Math Properties

$log_a b = \frac{log_c b}{log_c a} = \frac{1}{log_b a}$
$P(A|B) = \frac{P(A \cap B)}{P(B)} = \frac{P(B|A)P(A)}{P(B)}$

Pre-Processing

Data Cleaning

Deletion: Remove missing values, remove duplicates
Imputation: Fill in missing values, mean/median/mode, similar case, forward fill, interpolation, default values
Outliers: Legitimate unusual values with respect to rest of data
Noise: Random errors and variations

Feature Engineering

Feature Transformation: Transform features to make them more useful
Feature Creation: Create new features from existing features
Normalization: Scale features to be between 0 and 1; $\frac{x_i - x_{\min}}{x_{\max} - x_{\min}}$
Standardization: Scale features to $\mu = 0$ and $\sigma = 1$; $\frac{x_i - \mu}{\sigma}$
Binning: Continuous $\rightarrow$ Categorical
Encoding: Categorical $\rightarrow$ Numerical
Sampling: Reduce **dataset** size
Aggregation: $(x_i, y_i, z_i) \rightarrow w_i$
Dimensionality: Attributes/columns in single case
Curse of Dimensionality: More dimensions $\rightarrow$ More sparse
Dimensionality Reduction:
Feature Selection: Subset of attributes, Filter, Embedded
Feature Extraction: Decrease dimensions while keeping max variance, PCA, SVD, LDA

Decision Trees

Grown recursively by partitioning data into subsets based on attribute values
Forms axis-parallel hyperplanes
c = number of classes, p = probability/fraction of class
Entropy: $\sum_{i=1}^{c} -p_i \log_2 p_i$
Gini: $1 - \sum_{i=1}^{c} p_i^2$
r = parent, k = number of partitions, n$_i$ = number of records in partition$_i$, Impurity = (Entropy, Gini)
Entropy $\rightarrow$ Information Gain, Gini $\rightarrow$ Gini Gain
Gain: $Impurity(r) - (\sum_{i=1}^{k} \frac{n_i}{n} Impurity(i))$
Split Info: $-\sum_{i=1}^{k} \frac{n_i}{n} \log_2 \frac{n_i}{n}$
Gain Ratio: $\frac{InfoGain}{SplitInfo}$
Classification: Assign labels to objects
Descriptive Modeling: Explain, describe, summarize
Predictive Modeling: Predict label of unknown record
Split Conditions:
Continuous: Threshold value, binning
Nominal & Ordinal: Multiway, binary w/ grouping
Characteristics: Inexpensive to construct, fast to test, easy to interpret, robust to outliers(especially when pruned), redundant/irrelevant attributes don't affect tree structure, eager learner
Pre-Pruning: Stop growing tree before it's fully grown

Linear Regression

Finds best fit line through data
Use one-hot encoding/dummy encoding for categorical data
There's also polynomial and non-linear regression
Least Squares: Minimize SSE
$\hat{\beta}_0 = \bar{Y} - \hat{\beta}_1 \bar{X}$
$\hat{\beta}_1 = \frac{\sum (X_i - \bar{X})(Y_i - \bar{Y})}{\sum (X_i - \bar{X})^2}$
SSE: $\sum_{i=1}^{n} (y_i - \hat{y}_i)^2$
MSE: $\frac{SSE}{n}$
RMSE: $\sqrt{MSE}$
MAE: $\frac{\sum_{i=1}^{n} |y_i - \hat{y_i}|}{n}$
var(mean): $\frac{\sum_{i=1}^{n} (y_i - \bar{y_i})^2}{n}$
var(fit): $\frac{\sum_{i=1}^{n} (y_i - \hat{y_i})^2}{n}$
R^2: $\frac{var(mean) - var(fit)}{var(mean)}$
Multiple Linear Regression: $y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_d x_d$
Characteristics: Non-decreasing(More features never make R^2 worse), More features $\neq$ better model, but can be better SSE)
n = number of samples, p = number of features
Adjusted R^2: $1 - \frac{(1-R^2)(n-1)}{n-p-1}$

