MACHINE LEARNING
CHEAT-SHEET FOR BEGINNERS

FUNDAMETAL CONCEPTS

MACHINE LEARNING TYPES

- **Supervised Learning:** Training a model on labeled data to predict outcomes.
  - Regression: Predicting a continuous output variable.
  - Classification: Predicting a categorical output variable.
- **Unsupervised Learning:** Finding patterns and structure in unlabeled data.
  - Clustering: Grouping similar data points together.
  - Dimensionality Reduction: Reducing the number of variables while preserving essential information.
- **Reinforcement Learning:** Training an agent to make decisions in an environment to maximize cumulative rewards.
  - Agent: The learner and decision-maker.
  - Environment: The world or system the agent interacts with.
  - Reward: Feedback signal indicating the desirability of an action.
  - State: The current situation or configuration of the environment.
  - Action: What the agent does in a given state.
- **Semi-supervised Learning:** Training a model on a dataset with both labeled and unlabeled data, typically when labeled data is scarce.
- **Self-supervised Learning:** A form of unsupervised learning where the data provides the supervision. For example, predicting a part of the input from other parts of the input.
- **Transfer Learning:** Leveraging knowledge gained from one task to improve performance on a related task, often by using a pre-trained model.

BASIC TERMINOLOGY

- **Features:** Input variables used to make predictions.
- **Labels/Targets:** Output variables the model aims to predict.
- **Training Set:** Data used to train the model.
- **Validation Set:** Data used to tune hyperparameters and evaluate model performance during training.
- **Test Set:** Data used to evaluate the final model's performance on unseen data.
- **Overfitting:** When a model learns the training data too well, performing poorly on unseen data.
- **Underfitting:** When a model is too simple to capture the underlying patterns in the data.
- **Bias:** Error due to overly simplistic assumptions in the learning algorithm.
- **Variance:** Error due to the model's sensitivity to small fluctuations in the training data.
- **Hyperparameters**: Parameters set before training, controlling the learning process.
- **Parameters:** Internal model parameters learned during training.
- **Model:** A mathematical representation learned from data to make predictions.
- **Loss Function:** Measures the error between predicted and actual values during training (e.g., MSE, Cross-Entropy).
- **Cost Function:** The average loss over the entire training dataset.
- **Optimizer:** Algorithm that updates model parameters to minimize the loss function (e.g., Gradient Descent).
- **Evaluation Metrics**: Quantify model performance (e.g., Accuracy, F1-score, R-squared).
- **Cross-Validation:** Technique to assess model performance by splitting data into multiple folds and training/testing on different combinations.
- **Regularization:** Techniques to prevent overfitting by adding a penalty to the loss function.
- **Ensemble Learning:** Combining multiple models to improve overall performance.
  - Bagging (Bootstrap Aggregating): Training multiple models on different subsets of the training data and averaging their predictions.
  - Boosting: Training models sequentially, where each model focuses on correcting the errors of the previous ones.
  - Stacking: Training multiple models and then using another model (meta-learner) to combine their predictions.

DATA PROCESSING

- **Data Cleaning:** Handling missing values, outliers, and inconsistencies in the data.
  - Missing Values: Data points where values are not recorded. Strategies include imputation (mean, median, model-based) or removal.
  - Outliers: Extreme values that deviate significantly from other data points. Can be handled by removal, transformation, or using robust models.
  - Noise: Random errors or variations in the data.
- **Data Transformation:** Applying mathematical functions to change the distribution or scale of features.
  - Normalization: Scaling features to a specific range (e.g., 0 to 1).
  - Standardization: Transforming features to have zero mean and unit variance.
  - Log Transform: Applying the logarithm to reduce the impact of extreme values.
- **Feature Scaling:** Ensuring features have similar ranges to prevent features with larger values from dominating the learning process.
- **Feature Encoding:** Converting categorical features into numerical representations.
  - One-Hot Encoding: Creating binary features for each category.
  - Label Encoding: Assigning a unique integer to each category.
- **Feature Engineering:** Creating new features from existing ones to improve model performance.
- **Feature Selection:** Choosing the most relevant features for the model.
  - Filter Methods: Selecting features based on statistical measures (e.g., correlation, chi-squared).
  - Wrapper Methods: Evaluating different subsets of features using a specific model.
  - Embedded Methods: Feature selection is built into the model training process (e.g., Lasso regression).
- **Dimensionality Reduction:** Reducing the number of features while preserving important information.
  - PCA (Principal Component Analysis): Transforming features into a new set of uncorrelated features (principal components) that capture the most variance in the data.
  - t-SNE (t-distributed Stochastic Neighbor Embedding): A non-linear technique primarily used for visualization, preserving local neighborhood structures in lower dimensions.
  - LDA (Linear Discriminant Analysis): A supervised dimensionality reduction technique that maximizes the separation between different classes.
- **Data Splitting:** Dividing the data into training, validation, and test sets.
- Handling Imbalanced Datasets: Addressing datasets where one class significantly outnumbers others.
  - Oversampling: Duplicating samples from the minority class.
  - Undersampling: Removing samples from the majority class.
  - SMOTE (Synthetic Minority Over-sampling Technique): Creating synthetic samples for the minority class.

