# Machine learning algorithm

## Linear regression

**Statsmodels.api**

Adjusted R2: reflects fit of model  
Const. coefficient: Y-intercept  
Coefficients: change in output due to change in X\_i  
standard error: reflects level of accuracy of coefficients. Lower it is, the higher the accuracy  
: p-value  
Confidence interval: range in which the coefficients are likely to fail.

## Classification

Categorical target variable with at least 2 categories: binary VS multinomial classification

## Polynomial regression

* Simple linear regression
* Multiple linear regression
* Polynomial linear regression

## Gradient descent

* Takes gradient of loss function for each parameter
* Pick random values for parameters
* Plug parameters values into gradient
* Calculate step size = slope\*learning\_rate
* Calculate new parameter new = old-step
* Re-plug values into gradient and iterate until step\_size<threshold, or max\_iter

**Stochastic gradient descent**: uses randomly selected subset of data at everystep

* Reduces computation time of derivatives of loss function
* Useful when there is redundancy in data (clusters)

## Logistic regression

**Odds:**  ratio of event happening to event not happening  
**Probability:** ratio of event happening to everything that can happen

Logit function = -> basis of logistic regression

**Binary logistic regression**: categorical response has 2 possible outcomes  
**Multinomial LR**: 3 or more categories without ordering  
**Ordinal LR**: 3 or more categories with ordering

### Binary LR

Maximum likelihood estimation:  
- maximization method to determine the parameters that are more likely to produce observed data  
- sets mean and variance as parameters in determining the specific parametric values for a given model  
- assumes joint probability mass function

Advantages:

* Efficient
* Easy to implement
* Doesn’t require scaling of features
* Provide probability score of observations

Disadvantages:

* Cannot handle large number of categorical features
* Vulnerable to overfitting
* Cannot solve non-linear problems, requires transformations of non-linear features.

### Multinomial LR

(soft-max regression)

Generalizes logistic regression to multiclass problems. Assumes that classes are mutually exclusive.

## Generalized Linear Model

Flexible generalization of ordinary linear regression which allows for response variables that have error distributions models other than normal distributions.  
Linear model to be related to the response variable via a link function, and allows magnitude of the variance of each measurement to be a function of its predicted value.

For linear regression problems when noise distribution is not normal.

* Relationship X and Y does not look linear
* Variance of Y does not look constant with regards to X

Instead, can apply Poisson regression model (for example)

Link function: links linear predictor and parameters for probability distribution.

## Decision Trees

Non-parametric supervised learning method used for classification or regression.  
Learn from data to approximate a curve with if-then-else decision rules.  
Builds model in form of tree structure, breaking down dataset into smaller and smaller subsets, while an associated decision tree is incrementally developed. = Decision tree with leaf nodes

Gini impurity = 1 – p(True)^2 – p(False)^2

**Attribute selection measures:**heuristic for selecting splitting criterion that partitions the data in the best possible manner  
provides a rank for each feature, where the best score will be selected as splitting attribute.

**Information gain:**  
Entropy ~ measures the impurity of input sets  
Information gain is the decrease in entropy

**Gain ratio:**Information gain is biased for attribute with many outcomes, it may create useless partitioning in looking to maximize IG.  
Gain ratio normalizes the IG using split information.

**Gini index:**  
considers binary split for each attribute  
computes a weighted sum of impurity of each partition

**OPTIMIZING A DECISION TREE**  
Choose attribute selection measure (‘gini’, ‘entropy’)  
Choose splitting strategy (‘best’, ‘random’)  
Maximum depth of tree: high = overfitting, low = underfitting

## Random Forest

Flexible, easy to use, provides good results even without much hyperparameter tuning.

1. Create bootstrapped dataset
2. Create decision tree using bootstrap dataset, but only use a random subset of variables/columns at each step
3. Repeat 1-2 to make MANY trees

Variety of tree from bootstrapping and random subset makes forest flexible.

To predict, run data through every decision tree keeping track of outputs  
-> Bagging: bootstrapping the date while aggregating to make decision

Can assess accuracy of forest by passing out-of-bag datasets through trees from which they were excluded. -> out-of-bag error  
Step 2), typically start with square root of number of features.

Advantages of Random forests

* Highly accurate and robust due to number of decision trees
* Does not suffer from overfitting by taking average of all predictions, which dissipates biases
* Classification or regression
* Can handle missing value
  + Using median for continuous variables
  + Computing the proximity-weighted average of missing values
* Can get feature relative importance

Disadvantages

* Slow in generating predictions because of multiple trees
* Model is difficult to interpret

Shallow trees: few depths = less variance but higher bias (best for sequential methods)  
Deep trees: lot of depth = low bias but high variance (relevant for bagging methods)

Sampling over features makes trees look different and therefore reduces the correlation between outputs.

