Week #13: Statistical Learning and Stochastic Inference

Notes by: Francisco Richter

May 3, 2025

Aspect	**Structured (Traditional)**	**Unstructured (AI/ML)**
Input domain	$\mathcal{X} = \mathbb{R}^p$ with schema	$\mathcal{U}$ arbitrary (text, images, etc.), $\mathbf{U} \sim P_{\mathbf{U}}$
Mapping	Deterministic $f : \mathcal{X} \to \mathcal{Y}$	Learned $f_{\hat{\theta}}(\varphi(\mathbf{u}))$, $\hat{\theta} = \arg\min \widehat{R}_n$
Objective	None (hand-coded)	Empirical risk $\widehat{R}_n(\theta) = \frac{1}{n}\sum \ell(f_\theta(\varphi(\mathbf{u}_i)), \mathbf{y}_i)$
Correctness	$\sup_{\mathbf{x}} \lvert f(\mathbf{x}) - \mathbf{y}^*(\mathbf{x})\rvert = 0$	Statistical: $R(\hat{\theta}) \approx \min_\theta R(\theta)$
Output	Exact $\mathbf{y} = f(\mathbf{x})$, reproducible	Estimate $\hat{\mathbf{y}} = f_{\hat{\theta}}(\varphi(\mathbf{u}))$, has $\mathrm{Var}[\hat{\mathbf{y}}]$
Reproducibility	$\forall \mathbf{x}$, same $\mathbf{y}$ each run	Depends on random seed / retraining; may vary if $\hat{\theta}$ differs
Validation	Unit/integration tests	Estimate generalization gap $\lvert R(\hat{\theta}) - \widehat{R}_n(\hat{\theta})\rvert$
Monitoring	Uptime, exception rates	Drift: monitor $D(P_{\mathbf{U}} \Vert P_{\mathbf{U}}^{\text{prod}})$, test risk in production
Failure mode	Crash or logic bug	Elevated error rate $\mathrm{Err} = \mathbb{E}[\mathbf{1}\{f_{\hat{\theta}}(\mathbf{U}) \neq \mathbf{Y}\}]$

Table 1: Deterministic vs. Probabilistic Paradigms under Structured and Unstructured Inputs

1 Stochastic Program

To bridge the gap between deterministic and probabilistic paradigms, we introduce the stochastic program abstraction, which formally separates algorithmic logic, parameters, and randomness.

Definition. Program

A *program* is a triple $(M, \theta, P_\varepsilon)$ where

$$M : I \times \Omega \to O \quad \text{(deterministic map)}, \quad \theta \in \Theta \quad \text{(parameters)}, \quad \varepsilon \sim P_\varepsilon \quad \text{(randomness)},$$

so that for input $X = x$,

$$Y = M_\theta(x, \varepsilon), \qquad Y \mid X = x \sim p_\theta(\cdot \mid x).$$

Intuition: Stochastic program components

Think of a stochastic program as a recipe with three distinct parts: the cooking procedure (the deterministic map M), the ingredient proportions (parameters θ), and the inherent variability in cooking conditions (randomness ε). Even with the same recipe and proportions, slight variations in temperature or ingredient quality lead to different outcomes each time.

Remark.

This abstraction clarifies roles:

- M: the fixed computation or algorithm,
- θ: the unknown quantities to be *learned*,
- ε: the source of nondeterminism or noise.

"Learning" = estimating θ; "Inference" = assessing uncertainty in θ.

Example 1.1 (Binary Classifier)**.** *A convolutional neural network with weights θ and dropout mask ε defines*

$$p_\theta(y \mid x) = \mathit{softmax}(f_\theta(x; \varepsilon)).$$

Training maximizes the joint likelihood over data, while dropout regularizes by introducing randomness within M.