CORE ALGORITHMS & TECHNIQUES

SUPERVISED LEARNING

Regression:

- **Linear Regression (Simple, Multiple):** Modeling the relationship between a dependent variable and one or more independent variables using a linear equation.
- **Polynomial Regression:** Extending linear regression by adding polynomial terms to capture non-linear relationships.
- **Regularized Regression:** Adding a penalty term to the linear regression cost function to prevent overfitting.
  - Ridge Regression (L2 Regularization): Adds a penalty proportional to the square of the magnitude of coefficients.
  - Lasso Regression (L1 Regularization): Adds a penalty proportional to the absolute value of the magnitude of coefficients. Can perform feature selection.
  - Elastic Net: A combination of Ridge and Lasso regression.
- **Support Vector Regression (SVR):** Using Support Vector Machines (SVM) for regression tasks.
- **Decision Tree Regression:** Building a tree-like structure where each node represents a decision based on a feature, and each leaf node represents a predicted value.
- **Random Forest Regression:** An ensemble of decision trees for regression.
- **Gradient Boosting Regression:** An ensemble method that builds trees sequentially, each correcting the errors of the previous ones.
  - GBM (Gradient Boosting Machines): A general framework for gradient boosting.
  - XGBoost (Extreme Gradient Boosting): A highly efficient and popular implementation of gradient boosting.
  - LightGBM: Another fast and efficient gradient boosting framework, often faster than XGBoost.
  - CatBoost: A gradient boosting library that handles categorical features well.

Classification:

- **Logistic Regression:** A linear model for binary classification that uses the logistic function to predict the probability of a sample belonging to a particular class.
- **Support Vector Machines (SVM):** Finding the optimal hyperplane that separates different classes with the largest margin.
- **k-Nearest Neighbors (k-NN):** Classifying a sample based on the majority class of its k nearest neighbors in the training data.
- **Naive Bayes:** A probabilistic classifier based on Bayes' theorem, assuming independence between features.
- **Decision Trees:** Building a tree-like structure for classification, where each node represents a decision based on a feature, and each leaf node represents a class label.
- **Random Forests:** An ensemble of decision trees for classification.
- **Gradient Boosting Classifiers:** Using gradient boosting for classification tasks.
  - GBM, XGBoost, LightGBM, CatBoost: (See descriptions under Regression)
- **Neural Networks (Multilayer Perceptron):** A network of interconnected nodes (neurons) organized in layers, capable of learning complex non-linear relationships.

By Shailesh Shakya @Beginnersblog & OpenAILearning

UNSUPERVISED LEARNING

1. Clustering:

- **k-Means:** Partitioning data into k clusters, where each data point belongs to the cluster with the nearest mean (centroid).
    - **Hierarchical Clustering:** Building a hierarchy of clusters, either by agglomerative (bottom-up) or divisive (top-down) approach.
    - **DBSCAN (Density-Based Spatial Clustering of Applications with Noise):** Grouping data points based on their density, identifying clusters of high density separated by areas of low density.
    - **Gaussian Mixture Models (GMM):** Modeling the data distribution as a mixture of Gaussian distributions, where each Gaussian represents a cluster.

2. Dimensionality Reduction:

Principal Component Analysis (PCA): (See description under Data Preprocessing)

- **Linear Discriminant Analysis (LDA):** (See description under Data Preprocessing)
- **t-distributed Stochastic Neighbor Embedding (t-SNE):** (See description under Data Preprocessing)
- **Autoencoders:** Neural networks trained to reconstruct their input, learning a compressed representation of the data in a hidden layer.

