MACHINE LEARNING MODEL

COMPARISON BASED ON SOME METRICS

Safa ORHAN

Computer Engineering Student

Istanbul Kultur University

Istanbul, Turkey

Eyüp USTA

Computer Engineering Student

Istanbul Kultur University

Istanbul, Turkey

I. SUPPORT VECTOR MACHINE

Kernels = linear, poly, rbf, sigmoid

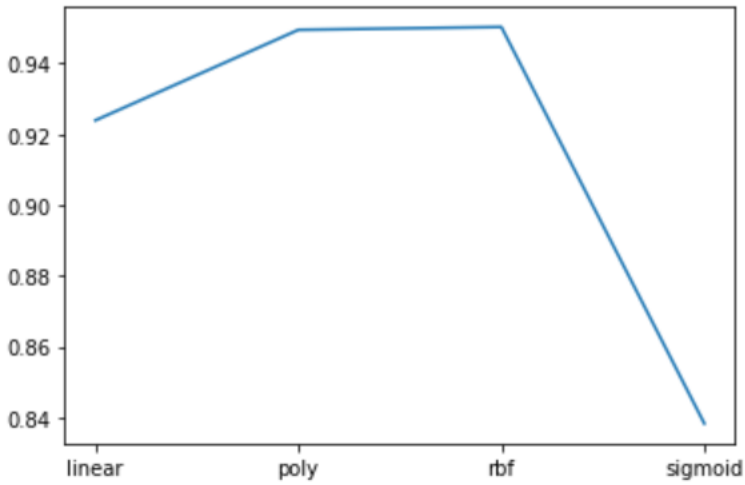
C = 1, 2, 3, 4, 5

Degree = 1, 2, 3, 4, 5, 6

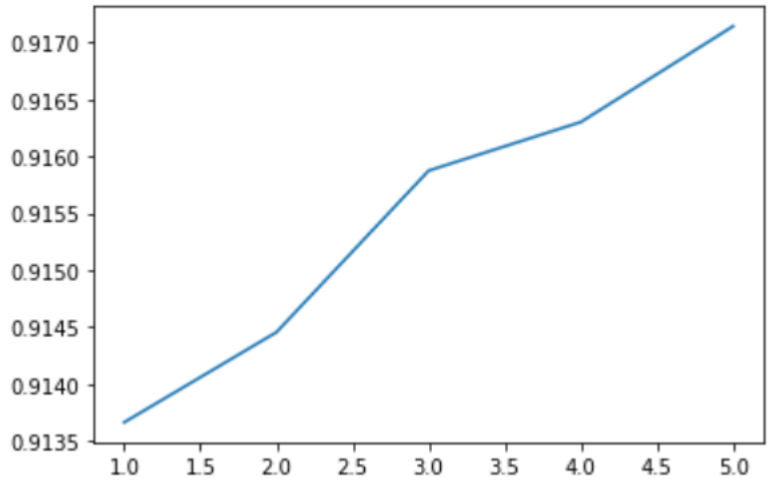
Gamma = scale, auto

Decision function shape = ovo, ovr

Kernel vs Accuracy



