BLG 527E - Machine Learning
Ömercan Susam – 504162517

Homework 3

Q1: How to run python code: “python hw3_q1.py”

Classifier Number	Method Name	Hyperparameters
0	DecisionTreeClassifier	DEFAULT
1		max_depth=3
2		criterion="entropy"
3		max_leaf_nodes=3
4		max_depth=3, max_leaf_nodes=3, criterion="entropy"
5	MLPClassifier	alpha=1
6		activation="logistic",alpha=1
7		solver='sgd',alpha=1
8		learning_rate="adaptive" ,alpha=1
9		activation="logistic",solver='sgd',learning_rate="adaptive" ,alpha=1

	Default Hyper Parameters
DecisionTreeClassifier	criterion="gini", splitter="best", max_depth=None, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0., max_features=None, random_state=None, max_leaf_nodes=None, min_impurity_decrease=0., min_impurity_split=None, class_weight=None, presort=False
MLPClassifier	hidden_layer_sizes=(100,), activation="relu", solver='adam', alpha=0.0001, batch_size='auto', learning_rate="constant", learning_rate_init=0.001, power_t=0.5, max_iter=200, shuffle=True, random_state=None, tol=1e-4, verbose=False, warm_start=False, momentum=0.9, nesterovs_momentum=True, early_stopping=False, validation_fraction=0.1, beta_1=0.9, beta_2=0.999, epsilon=1e-8

Table 1 Accuracies for Validation set comparison for different hyperparameters of DTC and MLP

Class	Classifier by Their Numbers									
	DecisionTreeClassifier					MLPClassifier				
	0	1	2	3	4	**5**	6	7	8	9
0	95.506%	93.258%	94.944%	97.753%	97.753%	99.438%	98.876%	98.876%	98.876%	98.876%
1	87.363%	0.000%	87.363%	100.000	0.000%	98.901%	96.703%	98.901%	100.000	95.055%
2	83.051%	1.130%	83.616%	0.000%	0.000%	96.610%	96.045%	98.305%	96.610%	97.740%
3	80.874%	90.164%	87.978%	0.000%	91.803%	93.443%	93.989%	93.443%	92.896%	92.350%
4	79.558%	4.972%	93.370%	0.000%	0.000%	97.790%	97.790%	97.790%	97.238%	97.238%
5	87.363%	9.890%	95.055%	0.000%	0.000%	98.352%	98.901%	98.901%	98.352%	98.352%
6	94.475%	96.685%	95.580%	0.000%	96.685%	97.790%	97.790%	97.790%	97.790%	98.343%
7	78.212%	55.307%	79.888%	70.391%	0.000%	92.179%	91.620%	92.179%	92.179%	90.503%
8	85.632%	0.000%	84.483%	0.000%	0.000%	91.954%	88.506%	90.805%	93.103%	86.782%
9	86.111%	29.444%	82.778%	0.000%	0.000%	98.889%	95.556%	97.778%	96.667%	95.556%
Average	85,81	38,09	88,51	26,81	28,62	**96,53**	95,58	96,48	96,37	95,08

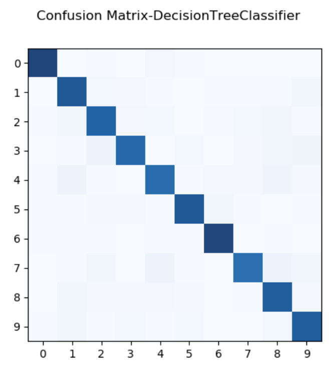
Max_depth or max_leaf_nodes values are highly critical for Decision Tree Classifier. Performance directly decreases (59%) when these are reduced. On the other hand, when criterion selection changed from ‘gini’ to ‘entropy’, performance improvement observed for 2,7 %. But these are only one run results, high scale experiments are needed to make a such deduction.
Best result for MLP are obtained when its alpha hyperparameter set to 1. Changes on hyperparameters are slightly changed the accuracy between 95,08% and 96,53%.
MLP has higher accuracy than DTC.

