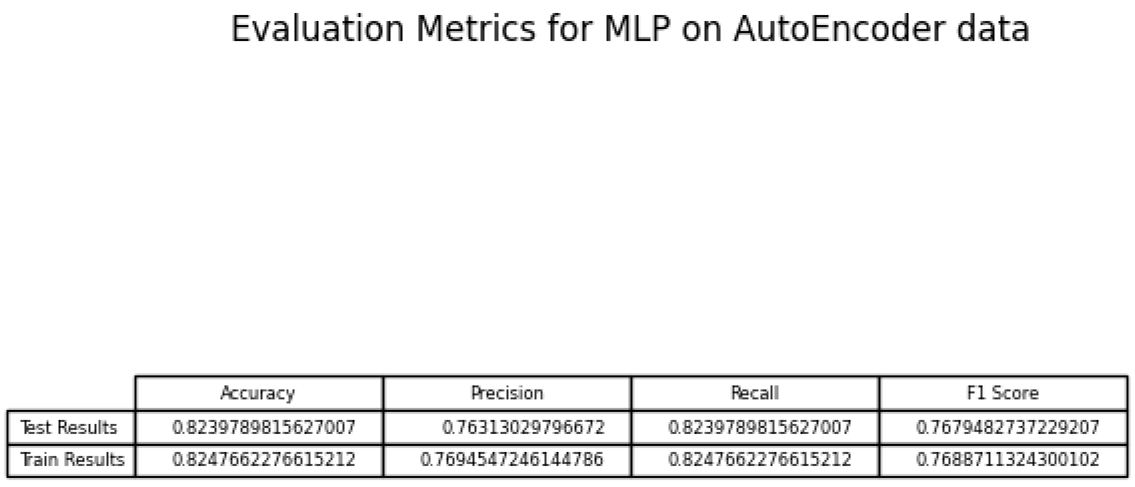
Confusion Matrix on test data



Confusion Matrix on train data

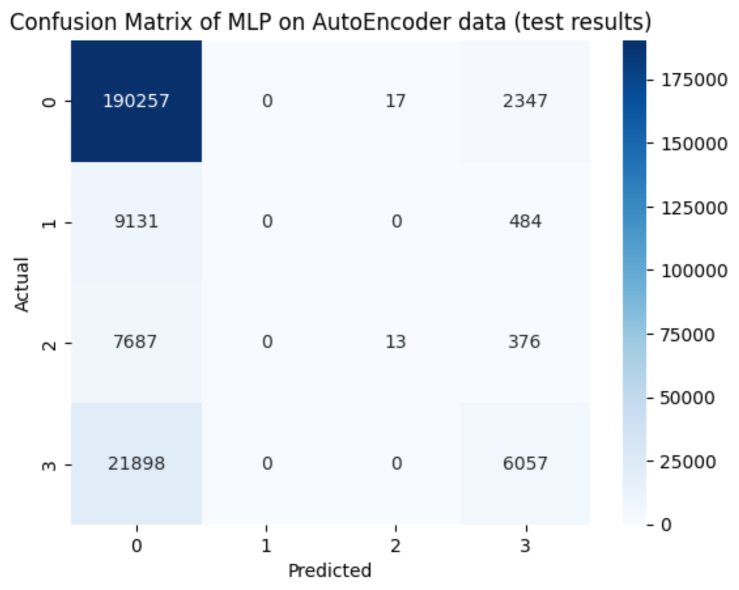


1.3. Evaluation Metrics on Auto Encoder

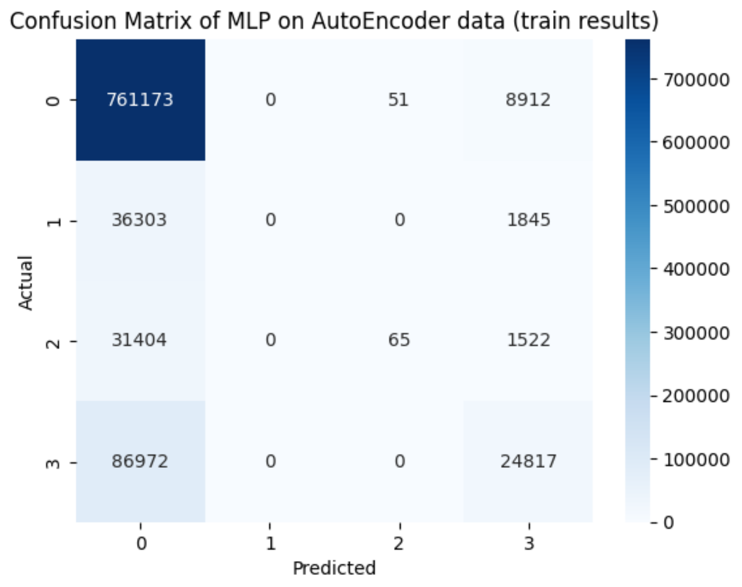

Evaluation Metrics for MLP on AutoEncoder data

	Accuracy	Precision	Recall	F1 Score
Test Results	0.8239789815627007	0.76313029796672	0.8239789815627007	0.7679482737229207
Train Results	0.8247662276615212	0.7694547246144786	0.8247662276615212	0.7688711324300102