Example 1.2 (Bayesian Linear Regression)**.** *In Bayesian linear regression, we have*

$$M_\theta(x, \varepsilon) = \theta^T x + \varepsilon, \quad \varepsilon \sim \mathcal{N}(0, \sigma^2),$$

which induces the conditional distribution

$$Y \mid X = x \sim \mathcal{N}(\theta^T x, \sigma^2).$$

The parameters θ are estimated from data, while the noise ε captures inherent variability in the relationship between X and Y.

The stochastic program abstraction provides several benefits:

- Separates concerns: Distinguishes between algorithmic logic, learned parameters, and randomness
- Enables formal analysis: Allows rigorous treatment of uncertainty and error
- Bridges paradigms: Connects traditional programming with statistical learning
- Clarifies learning objectives: Focuses on estimating parameters rather than designing algorithms

---

2 Estimation Principles

We now explore fundamental principles for estimating parameters in statistical models. These principles form the foundation for most machine learning algorithms.

2.1 Maximum Likelihood Estimation

Maximum Likelihood Estimation (MLE) is a method for estimating the parameters of a statistical model by maximizing the likelihood function.

Definition. Maximum Likelihood Estimation

Given independent and identically distributed (i.i.d.) observations $X_1, X_2, \ldots, X_n$ from a distribution with probability density function $f(x;\theta)$, the log-likelihood function is

$$\ell_n(\theta) = \sum_{i=1}^{n} \log f(X_i;\theta),$$

and the maximum likelihood estimator (MLE) is

$$\hat{\theta}_{\text{MLE}} = \arg\max_{\theta\in\Theta} \ell_n(\theta).$$

Intuition: Maximum likelihood

MLE finds the parameter values that make the observed data most probable. It's like adjusting the settings of a camera until the resulting photo looks as close as possible to what you're seeing with your own eyes. The "best" settings are those that produce the image most similar to reality.

Theorem 2.1 (Properties of MLE)**.** *Under regularity conditions (identifiability, smoothness, non-singular Fisher information $I(\theta)$), the MLE has the following properties:*

- Consistency*: $\hat{\theta}_n \to \theta_0$ in probability as $n \to \infty$.*
- Asymptotic normality*: $\sqrt{n}(\hat{\theta}_n - \theta_0) \xrightarrow{d} \mathcal{N}(0, I(\theta_0)^{-1})$ as $n \to \infty$.*
- Efficiency*: Achieves the Cramér-Rao lower bound among unbiased estimators.*

Remark.

The Fisher information

$$I(\theta) = \mathbb{E}_\theta\left[-\nabla^2 \log f(X;\theta)\right]$$

quantifies how much each observation informs about θ; its inverse gives the asymptotic covariance of $\hat{\theta}_n$.

Example 2.1 (MLE for Normal Distribution)**.** *Consider i.i.d. observations $X_1, X_2, \ldots, X_n$ from $\mathcal{N}(\mu, \sigma^2)$. The log-likelihood is*

$$\ell_n(\mu, \sigma^2) = -\frac{n}{2}\log(2\pi) - \frac{n}{2}\log(\sigma^2) - \frac{1}{2\sigma^2}\sum_{i=1}^{n}(X_i - \mu)^2.$$

Taking partial derivatives and setting them to zero:

$$\frac{\partial \ell_n}{\partial \mu} = \frac{1}{\sigma^2}\sum_{i=1}^{n}(X_i - \mu) = 0 \tag{1}$$

$$\frac{\partial \ell_n}{\partial \sigma^2} = -\frac{n}{2\sigma^2} + \frac{1}{2\sigma^4}\sum_{i=1}^{n}(X_i - \mu)^2 = 0 \tag{2}$$

Solving these equations yields the MLEs:

$$\hat{\mu}_{MLE} = \frac{1}{n}\sum_{i=1}^{n} X_i = \bar{X} \tag{3}$$

$$\hat{\sigma}^2_{MLE} = \frac{1}{n}\sum_{i=1}^{n}(X_i - \hat{\mu})^2 \tag{4}$$

2.2 Least Squares Estimation

Least Squares Estimation is a method for estimating parameters by minimizing the sum of squared differences between observed and predicted values.

