Subject: Course on Machine learning (ML)

Date: March 6, 2025

---

1 Table of content

1. **Intro to machine learning (ML)** [2, 9]
   - **Theory:** Deterministic and Stochastic Approaches to Modeling. Basic ideas of supervised learning, unsupervised learning, and reinforcement learning. Flowchart for an ML model design. Exploratory data analysis (EDA), feature encoding, and normalization. ML models intuition (regression, classification, image captioning, control). Performance measure intuition.
   - **Skills:** Python basics. Exploratory data analysis. Os, NumPy, Pandas, Scikit-learn, Matplotlib, and Seaborn libraries.
2. **Linear models: fitting and classification** [2, 9]
   - **Theory:** Performance measures and their gradients. Matrix form of a linear model forward and backward passes. Training algorithms. Particular cases of multiple variables and polynomial models. Decision boundary in classification.
   - **Skills:** Regression and classification models for tabular data. Loss functions, optimizers, forward pass and the models' weights update in Pytorch. Accuracy of a trained model.
3. **Metrics. Tips and tricks in ML: regularization, data splitting, cross-validation, batches** [9, 2, 5]
   - **Theory:** Metrics fundamentals in learning and in estimating results. Binary classification metrics: accuracy, precision, recall, F_1 score, receiver operating characteristic (ROC), area under curve (AUC). Generalization for multiclass classification problems. Regression metrics: mean squared error (MSE), mean absolute error (MAE), root MSE (RMSE). Data splitting with bootstrapping and cross-validation. L_1 and L_2 normalization. Batches.
   - **Skills:** pre-processing of tabular data and post-processing of ML results. Dataset and data loader classes for tabular data. Implementation of L_1 and L_2 regularization in models. Perform cross-validation on a dataset and compare the results with and without regularization. Grid search.
4. **Feedforward neural networks. Backpropagation** [9, 2, 5]
   - **Theory:** Comparison of a natural and artificial neuron, linear and non-linear operations in artificial neurons. Activation functions and their derivatives. A feedforward or fully-connected neural network architecture. Forward pass in an artificial neural network (ANN) and its implementation in a matrix form. Backward pass and backpropagation using a calculation graph.
   - **Skills:** Image dataset and dataloader classes in PyTorch. Data transforms. The simplest ANN architectures in PyTorch. Solution of an image classification task.
5. **Bayesian approach** [3, 2, 4]

- **Theory:** Probability and conditional probability, the sum and product rules in discrete and continuous forms. Bayes' theorem and Bayes' inference examples. Probability density functions, their expectations, and variance. A Bayesian framework in ML. Comparison of the frequentist and Bayesian approaches.
- **Skills:** Bayesian linear regression model using PyMC3 library or a similar library. Comparison with the results of a traditional linear regression model.

6. **Loss functions fundamentals. Uncertainty estimation in machine learning** [9, 5, 2, 7]

   - **Theory:** Confidence and overconfidence of models. Maximum likelihood and negative log-likelihood intuition. An algorithm for constructing a loss function. Loss functions foundations: homoscedastic and heteroscedastic regression losses, binary cross-entropy, and cross-entropy losses. Noise in data and label smoothing. Aleatoric and epistemic uncertainties. The heteroscedastic regression loss for multiclass classification. Dropout (DO) and ensembling as Bayesian methods for uncertainty estimation. Temperature scaling and expected calibration error.
   - **Skills:** Implementation of ANNs with different loss functions (e.g., MSE, MAE, cross entropy, etc.) and comparison of their performance on a classification task. Comparison of a homoscedastic and heteroscedastic linear regression models. Deep ensembling (DE) and Monte Carlo dropout (MC-DO) methods implementation.

7. **Convolutional neural networks (CNNs). Residual neural networks** [9, 2, 5, 8]

   - **Theory:** Convolutional neural networks (CNNs) intuition, filters, and kernels. Padding, pooling, and striding techniques. Image size transformation after convolution. Receptive fields. Residual connections. 2D CNN architectures: AlexNet, ResNet, DenseNet. A CNN training algorithm. 1D CNNs in application to signal processing.
   - **Skills:** Implementation of a simple CNN using PyTorch to classify images from MNIST or CIFAR-10 datasets. The use of different architectures.

