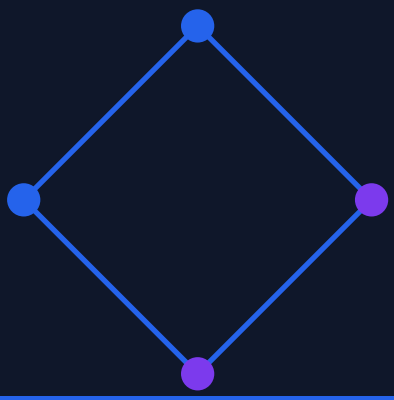
PMAI LAB

Mathematics • Python • Artificial Intelligence



LINEAR ALGEBRA FOR MACHINE LEARNING

Lecture 10

Integrated Machine-Learning Case Studies, Numerical Reliability, and Course Review

Lecture focus Combining the course methods into reliable machine-learning workflows for regression, recommendation, dimensionality reduction, and practical model evaluation.

Dr. Hira Benish

Associate Professor of Mathematics

Student Learning Handout | Includes NumPy practice and a capstone project

1 Lecture Overview

Machine-learning systems rarely use one linear-algebra method in isolation. A realistic workflow represents data as matrices, transforms features, fits a model, evaluates errors, and checks whether the computation is stable and meaningful.

This final lecture integrates the course through applied case studies and a reproducible Python mini-project. The emphasis is not only obtaining an answer, but selecting an appropriate method and defending the result.

Learning Outcomes

After completing this lecture, students should be able to:

- translate an applied problem into vectors, matrices, and dimensions;
- construct regression, recommendation, and PCA workflows;
- connect least squares, regularization, SVD, and embeddings;
- select a suitable numerical solver for a matrix problem;
- diagnose ill-conditioning, rank deficiency, and data leakage;
- evaluate residual, prediction, and reconstruction errors;
- build and interpret an end-to-end NumPy mini-project; and
- communicate mathematical choices, assumptions, and limitations.

1.1 A Reusable Applied Workflow

Stage	Central question
Represent	What do rows, columns, entries, and shapes mean?
Prepare	Must features be centered, scaled, encoded, or split?
Compute	Which factorization, solver, or transformation is appropriate?
Validate	Does the result generalize beyond the fitted data?
Diagnose	Are rank, conditioning, residuals, or leakage problematic?
Communicate	Can another person reproduce and interpret the result?

2 Case Study 1: Regression and Regularization

Suppose $X \in \mathbb{R}^{n \times d}$ contains n observations and d features, while $\mathbf{y} \in \mathbb{R}^n$ contains a continuous target. With an intercept column included, the linear model is

$$\mathbf{y} = X\boldsymbol{\beta} + \boldsymbol{\varepsilon}.$$

Least squares chooses

$$\widehat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}} \|X\boldsymbol{\beta} - \mathbf{y}\|_2^2.$$

The fitted values and residuals are

$$\widehat{\mathbf{y}} = X\widehat{\boldsymbol{\beta}}, \qquad \mathbf{r} = \mathbf{y} - \widehat{\mathbf{y}}.$$

2.1 Ridge Regularization

When predictors are strongly correlated or the problem is poorly conditioned, ridge regression solves

$$\widehat{\boldsymbol{\beta}}_\lambda = \arg\min_{\boldsymbol{\beta}} \left(\|X\boldsymbol{\beta} - \mathbf{y}\|_2^2 + \lambda \|\boldsymbol{\beta}\|_2^2 \right), \qquad \lambda \geq 0.$$

Its normal-equation form is

$$(X^T X + \lambda I)\widehat{\boldsymbol{\beta}}_\lambda = X^T \mathbf{y}.$$

The intercept is usually excluded from the penalty.

2.2 Evaluation

On an unseen test set of size m,

$$\text{RMSE} = \sqrt{\frac{1}{m} \sum_{i=1}^{m} (y_i - \widehat{y}_i)^2}.$$

Compare the model against a simple baseline, such as predicting the training-target mean.