Definition. Least Squares Estimation

Given a model $Y_i = f_\theta(X_i) + \varepsilon_i$ with $\varepsilon_i \sim \mathcal{N}(0, \sigma^2)$, the least squares estimator is

$$\hat{\theta}_{\text{LS}} = \arg\min_\theta \sum_{i=1}^{n} [Y_i - f_\theta(X_i)]^2.$$

Remark.

For models with normally distributed errors, least squares estimation is equivalent to maximum likelihood estimation.

Example 2.2 (Linear Regression)**.** *For the linear model $f_\theta(x) = \beta_0 + x^T\beta$, the least squares estimator is*

$$\hat{\beta} = (X^T X)^{-1} X^T y,$$

where X is the design matrix with rows X_i^T and y is the vector of responses Y_i.

Theorem 2.2 (Gauss-Markov Theorem)**.** *Under the assumptions of the linear model with uncorrelated errors having equal variance, the least squares estimator is the Best Linear Unbiased Estimator (BLUE), meaning it has the smallest variance among all linear unbiased estimators.*

2.3 Regularization

Regularization techniques address overfitting by adding a penalty term to the objective function, effectively shrinking parameter estimates toward zero or some other reference value.

Definition. Regularized Estimation

A regularized estimator minimizes the penalized negative log-likelihood or sum of squares:

$$\hat{\theta}_\lambda = \arg\min_\theta \left\{ -\ell_n(\theta) + \lambda R(\theta) \right\},$$

where $\lambda > 0$ is the regularization parameter and $R(\theta)$ is the penalty function.

Common penalty functions include:

- Ridge penalty: $R(\theta) = \|\theta\|_2^2$ (sum of squared parameters)
- Lasso penalty: $R(\theta) = \|\theta\|_1$ (sum of absolute parameters)
- Elastic Net: $R(\theta) = \alpha\|\theta\|_1 + (1-\alpha)\|\theta\|_2^2$ (combination of L1 and L2)

Intuition: Regularization

Regularization is like adding a cost for complexity. Without regularization, a model might create an overly complex explanation that perfectly fits the training data but fails to generalize. Regularization penalizes this complexity, forcing the model to find simpler explanations that might be slightly less accurate on training data but generalize better to new data.

Remark.

From a Bayesian viewpoint, regularization corresponds to imposing a prior distribution on the parameters:

- Ridge regression $\leftrightarrow$ Gaussian prior $\theta \sim \mathcal{N}(0, \tau^2 I)$
- Lasso regression $\leftrightarrow$ Laplace prior $p(\theta) \propto \exp(-\lambda\|\theta\|_1)$

The regularization parameter λ controls the bias-variance trade-off.

Example 2.3 (Ridge Regression)**.** *For the linear model with ridge penalty, the estimator is*

$$\hat{\beta}_{ridge} = (X^T X + \lambda I)^{-1} X^T y.$$

The ridge penalty shrinks all coefficients toward zero, but never exactly to zero.

Example 2.4 (Lasso Regression)**.** *For the linear model with lasso penalty, there is no closed-form solution, but efficient algorithms like coordinate descent can find the solution. The lasso has the property of variable selection: it can shrink some coefficients exactly to zero, effectively removing those features from the model.*

Practice Problems

1. Derive the MLE for the parameter p of a Bernoulli distribution based on n independent observations.
2. Show that for the linear regression model with normally distributed errors, the least squares estimator is identical to the maximum likelihood estimator.
3. Consider a ridge regression problem with design matrix X and response vector y. How does the effective degrees of freedom of the model change as the regularization parameter λ increases from 0 to ∞?

