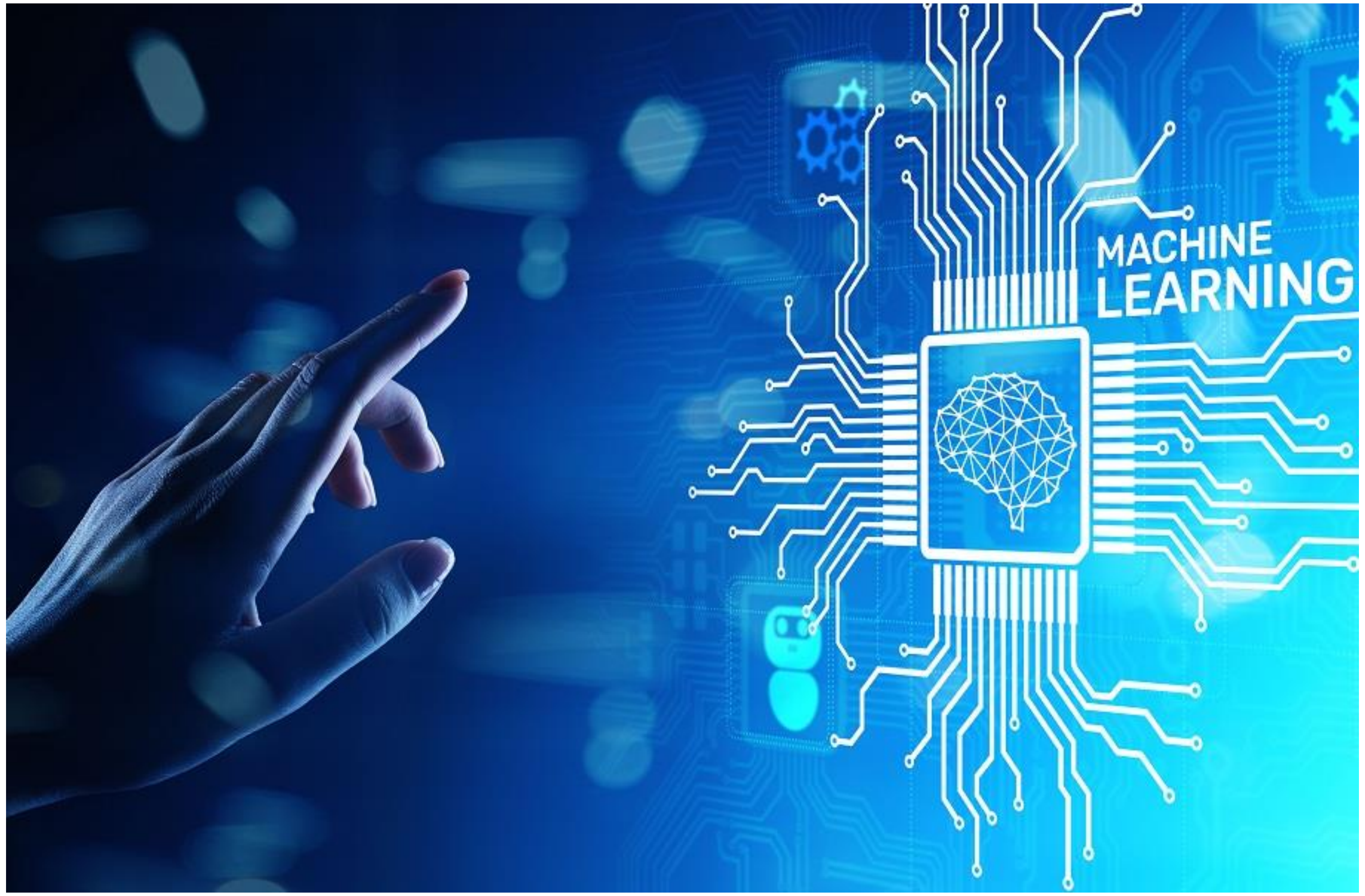
Duration: 100 min | BCSE424P | ML for Robotics Lab | Dr. Harini S

Hyper parameter Tuning – MLP, Linear SVM



Learning for Today (17.02.2026)

- MLP
  - Softmax activation, Cross – Entropy
  - MSE
  - GD Variants – SGD
  - Auto Hyperparameter Tuning Methods
- Linear SVM

Softmax activation function

- Softmax is an activation function used in the output layer of a Multi-Layer Perceptron (MLP) for multi-class classification.

$$\text{Softmax}(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{C} e^{z_j}}$$

- Example: If MLP outputs : [2.0,1.0,0.1], after softmax = [0.65,0.24, 0.11]
- Class 1 → 65% probability
- Class 2 → 24% probability
- Class 3 → 11% probability
- Implemented using **mlp.predict_proba(X_test) → did in previous lab**

Cross Entropy for Multi-Class

- If an MLP has C output neurons, softmax gives probabilities:

$$\hat{y}_i = \frac{e^{z_i}}{\sum_{j=1}^{C} e^{z_j}}$$

- Cross-Entropy Loss:

$$L = -\sum_{i=1}^{C} y_i \log(\hat{y}_i)$$

- When ML used for classifier → **mlp.loss_curve_ → did in last lab class**

Sample 1 :SGD in MLP

```
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.neural_network import MLPClassifier


# Load data


# Split Data

mlp_sgd = MLPClassifier(
   hidden_layer_sizes=(16, 8),
   activation='relu',
   solver=‘sgd”,
   learning_rate_init=0.01,
   momentum=0.9,
   max_iter=500,
   random_state=42
)

# Train Model

# Predict
```

Sample 2 : Hyperparameter Tuning

```
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split,
GridSearchCV
from sklearn.neural_network import MLPClassifier


# Load data


# Split Data


# Define model
```

```
# Hyperparameter grid
param_grid = {
'hidden_layer_sizes': [(10,), (20,), (10,10)],
'activation': ['relu', 'tanh'],
'learning_rate_init': [0.001, 0.01],
'alpha': [0.0001, 0.001]
}


# Grid search
grid = GridSearchCV(mlp, param_grid, cv=5)
grid.fit(X_train, y_train)


print("Best Parameters:", grid.best_params_)
print("Best Score:", grid.best_score_)
```

Exercises

1. Identify the hyper parameters for Wisconsin dataset
2. Identify the hyper parameters for Indian diabetes dataset

Sample 3: SVM for Classification (Linear Kernel)

```
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score,
classification_report

# Load dataset (iris)

# Split train and test

svm_model = SVC(kernel='linear', C=1)

svm_model.fit(X_train, y_train)

y_pred = svm_model.predict(X_test)
print("Accuracy:", accuracy_score(y_test, y_pred))
print("\nClassification Report:\n")
print(classification_report(y_test, y_pred))
```

```
print("Number of Support Vectors:",
model.n_support_)
print("Weights (w):", model.coef_)
print("Bias (b):", model.intercept_)
```

Exercises

1. Apply Linear SVM for classifiiying Survived (0/1) in titanic dataset
2. Apply Linear SVM for classifying (Benign / Malignant) in Wisconsin Dataset