Practical Decisions

Split the data before estimating means, standard deviations, or model parameters. Fit every preprocessing step using the training set only, then apply the learned transformation to the test set.

Machine-Learning Connection

Least squares provides the modeling objective, projections explain the fitted vector, and ridge regularization stabilizes weak or correlated feature directions.

3 Case Study 2: Recommendation Systems

Let $R \in \mathbb{R}^{m \times n}$ be a user-item interaction matrix. Rows represent users, columns represent items, and observed entries may contain ratings, purchases, clicks, or other preference signals.

3.1 Similarity-Based Recommendation

Two users represented by row vectors $\mathbf{r}_i$ and $\mathbf{r}_j$ can be compared using cosine similarity:

$$s_{ij} = \frac{\mathbf{r}_i^T \mathbf{r}_j}{\|\mathbf{r}_i\|_2 \|\mathbf{r}_j\|_2}.$$

Recommendations can be generated from items preferred by similar users. Sparse overlap and popularity effects must be handled carefully.

3.2 Latent-Factor Recommendation

A rank-k approximation is

$$R \approx U_k \Sigma_k V_k^T.$$

Define

$$P = U_k \Sigma_k^{1/2}, \qquad Q = V_k \Sigma_k^{1/2}.$$

Then

$$R \approx P Q^T, \qquad \widehat{r}_{ij} = \mathbf{p}_i^T \mathbf{q}_j.$$

The rows of P are user embeddings and the rows of Q are item embeddings. Their inner product estimates compatibility.

3.3 Real-World Caution

Missing interactions are not necessarily negative preferences. Ordinary SVD on a matrix filled with zeros may distort the meaning of unobserved entries. Production methods often minimize error only over observed interactions and may include bias and regularization terms.

Ranking Evaluation

Prediction error alone may not match the business objective. Recommendation systems are commonly evaluated with ranking measures such as precision at k, recall at k, coverage, and performance for new or less-active users.

Machine-Learning Connection

Recommendation systems connect vector similarity, low-rank approximation, embeddings, and dot-product scoring in one commercially important application.

4 Case Study 3: PCA in a Learning Pipeline

Let $X \in \mathbb{R}^{n \times d}$ contain correlated numerical features. Center the training data and compute

$$X_c = U\Sigma V^T.$$

The first k principal directions are the columns of V_k, and the reduced representation is

$$Z_k = X_c V_k \in \mathbb{R}^{n \times k}.$$

4.1 Selecting the Number of Components

The cumulative explained-variance ratio is

$$\text{CEV}(k) = \frac{\sum_{i=1}^{k} \sigma_i^2}{\sum_{i=1}^{r} \sigma_i^2}.$$

A threshold such as 90% or 95% can guide exploration, but the best k should also be checked using validation performance, interpretability, memory, and computation.

4.2 Reconstruction and Compression

Reconstruct the original feature space using

$$\widehat{X} = Z_k V_k^T + \mathbf{1}\boldsymbol{\mu}^T.$$

The relative reconstruction error is

$$\frac{\|X - \widehat{X}\|_F}{\|X\|_F}.$$

4.3 Correct Pipeline Order

1. Split observations into training and test sets.
2. Estimate centering and scaling values from the training set.
3. Fit PCA on the transformed training matrix.
4. Transform both sets using the training principal directions.
5. Fit the downstream model on training scores.
6. Evaluate on test scores without refitting preprocessing.

PCA maximizes retained input variance, not target prediction. A component with modest variance can still be useful for classification or regression, so downstream validation remains necessary.

Machine-Learning Connection

PCA can reduce storage and training time, remove correlated directions, support visualization, and sometimes improve numerical conditioning.

5 Numerical Reliability and Solver Choice

Mathematically equivalent formulas can behave differently in floating-point computation. Reliable machine learning requires attention to rank, scale, and conditioning.

