Linear Algebra and Optimization for Machine Learning
Lesson 1
Linear Regression, Vectors, and Matrices

Prof. Marcos M. Raimundo
Institute of Computing - UNICAMP

Linear Algebra for Machine Learning

- **Explore** the core concepts of linear algebra that empower machine learning algorithms.
- **Understand** how matrices, vectors, and linear transformations are the building blocks of data representation and model design.
- **Connect** theoretical knowledge to practical applications in machine learning, such as linear regression, dimensionality reduction, and more.

Vectors in Machine Learning

- A **vector** is an ordered list of numbers representing features or attributes of a data point.
- In machine learning, vectors are fundamental building blocks for representing and processing data.
- Each element within a vector corresponds to a specific characteristic or measurement of the data.

Examples of Structured Data Represented as Vectors:

- **Customer data:** (Age, Income, Purchase Frequency)
- **House data:** (Square Footage, Number of Bedrooms, Price)
- **Image data:** (Pixel Intensity Values)
- **Text data:** (Word Frequencies)

Vectors and Inner Product

Example:

Consider two vectors representing house data:

Vector $\mathbf{a} = (1500, 3, 250000)$ (Square footage, bedrooms, price) Vector $\mathbf{b} = (1200, 2, 180000)$

Inner Product (Dot Product):

The inner product (or dot product) of two vectors is calculated as:

$$\mathbf{a} \cdot \mathbf{b} = a_1 b_1 + a_2 b_2 + ... + a_n b_n$$

In our example:

$$\mathbf{a} \cdot \mathbf{b} = (1500)(1200) + (3)(2) + (250000)(180000) = 45,000,000,006$$

Vectors and Inner Product - ML Interpretations

The inner product often has geometric interpretations:

Similarity: In machine learning, a larger inner product between two vectors can indicate greater similarity between the data points they represent.

Projection: The inner product can be used to project one vector onto another, quantifying how much one vector lies in the direction of another.

Linear Regression: A Machine Learning Model

Goal: Predict a numerical output (dependent variable) based on one or more input features (independent variables) by fitting a linear equation to observed data.

Key Idea: Assume a linear relationship between the input features and the output:

$$\hat{y} = w_1 x_1 + w_2 x_2 + ... + w_n x_n + b$$

- $\hat{y}$ is the predicted output.
- $x_1, x_2, ..., x_n$ are the input features.
- $w_1, w_2, ..., w_n$ are the model's weights (coefficients) that determine the importance of each feature.
- b is the bias term (intercept).

Example (House Price Prediction): Predict house prices based on square footage, number of bedrooms, and other relevant features. The linear regression model would learn the relationship between these features and prices from historical data.

Matrices in Machine Learning

- A rectangular array of numbers, arranged in rows and columns.
- In machine learning, matrices are used to represent datasets, where each row typically represents a data point and each column represents a feature.

Example: House Price Prediction Imagine we have data for three houses, with features: square footage (sqft), number of bedrooms (bed), and number of bathrooms (bath):

Each row represents a house. Each column represents a feature.

$$\mathbf{X} = \begin{bmatrix} 1200 & 3 & 2 \\ 1500 & 4 & 2.5 \\ 1800 & 3 & 3 \end{bmatrix}$$

We also have the corresponding prices:

$$\mathbf{y} = \begin{bmatrix} 200000 \\ 250000 \\ 300000 \end{bmatrix}$$

Linear Regression using Matrices

We can write the linear regression model using matrices:

$$\hat{\mathbf{y}} = \mathbf{X}\mathbf{w}$$

where:

- $\hat{\mathbf{y}}$ is the vector of predicted prices.
- $\mathbf{X}$ is the matrix of features.
- $\mathbf{w}$ is the vector of weights (coefficients) that we want to learn.

Matrix Multiplication in Detail

Matrix multiplication is a fundamental operation in linear algebra with widespread applications in machine learning. It involves combining two matrices, let's call them **A** (with dimensions $m \times n$) and **B** (with dimensions $n \times p$), to produce a new matrix **C** (with dimensions $m \times p$). Here's how it works:

Key Points:

- **Dimensions Matter:** The number of columns in the first matrix (**A**) must equal the number of rows in the second matrix (**B**). This is essential for the multiplication to be defined.
- **Element-wise Calculation:** Each element c_{ij} in the resulting matrix **C** is calculated as the dot product of the ith row of **A** and the jth column of **B**.

$$c_{ij} = \sum_{k=1}^{n} a_{ik} b_{kj}$$

In simpler terms, to find the element in the ith row and jth column of the resulting matrix $\mathbf{C}$, you take the dot product of the corresponding row from $\mathbf{A}$ and column from $\mathbf{B}$.

$$\mathbf{A} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \quad \text{and} \quad \mathbf{B} = \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix}$$

To calculate $\mathbf{C} = \mathbf{AB}$, we perform the following:

$$c_{11} = (1)(5) + (2)(7) = 19; c_{12} = (1)(6) + (2)(8) = 22$$

$$c_{21} = (3)(5) + (4)(7) = 43; c_{22} = (3)(6) + (4)(8) = 50$$

Therefore $\mathbf{C} = \mathbf{AB} = \begin{bmatrix} 19 & 22 \\ 43 & 50 \end{bmatrix}$

Fundaments: Matrix Inverse

Concept:

The inverse of a square matrix $\mathbf{A}$, denoted as $\mathbf{A}^{-1}$, is a matrix that, when multiplied by $\mathbf{A}$, results in the identity matrix $\mathbf{I}$.