RF combines bagging and random feature subspace selection to create more robust models.

## Ensemble learning

Ensemble methods use multiple learning algorithms to obtain better predictive performance than could be obtained from any of the constituents learning algorithms alone.  
Combining weak learners correctly, can obtain accurate/robust models

Coherence of choice for weak learners  
If base models have low bias/high variance, we need to aggregate with a method that tends to reduce variance.

## Bagging

Considers homogeneous weak learners  
Learn independently from each other and then combines them following some deterministic averaging process  
goal is to get ensemble model with less variance than its components.  
parallel method, can be trained concurrently

## Boosting

Sequential methods fit models iteratively such that the training of a model at a given step depends on the models at previous steps.  
Produces an ensemble model that is less biased than its weak learners.  
Each model in the sequence is fitted giving more importance to observations in the dataset that were badly handled by previous models in the sequence  
Base model should have low variance but high bias, e.g. shallow decision tree

## Adaptive boosting

Having a weighted sum of weak learners is a difficult optimization problem -> addressed through an iterative optimization process. Instead of optimizing globally over all models, we approximate by optimizing locally.

## Gradient boosting

Casts problem into gradient descent one

At each iteration we fit a weak learner to the opposite of the gradient of the current fitting error with respect to the current ensemble model.

## Stacking

Learns to combine (heterogeneous) weak learners using a meta-model.  
Usually requires to split data for training models and meta model, but to avoid this, can follow k-fold cross training such that all observation can be used to train meta model.

## XGBoost

Special type of gradient boosting algorithm  
Popular because of speed and accuracy

Extreme Gradient Boosted Trees

Multiple trees built on top of each other that will improve on the performance of the sequence by correcting the errors of the previous trees before it.

* Regularized boosting
* Can handle missing values automatically
* Parallel processing
* Can cross-validate at each iteration
  + Enables early stopping, finding optimal number of iterations
* Incremental training
* Can plug in own optimization objectives
* Tree pruning
  + Generally results in deeper, but optimized trees

Hyperparameters:

Booster: ‘gbtree’ for classification, ‘gblinear’ for regression  
Objective: e.g. ‘multi:softmax’ …  
ETA: learning rate  
max\_depth: depth of tree (shallow = underfit, deep = overfit)  
min\_child\_weith: can control overfitting, but too high will underfit  
etc…

## Naïve Bayes classifier

Probabilistic machine learning model that is used for classification tasks  
assumes that variables in the dataset are uncorrelated.

* Fundamental Naïve Bayes assumptions: each feature makes an independent and equal contribution to the outcome.

## Gaussian Naïve Bayes classifier

Continuous values associated with each feature are assumed to be distributed according to a Gaussian distribution

## Multinomial Naïve Bayes

Feature vectors represent the frequencies with which certain events have been generated by a multinomial distribution, ex: document classification with term frequency

## Bernoulli Naïve Bayes

Multivariate Bernoulli event model, features are independent Booleans describing inputs, ex: document classification with term occurrence.

Naïve Bayes learners and classifiers can be extremely fast compared to more sophisticated methods.  
Decoupling of class conditional feature distributions means that each distribution can be independently estimated as 1D distribution.

## Support Vector Machine

Plot each item as a point in an n-dimensional space (where n is number of features) with the value of each feature being the value of the given coordinate. Classification is performed by finding the optimal hyper-plane that differentiates the 2 classes.

SVM uses Kernel functions to systematically find SVC in higher dimensions. E.g. Polynomial kernel, Radial kernel…

Hyperparameters

* Kernel: transforms given dataset input into required form
* Regularization: C is a penalty parameter which represents misclassification or error term.
* Gamma: number of nearby points to consider to find separation

Advantages

* Good accuracy and faster prediction times relative to Naïve Bayes
* Use less memory because they use a subset of training points in decision phase
* Works well with clean margin of separation and high-dimensional space

Disadvantages

* Not suitable for large datasets because of high training time
* Work poorly with overlapping classes
* Sensitive to types of kernel used.

# Evaluation

Generalization error: measure of how accurately an algorithm is able to predict the outcome value of data.

