Core Math for ML/Stats (PCA, SPCA, PLS, Tensor Algebra)

Contents

1 Notation & Symbols (with Examples) **1**

2 Core Statistical Building Blocks **2**
- 2.1 Mean and variance . . . 2
- 2.2 Gaussian density . . . 2
- 2.3 Bayes' rule . . . 2

3 Supervised Learning: Models, Losses, Optimizers **2**
- 3.1 Linear regression (OLS / Ridge / Lasso) . . . 2
- 3.2 Logistic regression (binary) & Softmax (multi-class) . . . 2
- 3.3 Gradient-based optimization . . . 2
- 3.4 Maximum likelihood & KL . . . 3
- 3.5 Loss summary . . . 3

4 SVM and Neural Networks (Quick) **3**

5 Bias–Variance and Cross-Validation **3**

6 PCA, Rank, Eigenvalues: From Matrices to Tensors **3**
- 6.1 Rank . . . 3
- 6.2 Eigenvalues/eigenvectors . . . 3
- 6.3 PCA via covariance and SVD . . . 3
- 6.4 Does PCA depend only on X? . . . 4

7 Tensor Algebra and Decompositions **4**
- 7.1 Tensor regression (example) . . . 4
- 7.2 HOSVD / Tucker (Tensor PCA) . . . 4
- 7.3 CP/PARAFAC . . . 4
- 7.4 Where the products appear . . . 4

8 How-To: Khatri–Rao and $\mathbf{1}_k$ **4**
- 8.1 Khatri–Rao step-by-step (column-wise Kronecker) . . . 4
- 8.2 What $\mathbf{1}_k$ does (not cumulative) . . . 5

9 Supervised PCA (SPCA) and Partial Least Squares (PLS) **5**
- 9.1 Supervised PCA (SPCA) . . . 5
- 9.2 Partial Least Squares (PLS) . . . 5
- 9.3 PCA vs SPCA vs PLS (at a glance) . . . 5

10 Task Map: Where Each Equation is Used **6**

11 Identities Reference (handy) **6**

12 Mini Worked Examples (Hand-Check Size) **6**
12.1 Khatri–Rao and $\mathbf{1}_k$ aggregation 6
12.2 PCA via SVD 6
12.3 SPCA toy intuition 6

13 Appendix: Python Snippets (shapes-first) **7**

List of Figures

List of Tables

1 Frequently used operators with concrete examples. 1
2 Common losses and where theyre used. 3
3 Supervised vs. unsupervised component methods. 5
4 Quick map from math objects to ML tasks. 6
5 Frequently used algebraic identities. 6

1 Notation & Symbols (with Examples)

Unless stated, vectors are bold lowercase ($\mathbf{x}$), matrices bold uppercase ($\mathbf{X}$), tensors calligraphic ($\mathcal{X}$).

Linear algebra & tensor operators

Symbol	Name	Description & Example
$\mathrm{vec}(A)$	Vectorization	Stacks entries of A column-wise into a single column vector. Example: $A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \Rightarrow \mathrm{vec}(A) = [1, 3, 2, 4]^\top$.
$\odot$	Khatri–Rao product	Column-wise Kronecker: if $A \in \mathbb{R}^{m \times k}, B \in \mathbb{R}^{n \times k}$, then $A \odot B \in \mathbb{R}^{mn \times k}$ with $(A \odot B)_{:,j} = A_{:,j} \otimes B_{:,j}$. Example with $A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$, $B = \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix}$: $A \odot B = \begin{bmatrix} 5 & 12 \\ 7 & 16 \\ 15 & 24 \\ 21 & 32 \end{bmatrix}$.
$\otimes$	Kronecker product	Full expansion: $(A \otimes B) = (a_{ij}B)_{ij}$. Example: $A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$, $B = \begin{bmatrix} 0 & 5 \\ 6 & 7 \end{bmatrix}$, then $A \otimes B = \begin{bmatrix} 0 & 5 & 0 & 10 \\ 6 & 7 & 12 & 14 \\ 0 & 15 & 0 & 20 \\ 18 & 21 & 24 & 28 \end{bmatrix}$.
$\circ$	Hadamard product	Elementwise product: $(A \circ B)_{ij} = A_{ij}B_{ij}$. Example: $\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \circ \begin{bmatrix} 10 & 20 \\ 30 & 40 \end{bmatrix} = \begin{bmatrix} 10 & 40 \\ 90 & 160 \end{bmatrix}$.
$\langle A, B \rangle$	Inner product	Sum of elementwise products: $\langle A, B \rangle = \sum_{ij} A_{ij}B_{ij} = \mathrm{tr}(A^\top B)$. Example with A, B above: $\langle A, B \rangle = 70$.
$\lVert \cdot \rVert_F$	Frobenius norm	$\lVert A \rVert_F = \sqrt{\sum_{ij} A_{ij}^2} = \sqrt{\langle A, A \rangle}$. Example: $\lVert \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \rVert_F = \sqrt{30}$.
$\mathbf{1}_k$	Vector of ones	$\mathbf{1}_k = [1, \ldots, 1]^\top \in \mathbb{R}^k$. Used to sum/aggregate columns: $M\mathbf{1}_k$ yields row sums of M.