5.1 Conditioning

For a full-rank matrix,

$$\kappa_2(A) = \frac{\sigma_{\max}(A)}{\sigma_{\min}(A)}.$$

A large condition number means that small changes in data may cause large changes in a solution. Forming normal equations can make the problem worse because

$$\kappa_2(A^T A) = \kappa_2(A)^2.$$

5.2 Choosing a Computation

Problem	Preferred tool	Reason
Square $A\mathbf{x} = \mathbf{b}$	`np.linalg.solve`	Solves directly without forming A^{-1}
Least squares	`np.linalg.lstsq`	Uses a stable factorization
Rank-deficient system	SVD or pseudoinverse	Reveals weak and null directions
Symmetric positive definite	Cholesky factorization	Efficient and structure-aware
Dimensionality reduction	Truncated SVD or PCA	Retains dominant directions
Repeated large problems	Iterative methods	Avoids expensive full factorizations

5.3 Reliability Checklist

- Scale features when units differ substantially.
- Inspect rank, singular values, and condition number.
- Avoid explicit matrix inversion when solving a system.
- Use tolerances rather than exact equality for floating-point results.
- Compare alternative solvers on small test cases.
- Report sensitivity, uncertainty, and known data limitations.

Regularization as Stabilization

Ridge regression increases weak eigenvalues of $X^T X$ by λ. This reduces sensitivity but introduces bias. The value of λ should be selected using validation data.

6 Validation, Diagnostics, and Communication

A result is useful only when it is mathematically correct, numerically reliable, and evaluated for the intended task.

6.1 Core Diagnostics

Diagnostic	Quantity	Question answered
System residual	$\|A\mathbf{x} - \mathbf{b}\|_2$	Does the solution satisfy the equations?
Relative residual	$\|A\mathbf{x} - \mathbf{b}\|_2 / \|\mathbf{b}\|_2$	Is the residual small relative to the data?
Prediction RMSE	$\sqrt{\text{mean}(\mathbf{y} - \widehat{\mathbf{y}})^2}$	How large are unseen prediction errors?
PCA reconstruction	$\|X - \widehat{X}\|_F / \|X\|_F$	How much matrix information was discarded?
Condition number	$\sigma_{\max}/\sigma_{\min}$	Is the computation sensitive?
Baseline comparison	Model versus simple rule	Does the model add practical value?

6.2 Frequent Failure Modes

- **Data leakage:** test information influences scaling, PCA, or fitting.
- **Shape mismatch:** rows and columns represent inconsistent objects.
- **Silent rank deficiency:** redundant features make parameters unstable.
- **Metric mismatch:** the reported measure does not reflect the application goal.
- **Over-interpretation:** correlation or vector proximity is treated as causation.

Minimum Reproducible Report

State the problem, data-matrix meaning, dimensions, preprocessing, method, parameter choices, evaluation split, numerical diagnostics, results, and limitations. Include executable code with a fixed random seed.

Machine-Learning Connection

Technical work is market-ready when another analyst can reproduce it, verify the assumptions, and explain why the selected method supports the decision being made.

7 Capstone: Applied Linear Algebra Portfolio Project

Select one dataset and solve a meaningful problem using at least two methods from the course. The project should demonstrate mathematical understanding, Python implementation, and professional communication.

7.1 Suggested Project Tracks

- **Regression:** compare least squares and ridge under correlated features.
- **Recommendation:** compare cosine neighbors with a low-rank latent-factor model.
- **Compression:** compare PCA or truncated SVD across several ranks.
- **Network data:** study an adjacency or graph-feature matrix using eigenvectors, similarity, or low-rank representations.

7.2 Required Deliverables

1. A one-paragraph problem statement and clearly defined objective.
2. A description of rows, columns, units, dimensions, and missing values.
3. Mathematical formulation of the selected methods.
4. Reproducible Python code using NumPy and relevant libraries.
5. At least one baseline and one appropriate evaluation measure.
6. Shape, rank, condition, or reconstruction diagnostics where relevant.
7. Two clear visualizations or tables and a concise interpretation.
8. A README explaining how to run the project and reproduce the results.