Loss function: measures how far an estimated value of a quantity is from the true value.  
-> Squared error loss (sensitive to outliers)  
-> Absolute error loss (not smooth at )  
->

Contributions to generalization error:   
-> approximation: pool functions that are too simple and don’t capture complexities  
-> estimation: from not enough data  
-> optimization: loss function too complex

## Regression metrics

* Mean squared error
  + Average of the squared difference between target and predicted value
  + Differentiable, can be optimized
* Root-mean squared error
  + Squared root of MSE
  + Good when large errors are undesirable
* Mean absolute error
  + Robust to outliers and does not penalize the errors as much as MSE (not applicable to applications where you want to pay attention to outliers)
* , coefficient of determination
  + Helps comparing models with a constant baseline

## Classification metrics

* Accuracy
  + # correct / # predictions = (True positive + True negatives)/ Predictions
  + Works when both possible outcomes are balanced
* Precision/Recall
  + True positives / (True positives + False positives) -> evaluates data on positive performance. Can miss many false negatives
  + True positives/(True positives + False negatives) -> evaluates performance on ground truth of positive outcomes. Misses false positives
* F1-score
  + Takes False positive and False negatives into account.
  + Useful in uneven class distributions
  + F1 = 2(precision\*recall)/(precision+recall)
* ROC curve/AUC score
  + Receiver operating characteristic curve -> shows performance of a classification model at all classification thresholds
  + Area under ROC curve [0,1]
  + ROC looks at True positive rate = (TP/(TP+FN)) VS False positive rate = (FP/(FP+TN)). We want maximum tpr while minimizing fpr
  + Most useful when comparing model to itself
* Log-loss
  + Takes into account the uncertainty of prediction based on how much it varies from actual label.
  + More nuanced view into performance of the model

**Grid Search**: process of performing hyperparameter tuning so as to determine the optimal value for a given model.

## Linear regression

Loss function:

Looking for vector *w* which minimizes our loss function

-> percentage of variation in *y* is explained by *X*

Adjusted R2 scales by the number of parameters (more parameters give more opportunities for randomness to reduce SS(fit).

F: tells us if R2 is significant

## Bias in Machine Learning

Bias: inability for a ML method to capture the true relationship.

Managing biases

* Choose the right learning model for the problem
* Choose representative training dataset
* Monitor performance using real data

Variance: difference in fits between datasets (training/testing)

Finding sweet-spot between under/over-fitting

* Regularization
* Boosting
* Bagging

## Regularization

Desensitization, technique to avoid over-fitting. Adds a penalty on different parameters of a model to reduce the freedom of the model.

**Ridge:** introduces small amount of bias into fit of training data, e.g., minimize sum of squared residuals + . The large lambda, the least sensitive our regression is to X. (L2 regularization).

**LASSO**: minimize sum of squared residuals + . Can exclude “useless” variables, which is good to reduce variance for models with lots of variables. (L1 regularization). Sparsity: when most coeffs. become 0.

## K-fold cross validation

Systematically creates and evaluates multiple models on multiple subsets of the dataset.  
-> can calculate mean of performance measures to get a sense of how well model performs on average.  
-> can calculate the standard deviation of performance measures to get a sense of much the results are expected to vary.

Data is divided into k-subset  
Hold-out method is repeated k-times such that each time, 1 of the k-subsets is used as the test/validation and other (k-1) subsets are put together to form a training dataset.

= Bias reduction because most of data is used for training (k-1) times, and reduces variance as all data points are used in validation set. K~5-10

## Stratified K-fold CV

Similar to k-fold, where each fold contains approx. the same percentage of samples of each target class as the complete set.

## Leave-p-out CV

Leaves p points out of training data (n-p), and this is repeated for all combination in which original sample can be separated this way.

## Bootstrap

Randomly draw datasets from training sample, each sample = same size as training sample, refit the model with bootstrap sample, examine model.  
Statistical properties: drawn directly from true underlying distribution and independently from each other.   
Only an approx. of representativity (N must be large enough to capture most of complexity of underlying distribution) and of independence (N should be large enough relative to B so that samples are not correlated).  
Can be used to evaluate variance or confidence intervals of a statistical estimator.

## Resampling methods

Statistical procedures for sampling a dataset and estimating an unknown quantity.

## Validation

Process of deciding whether the numerical results quantifying hypothesized relationships between variables are acceptable as description of the data.

## Overfitting

Signal: true underlying pattern that you wish to learn from the data  
Noise: irrelevant information or randomness in dataset

Overfitting = high variance in predictions  
Underfitting = high bias in predictions

**Preventing overfitting**-> Cross-validation  
-> Train with more data  
-> Remove features  
-> Early stopping (for iterative training algorithms)  
-> Regularization  
-> Ensembling:

bagging = training large numbers of strong learners in parallel, combining and smoothing their predictions  
boosting = training large numbers of weak learners, sequence of learning from previous mistakes, combine weak learners into strong one.