Unsupervised Learning

Category	Approach	Application	Description	Hyperparameters & Regularization	Strength	Weaknesses	Performance Metrics
Hard Clustering	K-Means-Clustering	Clustering Analysis (grouping instances to clusters)	- randomly assign initial centroids spaced out on data - assign points to their nearest centroids (euclidean & manhattan) - calculate center for all clusters - make center new centroid - re-assign data to clusters and repeat until convergence	- number of clusters	- good for large datasets- easy cluster a	- only works for spherical data- only wor	- Inertia (mean squared distance of all centroids to instances in cluster) - Elbow-Method (Inertia vs. number of clusters; “joint” is optimal) - Silhouette-Coefficient
Hard & Soft Clustering	Gaussian Mixture Models	Density Estimation	- assume number of clusters, mu and sigma for each cluster - Expectation: * Compute membership probability for each data record * Calculate expression for responsibility for likelihood of all data points - Maximization: * Calculate total likelihood for all clusters * Update mu and sigma for all clusters based on likelihood * Repeat until convergance - Assign instances to cluster with highest membership probability	- number of clusters - initial mu and sigma	- hard and soft clustering - shows probabilities for data points - good for more complex datasets (ellipsoids)	- slow and computationaly expensive - can get stuck in local optima - only numerical features	- Bayesian Information Criterion - Akaika Information Criterion
Hard Clustering	Hierarchical Clustering	Clustering Analysis (grouping instances to clusters)	Bottom-up or top-down approach; dendogram-representation: - calculate proximity matrix from one instance to the other - Bottom-Up (agglomerative): * initiate every data point as a single cluster * step-by-step, add closest datapoints to clusters and clusters to clusters (single linkage, average linkage or complete linkage) * repeat until all data points are within one cluster (universe cluster) - Top-Down (divisive): * all instances start in universe cluster * split clusters based on farthest distances * repeat until all data points are individual clusters - find longest vertical line from one cluster to the other (biggest distance)	- None	- non-parametric - easy to implement - informative cluster hierarchy	- very sensitive to outliers - not feasible for large datasets - clusters cannot be un-made	
Hard Clustering	DBSCAN	Outlier Detection	- set MinPTS and Surface Radius - start at any data record and check if it forms a cluster; if it does, make it a center point - check for all other points in the cluster if they are either center points or border points (don’t form their own cluster) - start at record that was not assigned before and restart - non-assigned data records are outliers	- MinPTS and Surface Radius	- handles outliers well - outlier detection - doesn’t need number of clusters - can handle arbitrarily shaped datasets	- hard to setup minPTS and radius (domain expert needed) - needs clusters of same density	

Regression

Category	Approach	Application	Description	Hyperparameters & Regularization	Strength	Weaknesses	Performance Metrics
Regression	(Multi-) Linear Regression	Predicting continuous value	- needs standard deviation of the data, since RSS is MLE for normal distribution - needs to pass test for linearity - learns linear relation of data to target variable and makes predictions for target variable based on new feature values	- lambda-Parameter for weights - Drastic: Lasso Regression: add lambda*(sum coefficients) to cost function; cancels out small weights - Less drastic: Ridge regression; adds lambda*(sum coefficients)^2 to the cost function, dampening large weights	- weight parameters are transparent and easy to understand for linear models	- prone to overfitting, always needs regularization - susceptible to outliers	- mean squared error (succeptible to outliers) - root mean squared error (RMSE – more interpretable like standard dev.) - mean absolute error - R^2 (how much of variance in dependent variable stems from independent variables; 0 to 1, where 1 is optimal)
Regression	Polynomial Regression	Predicting continuous value	-applicable to non-linear data; rest: see above	- see above	- works with non-linear data	- see above	- see above
Regression	Quantile Regression	Predicting continuous value	- seperate dependent variable into different segments (quantiles) to train the model individually for each segment	- see above	- works with multimodal or skewed data	- see above	- see above
Regression	Support Vector Machines	Predicting continuous value	- learns relation of data to target variable by fitting as many points as possible on the hyperplane - trys to minimize weights instead of minimizing error! - all points have to be within certain margin (E) - derived function is used to make predictions for new instances - works like the “inverse” of the large margin classifier - for non-linear problems, soft margins are introduced with slack variables ksi	- Parameter C: Increase to decrease regularization; decrease to increase regularization (allows more point on margin) - Slack Variables ksi allow for points outside the margin (soft margin)	- Not given in script	- Not given in script	- Not given in script
Regression	Regression Trees	Predicting target variable	Build the tree: - calculate standard deviation of target variable - calculate standard deviation of all manifestations of feature variable and weight it with its probability - calculate reduction in standard deviation of target variable by subtracting standard deviation of all feature variables - select the feature that causes the highest reduction in standard deviation as the root/ next node	- Pruning the tree: continue as long as coefficient of variation (sigma / mu) stays under a certain value	- little assumptions on the data - allows regression based on non-numerical data (e.g. categorial data)	- needs pruning or would overfit	- Not given in script

