Supervised Machine Learning: Regression and Classification

Week 1: Introduction to Machine Learning

Learning Objectives:

- Define machine learning
- Define supervised learning
- Define unsupervised learning
- Write and run Python code in Jupyter Notebooks
- Define a regression model
- Implement and visualize a cost function
- Implement gradient descent
- Optimize a regression model using gradient descent

Key Concepts:

Artificial intelligence (AI) is a set of technologies that enable computers to perform a variety of advanced functions, including the ability to see, understand and translate spoken and written language, analyze data, make recommendations, and more.

Machine Learning (ML) is a subset of AI that involves the use of algorithms and statistical models to enable computers to improve their performance on a specific task through experience (data) without being explicitly programmed.

Supervised learning is a type of ML where the model is trained on labeled data. It includes both inputs and their corresponding outputs, allowing the model to learn the mapping from inputs to outputs.

Types of Supervised Learning:

Regression: Predicts a continuous outcome.

Example: Predicting house prices.

Classification: Predicts a discrete label.

Example: Spam email detection.

Unsupervised learning is a type of ML where the model is trained on unlabeled data, allowing the model to find hidden patterns or intrinsic structures in the input data.

Types of Unsupervised Learning:

Clustering: Grouping similar data points together.

Example: Customer segmentation.

Anomaly Detection: Find unusual data points

Example: Fraud detection.

Dimensionality Reduction: Compress data using fewer numbers

A **cost function** measures the performance of a model by quantifying the error between the predicted values and the actual values. In regression, a common cost function is the Mean Squared Error (MSE).

MSE Formula:

$$J(\beta) = \frac{1}{2m} \sum_{i=1}^{m} (\hat{y}_i - y_i)^2$$

Gradient descent is an optimization algorithm used to minimize the cost function by iteratively adjusting the model parameters.

Gradient Descent Update Rule:

$$\beta_j := \beta_j - \alpha \frac{\partial J(\beta)}{\partial \beta_j}$$

Week 2: Regression with multiple input variables

Learning Objectives:

- Use vectorization to implement multiple linear regression
- Use feature scaling, feature engineering, and polynomial regression to improve model training
- Implement linear regression in code

Key Concepts:

NumPy is the fundamental package for scientific computing in Python. Essential for vectorized operations and handling arrays/matrices.

Multiple linear regression predicts a dependent variable using multiple independent variables

Formula:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_n x_n$$

Feature scaling standardizes data ranges. Common methods include:

- **Z-score Normalization**: Rescales data to have a mean of 0 and a standard deviation of 1. $z = \frac{x - \mu}{\sigma}$
- **Min-Max Scaling**: Scales data to a fixed range, usually 0 to 1.

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}}$$

Feature engineering creates new features to improve model performance.

The **learning rate (α)** controls gradient descent step size. If the learning rate is too high, the algorithm may overshoot the minimum; if too low, it may converge very slowly.

Polynomial regression models the relationship between the independent variable and the dependent variable as an n-th degree

Formula:

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \ldots + \beta_n x^n$$

Scikit-learn: Python library for implementing machine learning models.

Vectorization: Use matrix operations to speed up calculations.

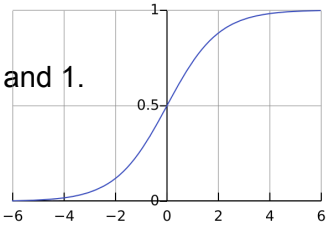
Week 3: Classification

Learning Objectives:

- Use logistic regression for binary classification
- Implement logistic regression for binary classification
- Address overfitting using regularization, to improve model performance

Key Concepts:

Sigmoid Function: Maps predicted values to probabilities between 0 and 1.



$$\sigma(z) = \frac{1}{1+e^{-z}}$$

Decision Boundary: Threshold for classifying predicted probabilities.

Overfitting: Model fits the training data too well and performs poorly on new data.

Regularization: Prevents overfitting by adding a penalty to the cost function.

Loss is a measure of the difference of a single example to its target value while the

Cost is a measure of the losses over the training set

$$loss(f_{\mathbf{w},b}(\mathbf{x}^{(i)}), y^{(i)}) = (-y^{(i)} \log\left(f_{\mathbf{w},b}\left(\mathbf{x}^{(i)}\right)\right) - \left(1 - y^{(i)}\right) \log\left(1 - f_{\mathbf{w},b}\left(\mathbf{x}^{(i)}\right)\right)$$