Table 1: Frequently used operators with concrete examples.

Useful identities. $\mathrm{vec}(AXB) = (B^\top \otimes A)\mathrm{vec}(X)$, $(A \odot B)^\top (A \odot B) = (A^\top A) \circ (B^\top B)$.

Extra numeric examples (quick check)

- **Inner/Frobenius:** With A, B as above, $\langle A, B\rangle = 70$, $\|A\|_F = \sqrt{30}$.
- **Aggregation with $\mathbf{1}_k$:** If $M = \begin{bmatrix}1 & 2 & 3\\ 4 & 5 & 6\end{bmatrix}$, $M\mathbf{1}_3 = \begin{bmatrix}6\\ 15\end{bmatrix}$ (row sums). Not a cumulative sum.

2 Core Statistical Building Blocks

2.1 Mean and variance

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i, \qquad \mathrm{Var}(X) = \frac{1}{n}\sum_{i=1}^{n}(x_i - \mu)^2.$$

Use: standardization, bias–variance analysis, Gaussian likelihoods.

2.2 Gaussian density

$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right).$$

Use: regression noise models, generative modeling, conjugacy.

2.3 Bayes' rule

$$p(y \mid x) = \frac{p(x \mid y)p(y)}{p(x)}.$$

Use: naive Bayes, Bayesian regression, probabilistic inference.

3 Supervised Learning: Models, Losses, Optimizers

3.1 Linear regression (OLS / Ridge / Lasso)

Model: $\mathbf{y} = \mathbf{X}\beta + \varepsilon$, loss $L(\beta) = \frac{1}{n}\|\mathbf{y} - \mathbf{X}\beta\|_2^2$. Closed-form OLS: $\hat{\beta} = (X^\top X)^{-1}X^\top y$ (if full rank).

$$\text{Ridge: } L = \tfrac{1}{n}\|y - Xw\|_2^2 + \lambda\|w\|_2^2, \qquad \text{Lasso: } L = \tfrac{1}{n}\|y - Xw\|_2^2 + \lambda\|w\|_1.$$

Task: regression. **Notes:** multicollinearity; sparsity.

3.2 Logistic regression (binary) & Softmax (multi-class)

Sigmoid: $\sigma(z) = \frac{1}{1+e^{-z}}$, probability $\hat{p}(y{=}1 \mid x) = \sigma(x^\top w + b)$.
Cross-entropy loss:

$$L = -\frac{1}{n}\sum_{i=1}^{n}\big[y_i \log \hat{p}_i + (1-y_i)\log(1-\hat{p}_i)\big].$$

Softmax: $\mathrm{softmax}(z)_k = \frac{e^{z_k}}{\sum_j e^{z_j}}$, multi-class loss $L = -\frac{1}{n}\sum_i \log \hat{p}_{i,y_i}$. **Task:** classification.

3.3 Gradient-based optimization

Update: $\theta_{t+1} = \theta_t - \eta\,\nabla_\theta L(\theta_t)$.
Variants: SGD, momentum, Adam, RMSProp.

3.4 Maximum likelihood & KL

$$\hat{\theta} = \arg\max_{\theta} \sum_{i=1}^{n} \log p(x_i \mid \theta), \qquad D_{\mathrm{KL}}(p\|q) = \sum_{x} p(x) \log \frac{p(x)}{q(x)}.$$

Use: derive losses; variational inference (VI) / VAEs.