Confusion Matrix on test data



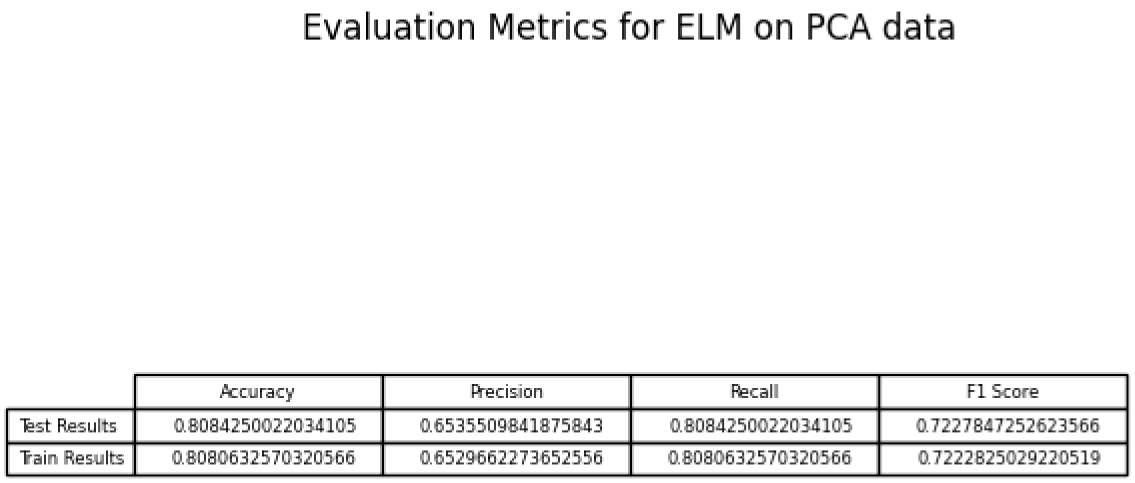
Confusion matrix on train data



2. **ELM**

In this section I used ***hpelm*** (stands for high performance extreme learning models) and fitted ELMs on my data, but before that I needed to apply one-hot encoding on my labels to pass labels to ELM instance from this library so, I did that. Here are the results.

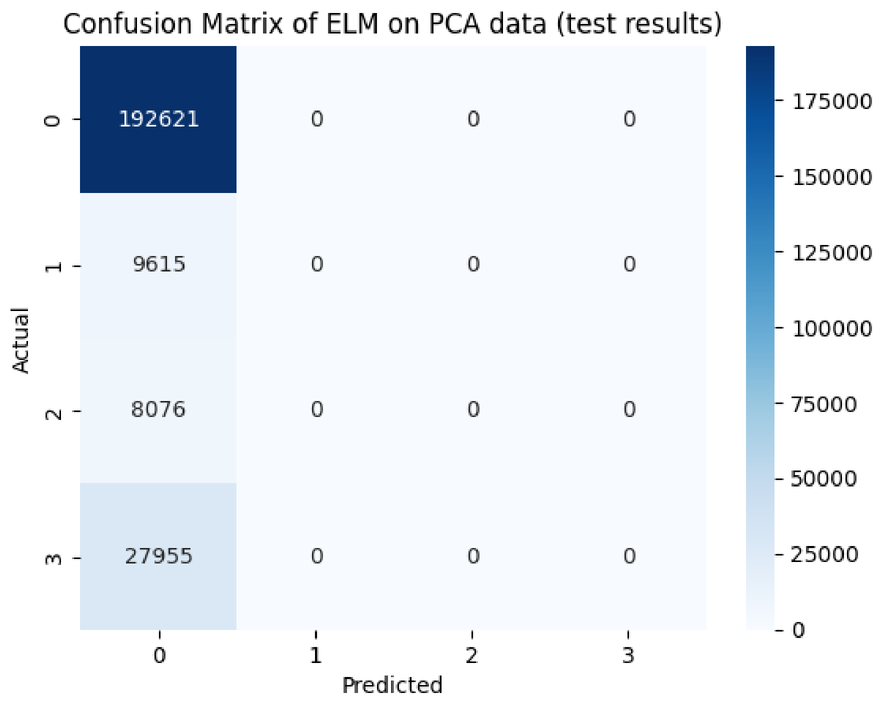
2.1. Evaluation Metrics on PCA



Evaluation Metrics for ELM on PCA data

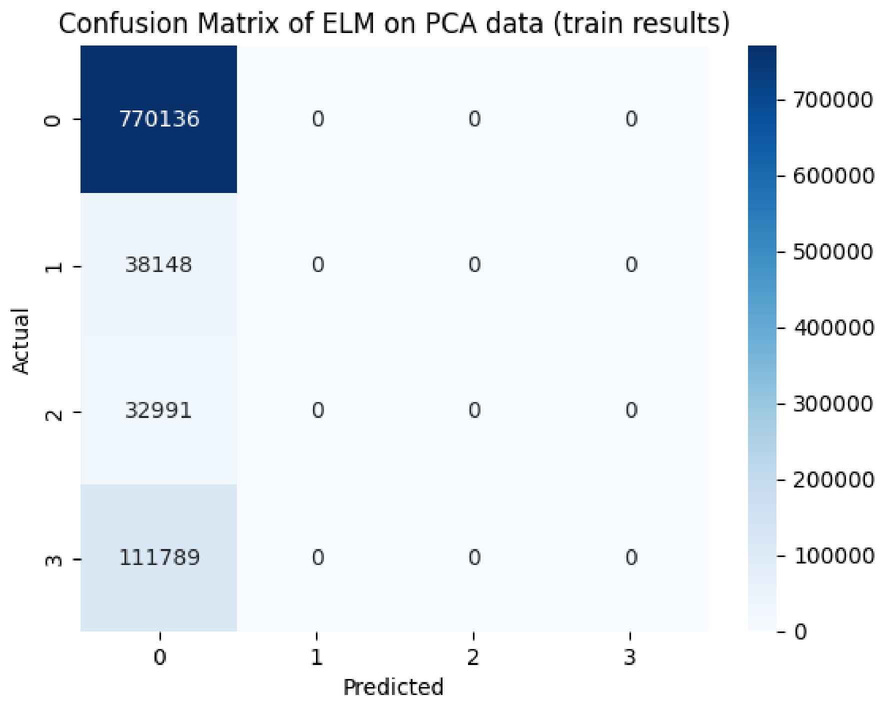
	Accuracy	Precision	Recall	F1 Score
Test Results	0.8084250022034105	0.6535509841875843	0.8084250022034105	0.7227847252623566
Train Results	0.8080632570320566	0.6529662273652556	0.8080632570320566	0.7222825029220519