BLG 527E - Machine Learning
Ömercan Susam – 504162517

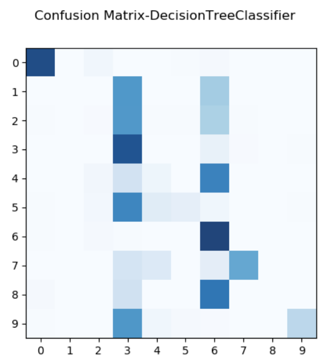
Homework 3

Classifier Number

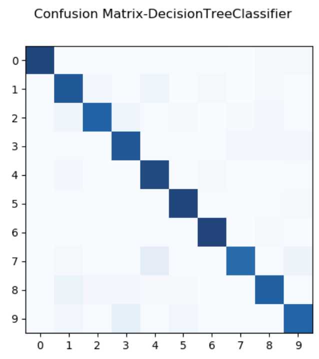
0



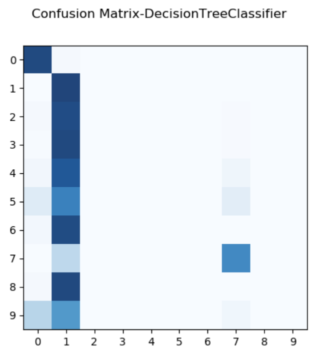
1



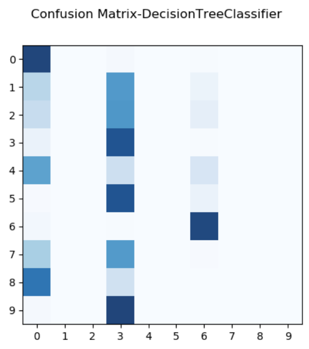
2



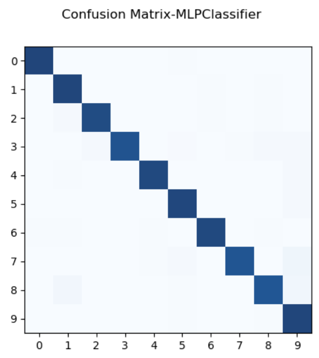
3



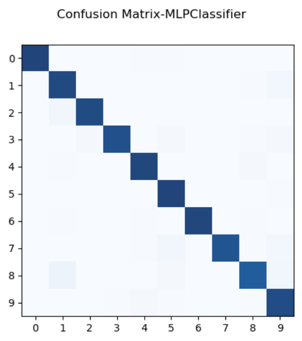
4



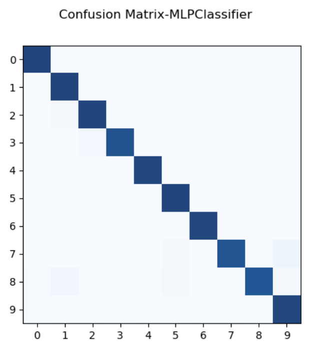
5



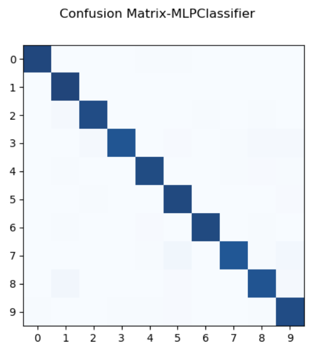
6



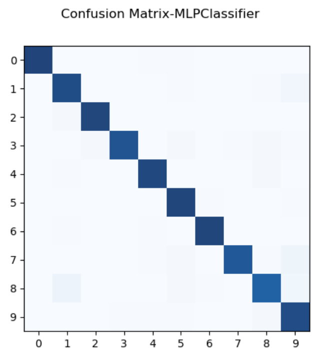
7



8



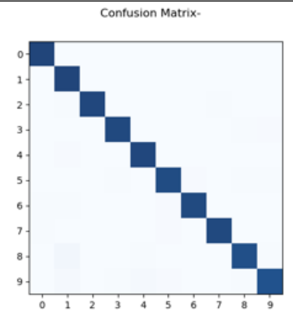
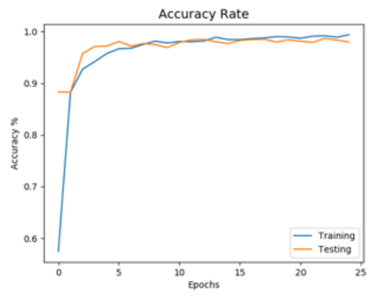
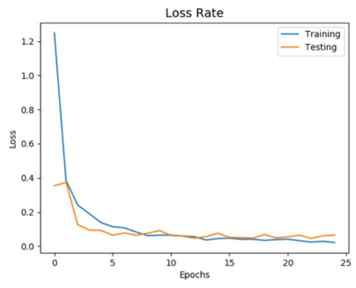