Solution:

For problem 1: Let $X_1, X_2, \ldots, X_n$ be i.i.d. Bernoulli(p) random variables. The likelihood function is

$$L(p) = \prod_{i=1}^{n} p^{X_i}(1-p)^{1-X_i} = p^{\sum X_i}(1-p)^{n-\sum X_i}$$

The log-likelihood is

$$\ell(p) = \left(\sum_{i=1}^{n} X_i\right) \log p + \left(n - \sum_{i=1}^{n} X_i\right) \log(1-p)$$

Taking the derivative with respect to p and setting it to zero:

$$\frac{d\ell}{dp} = \frac{\sum X_i}{p} - \frac{n - \sum X_i}{1-p} = 0$$

Solving for p gives the MLE:

$$\hat{p}_{\text{MLE}} = \frac{1}{n}\sum_{i=1}^{n} X_i = \bar{X}$$

---

3 Bootstrap Inference

The bootstrap is a powerful resampling technique for estimating the sampling distribution of a statistic and constructing confidence intervals when analytical expressions are unavailable or complex.

3.1 The Bootstrap Principle

Definition. Nonparametric Bootstrap

Given data $X_1, X_2, \ldots, X_n$ and an estimator $\hat{\theta} = t(X_1, X_2, \ldots, X_n)$:

1. For $b = 1, 2, \ldots, B$, draw a bootstrap sample $(X_1^{*(b)}, X_2^{*(b)}, \ldots, X_n^{*(b)})$ by sampling with replacement from $\{X_i\}$.
2. Compute the bootstrap replicate $\hat{\theta}^{*(b)} = t(X_1^{*(b)}, X_2^{*(b)}, \ldots, X_n^{*(b)})$.

The empirical distribution of $\{\hat{\theta}^{*(b)}\}_{b=1}^{B}$ approximates the sampling distribution of $\hat{\theta}$.

Intuition: Bootstrap

The bootstrap treats the observed data as a stand-in for the entire population. By resampling from this data (with replacement), we simulate drawing new samples from the population. The variation in statistics calculated from these resamples approximates how the statistic would vary if we could repeatedly sample from the true population.

3.2 Bootstrap Standard Error and Confidence Intervals

The bootstrap standard error is estimated as:

$$\widehat{\mathrm{SE}}_{\mathrm{boot}} = \sqrt{\frac{1}{B-1}\sum_{b=1}^{B}(\hat{\theta}^{*(b)} - \bar{\hat{\theta}}^{*})^2}, \quad \bar{\hat{\theta}}^{*} = \frac{1}{B}\sum_{b=1}^{B}\hat{\theta}^{*(b)}$$

Several methods exist for constructing bootstrap confidence intervals:

3.2.1 Percentile Method

The percentile method uses quantiles of the bootstrap distribution:

$$\mathrm{CI}_{1-\alpha} = [\hat{\theta}^{*(\alpha/2)}, \hat{\theta}^{*(1-\alpha/2)}]$$

where $\hat{\theta}^{*(q)}$ is the q-quantile of the bootstrap distribution.

3.2.2 Basic Bootstrap Method

The basic bootstrap method uses the bootstrap distribution to estimate the sampling distribution of $\hat{\theta} - \theta$:

$$\mathrm{CI}_{1-\alpha} = [2\hat{\theta} - \hat{\theta}^{*(1-\alpha/2)}, 2\hat{\theta} - \hat{\theta}^{*(\alpha/2)}]$$

3.2.3 BCa Method (Bias-Corrected and Accelerated)

The BCa method adjusts for bias and non-constant variance:

$$\mathrm{CI}_{1-\alpha} = [\hat{\theta}^{*(a_1)}, \hat{\theta}^{*(a_2)}]$$

where

$$a_1 = \Phi\left(z_0 + \frac{z_0 + z_{\alpha/2}}{1 - a(z_0 + z_{\alpha/2})}\right) \tag{5}$$

$$a_2 = \Phi\left(z_0 + \frac{z_0 + z_{1-\alpha/2}}{1 - a(z_0 + z_{1-\alpha/2})}\right) \tag{6}$$

Algorithm 1 Nonparametric Bootstrap for Confidence Intervals

Require: Data $\{X_1, X_2, \ldots, X_n\}$, estimator function $t(\cdot)$, number of bootstrap samples B, confidence level $1 - \alpha$