Confusion Matrix of test data



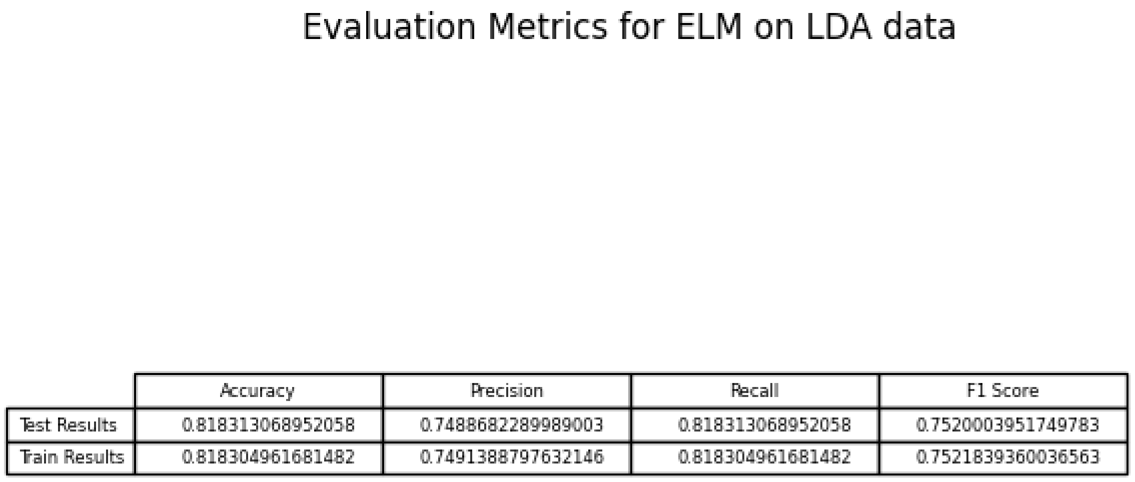
	0	1	2	3
0	192621	0	0	0
1	9615	0	0	0
2	8076	0	0	0
3	27955	0	0	0