CONVOLUTIONAL NEURAL NETWORKS (CNNS):

- **Convolutional Layers:** Apply filters to input data to extract features.
- **Pooling Layers:** Reduce the spatial dimensions of feature maps, reducing the number of parameters and computation.
- **Filters/Kernels:** Small matrices that slide across the input data, performing element-wise multiplication and summation to produce feature maps.
- **Padding:** Adding extra pixels around the borders of the input to control the output size.
- **Stride:** The number of pixels a filter moves across the input in each step.
    - Common Architectures:LeNet: One of the first successful CNN architectures, used for digit recognition.
    - AlexNet: A deeper CNN that achieved significant improvements in image classification.
    - VGG: A CNN architecture known for its simplicity and use of small 3x3 filters.
    - ResNet (Residual Network): Introduced residual connections to enable the training of very deep networks.
    - Inception: Uses modules with multiple filter sizes to capture features at different scales.

RECURRENT NEURAL NETWORKS (RNNS)

- **Recurrent Units:** Process sequential data by maintaining a hidden state that is updated at each time step.
- **Vanishing/Exploding Gradients**: Problems that can occur during training RNNs, where gradients become too small or too large, making it difficult to learn long-range dependencies.
- Long Short-Term Memory (LSTM): A type of RNN cell designed to address the vanishing gradient problem by using a memory cell and gates to control the flow of information.
- **Gated Recurrent Unit (GRU):** A simplified version of LSTM that also uses gates to control information flow.
- **Sequence-to-Sequence Models**: RNN architectures that map an input sequence to an output sequence, used in tasks like machine translation and text summarization.
- **Attention Mechanisms:** Allow RNNs to focus on specific parts of the input sequence when generating the output sequence.

OTHER DEEP LEARNING TOPICS

- **Generative Adversarial Networks (GANs):** Consist of two networks, a generator and a discriminator, that are trained adversarially to generate realistic data.
- **Autoencoders:** (See description under Unsupervised Learning).
    - Variational Autoencoders (VAEs): A type of autoencoder that learns a probabilistic representation of the input data, allowing for generating new samples.
- **Transformers**: A neural network architecture based on the self-attention mechanism, which has achieved state-of-the-art results in many NLP tasks.
- **Transfer Learning in Deep Learning:** Using a pre-trained deep learning model on a large dataset and fine-tuning it for a specific task, often with a smaller dataset.
- **Object Detection:** Identifying and locating objects within an image.
    - YOLO (You Only Look Once): A real-time object detection system that performs detection in a single pass.
    - Faster R-CNN: A two-stage object detection system that uses a region proposal network to generate candidate object regions.
- **Image Segmentation:** Partitioning an image into segments, where each segment corresponds to a different object or region.
    - U-Net: A CNN architecture commonly used for image segmentation, particularly in biomedical applications.

DEEP LEARNING

CNN | NEURAL NETWORK | RNN

NEURAL NETWORK

- **Perceptron**: The simplest form of a neural network, a single-layer model with a linear activation function.
- **Activation Functions:** Introduce non-linearity into neural networks, allowing them to learn complex patterns.
    - Sigmoid: Outputs values between 0 and 1, often used in the output layer for binary classification.
    - ReLU (Rectified Linear Unit): Outputs the input if it's positive, otherwise outputs 0. A common choice for hidden layers.
    - Tanh (Hyperbolic Tangent): Outputs values between -1 and 1.
    - Softmax: Outputs a probability distribution over multiple classes, often used in the output layer for multi-class classification.
- **Backpropagation:** Algorithm for computing the gradients of the loss function with respect to the network's weights, used to update the weights during training.
- **Gradient Descent:** An optimization algorithm that iteratively updates model parameters in the direction of the negative gradient of the loss function.
    - SGD (Stochastic Gradient Descent): Updates parameters using the gradient calculated from a single data point or a small batch of data points.
    - Adam (Adaptive Moment Estimation): An adaptive learning rate optimization algorithm that combines the benefits of RMSprop and Momentum.
    - RMSprop (Root Mean Square Propagation): An adaptive learning rate method that maintains a moving average of the squared gradients.
- **Loss Functions:** Measure the error between predicted and actual values in neural networks.
    - MSE (Mean Squared Error): Commonly used for regression tasks.
    - Cross-Entropy: Commonly used for classification tasks.
- **Regularization:** Techniques to prevent overfitting in neural networks.
    - Dropout: Randomly dropping out neurons during training, forcing the network to learn more robust features.
    - L1/L2 Regularization: Adding a penalty term to the loss function based on the magnitude of the weights (see descriptions under Regularized Regression).
- **Batch Normalization:** Normalizing the activations of each layer to have zero mean and unit variance, which can speed up training and improve performance.
- **Weight Initialization:** Setting initial values for the weights of a neural network. Proper initialization can help with faster and more stable training. Examples include Xavier/Glorot initialization and He initialization.
- **Learning rate:** Controls the size of the steps taken during gradient descent.
- **Momentum:** Helps accelerate gradient descent by accumulating past gradients.
- **Batch size:** The number of training examples used in one iteration of gradient descent.