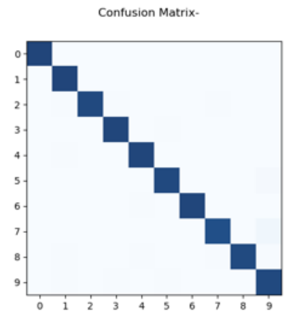
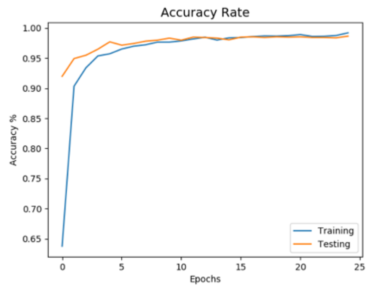
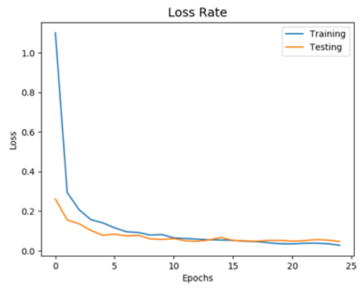
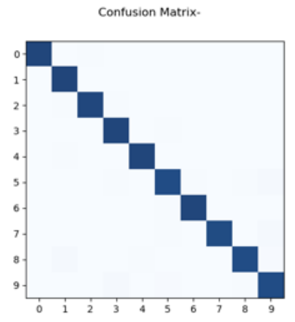
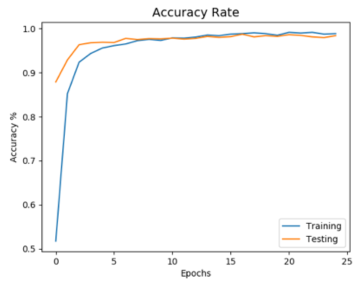
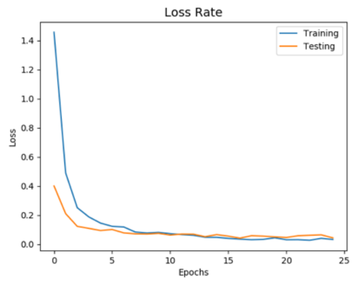
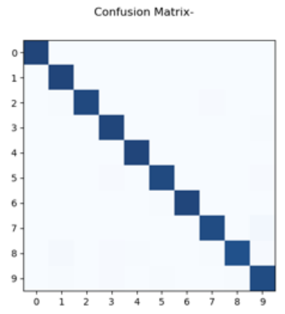
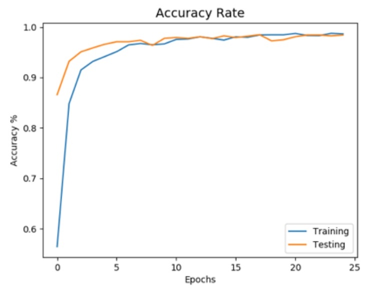
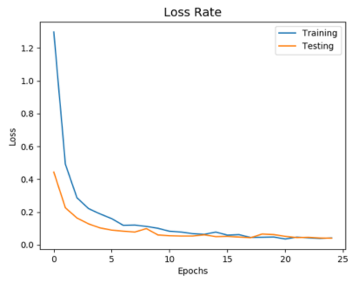
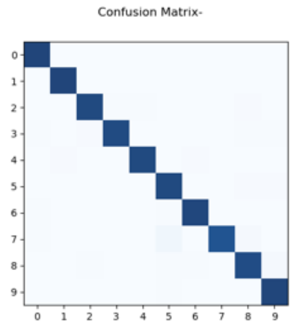
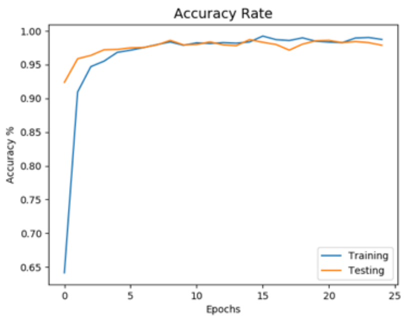
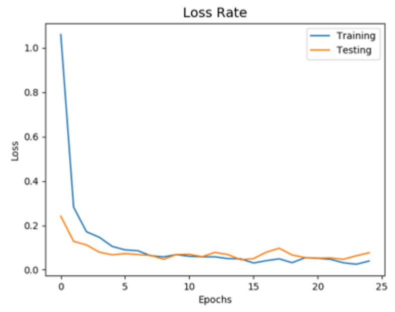
9



Homework 3

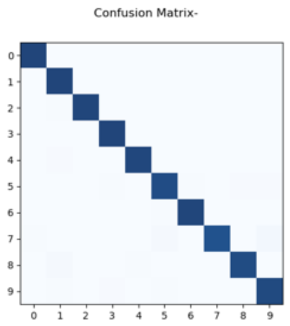
Q2: How to run python code: “python hw3_q2.py”