Confusion Matrix of train data



	0	1	2	3
0	770136	0	0	0
1	38148	0	0	0
2	32991	0	0	0
3	111789	0	0	0

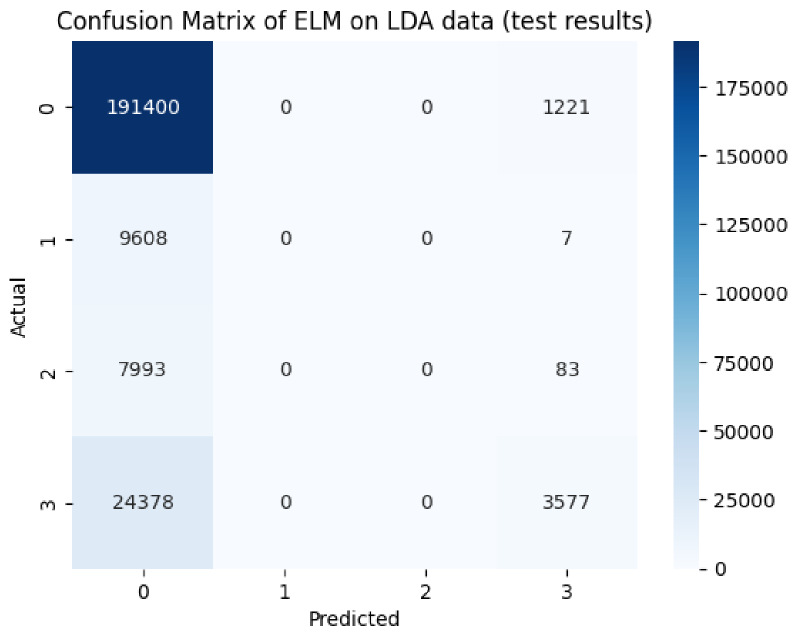
2.2. Evaluation Metrics on LDA



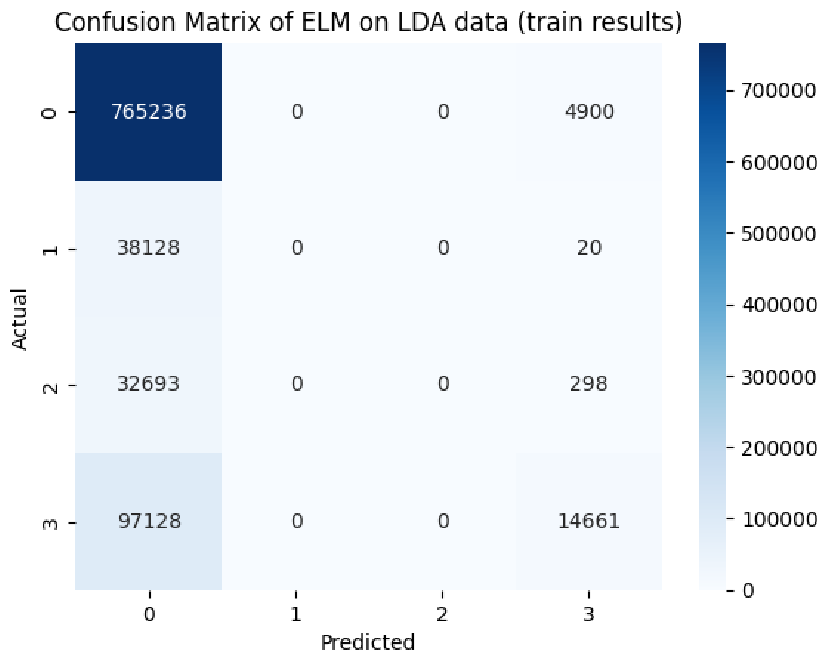
Evaluation Metrics for ELM on LDA data

	Accuracy	Precision	Recall	F1 Score
Test Results	0.818313068952058	0.7488682289989003	0.818313068952058	0.7520003951749783
Train Results	0.818304961681482	0.7491388797632146	0.818304961681482	0.7521839360036563