7.3 Suggested Assessment Rubric

Criterion	Marks
Problem formulation and data representation	15
Mathematical correctness and method selection	25
Python implementation and reproducibility	20
Validation and numerical diagnostics	20
Interpretation, limitations, and communication	20
Total	100

> A strong project does not need the largest dataset. A smaller, well-explained analysis with correct mathematics and honest validation is more valuable than complicated code without interpretation.

8 Python Mini-Project: Reliable Ridge Regression

The following example uses a built-in numerical dataset, keeps test data separate, standardizes features using training statistics, solves a ridge system, and compares test RMSE with a baseline.

```python
import numpy as np
from sklearn.datasets import load_diabetes
from sklearn.model_selection import train_test_split

X, y = load_diabetes(return_X_y=True)
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.25, random_state=7
)

# Learn preprocessing from the training set only.
mean = X_train.mean(axis=0)
std = X_train.std(axis=0)
Xtr = (X_train - mean) / std
Xte = (X_test - mean) / std

# Add an intercept column.
A = np.column_stack([np.ones(len(Xtr)), Xtr])
A_test = np.column_stack([np.ones(len(Xte)), Xte])

# Ridge system; do not penalize the intercept.
lam = 1.0
penalty = np.eye(A.shape[1])
penalty[0, 0] = 0.0
beta = np.linalg.solve(
    A.T @ A + lam * penalty, A.T @ y_train
)

prediction = A_test @ beta
rmse = np.sqrt(np.mean((prediction - y_test) ** 2))
baseline = np.full_like(y_test, y_train.mean())
baseline_rmse = np.sqrt(np.mean((baseline - y_test) ** 2))
r2 = 1 - np.sum((prediction - y_test) ** 2) / np.sum(
    (y_test - y_test.mean()) ** 2
)

print("Train/test shapes:", A.shape, A_test.shape)
print("Condition number:", np.linalg.cond(A))
print("Model RMSE:", rmse)
print("Baseline RMSE:", baseline_rmse)
print("Test R-squared:", r2)
```

Interpret the Output

The model should be judged on unseen test observations and against the baseline. The condition number describes sensitivity, while RMSE and R^2 describe predictive performance. This educational example is not a clinical decision system.

Extend the experiment by comparing several values of λ selected on a validation set, and compare the result with `np.linalg.lstsq`.

9 Course Review and Next Steps

No.	Core idea	Machine-learning connection
01	**Linear systems**	Constraints, data equations, solution structure
02	**Vectors and span**	Feature representation and combinations
03	**Basis and dimension**	Redundancy and compact coordinates
04	**Linear transformations**	Matrix mappings and model layers
05	**Orthogonal projections**	Similarity, decomposition, approximation
06	**Least squares**	Regression and residual minimization
07	**Eigenpairs**	Spectral directions and repeated dynamics
08	**SVD and PCA**	Low-rank models, reduction, compression
09	**Networks, embeddings, attention**	Modern learned representations
10	**Integrated workflows**	Reliable application and communication

9.1 Final Mastery Checklist

You are ready to apply the course when you can:

- identify the meaning and dimensions of every matrix in a problem;
- connect geometric interpretation with algebraic computation;
- choose a stable solver instead of relying on an inverse formula;
- implement and verify a method using NumPy;
- compare results with a baseline and an appropriate error measure; and
- explain assumptions, limitations, and practical consequences clearly.

Practice Tasks

Choose one formula from each lecture and explain: its dimensions, geometric meaning, computational method, and one machine-learning application. This creates a compact revision portfolio for the complete course.

Course completion: You have progressed from vectors and systems to SVD, PCA, neural networks, attention, and reliable applied workflows.