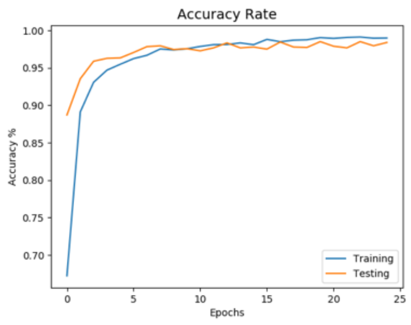
Results for 7 different optimizers with 25 Epochs for Validation set.

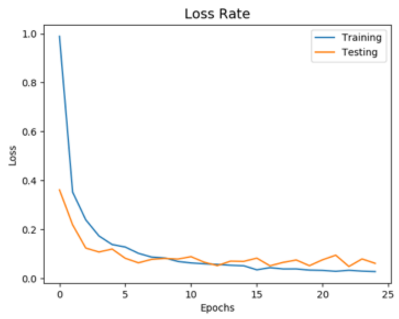
	Confussion Matrix	Accuracy Rate	Loss Rate
adadelta			
adagrad			
adam			
adamax			
nadam			

Homework 3

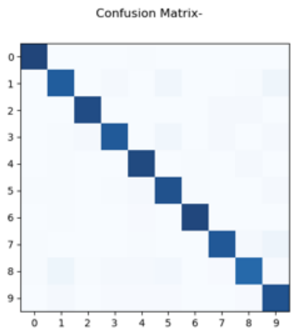
rmsprop

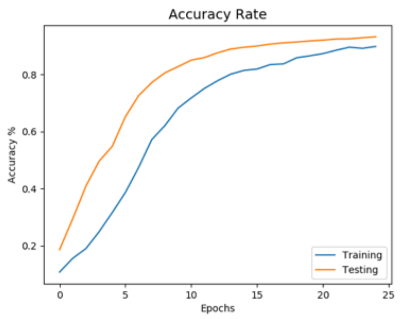


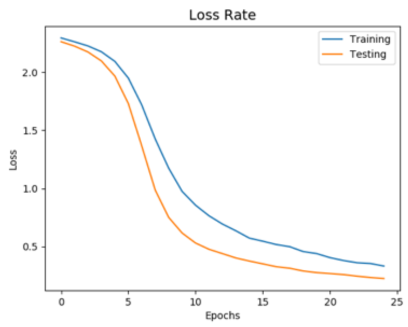




sgd







Winner of the optimizer competition is **adagrad** with 98,65% accuracy for 25 Epochs. Worst optimizer is **sgd** which is easily observable when accuracy rate and loss rate graphs are considered.

	Validation Set Accuracy (Epochs = 25)						
Class	nadam	adamax	adam	adadelta	**adagrad**	rmsprop	sgd
0	100.000%	100.000%	99.438%	100.000%	100.000%	100.000%	99.438%
1	100.000%	100.000%	100.000%	100.000%	100.000%	100.000%	89.011%
2	97.740%	98.305%	98.870%	98.870%	98.305%	99.435%	96.045%
3	97.268%	99.454%	99.454%	98.361%	99.454%	100.000%	90.164%
4	97.790%	100.000%	99.448%	98.895%	99.448%	98.895%	97.238%
5	97.802%	97.802%	96.703%	97.253%	98.352%	96.703%	93.956%
6	98.895%	99.448%	99.448%	97.790%	99.448%	99.448%	98.343%
7	93.296%	97.207%	97.207%	98.324%	95.531%	94.413%	91.061%
8	96.552%	94.828%	96.552%	95.402%	97.701%	97.126%	83.908%
9	99.444%	97.222%	96.667%	94.444%	98.333%	97.778%	93.333%
Average	97,8787	98,4266	98,3787	97,9339	**98,6572**	98,3798	93,2497

Comparison of accuracies for different kernel size in convolution for Adadelta. Same DNN structure with the best resulting as Figure 1 except Epochs = 3. Surprisingly, higher score obtained than with less Epochs.

	Accuracies for Validation Set with Different Kernel Sizes	
Class	3x3	2x2
0	99.438%	99.438%
1	95.055%	91.758%
2	96.610%	97.740%
3	93.443%	95.082%
4	93.370%	98.343%
5	98.352%	95.604%
6	98.343%	98.343%
7	92.737%	93.296%
8	89.080%	89.080%
9	98.889%	92.778%
Average	98,49	95,5317

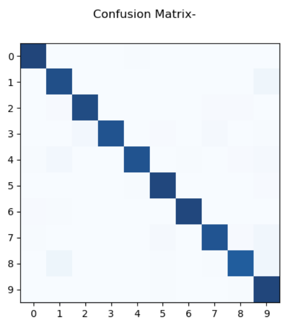


Figure 1 Kernel 3x3

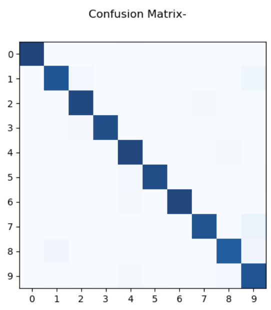


Figure 2 Kernel 2x2

Homework 3

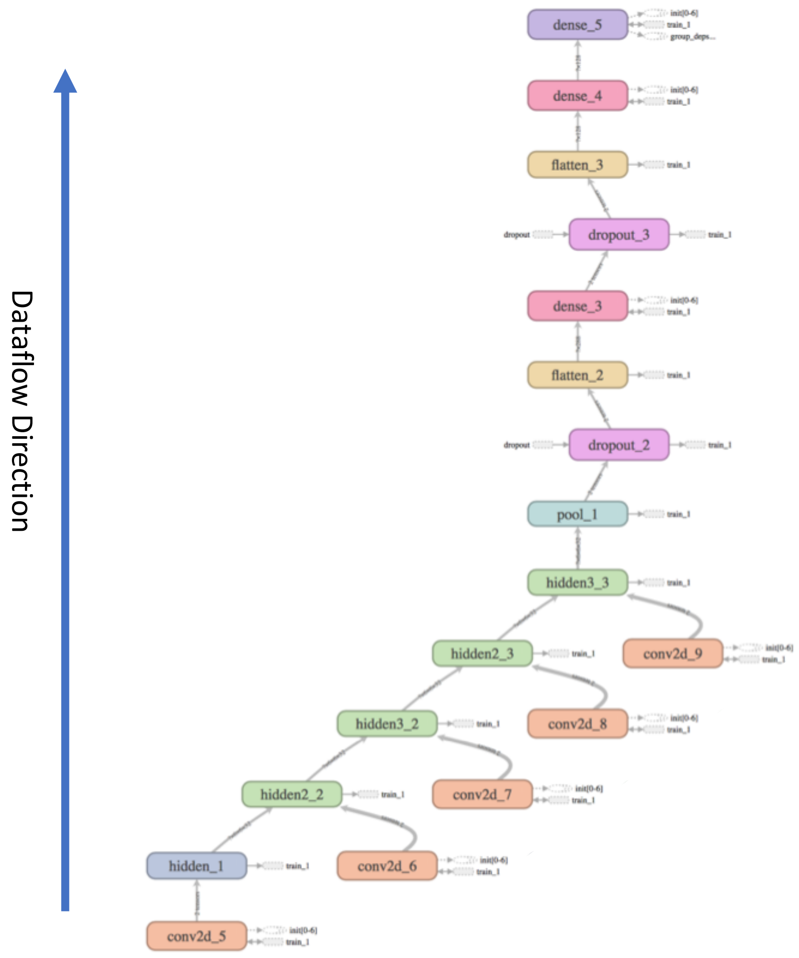


Figure 3 Visualization of the best resulting Deep Neural Network. Conv -> Conv -> Conv -> MaxPooling -> Dropout -> Flatten -> Dense -> Dropout -> Flatten -> Dense -> Dense. Kernel size for convolutions = (3x3).

<u>Comparison</u>

Both for DNN and MLP, accuracy rates are quite high. Best result for DNN is 98,65% and for MLP it is 96,53%. Besides, when layer and iteration number increased in DNN, performance directly boosts up.

On the other hand, MLP and DTC run times are under 1 second with 2,7 GHz Intel Core i5 CPU. Whereas, DNN nearly takes 10 minutes to complete 25 epochs.

BLG 527E - Machine Learning
Ömercan Susam – 504162517

Homework 3

Q3: How to run python code: “python hw3_q3.py”

	Accuracy	
Class	Test	Training
0	91,69%	92,34%
1	0,39%	0,70%
Average	46,04%	46,52%

<u>Comparison</u>

Results for HMM is very low when compared with the other methods. Almost %50 difference with the best algorithm in negative way but it is better than some DTC models. The reason may be implementation faults.