CLASSIFICATION

Category	Approach	Application	Description	Hyperparameters & Regularization	Strength	Weaknesses	Performance Metrics
Classification	Logistic Regression	Binary or categorial classification	- based on bernoulli distribution - uses coefficients to determine a probability - decision for classification is done based on threshold (e.g. 90%)	- lambda or inverse (C) - Lasso Regression (drastic) - Ridge Regression (less drastic)	- takes binomial categorial or discrete data - works with big samples	- not feasible for non-linearly seperable datasets	Confusion Matrix: - Precision (TP / TP + FP) - Recall (TP / TP + FN) - F1-Score [2*Precision*Recall / (Precision + Recall)] - Accuracy [(TP + TN) / (TP + TN + FP + FN)]
Classification	Support Vector Machines	Categorial or discrete classification	Large margin classifier: Maximizes the width between support vectors of different classes: - Hard Margin Problem (linearly seperable data): All data points have to be classified in the correct class; no room for error (overfitting?) - Soft Margin Problem (nonlineary seperable data): Model allows certain errors within the classification, making the model more robust (less overfitting) w. slack-variable zeta and parameter C - Dimensionality: To make non-linearly separable data linearly separable, one can increase its dimension → curse of dimensionality (overfitting and cost for calculation) - Kernel-Trick: To avoid this, one can use the kernel trick, because it only transfers the relevant datapoints more efficiently w. dot products → Polynomial Kernel → RBF (Gaussian) Kernel	- Regularization Parameter C (increasing --> less regularization) - for RBF (Gaussian) Kernel: gamma (increasing → less regularization)	- high discrimination power - takes continuous and categorical data - works with linear and nonlinear problems - memory efficient (only needs support vectors) - good generalization	- needs proper mathematical understanding - harder to understand than more simple models	Confusion Matrix: - Precision (TP / TP + FP) - Recall (TP / TP + FN) - F1-Score [2*Precision*Recall / (Precision + Recall)] - Accuracy [(TP + TN) / (TP + TN + FP + FN)]
Classification	Decision Trees	Non-parametric classification for all kinds of data	Asking binary question: Yes or no – trees are built on roots (initial question) nodes (decision points) and leafs (answers) Splits are made by the highest gain of information for the target variable: - ID 3: “Information Gain”: → Gain (f) = Entropy (dataset) – weightet Entropy (f) (Dataset) - ID 4.5: “Split Info”: Avoids completely homogenous datasets (poor generalization!) and can use continuous and discrete features, while handling missing values and decision tree pruning. - CART Reduction in Impurity by using the Gini Index, implements post-pruning	Pruning: Pre-Pruning: Only split a node if a certain condition is met (max depth, min members in a node …) Post-Pruning: Remove parts of a tree that hold little information with a leaf that holds the label of the most frequent feature in the leaf (implemented in CART)	- non-parametric model - easy to interpret - uses discrete and numerical features	- always overfits if not pruned (recreates the input data by design) - very susceptible to outliers	Confusion Matrix and: ROC (Receiver Operating Characteristic Curve): - shows performance at all classification thresholds - Recall on abzissa, False Positive Rate on ordinate: → Recall = TP / (TP + FN), FPR = FP / (FP + TN) AOC: Area under ROC (from 0 to 1)
Classification	Ensemble Models	Non-parametric classification for all kinds of data	Meta Method: Combine several homogen or heterogen (weak) learners that are strong at specific parts of input space → combination evens out errors but overall prediction stays strong Bagging (for homegen weak lerners): - all samples are thrown in a “Bag” and repeatedly drawn to train the weak lerners, afterwards “returned to the bag” (each sample can be drawn multiple times) - all weak learners vote for classification, decision by majority vote (“agg”regation step) Pasting (for homogen weak lerners): - all samples are thrown in a bag and drawn only once as a subset to train the weak lerners (each sample can only be used once!) Boosting: - iterative approach → performance of all weak learners is improved by training samples step-wise - all samples are drawn repeatedly for training - missclassified instances get a bigger probability to be re-drawn Adaptive Boosting: - like boosting, but successful learners get more important vote in final classification step	- depends on the chosen models	- combines several different ML approaches and evens out their weaknesses; usually a good choice	- results might become more opaque	Confusion Matrix and: ROC (Receiver Operating Characteristic Curve): - shows performance at all classification thresholds - Recall on abzissa, False Positive Rate on ordinate: → Recall = TP / (TP + FN), FPR = FP / (FP + TN) AOC: Area under ROC (from 0 to 1)
Classification	Random Forest	Non-parametric classification for all kinds of data	Forest creation and voting: - build trees by choosing subset of training samples and usually subset of training features - trees vote for classification - training on “bagging” method	- number of trees in forest - maximal depth of a tree - minimal samples per leaf	- big features spaces - random subset of features reduces correlation between learners; reducing variance	- ?	See above

Category	Approach	Application	Description	Hyperparameters & Regularization	Strength	Weaknesses	Performance Metrics
Optimization	Genetic Algorithm	Heuristic optimization methods for hyperparameter tuning	Creation of population: - individual problem solutions are encoded in a Chromosome with Genes (0 and 1 values) Evolution of fitness function: - putting individual solutions towards the fitness function Selection of fittest: - looking for a certain amount of highest scoring chromosomes Crossover: - re-combining chromosomes on single or dual crossover Mutation: - flipping one bit to prevent being stuck in local optima	None	None	None	None