C vs Accuracy



II. LINEAR SUPPORT VECTOR MACHINE

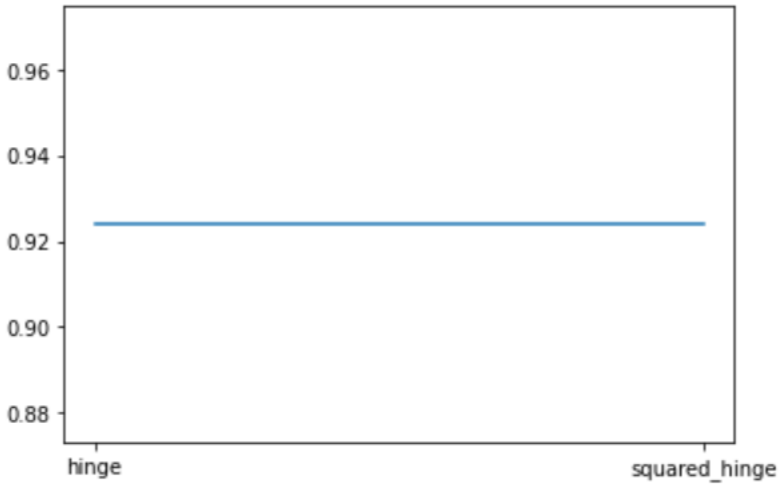
Losses = hinge, squared hinge

Penalty = l2

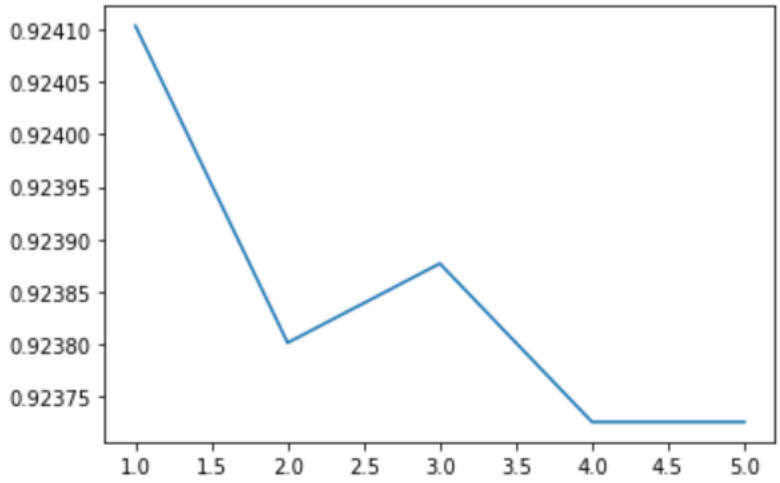
C = 1, 2, 3, 4, 5

Multi Class = ovr, crammer singer

Loss vs Accuracy



C vs Accuracy



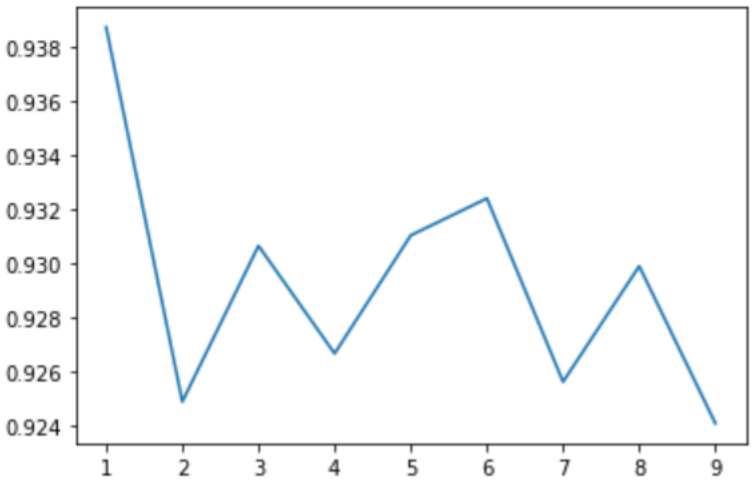
III. K – NEAREST NEIGHBORS

K = 1,2,3,4,5,6,7,8,9,10

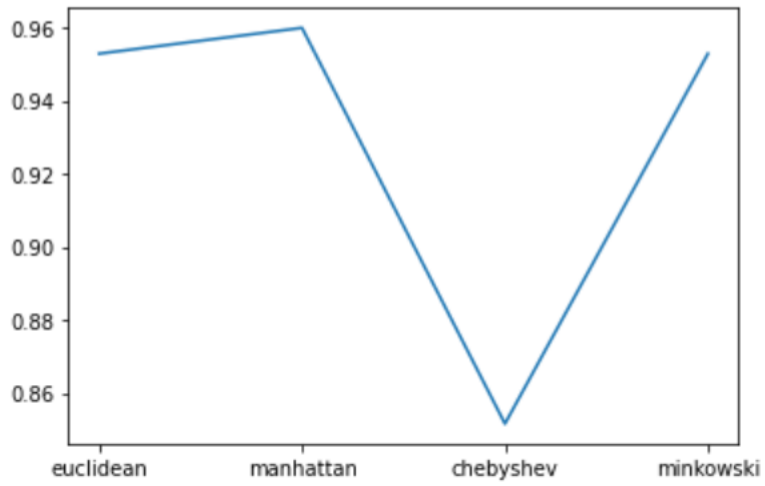
Weights = uniform, distance

Metric = euclidean, manhattan, chebyshev, minkowski

