Algorithmic Trading Bootcamp: Linear and Nonlinear Algorithms

By Deepak Kanungo
CEO, Hedged Capital LLC
February 9, 2023

Course Outline

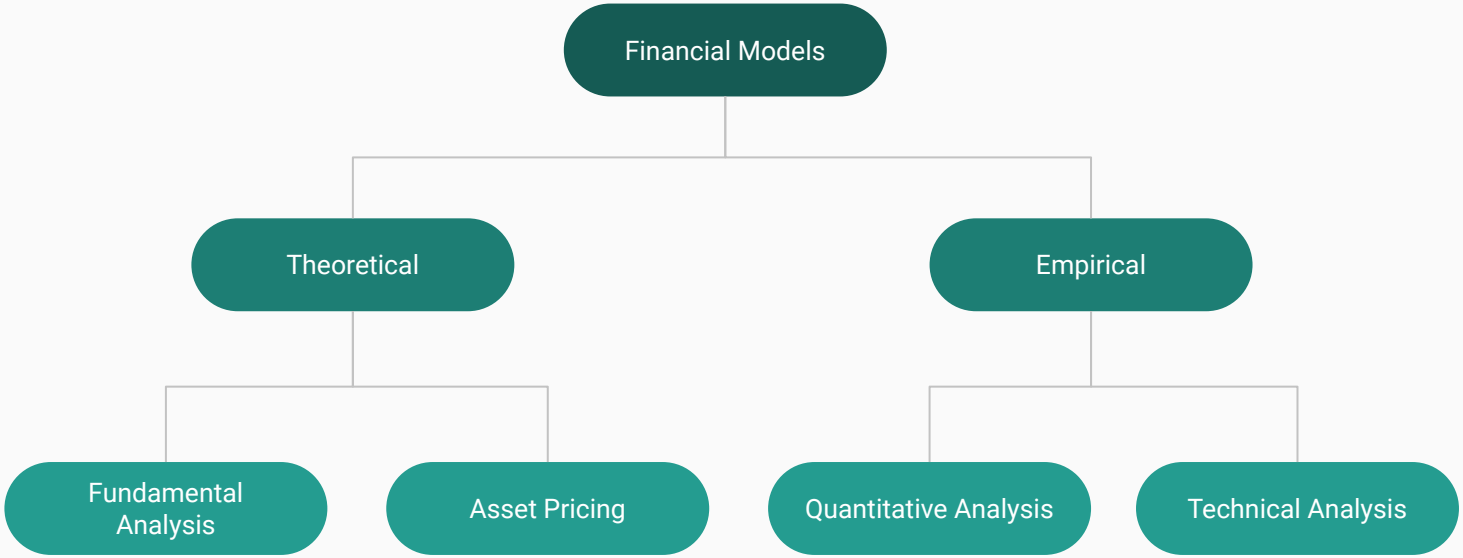
1. Linear Regression
2. Logistic Regression
3. Decision Trees
4. Random Forests and Gradient Boosted Machines

Linear Regression

Section 1

1. Ordinary least squares
2. Maximum likelihood estimates
3. Ridge regression
4. Lasso regression

Finance has been an early adopter of AI/ML



Linear Regression: The Market Model (MM)

$$(R - F) = a + b\,(M - F) + s$$

- Standard linear regression model where:
  - Y = (R - F) is the outcome variable or label, X = (M - F) is the predictor variable of feature
    - R is the realized return of a stock
    - F is the return on a risk-free asset (US treasury bill)
    - M is the realized return of a market portfolio (S&P 500)
    - **a** (alpha) is the expected stock-specific return
    - **b** (beta) is the level of systematic risk exposure to the market
    - **s** (sigma) is the unexpected stock-specific return

Model assumptions: Gauss-Markov theorem

- Linear regression makes the following assumptions:
  - Residuals are independent and identically distributed
  - Expected mean of the residuals is zero
  - Variance of the residuals is finite and constant
  - Residuals are normally distributed

Hands-on Exercise

- Simple linear regression with Statsmodels
  - Click here for Colab notebook

Linear regression algorithm

- Predicted response of a linear regression algorithm is a weighted sum of its input features:
- Y = w[0] + w[1]*x[1] + …..+ w[n]*x[n]
  - Y is predicted response
  - X[1]...x[n] are the input features
  - w[0]...w[n] are constant coefficients to be learned from training data
- Graphically it is a line in 2-dimensions, a plane in 3-dimensions and a hyperplane in n-dimension space

Performance measures for regression

- Error metrics are calculated based on the distance between predicted value and actual value
  - Mean squared error (MSE) : sensitive to outliers
  - Mean absolute errors (MAE): Not sensitive to outliers
  - Median absolute errors (MedAE): Not sensitive to outliers
  - R-squared: A standardized version of MSE