8. **Segmentation and object detection. Tips and tricks in ML: transfer learning and fine-tuning, augmentation, batch normalization, regularization.** [9, 2, 11]

   - **Theory:** Semantic segmentation intuition, applications and evaluation metrics. Upsampling and transposed convolutions. U-Net architecture with encoder and decoder parts. Loss functions: CE and Dice losses. Object detection and YOLO network intuition. Data augmentation. Transfer learning. Labeling tools.
   - **Skills:** Application of fine-tuning and transfer learning methods to ML models. The use of U-Net for unsupervised learning with patch augmentation.

9. **Recurrent neural networks (RNNs). Transformers** [2, 9]

   - **Theory:** Recurrent neural networks (RNNs) intuition, architecture, and limitations (e.g., vanishing gradient problem). Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) architectures. Transformer architecture, attention mechanisms, and self-attention. Encoder part of a transformer. Applications of RNNs and transformers in signals and images processing.
   - **Skills:** Implementation of simple RNNs and LSTMs using PyTorch for sequence prediction tasks. The using of pretrained transformers in signals, images and sequences processing.

10. **Reinforcement learning (RL)** [2, 9]

- **Theory:** Reinforcement learning intuition, Markov decision processes, Bellman equation. Policy-based methods (e.g., Policy Gradient, REINFORCE). Value-based methods (e.g., Q-learning, SARSA). Actor-Critic methods. Deep Q-Networks (DQN) and exploration strategies.
- **Skills:** Implementation of a simple RL agent using libraries like OpenAI Gym and PyTorch. Training and evaluation of the agent in a simulated environment.

11. **Unsupervised learning. Clustering. Autoencoders and variational autoencoders** [9, 2]
    - **Theory:** Unsupervised learning intuition and applications. K-Means clustering algorithm. Autoencoders intuition, architecture, and applications. A variational autoencoder intuition, algorithm and loss. Generative modeling, interpolation.
    - **Skills:** Implementation of a simple autoencoder and variational autoencoder using PyTorch.

12. **Physics informed neural networks (PINNs)** [6, 10]
    - **Theory:** Boundary value problem and variational problem intuition. Physics informed neural networks (PINNs) intuition and applications. Incorporating physical laws and constraints into neural network training. Applications of PINNs in solving differential equations and physical simulations.
    - **Skills:** Implementation of a simple PINN for solving a differential equation using PyTorch.

13. **Analysis of temporal and spatial data** [2, 9, 1]
    - **Theory:** Time series and multi-sensory measurements processing: signal filtering, fault diagnosis, prognosis, remaining useful life estimation. Point clouds. Measuring performance: noise, bias, variance, bias-variance decomposition and trade-off.
    - **Skills:** Simple models for multi-sensory data processing: filtering, fault diagnosis and (or) remaining useful life models.

References

[1] S. Ahmed, I. E. Nielsen, A. Tripathi, S. Siddiqui, R. P. Ramachandran, and G. Rasool. Transformers in time-series analysis: A tutorial. *Circuits, Systems, and Signal Processing*, 42 (12):7433–7466, 2023.

[2] M. Artemyev and A. Ashukha. *Handbook on Machine Learning (in Russian)*. Yandex, 2024. URL `https://education.yandex.ru/handbook/ml`.

[3] C. M. Bishop and N. M. Nasrabadi. *Pattern recognition and machine learning*, volume 4. Springer, 2006.

[4] Y. Gal and Z. Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059. PMLR, 2016.

[5] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[6] G. E. Karniadakis, I. G. Kevrekidis, L. Lu, P. Perdikaris, S. Wang, and L. Yang. Physics-informed machine learning. *Nature Reviews Physics*, 3(6):422–440, 2021.

[7] A. Kendall and Y. Gal. What uncertainties do we need in bayesian deep learning for computer vision? *Advances in neural information processing systems*, 30, 2017.

[8] F.-F. Li. Cs231n: Deep learning for computer vision, 2024. URL `https://cs231n.stanford.edu/`.

[9] S. J. Prince. *Understanding Deep Learning*. The MIT Press, 2023. URL `http://udlbook.com`.

[10] M. Raissi, P. Perdikaris, and G. E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational physics*, 378:686–707, 2019.

[11] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. Grad-cam: visual explanations from deep networks via gradient-based localization. *International journal of computer vision*, 128:336–359, 2020.