Madrid

footline[page number]

height

Outline

Contents

1 Introduction

Motivation & Core Research Question

- tk

2 Data Cleaning

Motivation & Core Research Question

- tk

3 Explorative Data Analysis (EDA)

Motivation & Core Research Question

- tk

4 Model Fitting

Model fitting is the process by which a machine learning model adjusts its internal parameters to minimize the discrepancy between its predictions and the observed data. In supervised learning, we define a loss function $L(\theta)$ that measures this error—where θ denotes all trainable parameters—and then optimize:

$$\theta \leftarrow \theta - \eta \, \nabla_\theta L(\theta),$$

using gradient-based methods such as (stochastic) gradient descent.

Within neural networks, backpropagation efficiently computes $\nabla_\theta L$ by applying the chain rule through each layer. Iterating this update over many epochs (and possibly minibatches) lets the model "fit" patterns in the training set. Although modern libraries automate these steps, understanding loss optimization and backpropagation is essential for diagnosing convergence issues, tuning hyperparameters (learning rate, batch size, etc.), and avoiding overfitting.

Model Fitting

- Definition: tuning model parameters to reduce prediction error
- Loss function $L(\theta)$ optimized via gradient descent
- Backpropagation computes gradients layer by layer
- Key hyperparameters: learning rate, batch size, epochs
- Understanding fitting helps with convergence diagnostics and generalization

5 Model Diagnostics

Once the network is trained, it's crucial to verify that it learned meaningful patterns rather than memorizing noise. In this section we cover loss-curve analysis, hyperparameter validation, error-propagation checks and final performance metrics.

5.1 Loss-Curve Analysis

We plot the training loss over iterations to ensure smooth decay and detect plateaus or oscillations:

$$L^{(t)} = \frac{1}{2N} \sum_{i=1}^{N} \big(y_i - \hat{y}_i^{(t)}\big)^2,$$

where $\hat{y}_i^{(t)}$ is the network's output at iteration t. A monotonically decreasing loss curve indicates stable learning; sudden spikes or flat regions suggest learning-rate issues or vanishing gradients.

```
plt.plot(clf.loss_curve_)
plt.title{Loss Curve}
plt.xlabel{Iterations}
plt.ylabel{Cost}
plt.show()
```

5.2 Hyperparameter Search

GridSearchCV systematically explores combinations of layer sizes, activation functions, solvers and regularization strengths. We examine cross-validation accuracy and standard deviation to choose a model that generalizes well:

- `hidden_layer_sizes`: (150,100,50), (120,80,40), (100,50,30)
- `activation`: logistic vs. relu
- `solver`: sgd vs. adam
- `alpha`: 1e–4, 5e–2, 1

- learning-rate schedules and initial rates

After fitting:

```
print(grid.best_params_)
print("Accuracy: {:.2f}".format(
    accuracy_score(y_test, grid.predict(X_test))
))
```

5.3 Custom Backpropagation Checks

To validate that gradients and weight updates behave correctly, we re-implement a minimal two-layer MLP. At each sample we compute:

$$\delta^2 = (y - a^2)\,\sigma'(z^2), \qquad \delta^1 = \big(W^2\,\delta^2\big) \circ \sigma'(z^1),$$

and update $W \leftarrow W + \eta\, a\, \delta,\ \ b \leftarrow b + \eta\, \delta.$ Convergence is signaled when the epoch loss falls below a small tolerance.

5.4 Final Performance Metrics

Beyond accuracy, inspect:

- Confusion matrix (precision, recall, F1-score)
- Receiver-Operating Characteristic (ROC) curve and AUC
- Calibration plots for predicted probabilities
- Profit or cost-benefit analysis if thresholds carry economic impact

—

- Plot loss curve $\rightarrow$ learning dynamics
- GridSearchCV $\rightarrow$ robust hyperparameters
- Backprop sanity-check $\rightarrow$ gradient weight updates
- Metrics: accuracy, confusion matrix, ROC/AUC, calibration
- Economic/profit evaluation via custom thresholds

6 Plotting & Discussion

Motivation & Core Research Question

- tk

7 Conclusion

Motivation & Core Research Question

- tk