Sample Study Notes - Introduction to Machine Learning

What is Machine Learning?

Machine Learning (ML) is a subset of artificial intelligence that enables computers to learn from data without being explicitly programmed. Instead of following hard-coded rules, ML systems improve their performance through experience.

Types of Machine Learning

1. Supervised Learning

Supervised learning uses labeled training data to learn the mapping between inputs and outputs. The algorithm learns from examples where the correct answer is already known.

Common Algorithms:

- Linear Regression: Predicts continuous values
- Logistic Regression: Binary classification
- Decision Trees: Tree-like model of decisions
- Support Vector Machines (SVM): Finds optimal decision boundary
- Neural Networks: Interconnected layers of neurons

Applications:

- Email spam detection
- Image classification
- Price prediction
- Medical diagnosis

2. Unsupervised Learning

Unsupervised learning works with unlabeled data to discover hidden patterns or structures. The algorithm must find relationships in the data on its own.

Common Algorithms:

- K-Means Clustering: Groups similar data points
- Hierarchical Clustering: Creates tree of clusters

- Principal Component Analysis (PCA): Dimensionality reduction
- Anomaly Detection: Identifies outliers

Applications:

- Customer segmentation
- Recommendation systems
- Data compression
- Fraud detection

3. Reinforcement Learning

Reinforcement learning trains agents to make sequences of decisions by rewarding desired behaviors and punishing undesired ones. The agent learns through trial and error.

Key Concepts:

- Agent: The learner/decision maker
- Environment: What the agent interacts with
- State: Current situation of the agent
- Action: What the agent can do
- Reward: Feedback from environment

Applications:

- Game playing (AlphaGo, chess engines)
- Robotics control
- Autonomous vehicles
- Resource management

The Machine Learning Pipeline

Step 1: Data Collection

Gather relevant data from various sources. Quality and quantity matter - more diverse data generally leads to better models.

Step 2: Data Preprocessing

Clean and prepare data for training:

- Handle missing values
- Remove outliers

- Normalize/standardize features
- Encode categorical variables
- Split into training/validation/test sets

Step 3: Feature Engineering

Create new features or transform existing ones to improve model performance. This requires domain expertise.

Step 4: Model Selection

Choose appropriate algorithm(s) based on:

- Problem type (classification, regression, clustering)
- Data size and dimensionality
- Interpretability requirements
- Computational resources

Step 5: Training

Feed training data to the model so it can learn patterns. The model adjusts its internal parameters to minimize prediction errors.

Step 6: Evaluation

Assess model performance using appropriate metrics:

- **Classification**: Accuracy, Precision, Recall, F1-Score, AUC-ROC
- **Regression**: Mean Squared Error (MSE), R-squared, Mean Absolute Error (MAE)
- **Clustering**: Silhouette Score, Davies-Bouldin Index

Step 7: Hyperparameter Tuning

Optimize model settings that aren't learned during training:

- Learning rate
- Number of layers/neurons
- Regularization parameters
- Tree depth

Common techniques:

- Grid Search: Try all combinations
- Random Search: Sample random combinations
- Bayesian Optimization: Smart parameter selection

Step 8: Deployment

Put the trained model into production where it can make predictions on new data.

Key Concepts

Overfitting vs Underfitting

Overfitting: Model learns training data too well, including noise. Performs poorly on new data.

- Symptoms: High training accuracy, low test accuracy
- Solutions: More data, regularization, simpler model, dropout

Underfitting: Model is too simple to capture data patterns.

- Symptoms: Low training and test accuracy
- Solutions: More complex model, better features, less regularization

Bias-Variance Tradeoff

Bias: Error from oversimplified assumptions. High bias leads to underfitting. **Variance**: Error from sensitivity to training data fluctuations. High variance leads to overfitting.

Goal: Find sweet spot that minimizes both.

Cross-Validation

Technique to assess model generalization by training/testing on different data subsets. K-fold cross-validation divides data into K parts, trains on K-1 parts, and tests on the remaining part K times.

Regularization

Techniques to prevent overfitting by adding penalty for complexity:

- **L1 (Lasso)**: Encourages sparsity, can eliminate features
- **L2 (Ridge)**: Shrinks coefficients toward zero
- **Elastic Net**: Combination of L1 and L2

Deep Learning

Deep learning uses neural networks with multiple hidden layers to learn hierarchical representations of data.

Neural Network Components

Neurons: Basic processing units that receive inputs, apply weights, add bias, and pass through activation function.

Layers:

- Input Layer: Receives raw data
- Hidden Layers: Extract features (more layers = "deeper")
- Output Layer: Produces final prediction

Activation Functions:

- ReLU (Rectified Linear Unit): f(x) = max(0, x)
- Sigmoid: Outputs between 0 and 1
- Tanh: Outputs between -1 and 1
- Softmax: Converts scores to probabilities

Loss Functions:

- Mean Squared Error (MSE): For regression
- Cross-Entropy: For classification
- Huber Loss: Robust to outliers

Optimization Algorithms:

- Gradient Descent: Basic optimizer
- Stochastic Gradient Descent (SGD): Faster, uses batches
- Adam: Adaptive learning rate, widely used
- RMSprop: Good for recurrent networks

Common Architectures

Convolutional Neural Networks (CNNs): Specialized for image data using convolutional layers that detect spatial patterns.

Recurrent Neural Networks (RNNs): Process sequential data by maintaining internal state/memory. Good for time series and text.

Transformers: Attention-based architecture that processes sequences in parallel. Basis for modern NLP models like GPT and BERT.

Model Evaluation Best Practices

1. **Never test on training data**: Always use separate test set
2. **Use appropriate metrics**: Different problems need different metrics
3. **Consider class imbalance**: Use precision/recall, not just accuracy
4. **Visualize results**: Confusion matrices, ROC curves, learning curves
5. **Monitor during training**: Track validation metrics to detect overfitting early

Ethical Considerations

- **Bias in data**: Models inherit biases from training data
- **Fairness**: Ensure equal treatment across demographic groups
- **Privacy**: Protect sensitive user information
- **Transparency**: Make model decisions explainable
- **Safety**: Consider potential harms from predictions