K vs Accuracy



Mertic vs Accuracy



VI. BERNOULLI NAIVE BAYES CLASSIFIER

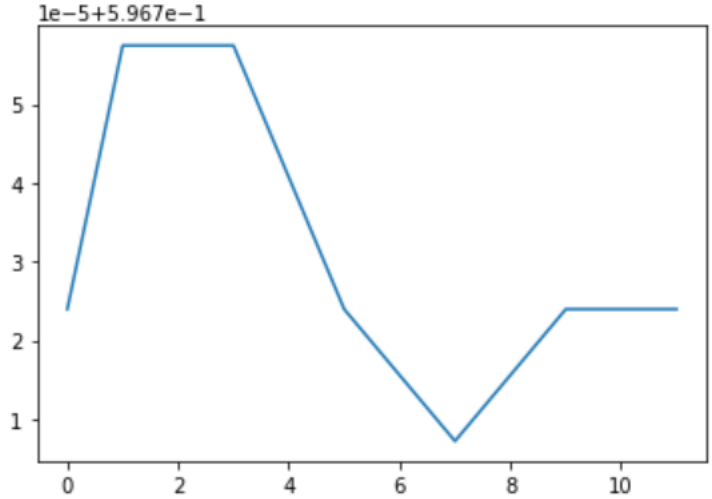
Bernoulli Naive Bayes Classifier:

Alpha = 0, 1, 2, 3, 4, 5, 7, 9, 11

Binarize = 0, 1, 2, 3, 4, 5, 7, 9, 11

Fit prior = True, False

Alpha vs Accuracy



VII. RANDOM FOREST CLASSIFIER

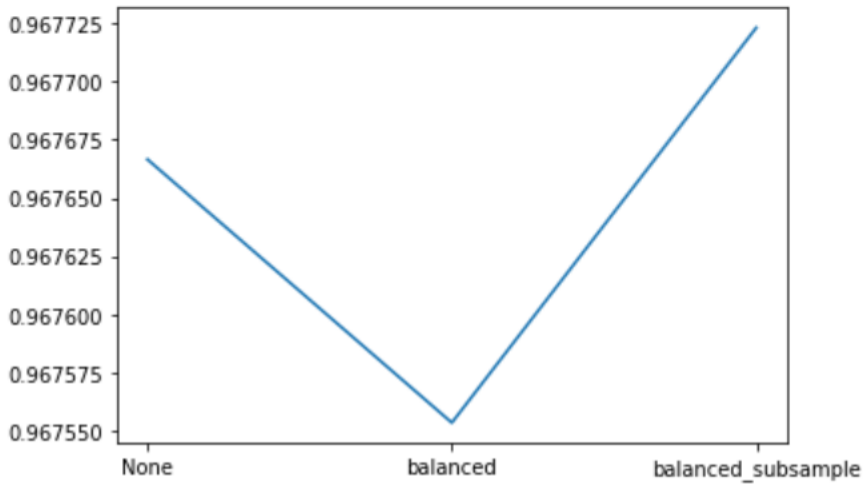
Max features = None, auto, sqrt, log2

Criterion = gini, entropy

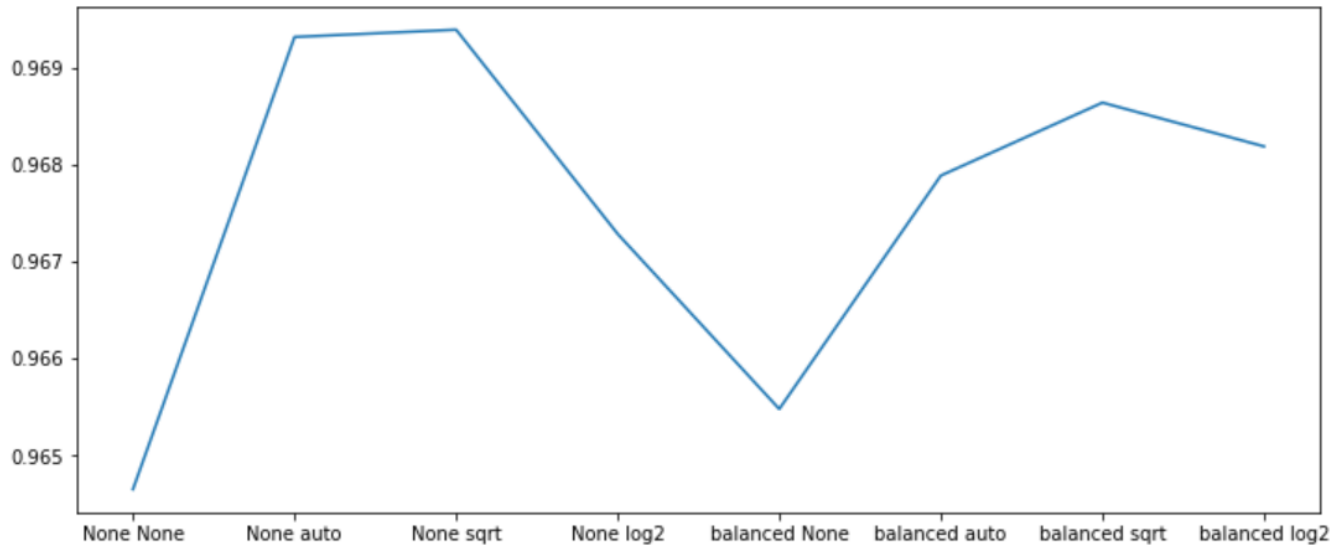
Class_weight = None, balanced, balanced_subsample