How a linear model learns

- Linear models learn the coefficient/weights from the training data by minimizing an error/performance metric:
  - Minimizes the sum of the squared distance between prediction and actual values
  - Assumes residual errors are normally distributed with constant variance
  - Ordinary least squares (OLS) regression algorithm is over 200 years old!
  - Same as maximum likelihood estimate (MLE) using Gauss-Markov theorem
- In higher dimensions, linear models are powerful baseline models
  - Easy to interpret and build
  - Fast to train and predict
  - Highly scalable on large datasets

Regularization reduces overfitting the data

- Financial data are abundant but have very low signal to noise ratio
  - Risk overfitting the data and learning from the noise instead of the signal
- Linear models that have highly correlated features tend to overfit data
  - Need methods to prevent overfitting of data
- Regularization uses two different types of shrinkage methods to:
  - Improve interpretation of predictions and inferences
  - Reduce generalization errors by reducing variance/overfitting

Lasso versus ridge regression models

- Two types of shrinkage methods that penalize complexity:
  - Lasso or L1 regularization: penalizes the sum of the absolute values of the coefficients
  - Ridge or L2 regularization: penalizes the sum of the coefficients squared
- In lasso regression, many of the coefficients are shrunk to zero
  - Used to eliminate features and increase interpretation of model
- In ridge regression, all coefficients are shrunk to near zero
  - Reduces the impact of any one feature
  - Coefficients with less variance are reduced more
- Features need to standardized prior to training

Hands-on Exercise

- Use linear regression models to forecast stock price returns
  - Click here for lasso and ridge regression models with Scikit-learn

Feature engineering is key to performance

- Features of a model enable good inference and prediction
  - Need systematic methods for selecting informative features
- Remove numerical features with low variances
  - Drop features that do not meet a variance threshold
- Drop some of the highly correlated features
  - Quantify the correlation among features
- L1/Lasso regularization can be used remove unimportant features
- Human expertise, judgement and experience are still important!

Logistic Regression Models

Section 3

1. Classification
2. Logistic algorithm
3. Hands-on exercise

Classification problems in finance

- For many problems in finance, you are interested in classifying an uncertain outcome as opposed to measuring its value:
  - Recession or growth
  - Bankruptcy or solvency
- Binary classes are easily extended to multiclass classifications
  - One versus rest method
- Also need to quantify the probability of belonging to a particular class
  - Probabilities need to add up to 1

Logistic/Sigmoid function

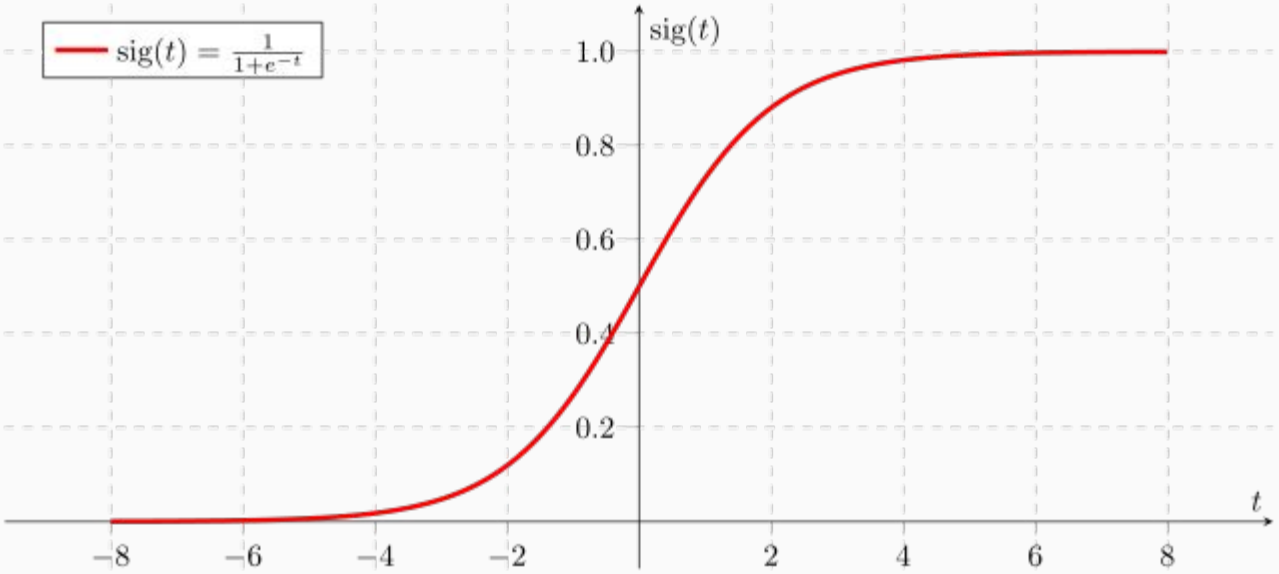


Image source: Creative Commons

Logistic regression algorithm

- A linear model is extracted from a logistic/sigmoid function via a logarithmic transformation
  - Coefficients are linear in the feature vector X
- The logistic function constrains the value of the function's output to between 0 and 1
  - Can be interpreted as the probability of belonging to a particular class
- A logistic regressor is the simplest neural network