3.5 Loss summary

Loss	Formula	Task	Notes
MSE	$\frac{1}{n}\sum_i (y_i - \hat{y}_i)^2$	Regression	Smooth, sensitive to outliers
MAE	$\frac{1}{n}\sum_i \lvert y_i - \hat{y}_i \rvert$	Regression	Robust to outliers
Cross-entropy	$-\sum_i y_i \log \hat{y}_i$	Classification	Likelihood-based
Hinge	$\max(0, 1 - y_i \hat{y}_i)$	Margin cls. (SVM)	Maximizes margin
Huber	piecewise quad./linear	Regression	Robust & smooth

Table 2: Common losses and where theyre used.

4 SVM and Neural Networks (Quick)

SVM (soft-margin):

$$\min_{w,b} \ \frac{1}{2}\|w\|_2^2 + C \sum_i \max(0, 1 - y_i(w^\top x_i + b)).$$

Neural layer: $h^{(\ell)} = f(W^{(\ell)} h^{(\ell-1)} + b^{(\ell)})$; **Use:** classification or regression depending on final activation.

5 Bias–Variance and Cross-Validation

$$\mathbb{E}\big[(\hat{f}(x) - f(x))^2\big] = \underbrace{(\mathrm{Bias})^2}_{\text{systematic error}} + \underbrace{\mathrm{Var}}_{\text{sensitivity}} + \mathrm{Noise}.$$

K-fold CV error $= \frac{1}{K}\sum_{k=1}^{K} L_k$. **Use:** model selection & hyperparameters.

6 PCA, Rank, Eigenvalues: From Matrices to Tensors

6.1 Rank

$\mathrm{rank}(A)$ = # linearly independent columns (or rows). Determines identifiability of OLS; caps # nonzero PCs.

6.2 Eigenvalues/eigenvectors

$Av = \lambda v$. In PCA, eigenvectors of covariance give principal directions and eigenvalues give explained variance.

6.3 PCA via covariance and SVD

With centered $X \in \mathbb{R}^{n \times d}$:

$$\Sigma = \tfrac{1}{n} X^\top X, \qquad \Sigma v_k = \lambda_k v_k.$$

Equivalently SVD: $X = U \Sigma V^\top$; columns of V are PCs and Σ^2/n are variances.

Short equivalence note (PCA $\leftrightarrow$ SVD) $X^\top X = V\Sigma^2 V^\top$ so eigenvectors of the covariance are the right singular vectors of X, and eigenvalues are σ_i^2.

6.4 Does PCA depend only on X?

Yes (standard PCA is unsupervised; depends solely on X via its covariance). Supervised variants below incorporate y.

7 Tensor Algebra and Decompositions

7.1 Tensor regression (example)

Scalar response:

$$y_i = \alpha + \Big\langle \mathrm{vec}(\tilde{\mathbf{B}}), \mathrm{vec}(\tilde{\mathbf{S}}_i) \Big\rangle + \sigma\epsilon_i = \alpha + \Big\langle (\tilde{B}_3 \odot \tilde{B}_2 \odot \tilde{B}_1)\mathbf{1}_k,\ \mathrm{vec}(\tilde{S}_i) \Big\rangle + \sigma\epsilon_i.$$

$\tilde{B}_1, \tilde{B}_2, \tilde{B}_3$ are factor matrices; $\odot$ aligns columns across modes; $\mathbf{1}_k$ aggregates k rank-1 components.

7.2 HOSVD / Tucker (Tensor PCA)

$$\mathcal{X} \approx \mathcal{G} \times_1 U_1 \times_2 U_2 \times_3 U_3, \qquad X_{(n)} = U_n S_n \Big(\bigotimes_{m \neq n} U_m \Big)^\top.$$

Kronecker ($\otimes$) ties other-mode bases in matricized form.

7.3 CP/PARAFAC

$$\mathcal{X} \approx \sum_{r=1}^{R} a_r \circ b_r \circ c_r, \qquad \mathrm{vec}(\mathcal{X}) \approx (C \odot B \odot A)\, \mathbf{1}_R.$$

Use: compact multiway factorization; $\odot$ is the natural column-aligned expansion.