Key Points:

- **Identity Matrix:** The identity matrix $\mathbf{I}$ is a square matrix with ones on the diagonal and zeros elsewhere. It acts like the number 1 in regular multiplication.
- **Square Matrix:** Only square matrices (same number of rows and columns) can have an inverse.
- **Not All Matrices are Invertible:** A matrix has an inverse only if its determinant is non-zero.

Formula:

$$\mathbf{A}\mathbf{A}^{-1} = \mathbf{A}^{-1}\mathbf{A} = \mathbf{I}$$

Example:

Consider the matrix:

$$\mathbf{A} = \begin{bmatrix} 2 & 1 \\ 4 & 3 \end{bmatrix}$$

Its inverse is:

$$\mathbf{A}^{-1} = \begin{bmatrix} 1.5 & -0.5 \\ -2 & 1 \end{bmatrix}$$

We can verify this by multiplying:

$$\mathbf{A}\mathbf{A}^{-1} = \begin{bmatrix} 2 & 1 \\ 4 & 3 \end{bmatrix} \begin{bmatrix} 1.5 & -0.5 \\ -2 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \mathbf{I}$$

Fundaments: Determinant

Determinant:

- A scalar value associated with a square matrix that provides important information about its properties.
- Denoted as $\det(\mathbf{A})$ or $|\mathbf{A}|$.

Geometric Interpretation: In two dimensions, the absolute value of the determinant represents the area of the parallelogram formed by the matrix's column vectors. In three dimensions, it represents the volume of the parallelepiped formed by the vectors.

Properties:

- If $\det(\mathbf{A}) = 0$, the matrix is singular (not invertible).
- If $\det(\mathbf{A}) \neq 0$, the matrix is non-singular (invertible).
- $\det(\mathbf{A}^T) = \det(\mathbf{A})$ (The determinant of the transpose is the same).
- $\det(\mathbf{AB}) = \det(\mathbf{A})\det(\mathbf{B})$ (The determinant of a product is the product of determinants).

Calculation (for a 2x2 matrix):

For a matrix $\mathbf{A} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$, the determinant is:

$$\det(\mathbf{A}) = ad - bc$$

Key Takeaway: The determinant tells us whether a matrix is invertible. In the context of linear regression, the determinant of $(\mathbf{X}^T\mathbf{X})$ being non-zero is crucial for finding a unique solution using the normal equation.

Solving Linear Regression with Matrix Operations

Normal Equation: To find the optimal weights, we can use the normal equation, which provides a closed-form solution:

$$\mathbf{w} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

where:

- $\mathbf{X}^T$ is the transpose of the feature matrix.
- $(\mathbf{X}^T\mathbf{X})^{-1}$ is the inverse of the matrix product $\mathbf{X}^T\mathbf{X}$.

Solving Linear Regression with Matrix Operations

Prepare the Data: (1) Organize your features into the matrix $\mathbf{X}$. (2) Ensure $\mathbf{X}$ includes a column of ones for the intercept. (3) Form the vector $\mathbf{y}$ of your actual outputs.

Calculate the Transpose and Inverse: (1) Compute $\mathbf{X}^T$. (2) Calculate $(\mathbf{X}^T\mathbf{X})^{-1}$. Ensure that the inverse exists (the matrix must be full rank).

Solve for Weights: Multiply $(\mathbf{X}^T\mathbf{X})^{-1}$ by $\mathbf{X}^T$, and then by $\mathbf{y}$ to obtain the optimal weight vector $\mathbf{w}$.

Key Points:

- **Efficiency:** The normal equation provides a direct solution, but it can be computationally expensive for large datasets.
- **Alternatives:** For very large datasets, iterative methods like gradient descent are often preferred.

System of Linear Equations:

* A set of two or more linear equations involving the same set of variables. * Each equation represents a linear relationship between the variables.

Example:

$$\begin{aligned} 2x + 3y - z &= 4 \\ x - y + 2z &= -3 \\ 3x + y + z &= 1 \end{aligned}$$

We can express a system of linear equations compactly using matrices:

$$\mathbf{Ax} = \mathbf{b}$$

where: * **A** is the coefficient matrix (coefficients of the variables). * **x** is the vector of unknowns (variables). * **b** is the vector of constants.

Solving using Matrix Inverse (if possible):

If the matrix **A** is square and invertible (non-zero determinant), we can solve for **x**:

$$\mathbf{x} = \mathbf{A}^{-1}\mathbf{b}$$

Connection to Linear Regression:

The normal equation for linear regression, $\mathbf{w} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$, is a specific case of solving a system of linear equations. In this case:

* $\mathbf{A}$ is analogous to $(\mathbf{X}^T\mathbf{X})$. * $\mathbf{x}$ is analogous to $\mathbf{w}$. * $\mathbf{b}$ is analogous to $\mathbf{X}^T\mathbf{y}$.

Other Solution Methods:

Besides using the inverse, there are other methods to solve systems of linear equations, such as:

* Gaussian elimination * LU decomposition * Iterative methods

The choice of method depends on the size and properties of the system.

Linear Independence and Linear Regression

Linear Independence:

- A set of vectors is linearly independent if no vector in the set can be expressed as a linear combination of the other vectors.
- In other words, no vector in the set is redundant; each vector contributes unique information.

Mathematical Definition:

A set of vectors $\{\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_n\}$ is linearly independent if the only solution to the equation:

$$c_1\mathbf{v}_1 + c_2\mathbf{v}_2 + ... + c_n\mathbf{v}_n = \mathbf{0}$$

is the trivial solution where $c_1 = c_2 = ... = c_n = 0$.

Example:

The vectors $\begin{bmatrix} 1 \\ 0 \end{bmatrix}$ and $\begin{bmatrix} 0 \\ 1 \end{bmatrix}$ are linearly independent.

Linear Dependence in Linear Regression

Problem:

When features (columns of the feature matrix $\mathbf{X}$) are linearly dependent in a linear regression model, it leads to a problematic situation called **multicollinearity**.

Consequences of Multicollinearity:

- **Unstable Estimates:** The estimated weights become highly sensitive to small changes in the data, making the model unreliable.
- **Difficult Interpretation:** It becomes challenging to isolate the individual effect of each feature on the prediction, as their contributions are entangled.
- **Inflated Standard Errors:** The standard errors of the estimated weights increase, making it harder to determine which features are statistically significant.
- **Normal Equation Failure:** In severe cases, the matrix $\mathbf{X}^T\mathbf{X}$ becomes singular (non-invertible), preventing the direct solution of the normal equation.

Linear Dependence in Linear Regression

When features are linearly dependent, the matrix $\mathbf{X}^T\mathbf{X}$ has a determinant of zero. This means it lacks a unique inverse, causing numerical instability when trying to compute $(\mathbf{X}^T\mathbf{X})^{-1}$ in the normal equation.

Example: Imagine predicting house prices using square footage and another feature that's simply double the square footage. These features are perfectly linearly dependent. The model won't be able to distinguish their individual effects, leading to unreliable weight estimates.

Solutions: Feature Selection/Removal: Identify and remove the redundant features that are causing the linear dependency. **Regularization:** Techniques like Ridge Regression or Lasso Regression can help mitigate the effects of multicollinearity by adding a penalty term to the loss function. **Dimensionality Reduction:** Methods like Principal Component Analysis (PCA) can create new, uncorrelated features that capture most of the information in the original features.

Regularization in the Normal Equation and Collinearity

When dealing with multicollinearity (linear dependence among features) or to prevent overfitting, we modify the normal equation by adding a regularization term.

For Ridge Regression (L2 regularization), the modified normal equation becomes:

$$\mathbf{w} = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y}$$

where:

- λ is the regularization strength (a hyperparameter you choose).
- $\mathbf{I}$ is the identity matrix of the same size as $\mathbf{X}^T\mathbf{X}$.

Intuition: Think of the regularization term as adding a "penalty" for large weights. This encourages the model to find a simpler solution that doesn't rely too heavily on any single feature, thus mitigating the negative effects of multicollinearity.

Regularization in the Normal Equation and Effects on Collinearity

- **Improves Numerical Stability:** Adding the $\lambda \mathbf{I}$ term ensures that the matrix $(\mathbf{X}^T\mathbf{X} + \lambda \mathbf{I})$ is always invertible, even if $\mathbf{X}^T\mathbf{X}$ is singular due to collinearity. This makes the solution more numerically stable.
- **Shrinks Coefficient Estimates:** The regularization term penalizes large weight values, effectively shrinking the coefficients towards zero. This reduces the impact of multicollinearity, making the model less sensitive to small changes in the data.
- **Bias-Variance Tradeoff:** Regularization introduces a slight bias into the model to reduce variance. By shrinking the coefficients, we sacrifice some accuracy on the training data in exchange for better generalization performance on new data.
- **Choosing λ:** The value of λ controls the strength of regularization. A larger λ means stronger regularization (more shrinkage of coefficients), which can help address severe multicollinearity but may also lead to underfitting if too large. You need to tune λ through techniques like cross-validation to find the optimal balance.

Thank you for attending!

- **Key Takeaways:**
  - Linear algebra is the mathematical backbone of machine learning.
  - Matrices, vectors, and linear transformations are essential tools for representing and manipulating data.
  - Concepts like linear independence, matrix inverses, and determinants play crucial roles in solving problems and understanding models.

- **Further Exploration:**
  - Delve deeper into specific machine learning algorithms and their reliance on linear algebra.
  - Practice applying these concepts to real-world datasets and problems.
  - Continue your learning journey in the fascinating world of machine learning and data science.

Linear Algebra and Optimization for Machine Learning
Lesson 1
Linear Regression, Vectors, and Matrices

Prof. Marcos M. Raimundo
Institute of Computing - UNICAMP