MODEL EVALUATION & TUNING

EVALUATION METRICS

Regression:

- **Mean Squared Error (MSE):** Average squared difference between predicted and actual values.
- **Root Mean Squared Error (RMSE):** Square root of MSE, provides an error measure in the same units as the target variable.
- **Mean Absolute Error (MAE):** Average absolute difference between predicted and actual values.
- **R-squared (Coefficient of Determination):** Proportion of variance in the target variable explained by the model.

2. Classification:

- **Accuracy:** Proportion of correctly classified samples.
- **Precision:** Proportion of true positives among predicted positives (TP / (TP + FP)). Measures the ability of the classifier not to label a negative sample as positive.
- **Recall:** Proportion of true positives among actual positives (TP / (TP + FN)). Measures the ability of the classifier to find all the positive samples.
- **F1-score:** Harmonic mean of precision and recall, balances both metrics.
- **AUC-ROC:** Area under the Receiver Operating Characteristic curve, measures the model's ability to distinguish between classes.
- **Confusion Matrix:** Table summarizing the performance of a classification model, showing counts of true positives, true negatives, false positives, and false negatives.

3. Clustering:

- Silhouette Score: Measures how similar a data point is to its own cluster compared to other clusters. Ranges from -1 to 1, with higher values indicating better clustering.
- Davies-Bouldin Index: Measures the average similarity between each cluster and its most similar cluster. Lower values indicate better clustering.

By Shailesh Shakya @Beginnersblog & OpenAILearning

CROSS VAILDATION

- **k-Fold Cross-Validation:** Dividing data into k folds, training on k-1 folds, and testing on the remaining fold, repeating k times.
- **Stratified k-Fold Cross-Validation:** Ensures that each fold has approximately the same proportion of samples from each class as the
- **Leave-One-Out Cross-Validation (LOOCV):** Using each data point as a test set and the remaining data as the training set, repeating for all data points. Computationally expensive but useful for small datasets.

HYPERPARAMETER TUNING

- **Grid Search:** Evaluating all possible combinations of hyperparameter values within a specified range.
- **Random Search:** Evaluating a random sample of hyperparameter combinations from a specified distribution. Often more efficient than grid search.
- **Bayesian Optimization:** Building a probabilistic model of the objective function (e.g., performance metric) and using it to select the most promising hyperparameter combinations to evaluate.

TOOLS & LIBRARIES

PYTHON LIBRARIES

- **NumPy:** Fundamental library for numerical computing in Python, providing support for arrays, matrices, and mathematical functions.
- **Pandas:** Powerful library for data manipulation and analysis, offering data structures like DataFrames for efficient data handling.
- **Scikit-learn:** Comprehensive machine learning library with a wide range of algorithms, tools for model selection, evaluation, and preprocessing.
- **TensorFlow:** Open-source deep learning framework developed by Google, known for its flexibility and scalability.
- **Keras:** High-level neural networks API that can run on top of TensorFlow, PyTorch, or Theano, simplifying the process of building and training deep learning models.
- **PyTorch:** Open-source deep learning framework developed by Facebook, known for its dynamic computation graphs and ease of use.
- **Matplotlib:** Plotting library for creating static, animated, and interactive visualizations in Python.
- **Seaborn:** Statistical data visualization library based on Matplotlib, providing a high-level interface for creating attractive and informative statistical graphics.
- **Statsmodels:** Library focused on statistical modeling, including regression analysis, time series analysis, and hypothesis testing.

OTHER TOOLS

- **Jupyter Notebook/Lab:** Interactive web-based environment for creating and sharing documents that contain live code, equations, visualizations, and narrative text.
- **Google Colab:** Free cloud-based Jupyter notebook environment with GPU and TPU support, provided by Google.
- **Git/GitHub:** Version control system (Git) and web-based hosting service (GitHub) for tracking changes to code and collaborating on projects.
- **SQL:** Structured Query Language, used for managing and querying relational databases.
  - Cloud Platforms (AWS, GCP, Azure) - ML Services:AWS (Amazon Web Services): Offers SageMaker for building, training, and deploying machine learning models.
  - GCP (Google Cloud Platform): Provides AI Platform for similar functionalities as SageMaker.
  - Azure (Microsoft Azure): Offers Azure Machine Learning for building, deploying, and managing machine learning models.
- **MLflow:** Open-source platform for managing the end-to-end machine learning lifecycle, including experiment tracking, model packaging, and deployment.
- **Weights & Biases:** A tool for tracking and visualizing machine learning experiments, providing insights into model performance and hyperparameter tuning.