7.4 Where the products appear

- **Kronecker $\otimes$:** separable covariances ($\Sigma_t \otimes \Sigma_s$), matrix-variate normals, vectorization identities $\mathrm{vec}(AXB) = (B^\top \otimes A)\mathrm{vec}(X)$.
- **Khatri–Rao $\odot$:** CP/Tucker updates, tensor regression design matrices, feature interaction with aligned ranks.
- **Hadamard $\circ$:** elementwise weighting, covariance identities, attention masks.

8 How-To: Khatri–Rao and $\mathbf{1}_k$

8.1 Khatri–Rao step-by-step (column-wise Kronecker)

Let $A = [a_1\, a_2\, \ldots a_K] \in \mathbb{R}^{I\times K}$, $B = [b_1\, b_2\, \ldots b_K] \in \mathbb{R}^{J\times K}$.

$$A \odot B = \big[\, a_1 \otimes b_1 \;\; a_2 \otimes b_2 \;\; \ldots \;\; a_K \otimes b_K \,\big] \in \mathbb{R}^{(IJ)\times K}.$$

Example: $A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$, $B = \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix}$:

$$(A \odot B)_{:,1} = \begin{bmatrix} 1 \\ 3 \end{bmatrix} \otimes \begin{bmatrix} 5 \\ 7 \end{bmatrix} = \begin{bmatrix} 5 \\ 7 \\ 15 \\ 21 \end{bmatrix}, \quad (A \odot B)_{:,2} = \begin{bmatrix} 2 \\ 4 \end{bmatrix} \otimes \begin{bmatrix} 6 \\ 8 \end{bmatrix} = \begin{bmatrix} 12 \\ 16 \\ 24 \\ 32 \end{bmatrix}.$$

8.2 What $\mathbf{1}_k$ does (not cumulative)

$\mathbf{1}_k$ is a column of ones used to *sum* components, not to produce cumulative sums. If $Z = [z_1 \dots z_K]$ collects K rank-1 terms, then $Z\mathbf{1}_k = \sum_{j=1}^{K} z_j$ (aggregation). For tensors, $(C \odot B \odot A)\mathbf{1}_K$ sums rank-1 parts into a single vector.

9 Supervised PCA (SPCA) and Partial Least Squares (PLS)

9.1 Supervised PCA (SPCA)

Find directions v in X that correlate with y:

$$v = \arg\max_{\|v\|=1} \ \mathrm{Cov}^2(Xv, y).$$

A common formulation solves the eigenproblem on

$$C = X^\top y y^\top X \quad (\text{or } X^\top Y Y^\top X \text{ for multi-response}), \qquad Cv = \lambda v.$$

Use: supervised dimensionality reduction when only components predictive of y are desired. For classification, let Y be one-hot labels.

9.2 Partial Least Squares (PLS)

Find paired latent scores (t, u) with $t = Xw,\ u = yq$ maximizing covariance:

$$\max_{w,q} \ \mathrm{Cov}^2(Xw, yq), \quad \text{with orthogonality across components.}$$

One PLS1 iteration (single response):

$$w \propto X^\top y, \quad t = Xw, \quad q = \frac{y^\top t}{t^\top t}, \quad p = \frac{X^\top t}{t^\top t}, \quad X \leftarrow X - tp^\top,\ y \leftarrow y - tq.$$

Final regression: $\hat{y} = XW(P^\top W)^{-1}Q^\top$.
Use: high-dimensional regression where predictors are collinear; chemometrics, genomics.

9.3 PCA vs SPCA vs PLS (at a glance)

Method	Uses y?	Objective	Output	Typical Task
PCA	No	$\max_v \mathrm{Var}(Xv)$	Unsupervised PCs	Compression/denoising
SPCA	Yes	$\max_v \mathrm{Cov}^2(Xv, y)$	Supervised PCs	Predictive features
PLS	Yes	$\max_{w,q} \mathrm{Cov}^2(Xw, yq)$	Latent scores (t, u)	Regression

Table 3: Supervised vs. unsupervised component methods.