How a logistic regression model learns

- Logistic regression models learn the coefficient/weights from the training data by using maximum likelihood estimation methods
  - You can also use non-linear least squares method
- Lasso/L1 regularization reduces features and overfitting
- Ridge/L2 regularization reduces overfitting
- Features need to be rescaled or standardized

The Confusion Matrix

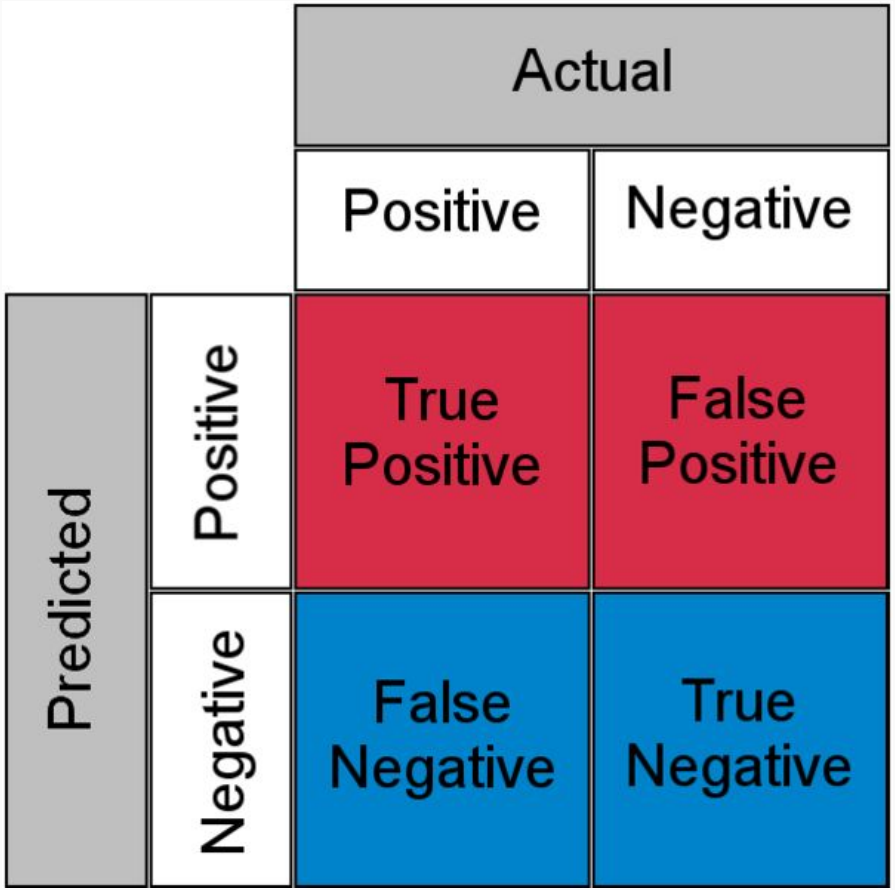


Image source: Creative Commons

Performance metrics for classifiers

- Accuracy = (TP + TN)/(TP + TN + FP + FN)
  - Misleading metric in imbalanced datasets
- Precision = TP/(TP + FP)
  - Focuses on the number of false positives
- Sensitivity or recall = TP/(TP + FN)
  - Focuses on the number of false negatives
- F-measure = 2 * (precision * sensitivity)/(precision + sensitivity)
  - Harmonic mean balances precision and sensitivity

Hands-on exercise

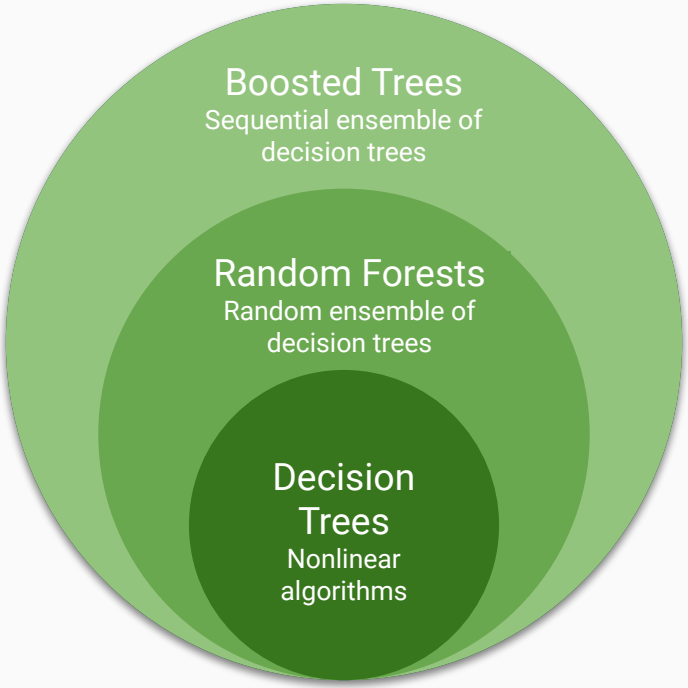
- Click here to use logistic regression models to forecast an economic recession

Decision Tree Based Models

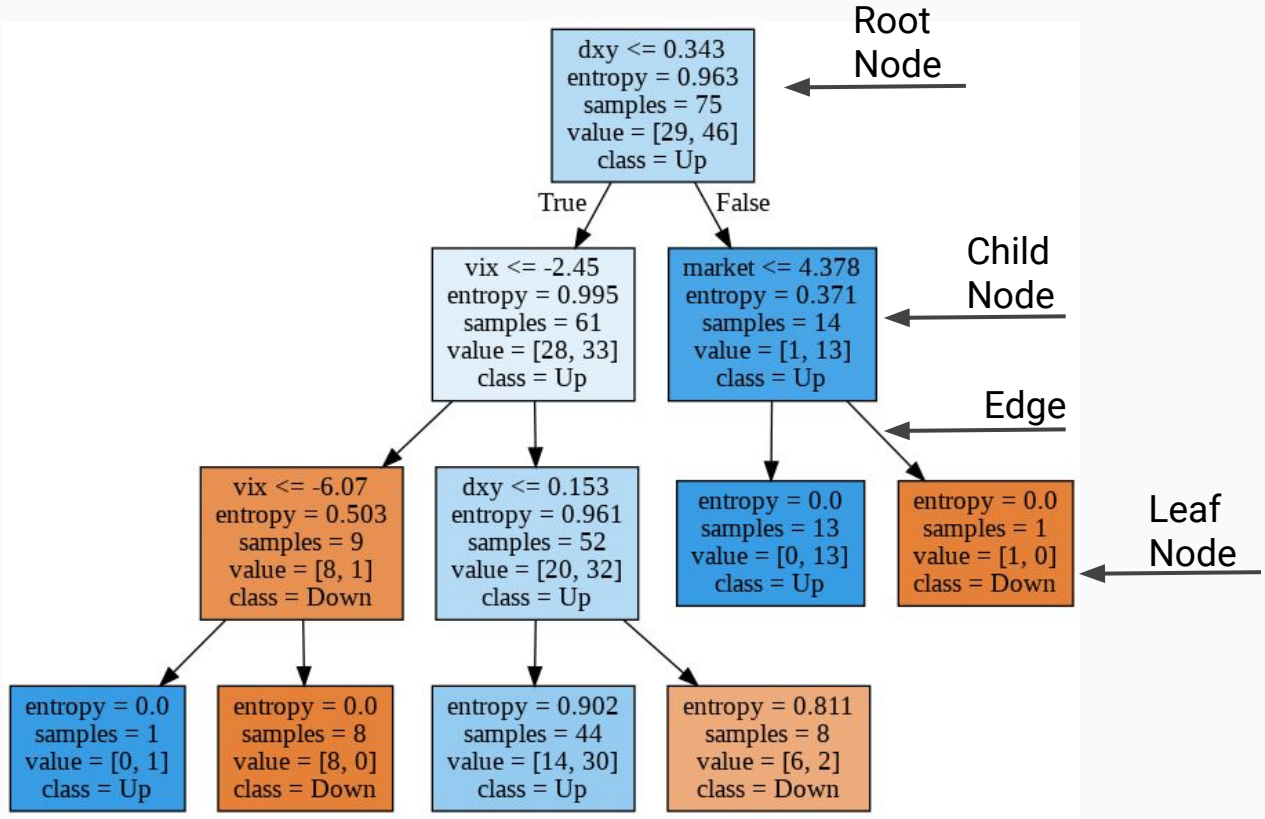
Section 3

1. Decision Trees
2. Hands-on Exercise
3. Regularizing Trees

Decision Tree (DT) based models use powerful nonlinear algorithms



A Decision Tree algorithm is a feature based series of true/false questions



Why use Decision Tree models

- One of the most powerful nonlinear base learners
  - Robust and scalable
- They are intuitive, white box models
  - Easy to interpret and visualize
- Not much data preprocessing is required
  - No need to standardize, normalize or scale data
- Quantifies uncertainty in predictions using probabilities
  - Only available in the classification algorithm

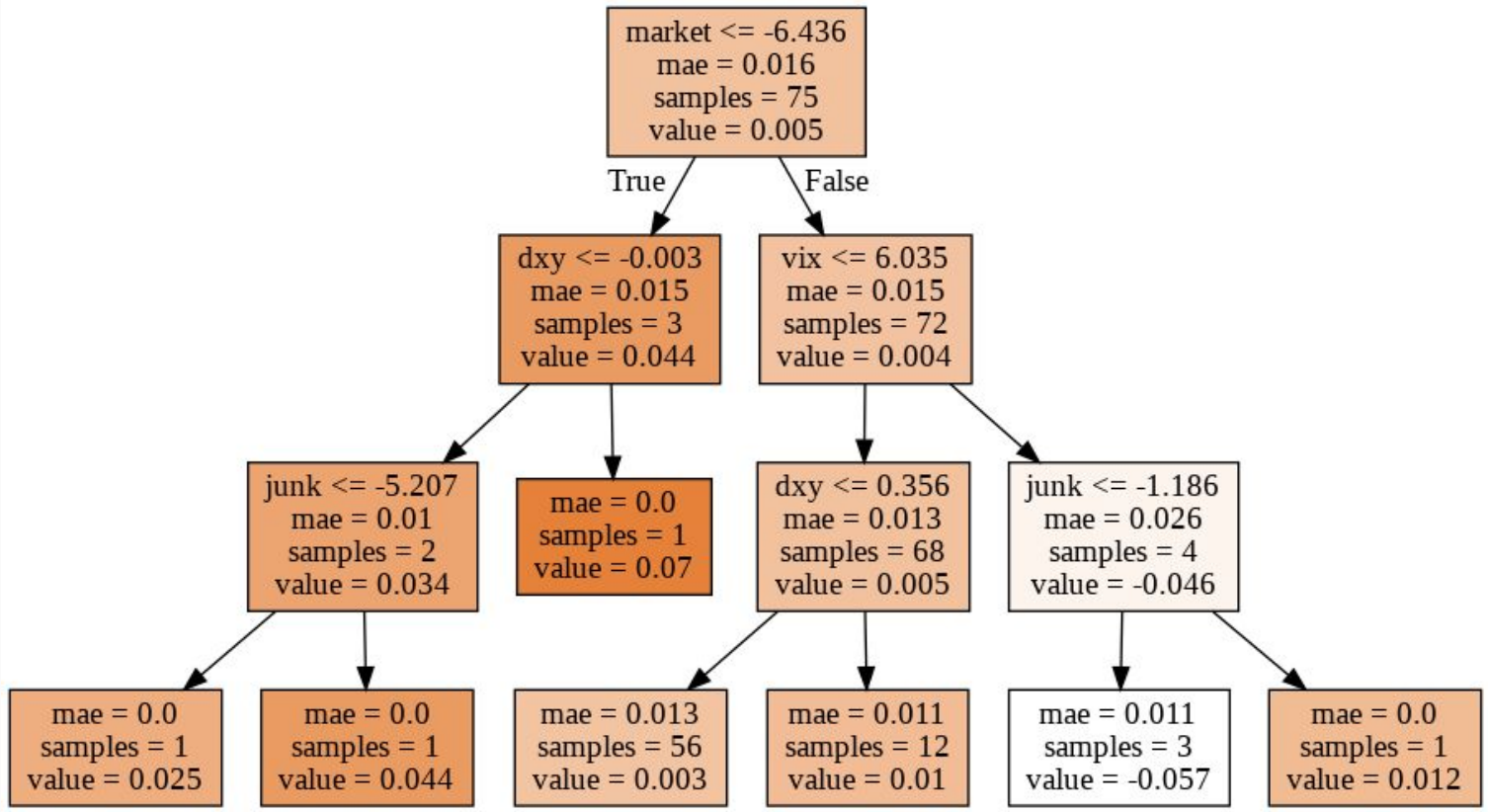
Scikit-Learn uses the CART algorithm

- Classification and Regression Tree (CART) algorithm
  - Recursively splits training set into two subsets based on a threshold value of a feature
  - Stops when it reaches its maximum depth or cannot split the data anymore
  - Produces a sequence of true/false questions leading to a most likely answer
  - Because questions have binary outcomes, non-leaf nodes will always have two children
- Solution produced by CART algorithm is not guaranteed to be optimal
  - Optimal tree is not solvable even for small datasets
- Uncertainty of classifications is quantified probabilistically
  - Simple ratio of each class divided by total sample size in the terminal node
  - Classification error rate is a similar ratio

How a DT classification model learns

- DT classification models learn feature threshold values recursively by minimizing a performance metric:
  - Gini Impurity = 1 - (R1^2 + …+ Rn^2)
    - where R is the ratio of the number of instances of a class to the training sample size of a node
  - Entropy = - {R1 * log(R1) + …. + Rn * log(Rn)}
    - where R is the ratio as defined above and log is the binary logarithm (base 2)
- Recursion stops automatically when the chosen metric equals zero
  - Gini impurity is the default metric and is faster to compute
  - Entropy produces slightly more balanced trees

Regression trees predict target values instead of classes at each node



How a DT regression model learns

- DT regression models learn feature threshold values recursively by minimizing a performance metric:
  - Mean squared error: {(A - Y1)^2 + …+(A - Yn)^2}/n
    - where A is the target value averaged over the entire sample size and Ys are target values of each training instance of the node
  - Mean absolute error: (|A -Y1| + ….+ |A - Yn|)/n ,
    - where A and Ys are the same as above
- Recursion stops automatically when the chosen metric equals zero
  - Mean squared error is sensitive to outlier values
  - Mean absolute error is not as sensitive to outlier values

Hands-on exercise

- Use built-in Colab document to use DT models to predict an economic recession and percentage loss in GDP
  - Click here for DT classification and regression models with Scikit-learn

Analyzing the importance of features in DT

- DT models can get overwhelming complex
  - Feature importance function helps summarize the decision-making process of a tree
- Quantifies how much a feature contributed in predicting the target value
  - Values are between 0 and 1 and the total sum of feature importances equals 1
- Nonlinear relationship between feature and target value
  - High importance value does not indicate a linear correlation with target value
  - Low importance value means that information in the feature is likely redundant

Regularization reduces overfitting the data

- Financial data are abundant but have very low signal to noise ratio
  - Risk is overfitting the data and learning its noisy structure
- Decision Trees can easily overfit training data with training scores of 100%
  - Need methods to prevent overfitting of data
- Regularization uses shrinkage methods to:
  - Improve interpretation of predictions and inferences
  - Reduce generalization errors by reducing variance/overfitting

Regularizing DTs with hyperparameters

- The following pre-pruning hyperparameters reduce overfitting of data
  - max_depth: reduces the complexity level of a tree
  - max_features: maximum number of features to be evaluated per node before a split
  - max_leaf_nodes: maximum number of leafs in a tree
  - min_samples_split: minimum sample size a node must have before a split
  - min_samples_leaf: minimum sample size required in a leaf node

Weaknesses of DT models

- DT models can get complex very fast even with a small set of features
  - Regression trees can get very deep
- Overfitting of data even with regularization
  - Poor out of sample performance, especially in regression models
- Highly sensitive to small variations in training data
  - Predictions of a single tree are unstable

Advanced Decision Trees

Section 4

1. Random Forest Algorithm
2. Hands-on Exercise
3. Gradient Boosted Machines
4. Hands-on Excercise

Why use Random Forest (RF) models

- One of the most powerful ensemble learning methods
  - Highly robust and scalable to very large datasets
- RF average has similar bias but lower variance than any single DT
  - Better generalization of predictions than DT models
- Like DT models, not much data preprocessing is required
  - No need to standardize, normalize or scale data
- Like DT models, quantifies uncertainty in predictions using probabilities
  - Only available in the classification algorithm

RF algorithm reduces variance of DTs

- Each DT is built uniquely using randomness:
  - Random sampling of training data with replacement (Bootstrapping)
  - Random subset of features
- DT algorithm is modified so it doesn't look for the best test for each node
  - It selects a random subset of features for each node
  - Selects the best possible test for one of the features in the subset above
- RF takes the prediction of each DT and averages them (Aggregating)
  - Each trained DT is still a weak learner but the ensemble is a much stronger learner
- Two different ways of aggregating classifier predictions
  - Soft voting averages predictions and hard voting takes the majority vote of classifiers

RF algorithm is trained using bagging (bootstrapping and aggregating)

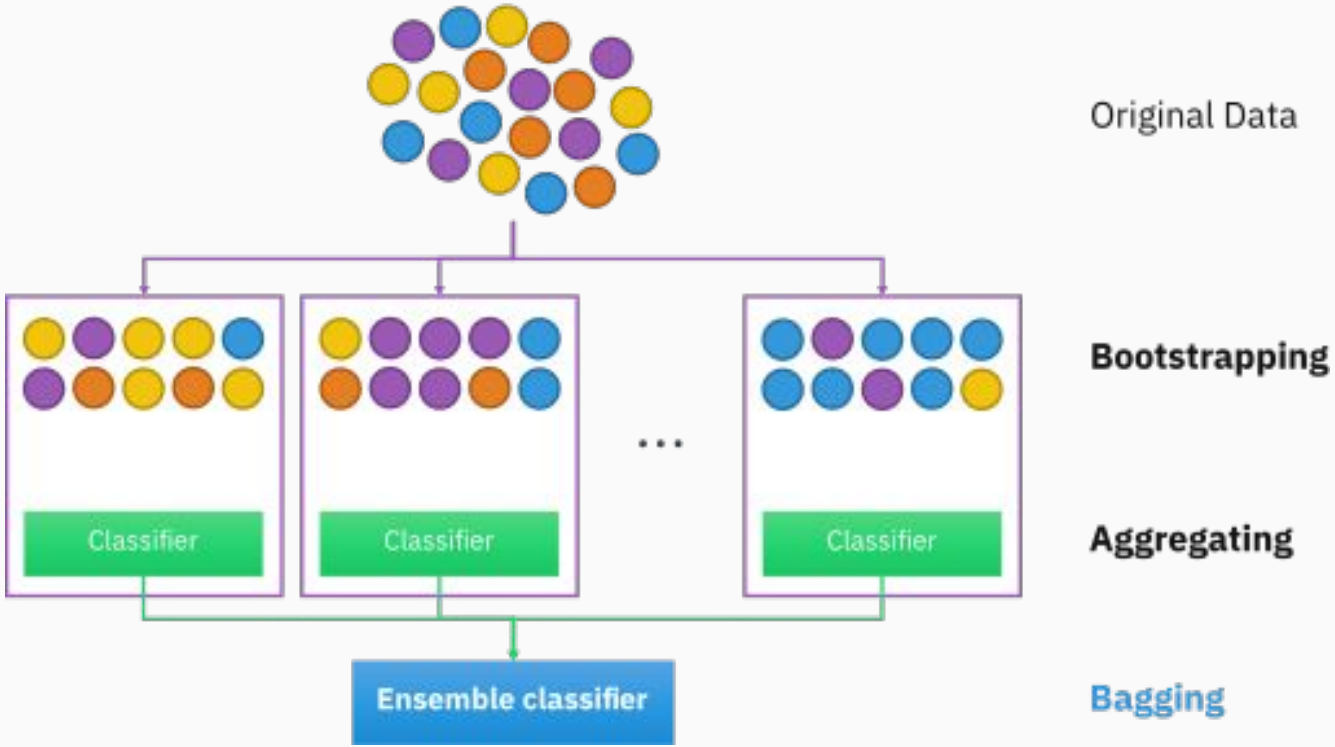


Image source: Wikimedia Commons

Using out-of-bag (OOB) data for validation

- RF trained using random sampling with replacement leaves out approximately 37% of the training data of size N:
  - Probability of not picking a training instance = (N-1)/N
  - Probability of not picking a training instance in N random draws = {(N-1)/N}^N
  - As N gets larger, {(N-1)/N}^N converges to exp(-1) = 0.368
- An OOB sample can be used as a validation dataset for a classification or regression tree not built using that dataset
  - The prediction average produces an unbiased estimate of the test error
- Cross validating large datasets can be costly
  - Validating using OOB is much faster and cheaper

Hands-on exercise

- Use built-in Colab to use Random Forest models to forecast stock price direction and returns

Analyzing the importance of features in RF

- RF models are black boxes and cannot be visualized
  - Feature importance function helps summarize factors driving decisions
  - Many more possible combinations are evaluated, so it's more reliable than DT
- Feature importance in RF models is a weighted average
  - The weight of each node that uses the feature to reduce impurity is equal to training sample size associated with the node
  - Feature importances of each tree in the forest are aggregated and normalized
- Nonlinear relationship between feature and target value still holds
  - High importance value does not indicate a linear correlation with target value
  - Low importance value means that information in the feature is likely redundant

Regularizing RFs with hyperparameters

- The following hyperparameters are key to reducing variance:
  - n_estimators: increase number of trees that populate the forest
  - max_features: reduce number of features included in the random subset
  - n_samples: increase bootstrap training sample size
- Use pre-pruning DT hyperparameters that reduce overfitting of data

Weaknesses of RF models

- RF models are complex, black box models
  - Forests can be dense and trees can be deep
- More memory and time required to train RF models
  - Physical constraints on the complexity of RF models
- Poor performance in sparse, high dimensional datasets
  - For example, text data

Why use Gradient Boosting?