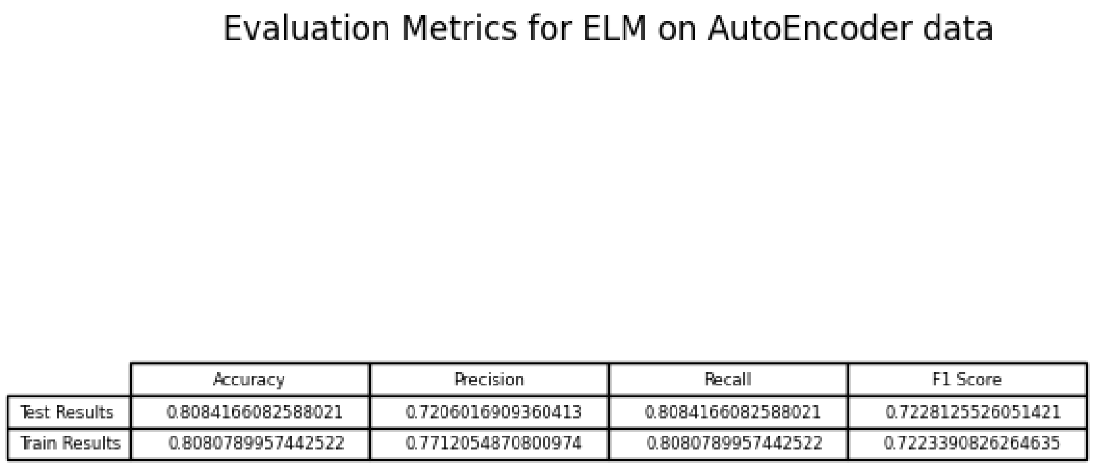
Confusion Matrix on test data



Confusion Matrix on train data



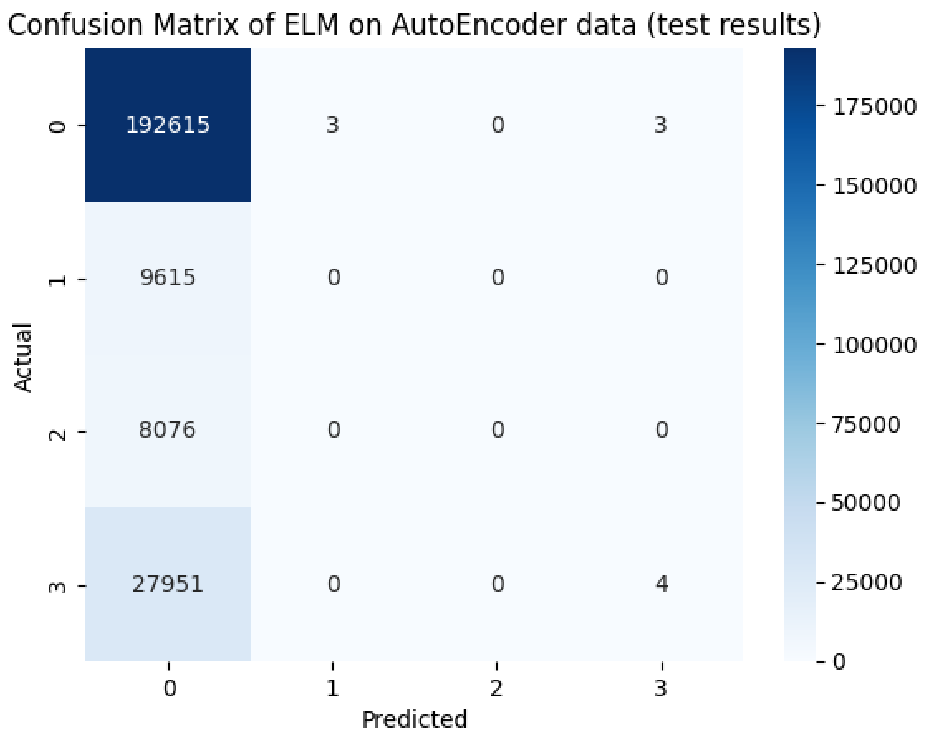
2.3. Evaluation Metrics on Auto Encoder



Evaluation Metrics for ELM on AutoEncoder data

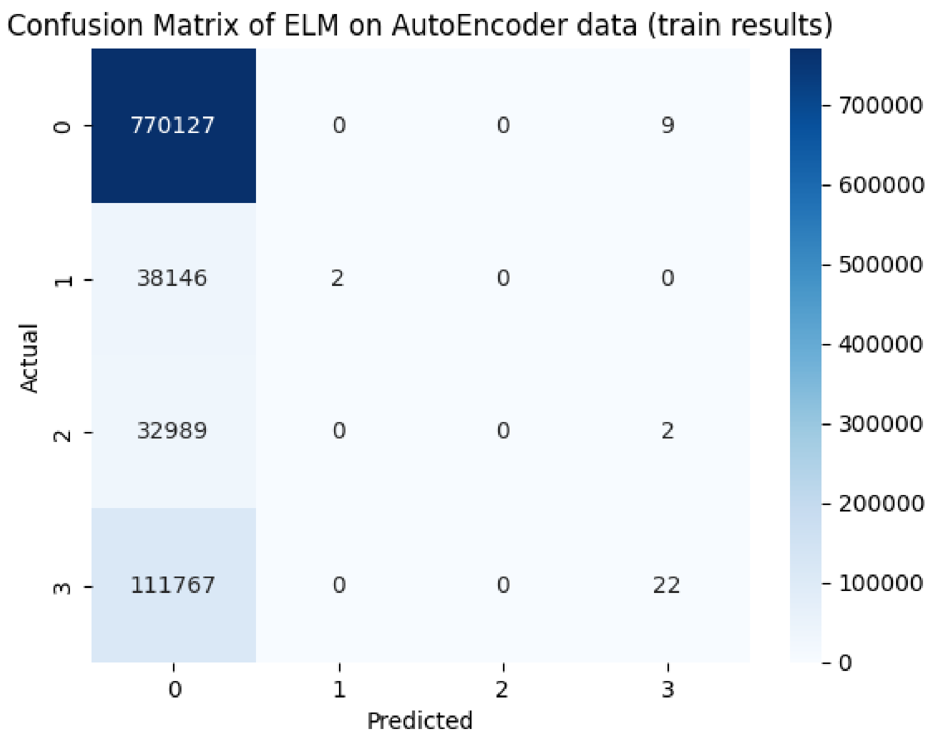
	Accuracy	Precision	Recall	F1 Score
Test Results	0.8084166082588021	0.7206016909360413	0.8084166082588021	0.7228125526051421
Train Results	0.8080789957442522	0.7712054870800974	0.8080789957442522	0.7223390826264635

Confusion Matrix on test data



Actual \ Predicted	0	1	2	3
0	192615	3	0	3
1	9615	0	0	0
2	8076	0	0	0
3	27951	0	0	4

Confusion Matrix on train data

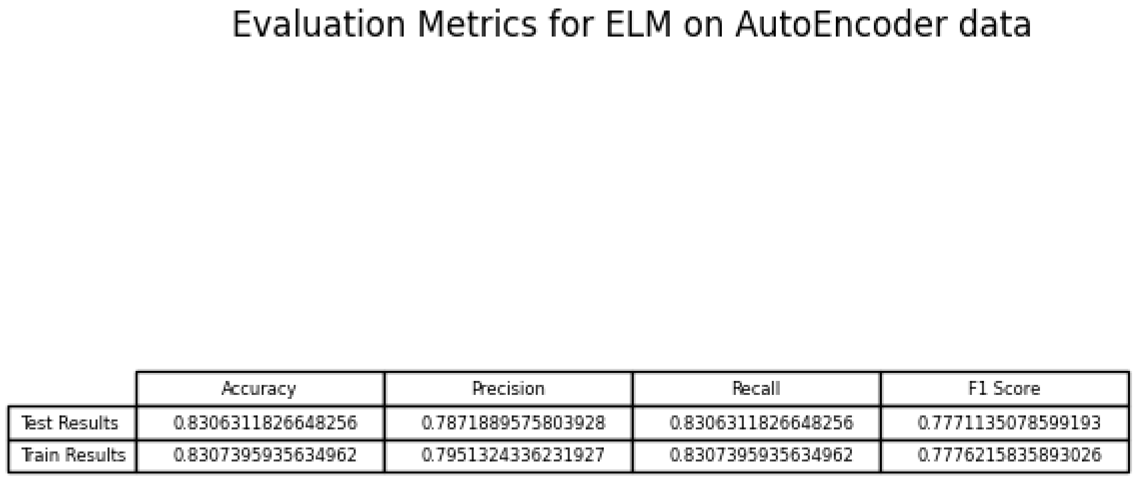


Actual \ Predicted	0	1	2	3
0	770127	0	0	9
1	38146	2	0	0
2	32989	0	0	2
3	111767	0	0	22

3. Decision Tree

In this section as I mentioned before I just implement a decision tree to compare the results at the end. Here are the results.