1: Compute the original estimate $\hat{\theta} = t(X_1, X_2, \ldots, X_n)$
2: **for** $b = 1$ to B **do**
3: Draw a bootstrap sample $\{X_1^{*(b)}, X_2^{*(b)}, \ldots, X_n^{*(b)}\}$ by sampling with replacement from $\{X_i\}$
4: Compute bootstrap replicate $\hat{\theta}^{*(b)} = t(X_1^{*(b)}, X_2^{*(b)}, \ldots, X_n^{*(b)})$
5: **end for**
6: Sort the bootstrap replicates: $\hat{\theta}^{*(1)} \leq \hat{\theta}^{*(2)} \leq \ldots \leq \hat{\theta}^{*(B)}$
7: Compute percentile confidence interval: $[\hat{\theta}^{*(\lfloor B\alpha/2 \rfloor)}, \hat{\theta}^{*(\lceil B(1-\alpha/2) \rceil)}]$
8: **return** Bootstrap distribution $\{\hat{\theta}^{*(b)}\}_{b=1}^{B}$ and confidence interval

with z_0 and a being the bias-correction and acceleration parameters.

Remark.

The bootstrap is model-agnostic and often quite accurate for moderate sample sizes. However, it can be computationally intensive, and adjustments (bias correction, BCa intervals) improve accuracy in small samples or when the statistic has high bias or skewness.

3.3 Parametric Bootstrap

The parametric bootstrap assumes a parametric model for the data:

Definition. Parametric Bootstrap

Given data $X_1, X_2, \ldots, X_n$ and a parametric model F_θ:

1. Estimate $\hat{\theta}$ from the original data.
2. For $b = 1, 2, \ldots, B$, generate a bootstrap sample $(X_1^{*(b)}, X_2^{*(b)}, \ldots, X_n^{*(b)})$ from the fitted model $F_{\hat{\theta}}$.
3. Compute the bootstrap replicate $\hat{\theta}^{*(b)}$ from the bootstrap sample.

Example 3.1 (Bootstrap for Regression Coefficients)**.** *Consider a linear regression model $Y = X\beta + \varepsilon$ with $\hat{\beta} = (X^T X)^{-1} X^T y$. To construct a bootstrap confidence interval for β_j:*

1. Compute residuals $\hat{\varepsilon}_i = y_i - x_i^T \hat{\beta}$ 2. For $b = 1$ to B: a. Resample residuals $\hat{\varepsilon}_i^{*(b)}$ with replacement b. Generate bootstrap responses $y_i^{*(b)} = x_i^T \hat{\beta} + \hat{\varepsilon}_i^{*(b)}$ c. Compute $\hat{\beta}^{*(b)} = (X^T X)^{-1} X^T y^{*(b)}$ 3. Use the distribution of $\{\hat{\beta}_j^{*(b)}\}_{b=1}^{B}$ to construct confidence intervals for β_j

Practice Problems

1. Implement the nonparametric bootstrap to estimate the standard error and construct a 95% confidence interval for the median of a dataset.
2. Compare the coverage properties of percentile, basic, and BCa bootstrap confidence intervals for the correlation coefficient between two variables with a sample size of $n = 30$.
3. Explain why the bootstrap might fail for estimating the maximum of a uniform distribution. Propose a modification to improve its performance in this case.

4 Model Gallery

We now present a spectrum of statistical models, each with increasing flexibility and complexity. These models form the foundation of modern statistical learning and inference.

4.1 Linear Regression

Linear regression models the relationship between a response variable and one or more predictor variables as a linear function.

Definition. Linear Regression Model

The linear regression model is defined as:

$$Y = X\beta + \varepsilon, \quad \varepsilon \sim \mathcal{N}(0, \sigma^2 I)$$

where $Y \in \mathbb{R}^n$ is the response vector, $X \in \mathbb{R}^{n \times p}$ is the design matrix, $\beta \in \mathbb{R}^p$ is the coefficient vector, and $\varepsilon \in \mathbb{R}^n$ is the error vector.