- Gradient boosted regression trees (GBRT), also known as Gradient boosting machines (GBM), are the most powerful nonlinear algorithms
  - Winning entries in most public contests use this algorithm
  - Used for regression and classification problems
- Gradient boosting enhances performance of RF models
  - Uses shallow trees that require less memory
  - Enables faster predictions of learning ensembles

How an adaptive boosting algorithm learns

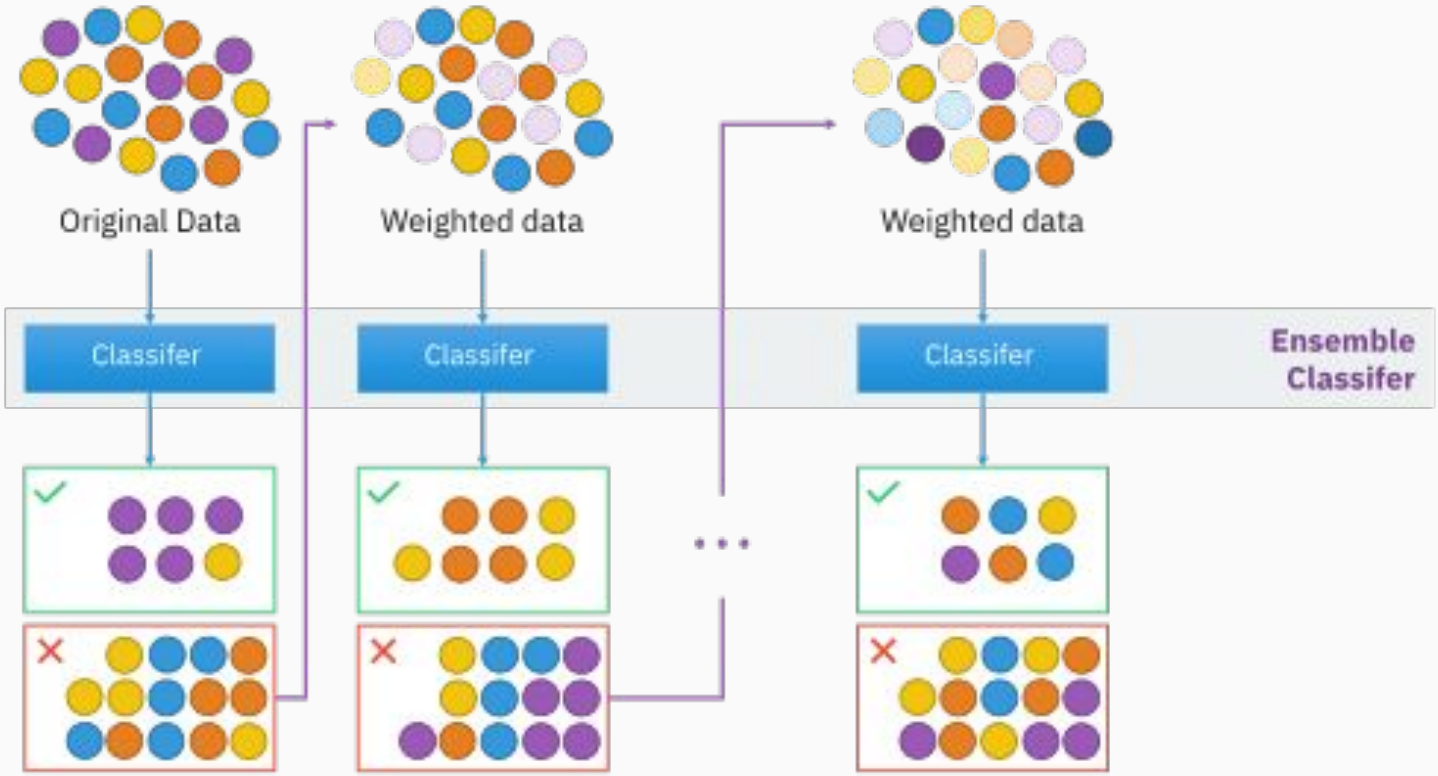


Image source: Wikimedia Commons

How gradient boosting algorithm learns

- Sequentially adds DTs to the ensemble, with each tree trying to correct the errors of the preceding ensemble
  - Unlike the RF algorithm, randomness is generally not used in building the ensemble
- At every iteration, algorithm tries to fit a new DT to the residual error of the previous ensemble
  - Unlike the adaptive boosting algorithm which modifies the weights of the training data
- Learning rate of the model controls the severity of the corrections
  - Higher learning rate leads to more complex GBM models and overfitting

Regularizing GBMs with hyperparameters

- The following hyperparameters are key to reducing variance:
  - n_estimators: reduce the number of trees that populate the ensemble
  - learning_rate: reduce the learning rate
- Use pre-pruning DT hyperparameters that reduce overfitting of data

Hands-on exercise

- Use built-in Colab to use GBM models to forecast stock price returns and probability of recession

Weaknesses of GBM models

- GBM models are complex, black box models
  - GBM ensembles can be dense
- Poor performance in data with a lot of statistical noise
  - For example, financial data
- Sensitive to outliers in the data set
  - Each new DT is required to fix the errors of its preceding ensemble