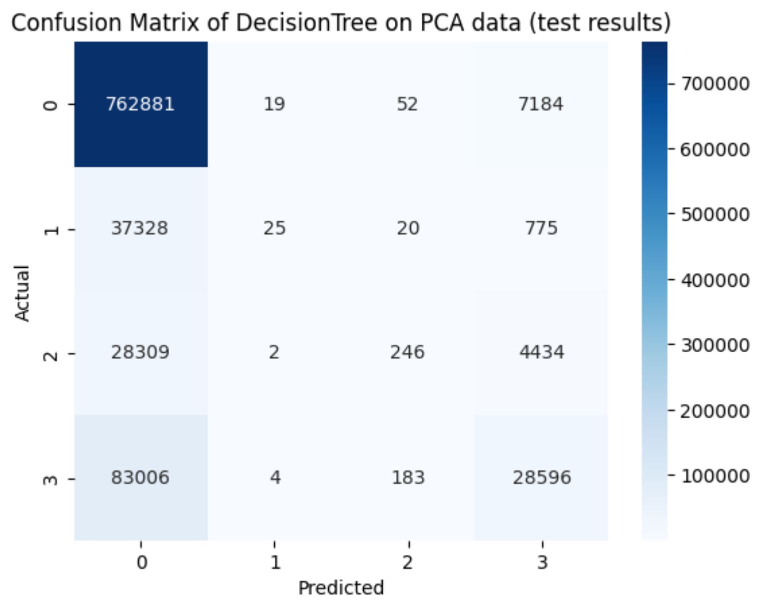
3.1. Evaluation Metrics on PCA



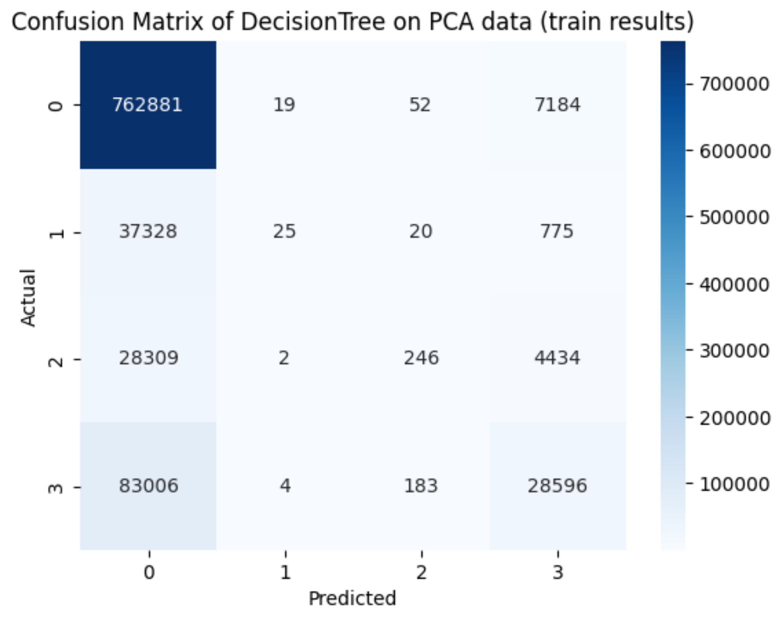
Evaluation Metrics for ELM on AutoEncoder data

	Accuracy	Precision	Recall	F1 Score
Test Results	0.8306311826648256	0.7871889575803928	0.8306311826648256	0.7771135078599193
Train Results	0.8307395935634962	0.7951324336231927	0.8307395935634962	0.7776215835893026

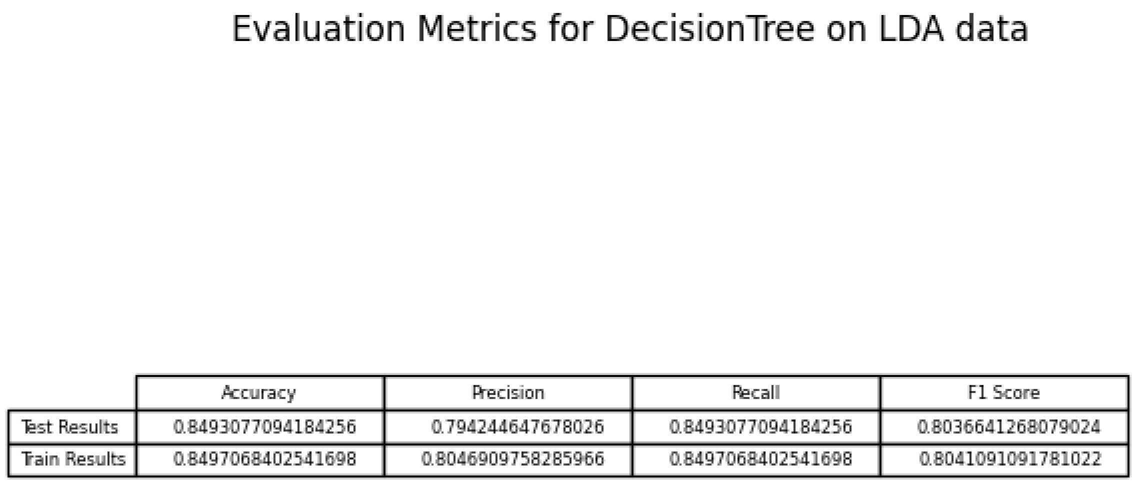
Confusion Matrix on test data



Confusion Matrix on train data

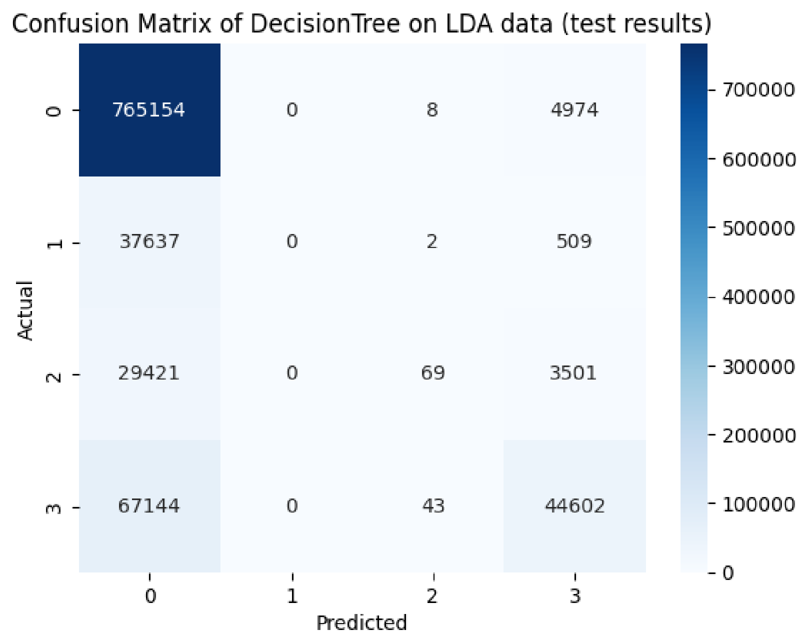


3.2. Evaluation Metrics on LDA

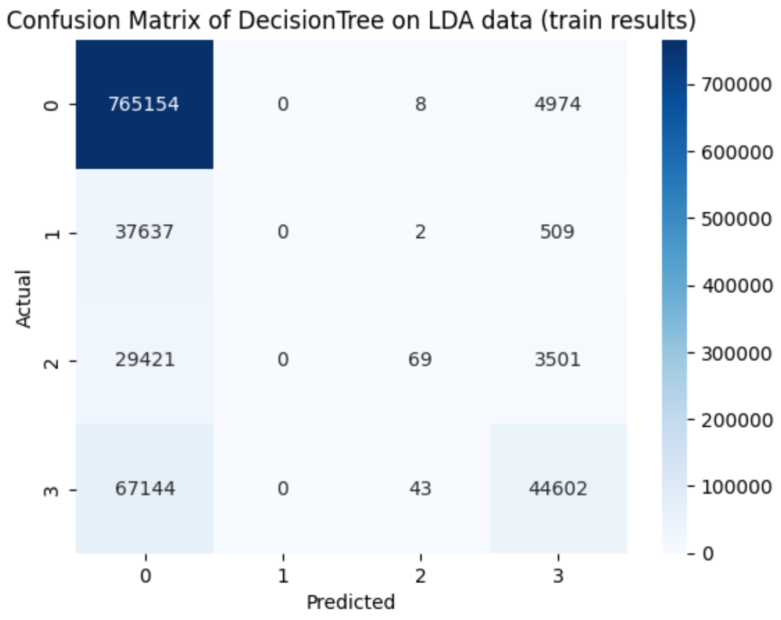

Evaluation Metrics for DecisionTree on LDA data

	Accuracy	Precision	Recall	F1 Score
Test Results	0.8493077094184256	0.794244647678026	0.8493077094184256	0.8036641268079024
Train Results	0.8497068402541698	0.8046909758285966	0.8497068402541698	0.8041091091781022