Equation/Concept	Primary Use	Notes
Linear regression (OLS/Ridge/Lasso)	Regression	L2 for stability, L1 for sparsity
Logistic/Softmax + Cross-Entropy	Classification	Probabilistic interpretation
MLE / Log-likelihood	Estimation	Derives many learning objectives
KL divergence	Regularization / VI	Distance between distributions
PCA (SVD)	Unsupervised DR	Depends only on X
SPCA / PLS	Supervised DR	Aligns components with y
SVM (hinge)	Classification	Margin maximization
Gradient descent/Adam	Optimization	Ubiquitous in deep learning
Kronecker $\otimes$	Multiway lin. maps, covariances	Separable structures
Khatri–Rao $\odot$	Tensor decompositions	Column-aligned Kronecker
Hadamard $\circ$	Elementwise modeling	Masks/weights, covariance algebra

Table 4: Quick map from math objects to ML tasks.

Identity	Comment
$\mathrm{vec}(AXB) = (B^\top \otimes A)\mathrm{vec}(X)$	Vectorization + Kronecker
$(A \odot B)^\top (A \odot B) = (A^\top A) \circ (B^\top B)$	Appears in CP/ALS updates
$\|A\|_F^2 = \langle A, A\rangle = \mathrm{tr}(A^\top A)$	Frobenius norm
$X^\top X = V\Sigma^2 V^\top$ (SVD of X)	PCASVD equivalence

Table 5: Frequently used algebraic identities.

10 Task Map: Where Each Equation is Used

11 Identities Reference (handy)

12 Mini Worked Examples (Hand-Check Size)

12.1 Khatri–Rao and $\mathbf{1}_k$ aggregation

Let $A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$, $B = \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix}$. Then $A \odot B = \begin{bmatrix} 5 & 12 \\ 7 & 16 \\ 15 & 24 \\ 21 & 32 \end{bmatrix}$. If each column is a rank-1 component, multiplying by $\mathbf{1}_2$ sums the two components: $(A \odot B)\mathbf{1}_2 = \begin{bmatrix} 17 \\ 23 \\ 39 \\ 53 \end{bmatrix}$.

12.2 PCA via SVD

Centered $X = U\Sigma V^\top$. The top k principal directions are columns of $V_{:,1\ldots k}$; the projected scores are $Z = XV_{:,1\ldots k} = U_{:,1\ldots k}\Sigma_{1\ldots k}$.

12.3 SPCA toy intuition

If y varies mainly with a subset of features, $X^\top yy^\top X$ emphasizes those directions even when overall variance in X is driven by others.

13 Appendix: Python Snippets (shapes-first)

The following compact demo reproduces our sessions computations (rank, PCA, SPCA, PLS, Kronecker, Khatri–Rao, Hadamard, plus vec/inner/Frobenius). Center X (and y) for PCA/SPCA/PLS.

```
import numpy as np
from numpy.linalg import matrix_rank, lstsq, norm
from sklearn.decomposition import PCA
from sklearn.cross_decomposition import PLSRegression

# OPTIONAL: SciPy for KhatriRao (fallback to manual if unavailable)
try:
    from scipy.linalg import khatri_rao as scipy_khatri_rao
    HAS_SCIPY = True
except Exception:
    HAS_SCIPY = False

# ---------- Toy data ----------
X = np.array([
    [0.0, 1.0, 2.0],
    [1.0, 2.0, 3.0],
    [2.0, 3.0, 4.0],
    [3.0, 4.0, 5.0],
])
y = np.array([1.0, 3.0, 5.0, 7.0])

A = np.array([[1, 2],
              [3, 4]])
B = np.array([[5, 6],
              [7, 8]])

# ---------- Rank ----------
print("rank(A) =", matrix_rank(A), " | rank(X) =", matrix_rank(X))

# ---------- PCA (unsupervised; depends only on X) ----------
Xc = X - X.mean(axis=0, keepdims=True)
pca = PCA(n_components=2)
Z_pca = pca.fit_transform(Xc) # scores (n x k)
V_pca = pca.components_.T # loadings (d x k)
print("\nPCA loadings V:\n", V_pca)
print("PCA scores Z:\n", Z_pca)
print("PCA explained variances:", pca.explained_variance_)

# ---------- SPCA (simple eigen formulation) ----------
yc = y - y.mean()
C_spca = Xc.T @ np.outer(yc, yc) @ Xc # d x d
evals, evecs = np.linalg.eigh(C_spca) # symmetric eig
idx = np.argsort(evals)[::-1]
V_spca = evecs[:, idx[:2]]
Z_spca = Xc @ V_spca
print("\nSPCA directions:\n", V_spca)
print("SPCA scores:\n", Z_spca)

# ---------- PLS (supervised; maximizes covariance with y) ----------
pls = PLSRegression(n_components=2)
T_pls, U_pls = pls.fit_transform(Xc, yc)
print("\nPLS X-scores T:\n", T_pls)
```

```
print("PLS Y-scores U:\n", U_pls)
print("PLS X-weights W:\n", pls.x_weights_)

# ---------- Least Squares (OLS) ----------
beta, residuals, rank, svals = lstsq(X, y, rcond=None)
print("\nOLS beta:\n", beta)
print("OLS residual sum of squares:", residuals.sum() if residuals.size else 0.0)

# ---------- Kronecker, KhatriRao, Hadamard ----------
kron_AB = np.kron(A, B)
print("\nKronecker A  B:\n", kron_AB)

def khatri_rao(A, B):
    m, k1 = A.shape
    n, k2 = B.shape
    assert k1 == k2
    out = np.zeros((m*n, k1), dtype=np.result_type(A, B))
    for k in range(k1):
        out[:, k] = np.kron(A[:, k], B[:, k])
    return out

A2 = np.array([[1, 2],
               [3, 4]])
B2 = np.array([[5, 6],
               [7, 8]])
KR = scipy_khatri_rao(A2, B2) if HAS_SCIPY else khatri_rao(A2, B2)
print("\nKhatriRao A  B:\n", KR)

H = A * B
print("\nHadamard A  B:\n", H)

# ---------- vec, inner product, Frobenius, ones ----------
vecA_col_major = A.reshape(-1, order='F') # vec in column-major sense
inner_AB = np.tensordot(A, B, axes=2)
froA = norm(A, 'fro')
ones2 = np.ones((2, 1))
print("\nvec(A) (col-major):", vecA_col_major)
print("A,B =", inner_AB, " == tr(A^T B)")
print("||A||_F =", froA)
print("1_k (k=2):\n", ones2)
```