DEPLOYMENT & MLOPS

MODEL DEPLOYME NT

- **REST APIs (Flask, FastAPI):** Creating web services that allow other applications to interact with a trained model.
  - Flask: A lightweight and popular web framework for building REST APIs in Python.
  - FastAPI: A modern, high-performance web framework for building APIs with Python 3.7+, based on standard Python type hints.
- **Containerization (Docker):** Packaging a model and its dependencies into a container for consistent and reproducible deployment across different environments.
- **Cloud Deployment (AWS SageMaker, Google AI Platform, Azure ML):** Deploying models on cloud platforms using managed services.
- **Serverless Deployment:** Deploying models as functions that are triggered by events, without managing servers (e.g., AWS Lambda, Google Cloud Functions, Azure Functions).

MLOPS

- **Model Versioning:** Tracking different versions of a model, including its code, data, and hyperparameters.
- Model Monitoring: Continuously tracking the performance of a deployed model and detecting issues like data drift or model degradation.
- **CI/CD for Machine Learning:** Applying Continuous Integration/Continuous Deployment principles to automate the process of building, testing, and deploying machine learning models.
- **A/B Testing:** Comparing different versions of a model in a production environment to determine which one performs better.

SPECIFIC TOPICS

Computer Vision:

- Image Processing: Manipulating and analyzing images using techniques like filtering, edge detection, and morphological operations.
- Object Detection: (See description under Deep Learning)
- Image Segmentation: (See description under Deep Learning)
- Image Classification: Assigning a label or category to an entire image.
- Transfer Learning in CV: Using pre-trained CNN models on large datasets (e.g., ImageNet) and fine-tuning them for specific computer vision tasks.

Time Series Analysis:

- Autocorrelation: Correlation of a time series with its own past values.
- Partial Autocorrelation: Correlation between a time series and its past values, removing the influence of intermediate lags.
- ARIMA (Autoregressive Integrated Moving Average): A class of statistical models for forecasting time series data.
- Exponential Smoothing: A family of forecasting methods that use weighted averages of past observations.
- Prophet: A forecasting procedure developed by Facebook, designed for business time series data with seasonality and trend changes.

Recommender Systems:

- Collaborative Filtering: Making recommendations based on the preferences of similar users or items.
- Content-Based Filtering: Making recommendations based on the characteristics of items and user profiles.
- Hybrid Approaches: Combining collaborative and content-based filtering techniques.

Anomaly Detection:

- One-Class SVM: A variation of SVM used for anomaly detection, where the model learns a boundary around the normal data points.
- Isolation Forest: An algorithm that isolates anomalies by randomly partitioning the data space.
- Autoencoders: (See description under Unsupervised Learning and Deep Learning). Can be used for anomaly detection by measuring the reconstruction error.

NATURAL LANGUAGE PROCESSING (NLP)

- **Tokenization:** Splitting text into individual words or units (tokens).
- **Stemming:** Reducing words to their root form (e.g., running, runs, ran -> run).
- **Lemmatization:** Reducing words to their base or dictionary form (e.g., better -> good).
- **Bag-of-Words:** Representing text as a collection of unique words and their frequencies.
- **TF-IDF (Term Frequency-Inverse Document Frequency):** A numerical statistic that reflects how important a word is to a document in a collection of documents.
- **Word Embeddings:** Representing words as dense vectors that capture semantic relationships.
  - Word2Vec: A popular technique for creating word embeddings by training a neural network on a large corpus of text.
  - GloVe (Global Vectors for Word Representation): Another method for generating word embeddings based on word co-occurrence statistics.
  - FastText: An extension of Word2Vec that also considers subword information.
- **Sentiment Analysis:** Determining the emotional tone or opinion expressed in text.
- Topic Modeling (LDA): Discovering abstract "topics" that occur in a collection of documents.
  - LDA (Latent Dirichlet Allocation): A probabilistic model that assumes each document is a mixture of topics and each topic is a distribution over words.
- **Text Classification:** Categorizing text into predefined classes (e.g., spam detection, news categorization).
  - Transformers (BERT, RoBERTa, GPT):BERT (Bidirectional Encoder Representations from Transformers): A powerful transformer-based model for various NLP tasks, pre-trained on a massive amount of text data.
  - RoBERTa (A Robustly Optimized BERT Pretraining Approach): An improved version of BERT with optimized training procedures.
  - GPT (Generative Pre-trained Transformer): A transformer-based model primarily used for text generation, also capable of other NLP tasks.

By Shailesh Shakya @Beginnersblog & OpenAILearning