Warm start = True, False

Class Weight vs Accuracy



(Max Features + Class Weight) vs Accuracy



VIII. DEEP LEARNING WITH TENSORFLOW

Optimizers = 'sgd', 'rmsprop', 'adam', 'adadelta', 'adagrad', 'adamax', 'nadam', 'ftrl'

Loss = 'binary_crossentropy', 'categorical_crossentropy', 'hinge', 'squared_hinge', 'huber'

Activation = 'softplus', 'softsign', 'selu', 'elu', 'exponential', 'tanh', 'sigmoid', 'relu'

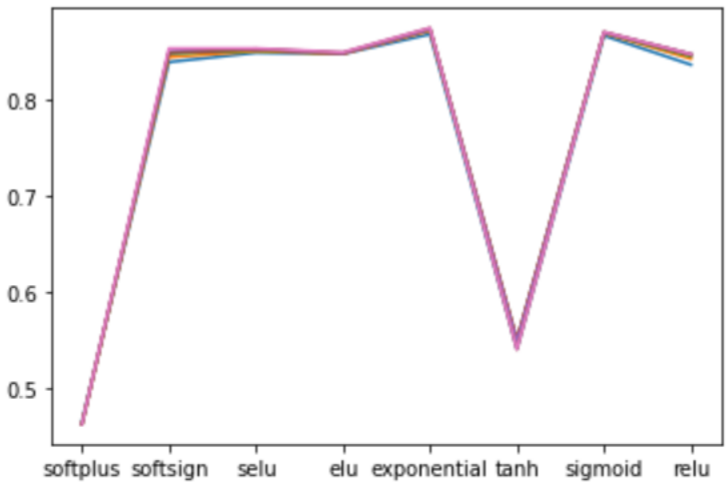
A. Results of the Deep Network:

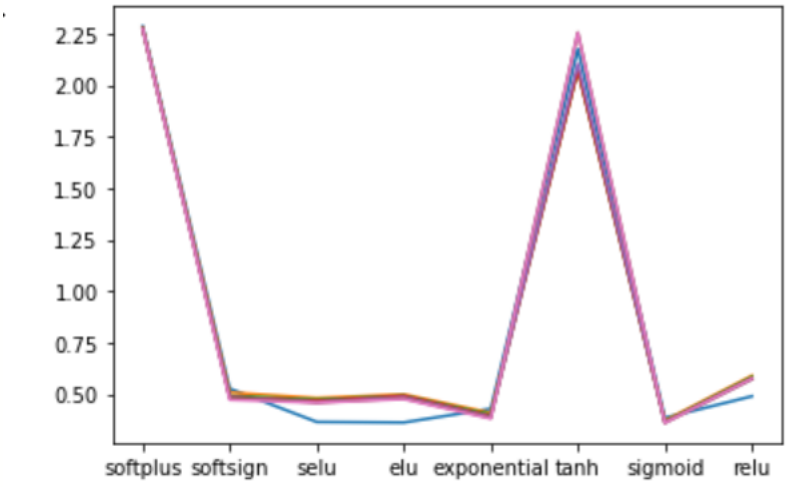
Input Layer: Dense (64, kernel_regularizer=l2, activation = relu, input_shape = (30,2))

Deep Layers: Dense (128, kernel_regularizer=l2, activation = relu), Dense (128, kernel_regularizer=l2, activation = relu), Dense (128, kernel_regularizer=l2, activation = relu), Dense (256, kernel_regularizer=l2, activation = relu),