Cross Validation

Training Error: Percentage misclassified(ex. SSE, R^2) on training set
Test/Generalization Error: Percentage misclassified on test/unseen set
Holdout Method: Split data into training and test sets; Issues- Less training data, overrepresentation/underrepresentation, varying performance

K-Fold Cross Validation

Split data into k folds, build model on k-1 folds, test on 1 fold, repeat k times, used **ONLY** to evaluate process
Error = $\frac{1}{k}\sum_{i=1}^{k} Error_i$
Build final model using **ALL** data
Cross Validation: Preprocess training set and apply exact same preprocessing to test set
Overfitting: Low training error but high test error, bad generalizations
Hyperparameter Tuning: Use validation set by partitioning training set
Nested Cross Validation: Use cross validation to tune hyperparameters, pick lowest error then test using entire training set
Get best hyperparameter using hyperparameter tuning on entire dataset, then build final model using the chosen hyperparameter and entire dataset

Nearest Neighbor

Usually Euclidean distance, can use weight factor = $\frac{1}{d^2}$
KNN: Find k nearest neighbors, assign majority class
k too small → overfit, k too large → underfit
Find best k using cross validation
Characteristics: Instance-based learning, lazy learner, no training(just retune k), slow testing, curse of dimensionality, feature selection critical

Naive Bayes

X = test record(x_1, x_2, ..., x_d), C = class
$P(C|X) \propto P(x_1|C) \cdot P(x_2|C) \cdot \ldots \cdot P(x_d|C) \cdot P(C)$
Requirements: Naïve → Independence, Binning
Laplace Smoothing: Used for multinomial/categorical data, add 1 to numerator and v(options for the feature) to the denominator
Characteristics: Fast, simple, scales with higher dimensions, independence assumption not always true

Evaluating Classifiers

Error Rate: Fraction of incorrect predictions on testing set; $\frac{FP+FN}{n}$
Accuracy: Fraction of correct predictions on test set; $\frac{TP+TN}{n}$
Confusion Matrix: P = predicted, A = actual
For cross validation, sum all confusion matrices

	$+_P$	$-_P$
$+_A$	TP	FN
$-_A$	FP	TN

TPR/Sensitivity/Recall/Coverage: $\frac{TP}{TP+FN}$, higher better
TNR/Specificity: $\frac{TN}{FP+TN}$, higher better
FPR: $\frac{FP}{FP+TN}$, lower better
FNR: $\frac{FN}{TP+FN}$, lower better
Precision: $\frac{TP}{TP+FP}$, higher better
F-Measure: $\frac{2 \times precision \times recall}{precision + recall}$, higher better
Fixing Class Imbalance:
Undersampling: Remove some majority class samples
Oversampling: Duplicate some minority class samples

Support Vector Machines

Hyperplane that maximizes margin between classes
Binary classifier, can be extended to multi-class
One-hot encoding for categorical data
Support Vectors: Changes hyperplane if moved, points on margin boundary or on the wrong side for its class
Goal: Minimize $\frac{||w||}{2}$ subject to $y_i(w \cdot x_i + b) \geq 1$
Soft Margin SVM: Minimize $\frac{||w||}{2} + C(\sum_{i=1}^{N} \xi_i)$ subject to $y_i(w \cdot x_i + b) \geq 1 - \xi_i$
$C \to \infty$ = Hard Margin SVM
Too little slack → overfit, too much slack → underfit
Trade-off between margin width and incorrect classification
Kernel Methods: Transform data into higher dimensions for linear seperability
Characteristics: Global optimization, requires feature scaling, extendable to multi-class, no curse of dimensionality, needs cross validation(hyperparameters, kernel function, cost)

Non-Linear Regression

Regression Tree: Decision tree with continuous output rather than categorical, recursive partitioning, axis-parallel
KNN Regression: Continuous KNN with neighbors means as output
Support Vector Regression: Fit hyperplane to minimize error, aka. points outside tube
Ensembles: Bagging/Boosting, average of the base classifiers