Index

activation function, 3
Adam, 2

Bayes' rule, 2
Bayesian regression, 2
bias–variance decomposition, 2, 3

chemometrics, 5
classification, 2
conjugacy, 2
covariance matrix, 3
CP decomposition, 4
cross-entropy, 2, 3
cross-validation, 3
not cumulative, 2

eigenvalues, 3
eigenvectors, 3
examples, 6
    numeric, 2

Frobenius norm, 1
full rank, 2

Gaussian
    density, 2
    likelihood, 2
generative modeling, 2
genomics, 5
gradient descent, 2
Gram matrix, 1

Hadamard identity, 1
Hadamard product, 1, 4
hinge loss, 3
HOSVD, 4
Huber loss, 3
hyperparameter tuning, 3

identities, 6
inference, 2
inner product, 1

Khatri–Rao product, 1, 4, 6
KL divergence, 3
Kronecker product, 1, 4

lasso, 2
linear algebra, 1
linear regression, 2
logistic regression, 2

loss
    cross-entropy, 2
loss functions, 3

MAE, 3
matrix, 1
maximum likelihood, 3
mean, 2
model selection, 3
momentum, 2
MSE, 3
multicollinearity, 2

naive Bayes, 2
neural networks, 3
normal distribution, 2
notation, 1

optimization, 2
OLS (ordinary least squares), 2

PARAFAC, 4
PCA, 3, 6
    SVD equivalence, 4
    unsupervised, 4
    vs SPCA vs PLS, 5
PLS, 5
posterior, 2
prior, 2
Python code, 7

rank, 3
ridge regression, 2
RMSProp, 2

SGD, 2
sigmoid, 2
softmax, 2
sparsity, 2
SPCA, 5, 6
standardization, 2
statistics
    core, 2
supervised learning, 2
SVD, 3, 6
SVM, 3
symbols, 1

task map, 6
tensor, 1
tensor algebra, 1, 4

tensor decomposition, 4
tensor PCA, 4
tensor regression, 4
trace, 1
Tucker decomposition, 4

variance, 2
variational autoencoder, 3
variational inference, 3
vector, 1
$\mathbf{1}_{\boldsymbol{k}}$, 4, 6
$\mathbf{1}_{\boldsymbol{k}}$ (vector of ones), 1
`vec` (vectorization), 1
vectorization identity, 1