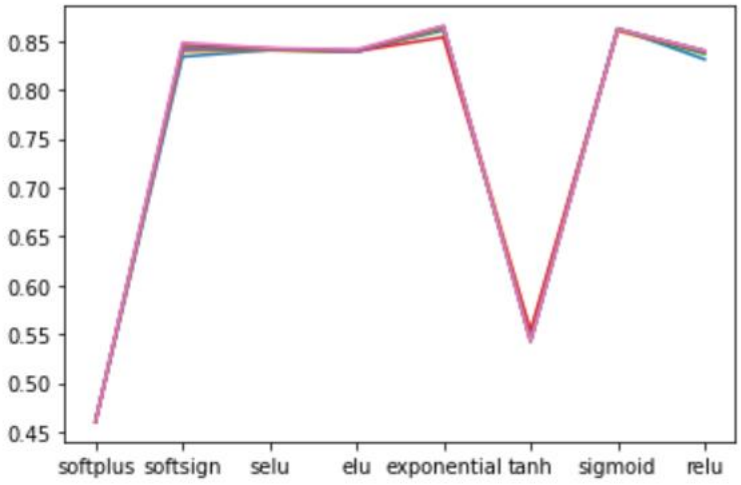
Output Layer: Dense (1, activation=softplus)

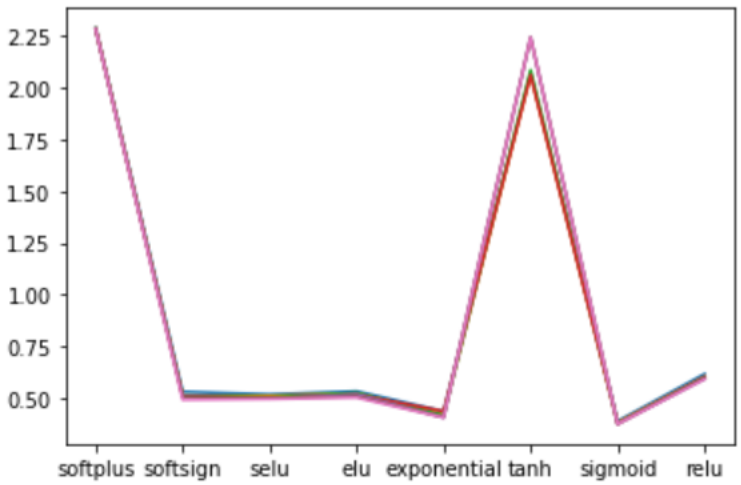
B. Validation Accuracy and Losses:





C. Validation Accuracy and Losses:





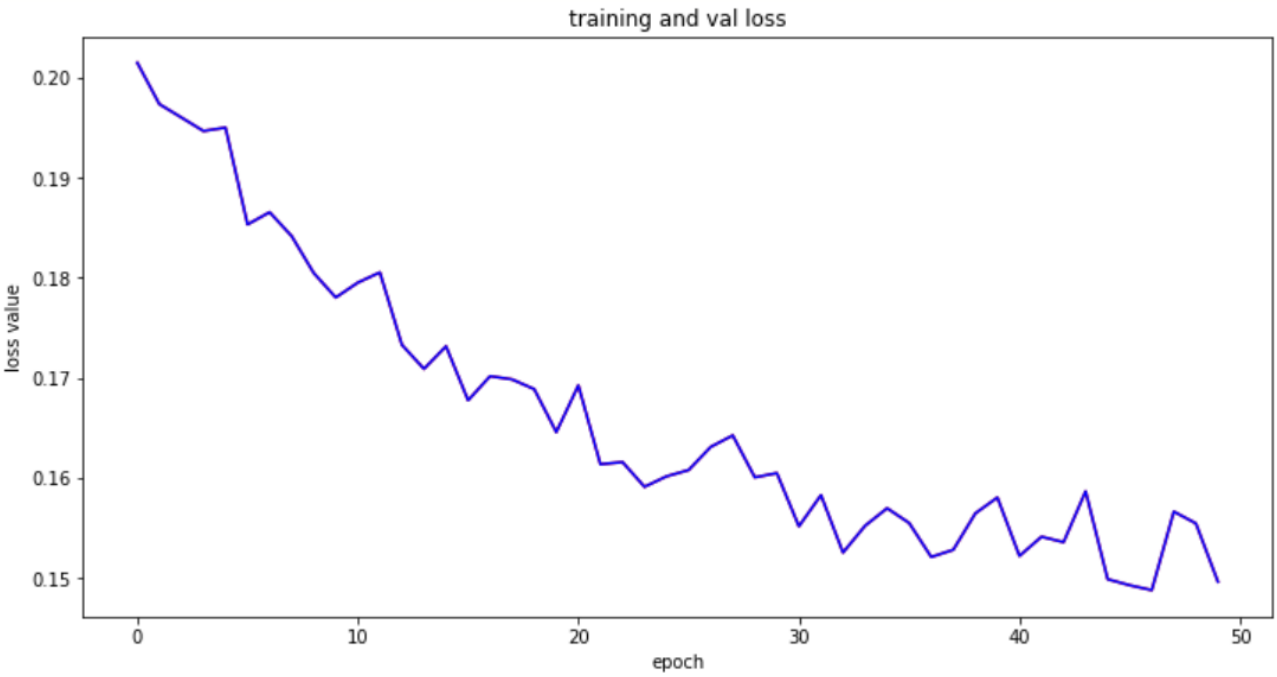
IX. CONVOLUTIONAL NEURAL NETWORKS

```
keras.layers.Conv2D(32,(1,1), activation='relu',input_shape=(2,5,3))

keras.layers.MaxPool2D(2,2)

keras.layers.Conv2D(64,(1,1),activation='relu')

keras.layers.Conv2D(128,(1,1),activation='relu')

keras.layers.Conv2D(128,(1,1),activation='relu')

keras.layers.Flatten()

keras.layers.Dense(512,activation='relu')

keras.layers.Dense(1,activation='sigmoid')
```