- Estimation: Ordinary Least Squares (OLS) $\hat{\beta} = (X^T X)^{-1} X^T y$
- Inference: $\hat{\beta} \sim \mathcal{N}(\beta, \sigma^2 (X^T X)^{-1})$
- Prediction: $\hat{y} = x^T \hat{\beta}$ with prediction variance $\mathrm{Var}(\hat{y}) = \sigma^2 x^T (X^T X)^{-1} x$

Example 4.1 (Housing Price Prediction)**.** *To predict housing prices, we might use a linear regression model with predictors such as square footage, number of bedrooms, and neighborhood median income. The coefficient for square footage represents the expected increase in price for each additional square foot, holding other variables constant.*

4.2 Logistic Regression

Logistic regression models the probability of a binary outcome as a function of predictor variables.

Definition. Logistic Regression Model

The logistic regression model is defined as:

$$\Pr(Y = 1 \mid x) = \frac{1}{1 + e^{-x^T \beta}} = \sigma(x^T \beta)$$

where $\sigma(z) = \frac{1}{1+e^{-z}}$ is the logistic (sigmoid) function.

- Estimation: Maximum Likelihood using Iteratively Reweighted Least Squares (IRLS):

  $$\beta \leftarrow (X^T W X)^{-1} X^T W z$$

  where $W_{ii} = \pi_i(1 - \pi_i)$ and $z_i = \eta_i + \frac{y_i - \pi_i}{\pi_i(1-\pi_i)}$
- Inference: $\hat{\beta} \approx \mathcal{N}(\beta, (X^T W X)^{-1})$
- Prediction: $\hat{\pi} = \sigma(x^T \hat{\beta})$

Example 4.2 (Credit Default Prediction)**.** *Logistic regression can predict the probability of a customer defaulting on a loan based on features like credit score, income, and debt-to-income ratio. The coefficients indicate how each feature affects the log-odds of default.*

4.3 Generalized Linear Models (GLMs)

GLMs extend linear regression to handle response variables with non-normal distributions through a link function.

Definition. Generalized Linear Model

A GLM consists of three components:

1. A random component: The response variable Y follows an exponential family distribution
$$p(y;\theta_i,\phi) = \exp\left\{\frac{y\theta_i - b(\theta_i)}{\phi} + c(y,\phi)\right\}$$
2. A systematic component: The linear predictor $\eta_i = x_i^T\beta$
3. A link function: $g(\mu_i) = \eta_i$ where $\mu_i = \mathbb{E}[Y_i] = b'(\theta_i)$

Common GLMs include:

- Linear regression: Normal distribution with identity link
- Logistic regression: Binomial distribution with logit link
- Poisson regression: Poisson distribution with log link
- Gamma regression: Gamma distribution with inverse link
- Estimation: Maximum Likelihood using IRLS
- Model assessment: Deviance $D = 2[\ell_{\text{sat}} - \ell(\hat{\beta})]$, which is asymptotically χ^2 under the correct model

Example 4.3 (Insurance Claim Modeling)**.** *For modeling insurance claim amounts, which are always positive and often right-skewed, a Gamma GLM with log link might be appropriate. Predictors could include driver age, vehicle type, and driving history.*

4.4 Generalized Additive Models (GAMs)

GAMs extend GLMs by allowing nonlinear relationships between predictors and the response through smooth functions.

Definition. Generalized Additive Model

A GAM has the form:
$$g(\mu_i) = \beta_0 + \sum_{j=1}^{p} f_j(x_{ij})$$
where f_j are smooth functions, typically represented as:
$$f_j(x) = \sum_{k=1}^{K_j} \theta_{jk} B_{jk}(x)$$
with B_{jk} being basis functions (e.g., splines).

- Estimation: Penalized likelihood maximization:

$$\ell(\theta) - \sum_{j=1}^{p} \lambda_j \theta_j^T \Omega_j \theta_j$$

  where Ω_j penalizes roughness via $\int [f_j''(t)]^2 dt$

- Model selection: Generalized Cross-Validation (GCV) or Restricted Maximum Likelihood (REML)

Example 4.4 (Environmental Modeling)**.** *GAMs are widely used in environmental science to model relationships between species abundance and environmental variables. For instance, fish abundance might have a nonlinear relationship with water temperature, with an optimal range and decreasing abundance at both higher and lower temperatures.*

4.5 Neural Networks

Neural networks are flexible models capable of learning complex, nonlinear relationships through compositions of simple functions.

Definition. Feedforward Neural Network

A feedforward neural network with L layers computes:

$$h^{(0)} = x \tag{7}$$

$$h^{(\ell)} = \sigma(W^{(\ell)} h^{(\ell-1)} + b^{(\ell)}) \quad \text{for } \ell = 1, 2, \ldots, L-1 \tag{8}$$

$$\hat{y} = f^{(L)}(h^{(L-1)}) \tag{9}$$

where σ is an activation function (e.g., ReLU, sigmoid), $W^{(\ell)}$ and $b^{(\ell)}$ are the weights and biases of layer ℓ, and $f^{(L)}$ is the output function.

- Training: Backpropagation with stochastic gradient descent:

$$\delta^{(L)} = \nabla_{\hat{y}} \mathcal{L} \odot f^{(L)\prime} \tag{10}$$

$$\delta^{(\ell)} = (W^{(\ell+1)T} \delta^{(\ell+1)}) \odot \sigma'(z^{(\ell)}) \tag{11}$$

$$\nabla_{W^{(\ell)}} \mathcal{L} = \delta^{(\ell)} h^{(\ell-1)T} \tag{12}$$

- Regularization: Dropout, weight decay, early stopping
- Architecture selection: Cross-validation, grid search

Theorem 4.1 (Universal Approximation Theorem)**.** *A feedforward neural network with a single hidden layer containing a finite number of neurons can approximate any continuous function on a compact domain to arbitrary precision, given sufficient neurons.*

Example 4.5 (Image Classification)**.** *Convolutional neural networks, a specialized type of neural network, excel at image classification tasks. They use convolutional layers to automatically learn hierarchical features from images, from simple edges and textures in early layers to complex objects in deeper layers.*

Practice Problems

1. Derive the maximum likelihood estimator for the parameters of a Poisson regression model where $\log(\mu_i) = x_i^T \beta$.
2. Compare the flexibility and interpretability trade-offs between linear models, GAMs, and neural networks. Provide examples of situations where each model would be most appropriate.

3. Implement a simple neural network with one hidden layer to approximate the function $f(x) = \sin(x)$ over the interval $[-\pi, \pi]$. Experiment with different numbers of hidden units and activation functions.

References

[R1] Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The Elements of Statistical Learning*. Springer.

[R2] Murphy, K. P. (2012). *Machine Learning: A Probabilistic Perspective*. MIT Press.

[R3] McCullagh, P., & Nelder, J. A. (1989). *Generalized Linear Models*. Chapman and Hall.

[R4] Hastie, T., & Tibshirani, R. (1990). *Generalized Additive Models*. Chapman and Hall.

[R5] Efron, B., & Tibshirani, R. J. (1994). *An Introduction to the Bootstrap*. Chapman and Hall/CRC.

[R6] Cybenko, G. (1989). Approximation by Superpositions of a Sigmoidal Function. *Mathematics of Control, Signals, and Systems*, 2(4), 303-314.

[R7] Hornik, K. (1991). Approximation Capabilities of Multilayer Feedforward Networks. *Neural Networks*, 4(2), 251-257.

[R8] Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press.

[R9] Park, T., & Casella, G. (2008). The Bayesian Lasso. *Journal of the American Statistical Association*, 103(482), 681-686.

[R10] Wood, S. N. (2017). *Generalized Additive Models: An Introduction with R*. Chapman and Hall/CRC.

[R11] Glorot, X., & Bengio, Y. (2010). Understanding the Difficulty of Training Deep Feedforward Neural Networks. *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, 249-256.