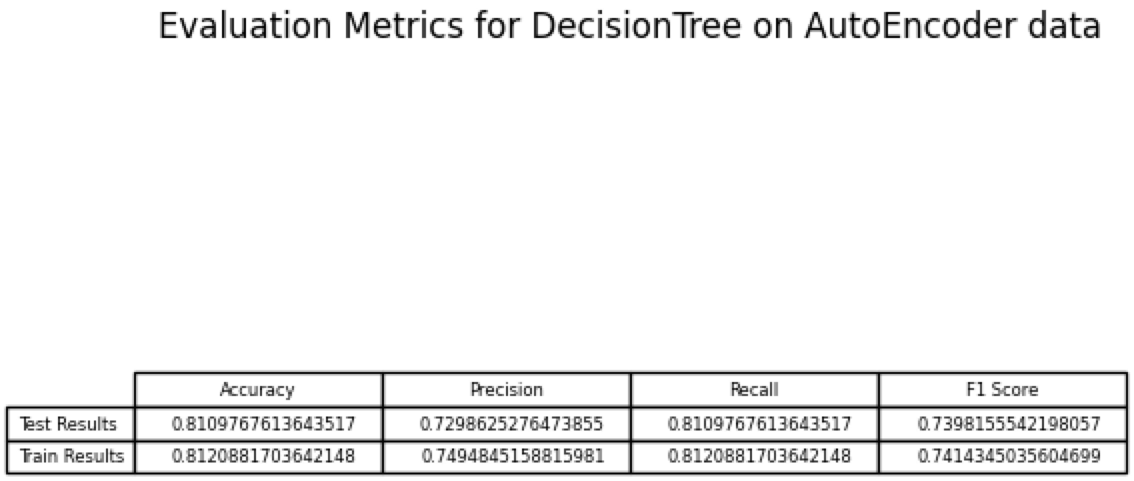
Confusion Matrix on test data



Confusion Matrix on train data

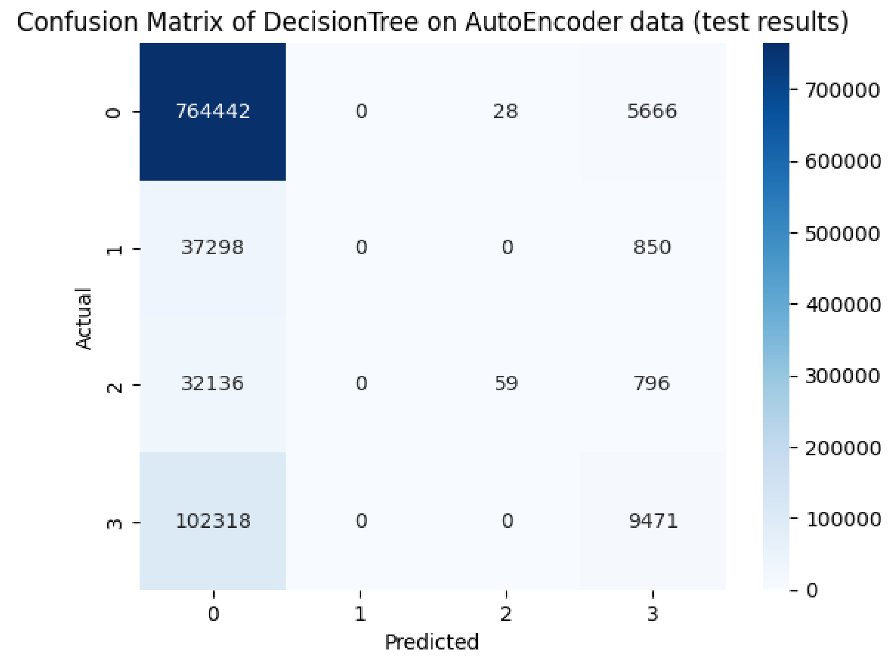


3.3. Evaluation Metrics on Auto Encoder


Evaluation Metrics for DecisionTree on AutoEncoder data

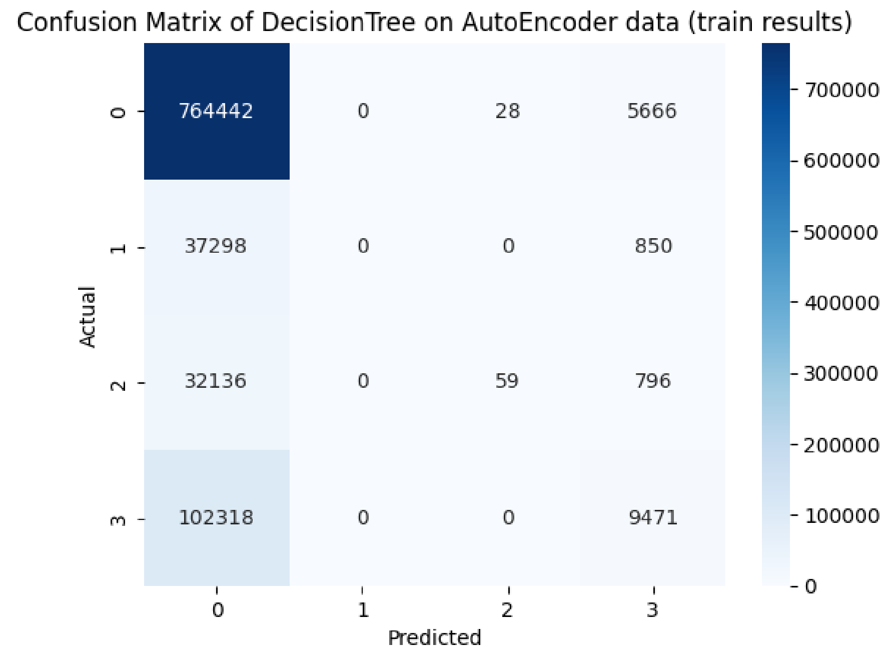
	Accuracy	Precision	Recall	F1 Score
Test Results	0.8109767613643517	0.7298625276473855	0.8109767613643517	0.7398155542198057
Train Results	0.8120881703642148	0.7494845158815981	0.8120881703642148	0.7414345035604699

Confusion Matrix on test data



Actual \ Predicted	0	1	2	3
0	764442	0	28	5666
1	37298	0	0	850
2	32136	0	59	796
3	102318	0	0	9471

Confusion Matrix on train data



Actual \ Predicted	0	1	2	3
0	764442	0	28	5666
1	37298	0	0	850
2	32136	0	59	796
3	102318	0	0	9471

Conclusion

as you can see on the confusion matrices, labels 1 and 2 are really bad predicted and that is the cause of the definition of these labels, that means departure had delay but not arrival and departure had not delay but arrival had which are really rare and it is normal that our data set is not balanced on all labels so, all of the models had problem on predicting labels 1 and 2. In the LDA and Auto Encoder data we had better results compare to PCA, but results of all models on auto encoder is roughly better than results on LDA.

The functionality of all three models is almost the same but MLP in this case is nearly better that others in every situation.