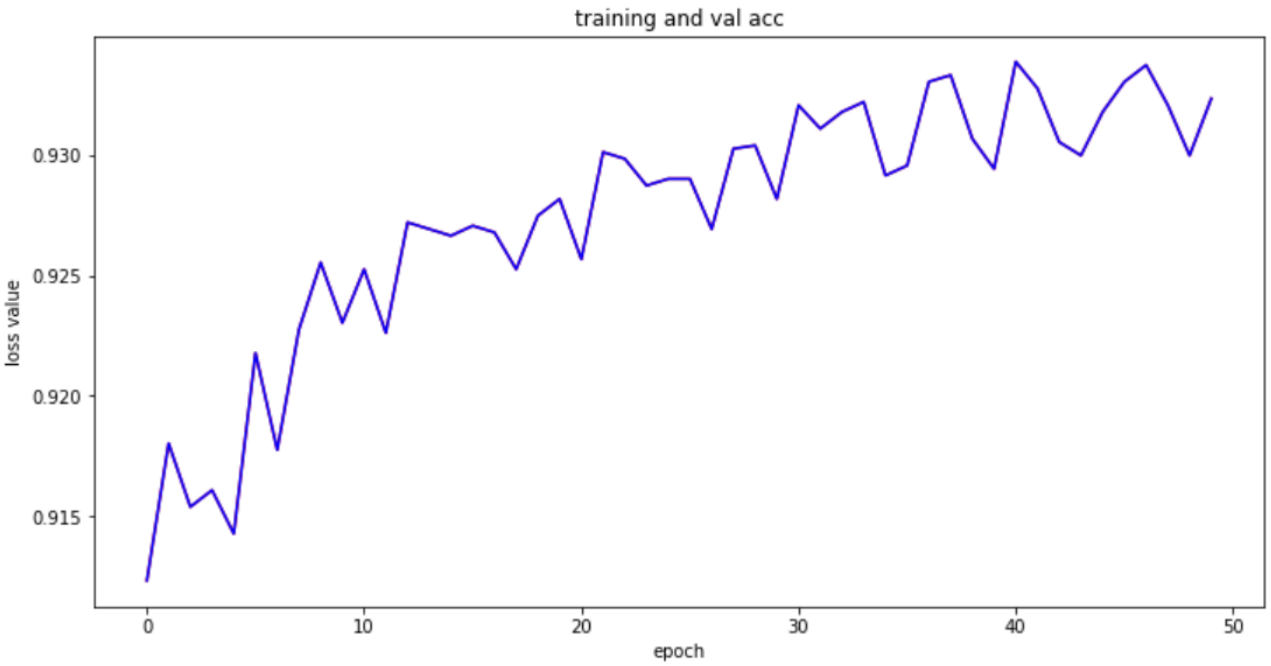
Best Accuracy = 83.89%

Average Accuracy = 83.40%

A. Loss:



B. Accuracy:



C. Adaboost Classifier:

N_estimators = 1000

Accuracy = 94.26%

D. Ensemble Learning:

Stacking Classifier

Parameters

Estimators = RandomForestClassifier(), LinearSVC()

Final Estimator = KNNClassifier()

Accuracy = 96.16%

E. Class Distribution Visualization: