Road map!

- Module 1- Introduction to Deep Forecasting
- Module 2- Setting up Deep Forecasting Environment
- Module 3- Exponential Smoothing
- Module 4- ARIMA models
- Module 5- Machine Learning for Time series Forecasting
- Module 6- Deep Neural Networks
- Module 7- Deep Sequence Modeling (RNN, LSTM)
- Module 8- Prophet and Neural Prophet

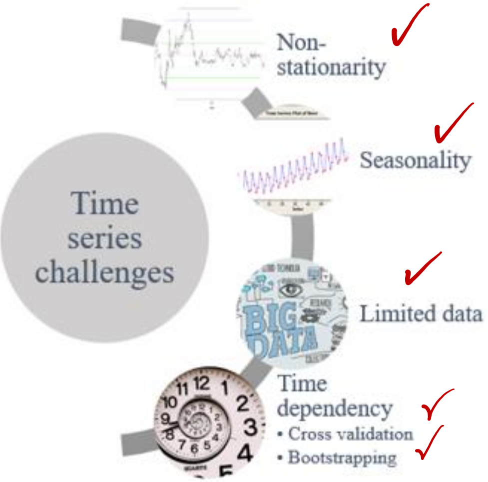
Module 5 – Part I
Machine Learning For timeseries Forecasting

Module 5 – Part II
Machine Learning for timeseries Decision Tree based Models

dmlc XGBoost | CatBoost | LightGBM

Aspect	Decision Tree (DT)	Random Forest (RF)	XGBoost	CatBoost	LightGBM
1. Sampling Process	Naïve	Naïve	Histogram-based	Feature Binning (Quantization)	GOSS
Method					Greedy
wth					Leaf-wise
ng Trees					Boosting

Module 5 – Part III
Challenges in Time Series Machine Learning



Module 5 – Part IV
ML for timeseries in Python

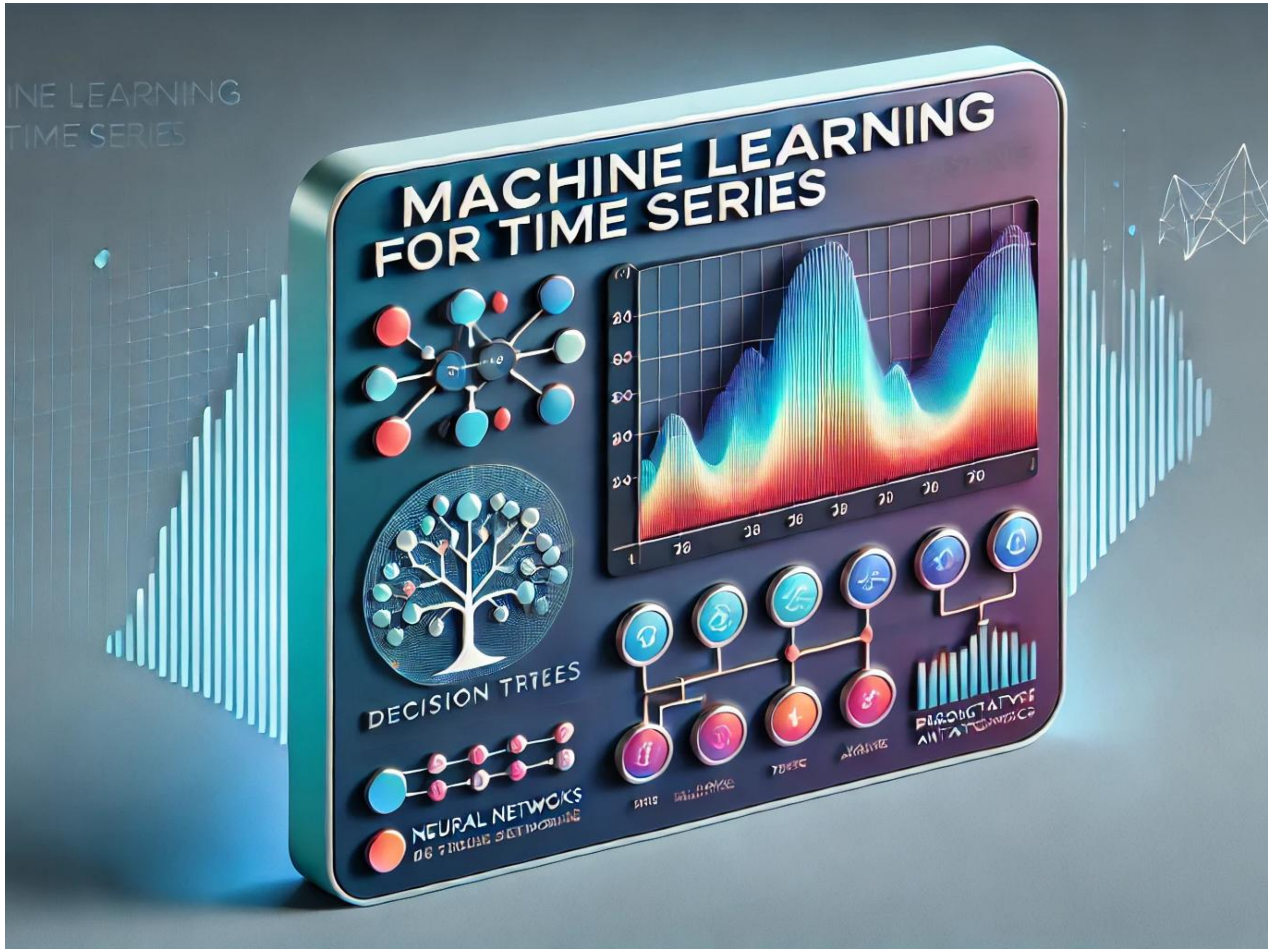
PYCARET

scikit learn

JON M. HUNTSMAN SCHOOL OF BUSINESS
UtahStateUniversity

Pedram Jahangiry

Module 5 – Part I
Machine Learning For timeseries Forecasting

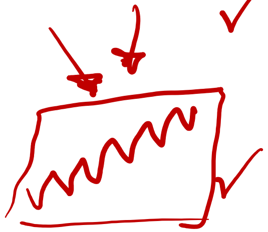


Jon M. Huntsman School of Business
Utah State University

Pedram Jahangiry

Model Comparison

Aspect	Econometric Models	Machine Learning (ML)	Deep Learning (DL)
Feature Engineering	lbs Requires explicit modeling of seasonality and trend ARIMA	Captures complex patterns with less need for manual engineering	Often automates feature engineering
Model Complexity	**Lower**; focuses on data generation process	**Moderate**; can handle non-linear interactions	**High**; suited for complex and high-dimensional data
Interpretability	**High**; interpretable parameters, statistical tests	Moderate; provides feature importance	Low; considered a 'black box' approach
Computational Intensity	Generally lower	Varies; dependent on model complexity and data size	High; requires significant computational resources
Data Suitability	Works well when the underlying process is **well understood**	Effective for structured datasets with **complex relationships**	Ideal for large datasets, including **unstructured** data
Core Models	ARIMA, Exponential Smoothing (ETS)	Random Forest, Gradient Boosting	RNNs, LSTMs, CNNs

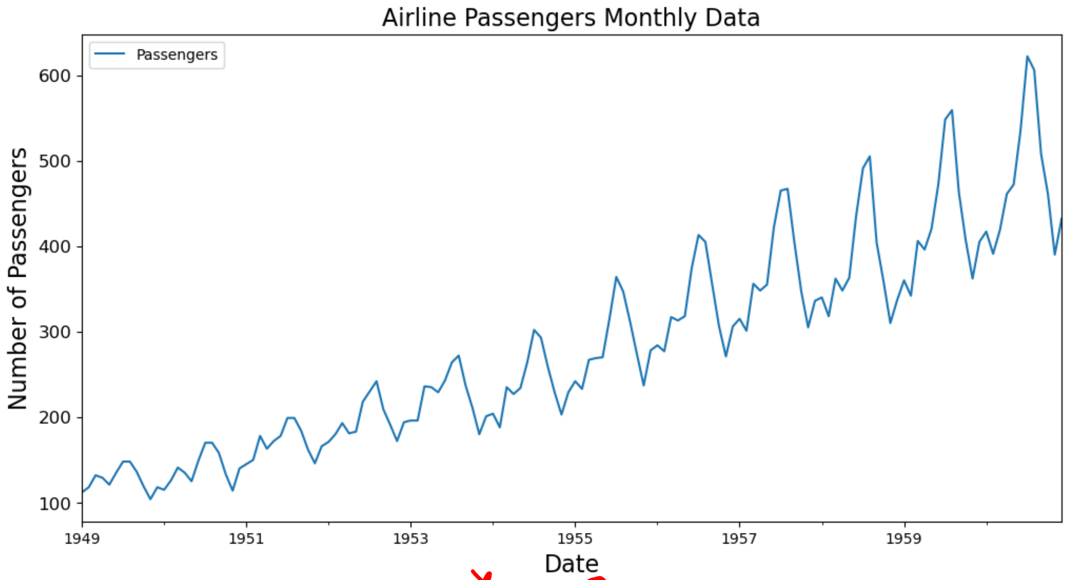


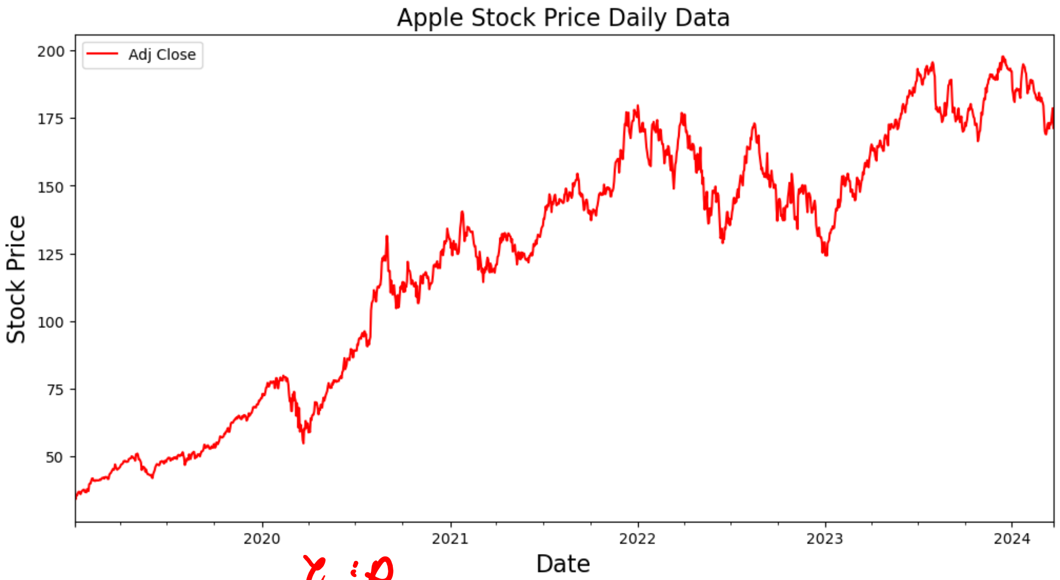
Jon M. Huntsman School of Business
Utah State University

Pedram Jahangiry

Pre-processing the data for machine learning

- How to convert time series problems into machine learning problems?





1949 Q_1 ... 1960 Q_4

Y_t : Q

Y_1, Y_2, ..., Y_{144}

ETS

ARIMA

ML

X_t | Y_t

Y_t : D

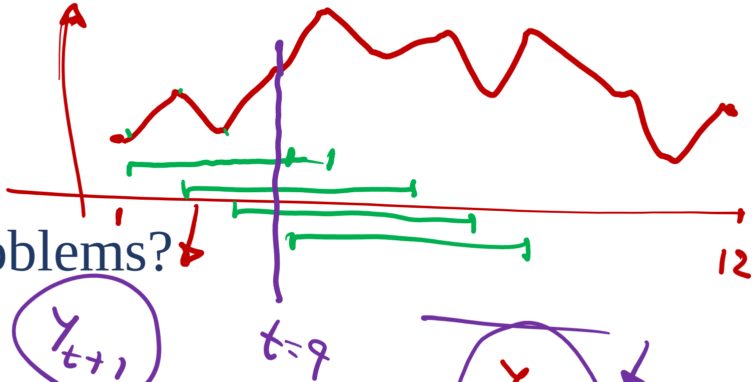
Y_1, ..., Y_{1350}

JON M. HUNTSMAN SCHOOL OF BUSINESS
UtahStateUniversity

Pedram Jahangiry

Preparing data for machine learning

- How to convert time series problems into machine learning problems?



y_{t-3} y_{t-2} y_{t-1} y_t y_{t+1}

8 obs

Features (X)				Target (y)
y_1	y_2	y_3	y_4	y_5
y_2	y_3	y_4	y_5	y_6
y_3	y_4	y_5	y_6	y_7
y_4	y_5	y_6	y_7	y_8
y_5	y_6	y_7	y_8	y_9
y_6	y_7	y_8	y_9	y_{10}
y_7	y_8	y_9	y_{10}	y_{11}
y_8	y_9	y_{10}	y_{11}	y_{12}

Y_1
Y_2
⋮
Y_{12}

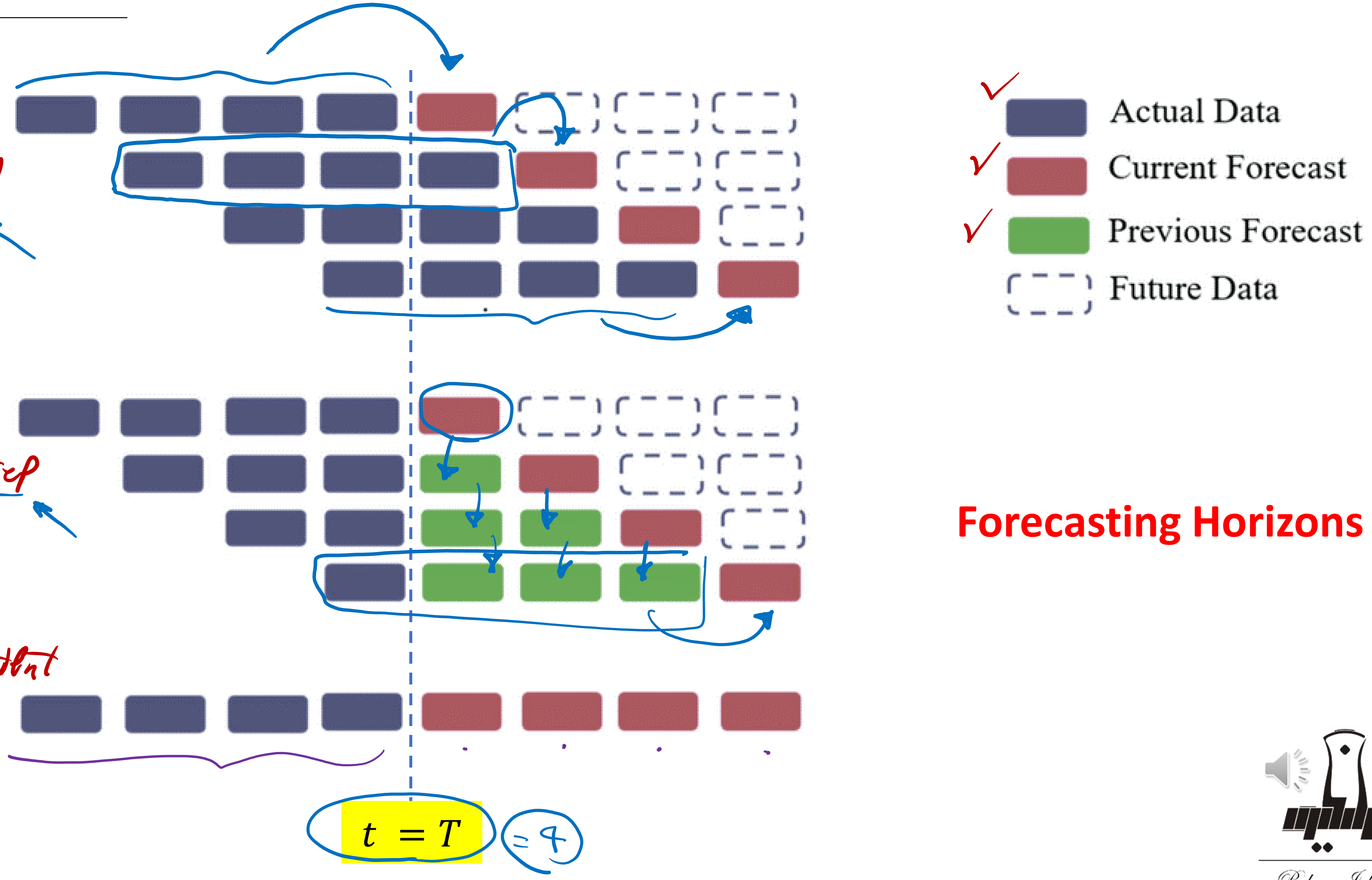
JON M. HUNTSMAN SCHOOL OF BUSINESS
UtahStateUniversity

Pedram Jahangiry

One-step, Multi-step, Multi-output Forecasts

Single output
$\hat{y}_{t+1}$

Multi-output
$\hat{y}_{t+1}$ $\hat{y}_{t+2}$ $\hat{y}_{t+3}$ $\hat{y}_{t+4}$



Forecasting Horizons

JON M. HUNTSMAN SCHOOL OF BUSINESS
UtahStateUniversity
Pedram Jahangiry

Single-Output (one-step ahead)

Y_{t-3} Y_{t-2} Y_{t-1} Y_t Y_{t+1}

Input Features (X)				Single-Output (y)
Lag 3	Lag 2	Lag 1	Lag 0	Lead 1
y_1	y_2	y_3	y_4	y_5
y_2	y_3	y_4	y_5	y_6
y_3	y_4	y_5	y_6	y_7
y_4	y_5	y_6	y_7	y_8
y_5	y_6	y_7	y_8	y_9
y_6	y_7	y_8	y_9	y_{10}
y_7	y_8	y_9	y_{10}	y_{11}
y_8	y_9	y_{10}	y_{11}	y_{12}
y_9	y_{10}	y_{11}	y_{12}	y_{13}
y_{10}	y_{11}	y_{12}	y_{13}	y_{14}

✓ y_t: Actuals

✓ y_t: Predictions

Jon M. Huntsman School of Business
Utah State University

Pedram Jahangiry

Single-Output (multiple-step ahead)

ETS
ARIMA

$\hat{Y}_{t+1}$

Y_{t-3}	Y_{t-2} Input Features (X)	Y_{t-1}	Y_t	Single-Output (y)
Lag 3	Lag 2	Lag 1	Lag 0	Lead 1 $\hat{Y}_{t+1}$
y_1	y_2	y_3	y_4	y_5
y_2	y_3	y_4	y_5	y_6
y_3	y_4	y_5	y_6	y_7
y_4	y_5	y_6	y_7	y_8
y_5	y_6	y_7	y_8	y_9
y_6	y_7	y_8	y_9	y_{10}
y_7	y_8	y_9	y_{10}	y_{11}
y_8	y_9	y_{10}	y_{11}	y_{12}
y_9	y_{10}	y_{11}	y_{12}	y_{13}
y_{10}	y_{11}	y_{12}	y_{13}	y_{14}

y_t: Actuals
y_t: Predictions

Multi-Output

ETS, ARIMA

ETS 1   ETS 2   ETS 3

y_{t-3} y_{t-2} y_{t-1} y_t

Input Features (X)				Multi-Output (Y)		
Lag 3	Lag 2	Lag 1	Lag 0	Lead 1	Lead 2	Lead 3
y_1	y_2	y_3	y_4	y_5	y_6	y_7
y_2	y_3	y_4	y_5	y_6	y_7	y_8
y_3	y_4	y_5	y_6	y_7	y_8	y_9
y_4	y_5	y_6	y_7	y_8	y_9	y_{10}
y_5	y_6	y_7	y_8	y_9	y_{10}	y_{11}
y_6	y_7	y_8	y_9	y_{10}	y_{11}	y_{12}
y_7	y_8	y_9	y_{10}	y_{11}	y_{12}	y_{13}
y_8	y_9	y_{10}	y_{11}	y_{12}	y_{13}	y_{14}
y_9	y_{10}	y_{11}	y_{12}	y_{13}	y_{14}	y_{15}
y_{10}	y_{11}	y_{12}	y_{13}	y_{14}	y_{15}	y_{16}

1

10

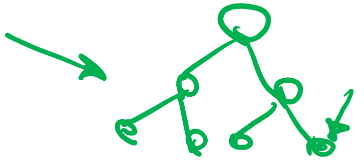
$\hat{y}_{t+1}$ $\hat{y}_{t+2}$ $\hat{y}_{t+3}$

y_t: Actuals

y_t: Predictions

Multi-Output

LR X

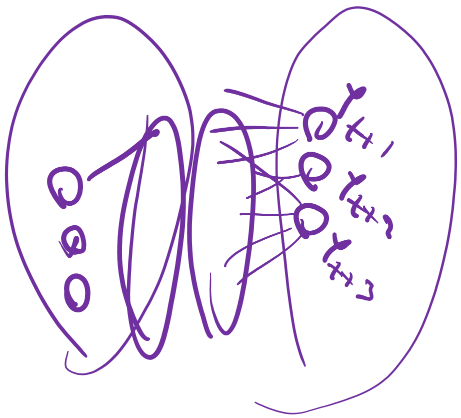


$[\hat{y}_{t+1}, \hat{y}_{t+2}, \hat{y}_{t+3}]$

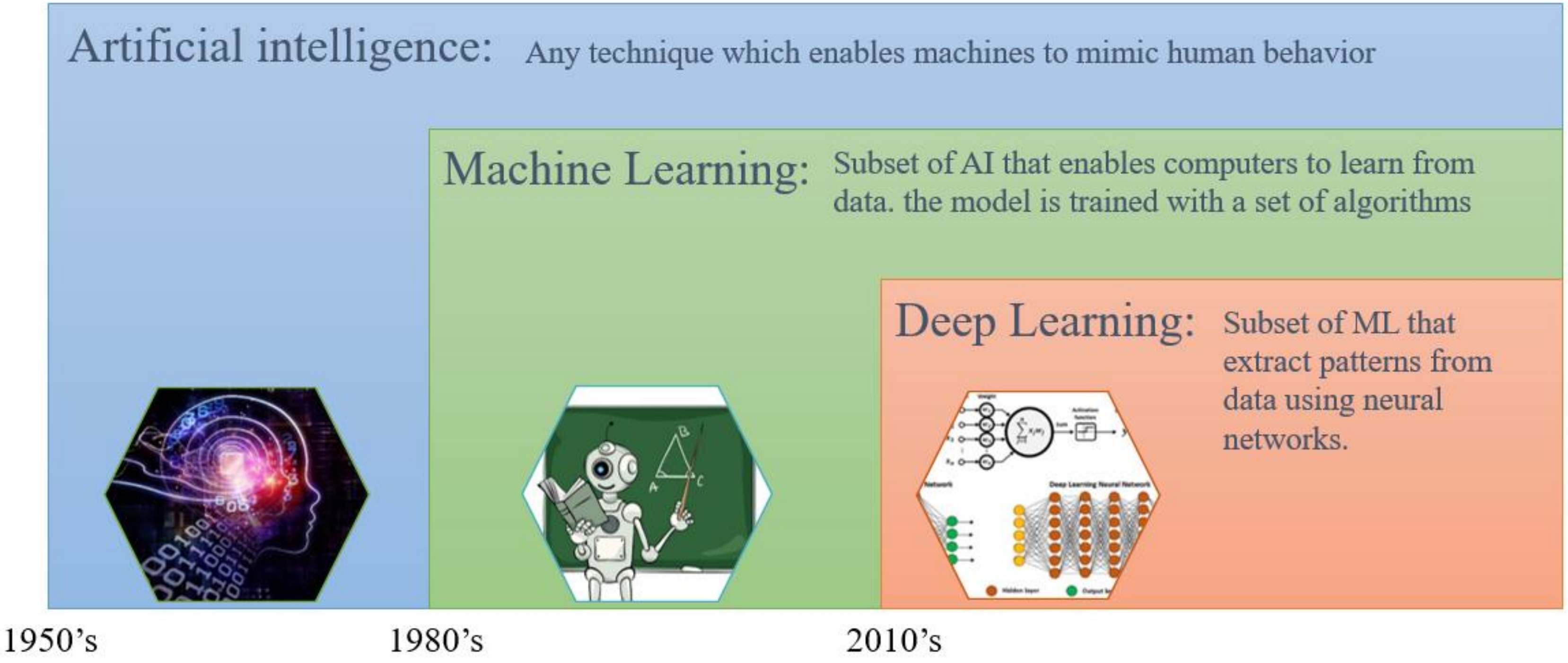
Input Features (X)				Multi-Output (Y)		
Lag 3	Lag 2	Lag 1	Lag 0	Lead 1	Lead 2	Lead 3
y_1	y_2	y_3	y_4	y_5 y_{t+1}	y_6 y_{t+2}	y_7 y_{t+3}
y_2	y_3	y_4	y_5	y_6	y_7	y_8
y_3	y_4	y_5	y_6	y_7	y_8	y_9
y_4	y_5	y_6	y_7	y_8	y_9	y_{10}
y_5	y_6	y_7	y_8	y_9	y_{10}	y_{11}
y_6	y_7	y_8	y_9	y_{10}	y_{11}	y_{12}
y_7	y_8	y_9	y_{10}	y_{11}	y_{12}	y_{13}
y_8	y_9	y_{10}	y_{11}	y_{12}	y_{13}	y_{14}
y_9	y_{10}	y_{11}	y_{12}	y_{13}	y_{14}	y_{15}
y_{10}	y_{11}	y_{12}	y_{13}	y_{14}	y_{15}	y_{16}

y_t: Actuals

y_t: Predictions



Machine Learning Fundamentals (review)



JON M. HUNTSMAN SCHOOL OF BUSINESS
UtahStateUniversity

Pedram Jahangiry

Machine Learning Fundamentals

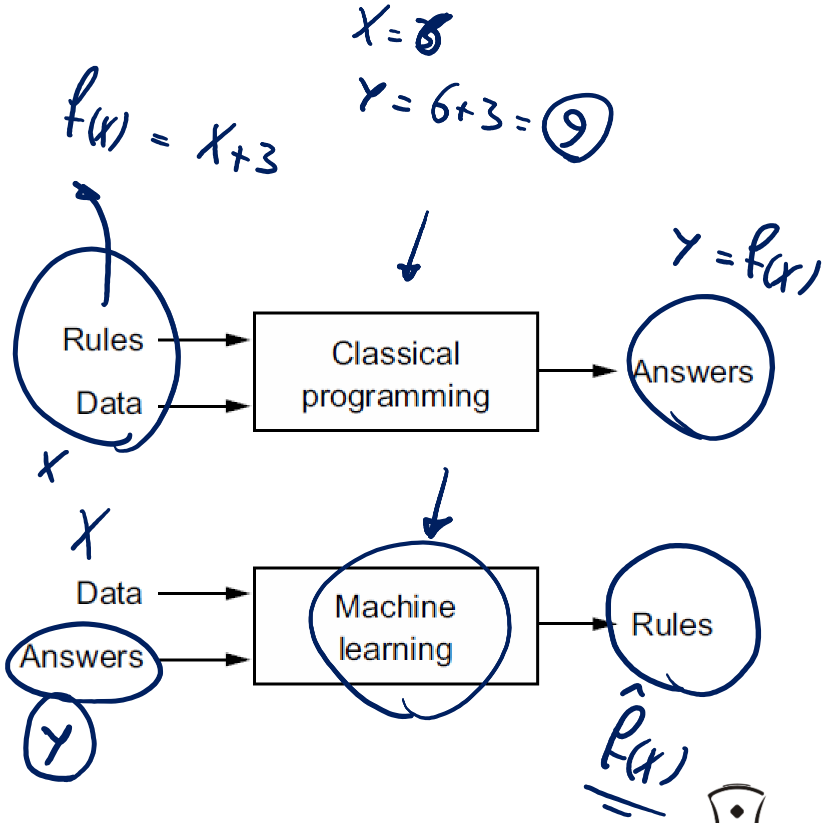
- ML vs traditional programming
- Types of ML
- The Model
- Evaluation metrics RMSE
- Bias-Variance tradeoff, overfitting
- Train, Test, Validation
- Resampling methods
- Cost Function
- Solvers/learners (GD, SGD)
- How do machines actually learn?

Jon M. Huntsman School of Business
Utah State University

Pedram Jahangiry

What is Machine Learning?

- Machine Learning is a subset of AI that enables computers to learn from data.
- A machine learning system is **trained** (with algorithms) rather than explicitly **programmed**.
- ML involves automated detection of meaningful patterns in data and apply the pattern to make predictions on unseen data!
- The goal is to maximize the performance on the unseen data. The purpose is to generalize.



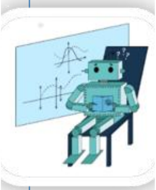
JON M. HUNTSMAN SCHOOL OF BUSINESS
UtahStateUniversity

Pedram Jahangiry

Types of Machine Learning

Supervised

- Regression ✓
- Classification ✓



Unsupervised

- Clustering ✓
- Anomaly detection ✓
- Dimensionality reduction ✓

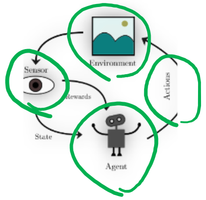
X: 1000s

PC1, PC2

Semi-supervised

meta_labels

Self-supervised



Reinforcement Learning

JON M. HUNTSMAN SCHOOL OF BUSINESS
UtahStateUniversity

Pedram Jahangiry

The Model

$$y = f(X, \theta) + \epsilon = f(X_1, X_2 \ \dots, X_m, \theta_1, \theta_2, \dots, \theta_k) + \epsilon$$

- y : response, dependent variables, output, Target
- X: predictors, independent variables, input, Features
- θ: estimates, specifications, Parameters
  - It is all about estimating f by $\hat{f}$ for two purposes:
    1) Inference (interpretable ML)
    2) Prediction

Jon M. Huntsman School of Business
UtahStateUniversity

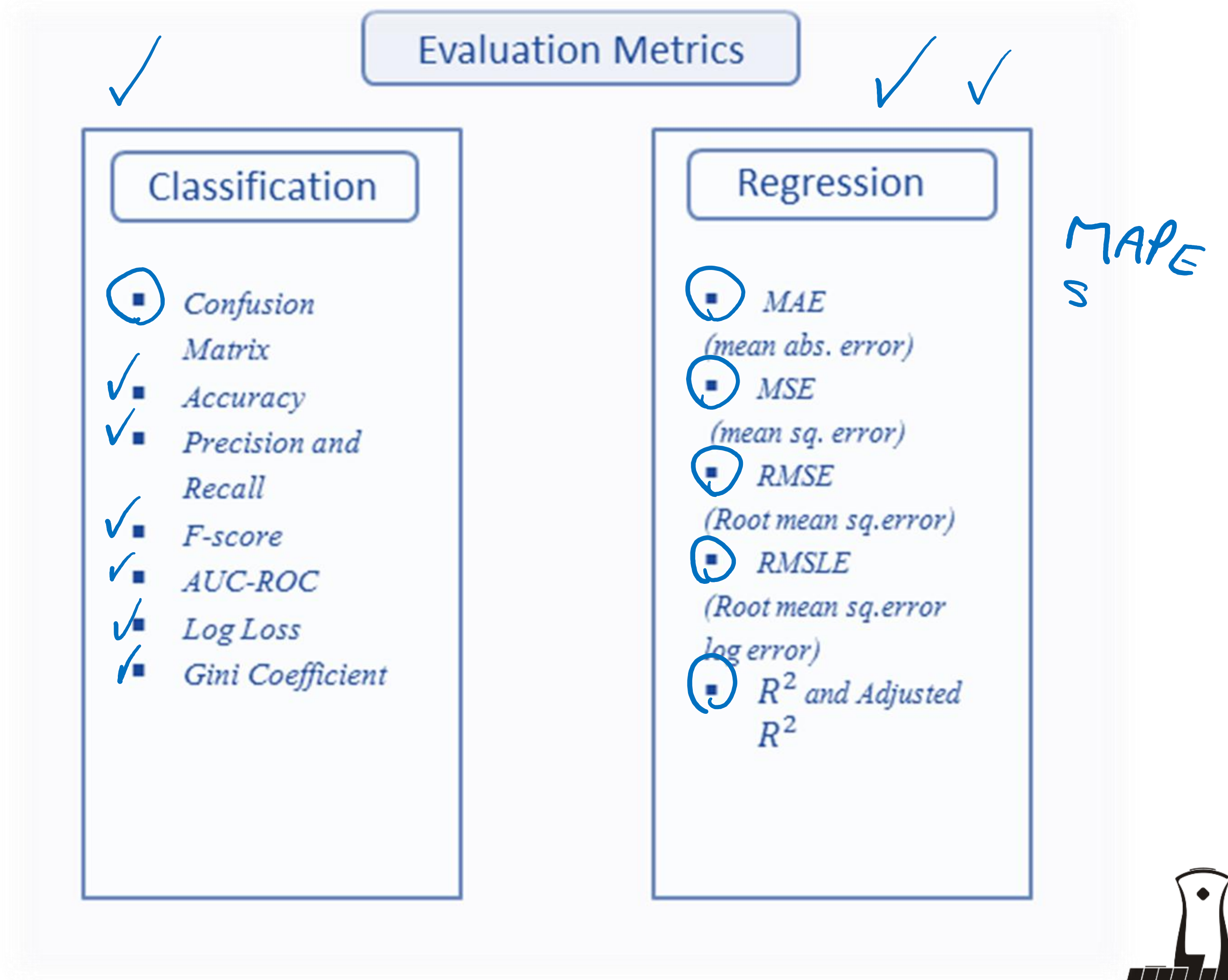
Pedram Jahangiry

Evaluation metrics

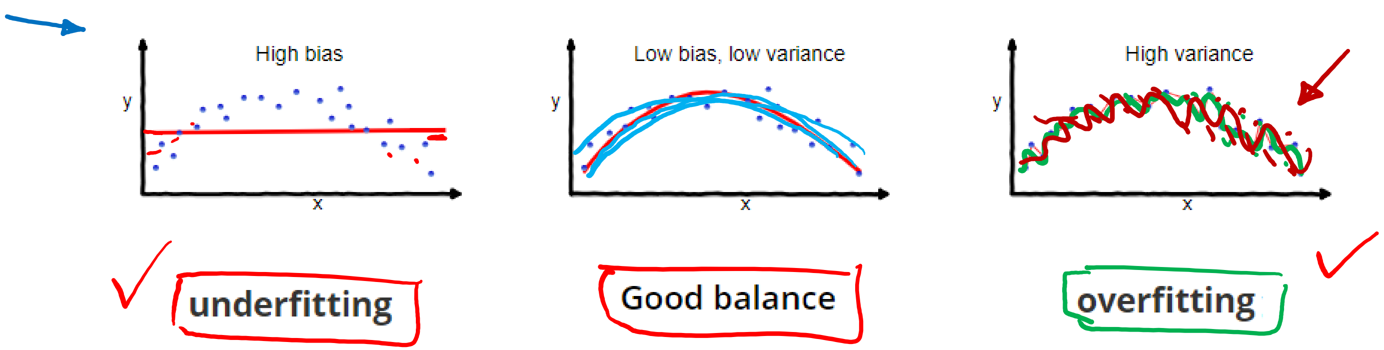
In general, we want to compare how close are the predictions to the actual numbers in the test set.

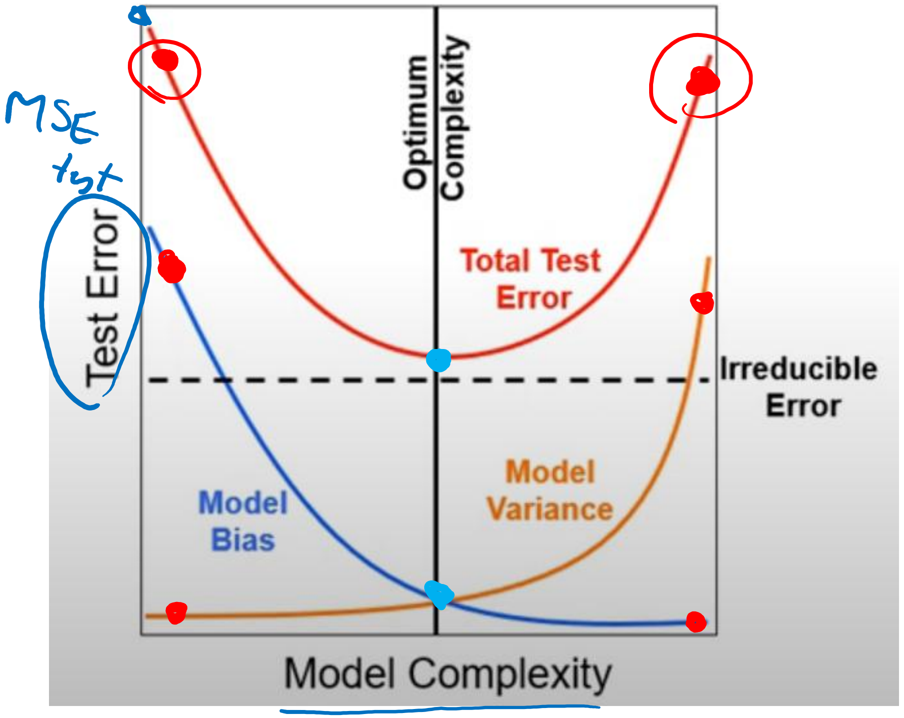
This is typically assessed using

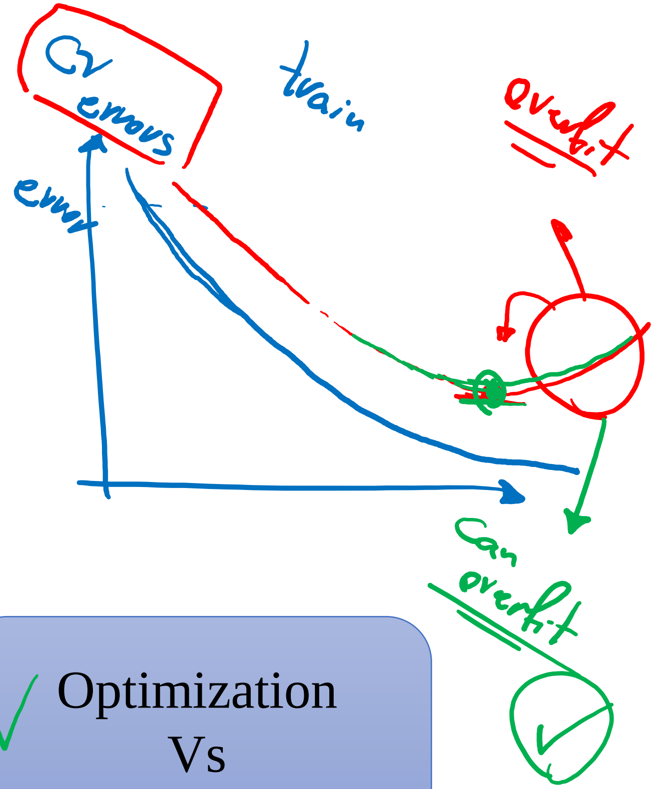
- MSE for quantitative response
- Misclassification rate for qualitative response



Representations of the bias-variance tradeoff

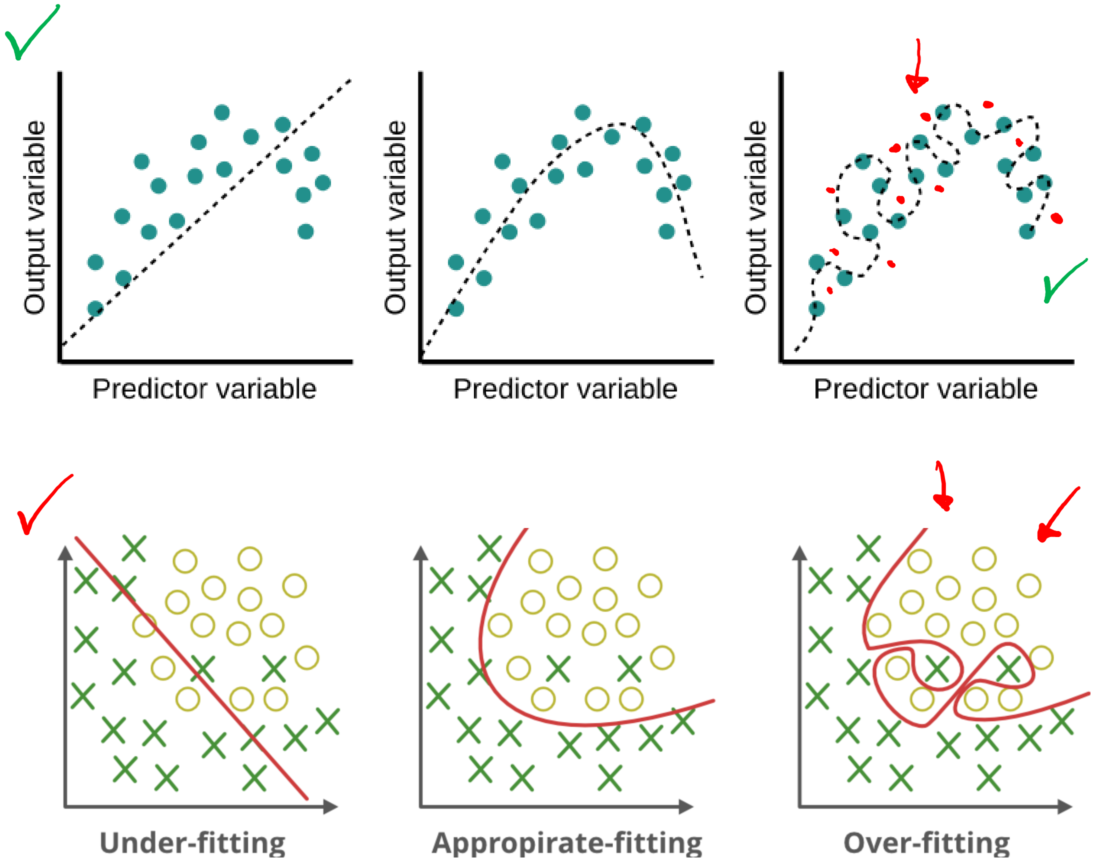






Optimization Vs Generalization

Overfitting



JON M. HUNTSMAN SCHOOL OF BUSINESS
UtahStateUniversity

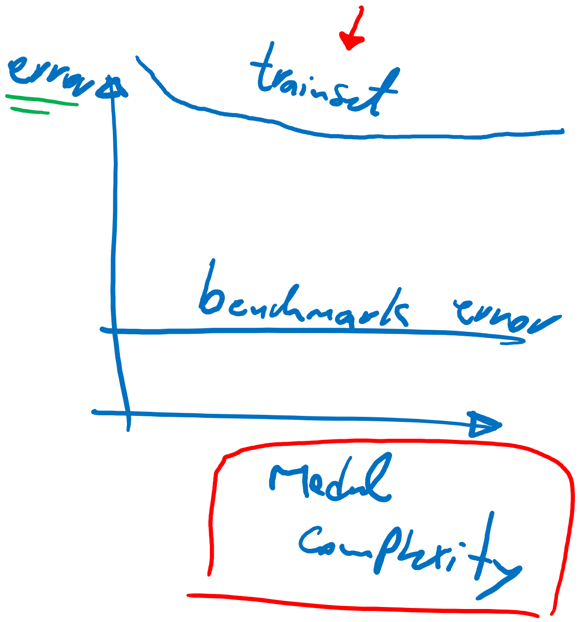
Mitigate overfitting

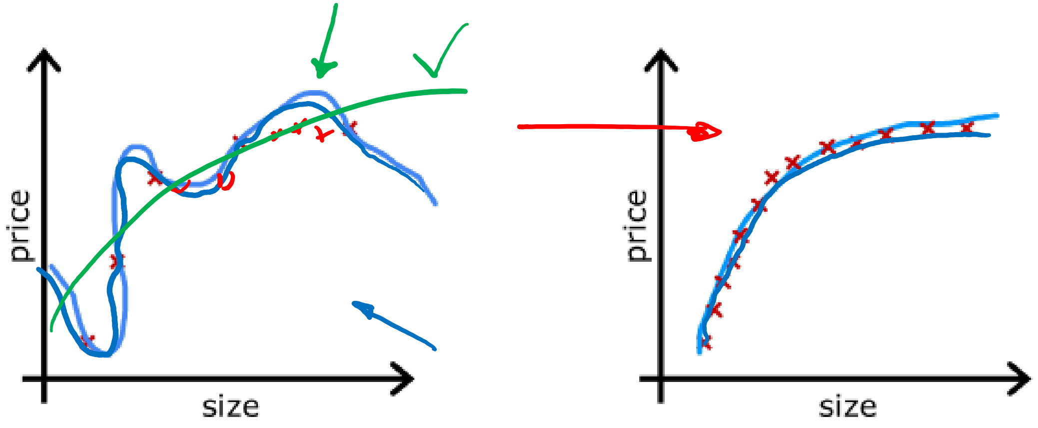
The main techniques used to mitigate overfitting risk in a model construction are:

1) Collect more data (Can reduce bias AND variance)

2) Complexity reduction (regularization, feature selection)

3) Cross validation (estimate the performance in test set)



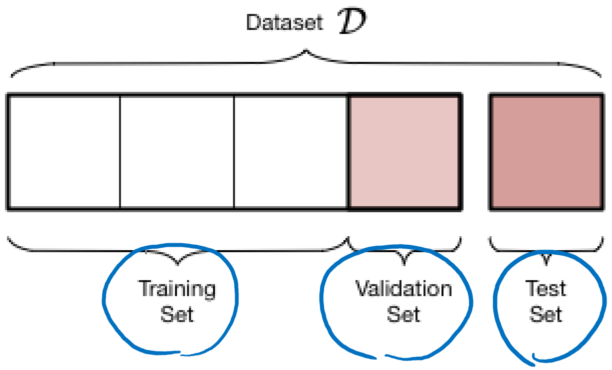


With more training example

Partitioning of the dataset

The data set is typically divided into three non-overlapping samples:

1) Training set: to train the model
2) Validation set: to validate and tune the model
3) Test set: to test the model's ability to predict well on new data (generalize)

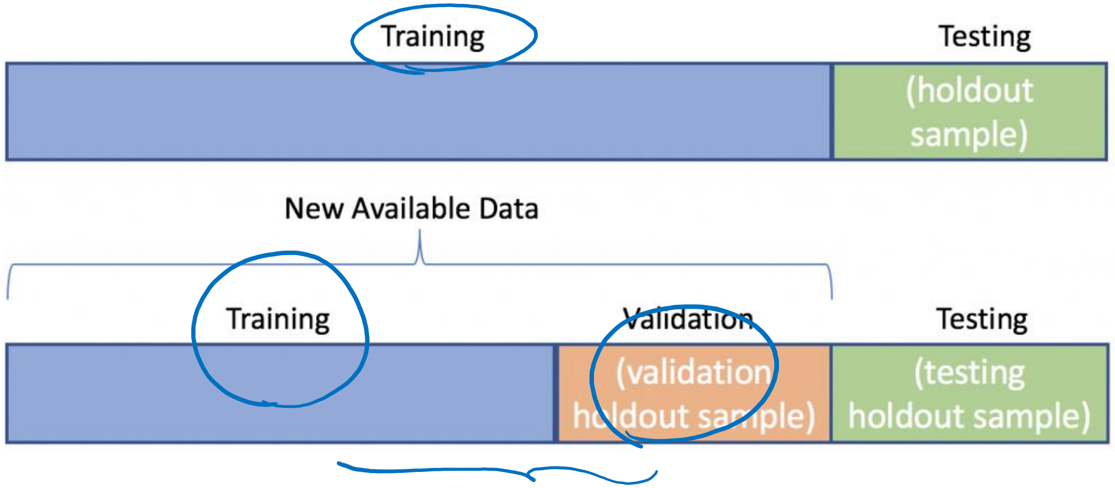


To be valid and useful, any supervised machine learning model must generalize well beyond the training data.

Large dataset is needed! But what if we don't have it?

Resampling methods

- Sometimes we cannot afford to split the data in three because the algorithm may not learn anything from a small training dataset!
- Small validation set is also problematic because we cannot tune the hyperparameters properly! Unstable model performance in validation set!
- Solution: combining the training and validation sets and use cross validation!



K-fold Cross Validation

1) Divide the training data into K roughly equal-sized non-overlapping groups. Leave out k^{th} fold and fit the model to the other $k-1$ folds. Finally, obtain predictions for the left-out k^{th} fold.

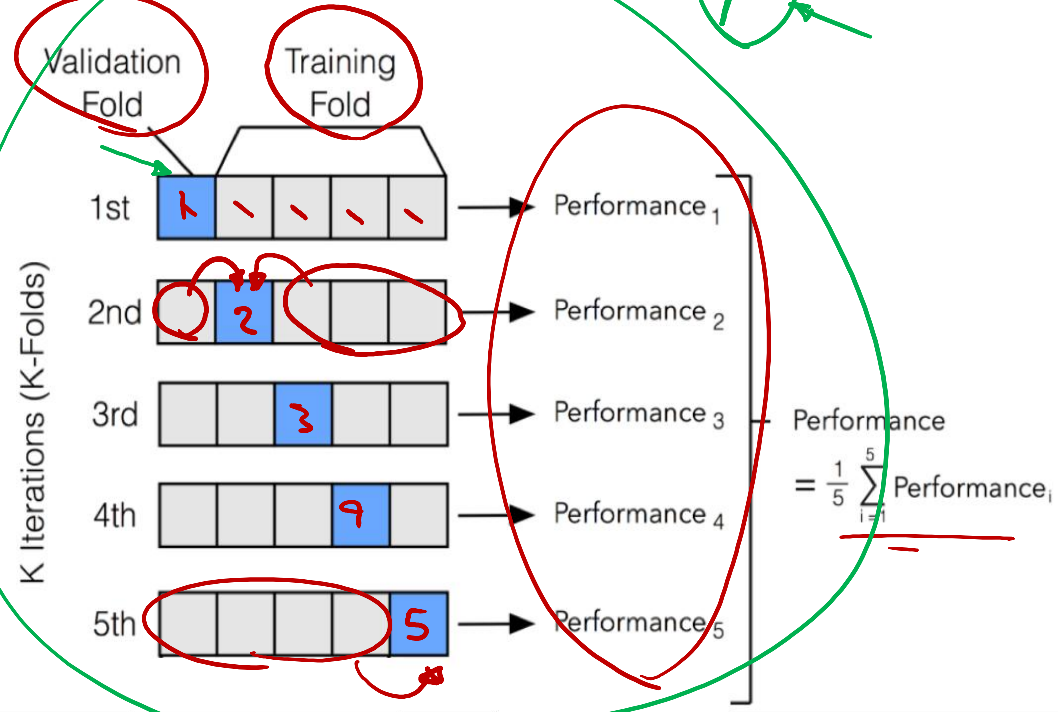
2) This is done in turn for each part k and then the results are combined.

k=5

CV

Var(MSE CV

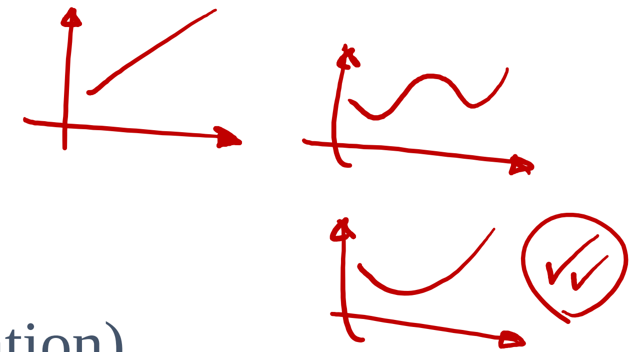
P



JON M. HUNTSMAN SCHOOL OF BUSINESS
UtahStateUniversity

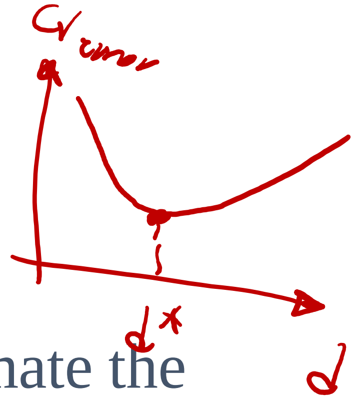
Pedram Jahangiry

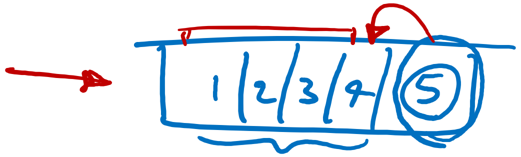
Why do we use Cross Validation?



Cross validation is mainly used for two purposes:

1. Model architecture selection (optimization vs generalization)
2. Estimation of model performance in the test set

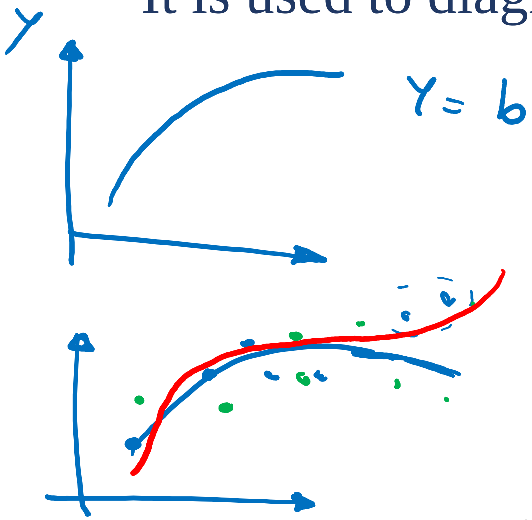




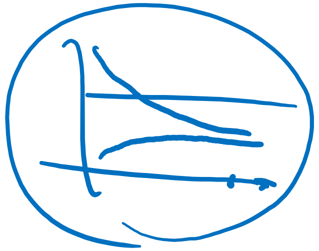
- After selecting the best model architecture, we estimate the generalization error using the test set.
- Different model comparison is based on test set performance!

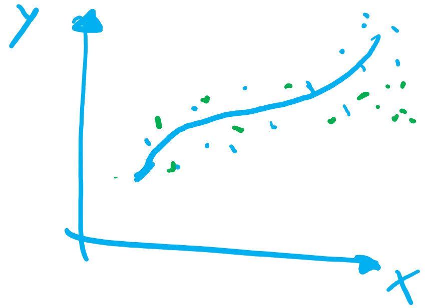
The Learning Curve: Do we need to collect more data?

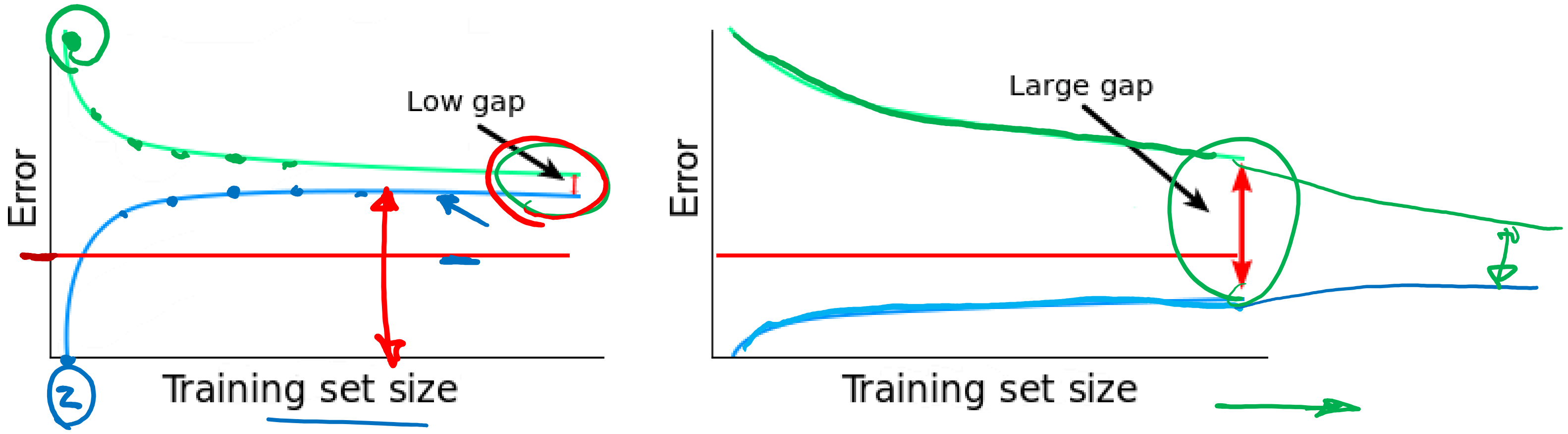
- A learning curve is a plot that shows the relationship between the amount of training data and the performance of a machine learning model.
- It is used to diagnose whether a model has high bias, high variance, or is just right.

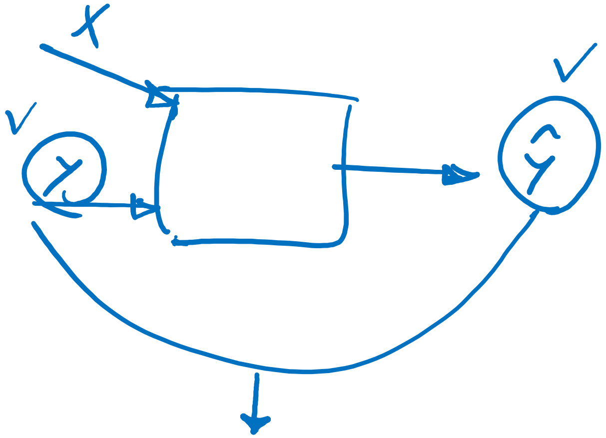


$Y = b + w_1 x_1 + w_2 x_1^2 + 4$









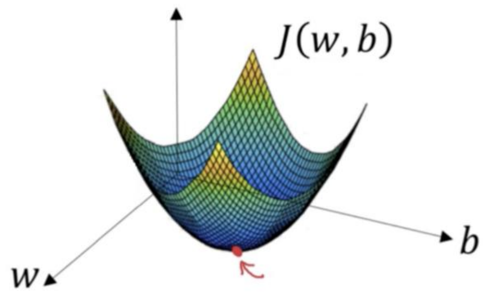
How do Machines Learn?

Jon M. Huntsman School of Business
UtahStateUniversity

Pedram Jahangiry

Terminology

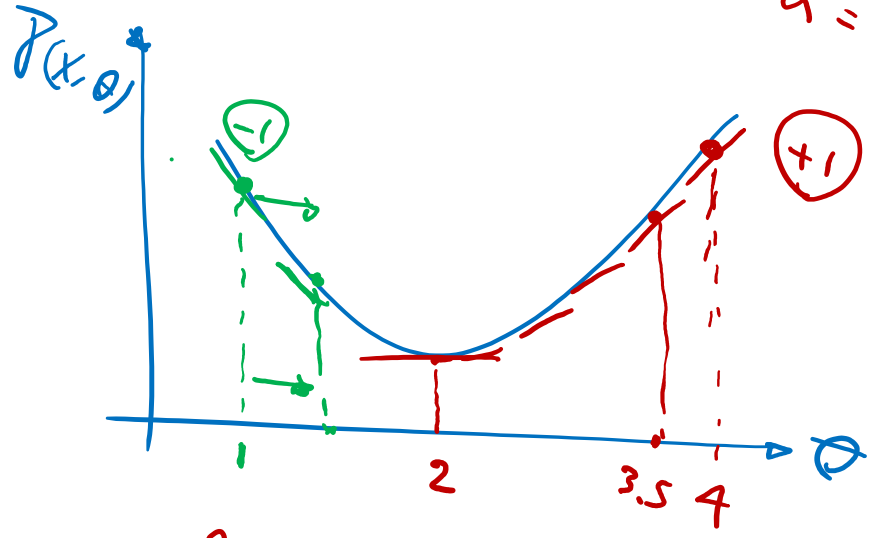
- Learning: Finding the model weights (parameters' values)
- Cost Function: Tells us "how good" our model is at making predictions for a given set of parameters.
- The cost function has its own curve and its own gradients. The slope of this curve tells us how to update our parameters to make the model more accurate.
- The two most frequently used optimization algorithms when the cost function is continuous and differentiable are Gradient Descent (GD) and Stochastic GD.



Solvers (learners)!

- Gradient Descent: is an iterative optimization algorithm for finding the minimum of a function.
- We starts at some random point and take steps proportional to the negative of the gradient of the function at the current point.

$\alpha = 0.5$



0.5   +1

$$\theta_j := \underline{\theta_j} - \underbrace{\alpha \frac{\partial}{\partial \theta_j} J(\theta)}$$

4    $-0.5 = 3.5$

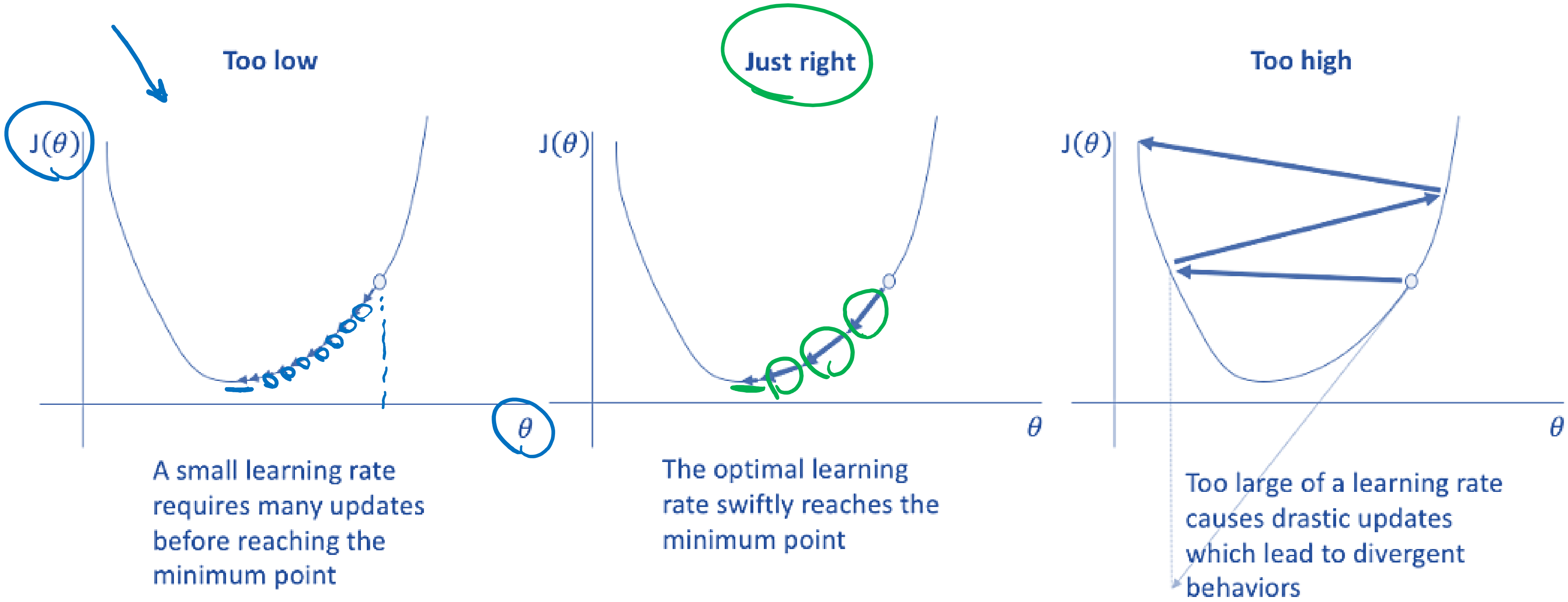
$\theta = 2 - 0.5(0) = 2$

$\theta = 1 - (0.5)(-1) = 1.5$

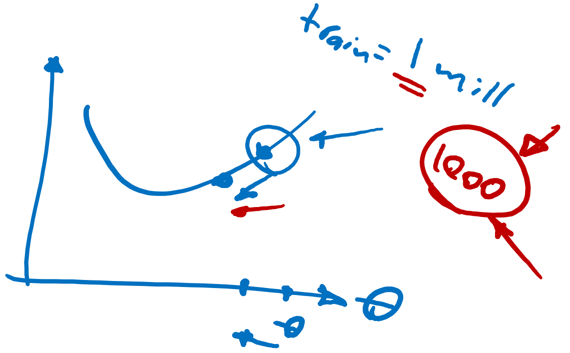
- θ_j is the model's j^{th} parameter
- α is the learning rate
- $J(\theta)$ is the cost function (which is differentiable)

Choice of learning rate

- If α is too small, gradient descent can be slow
- If α is too large, the gradient descent can even **diverge**.



Beyond Gradient Descent?



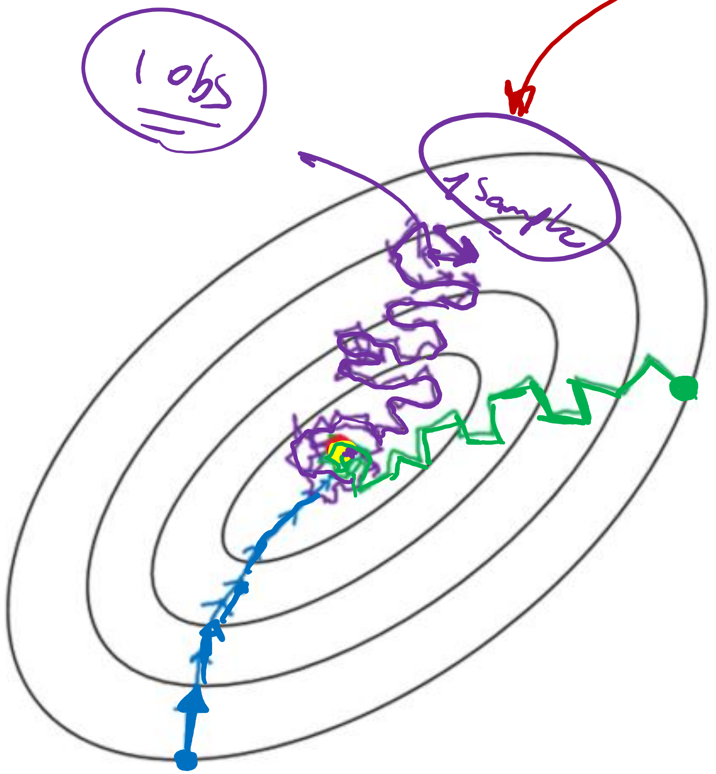
Disadvantages of gradient descent:

- Single batch: use the entire training set to update parameters!
- Sensitive to the choice of the learning rate
- Slow for large datasets

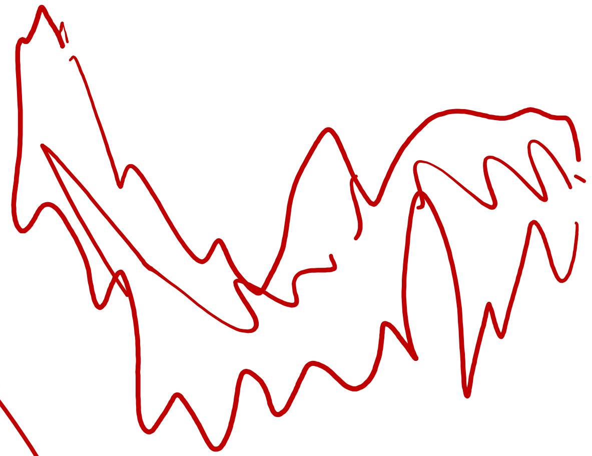
(Minibatch) Stochastic Gradient Descent: is a version of the algorithm that speeds up the computation by approximating the gradient using smaller batches (subsets) of the training data. SGD itself has various “upgrades”.

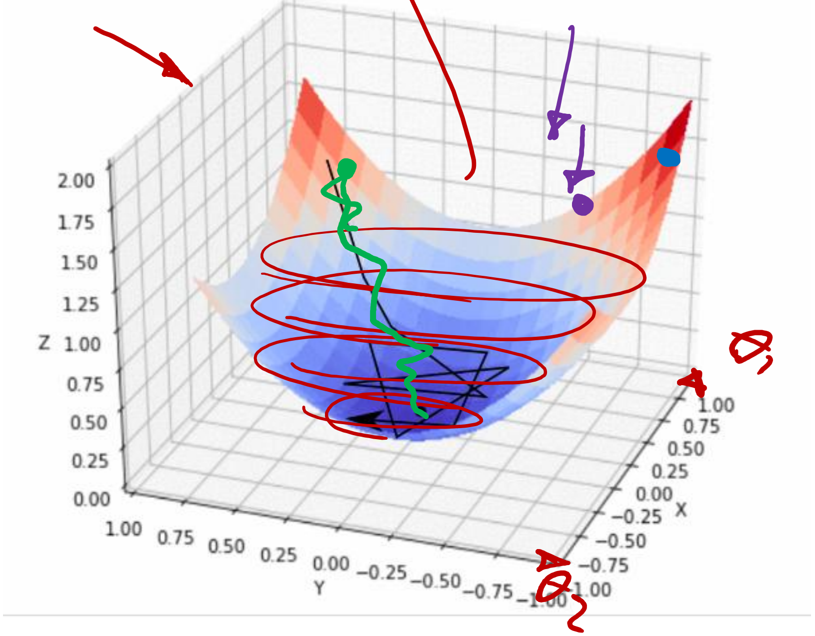
b = 1

SGD vs GD

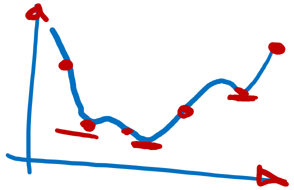


Batch gradient descent
Mini-batch gradient Descent
Stochastic gradient descent

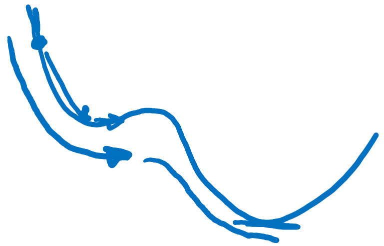


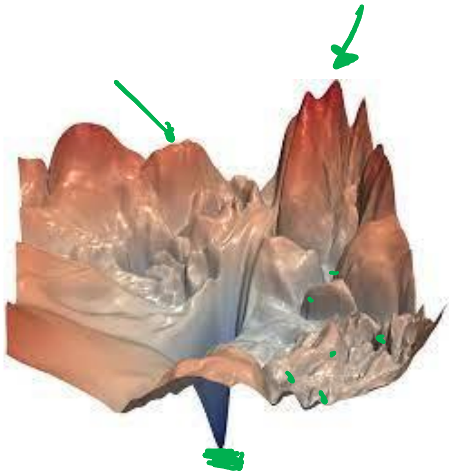


Beyond SGD?



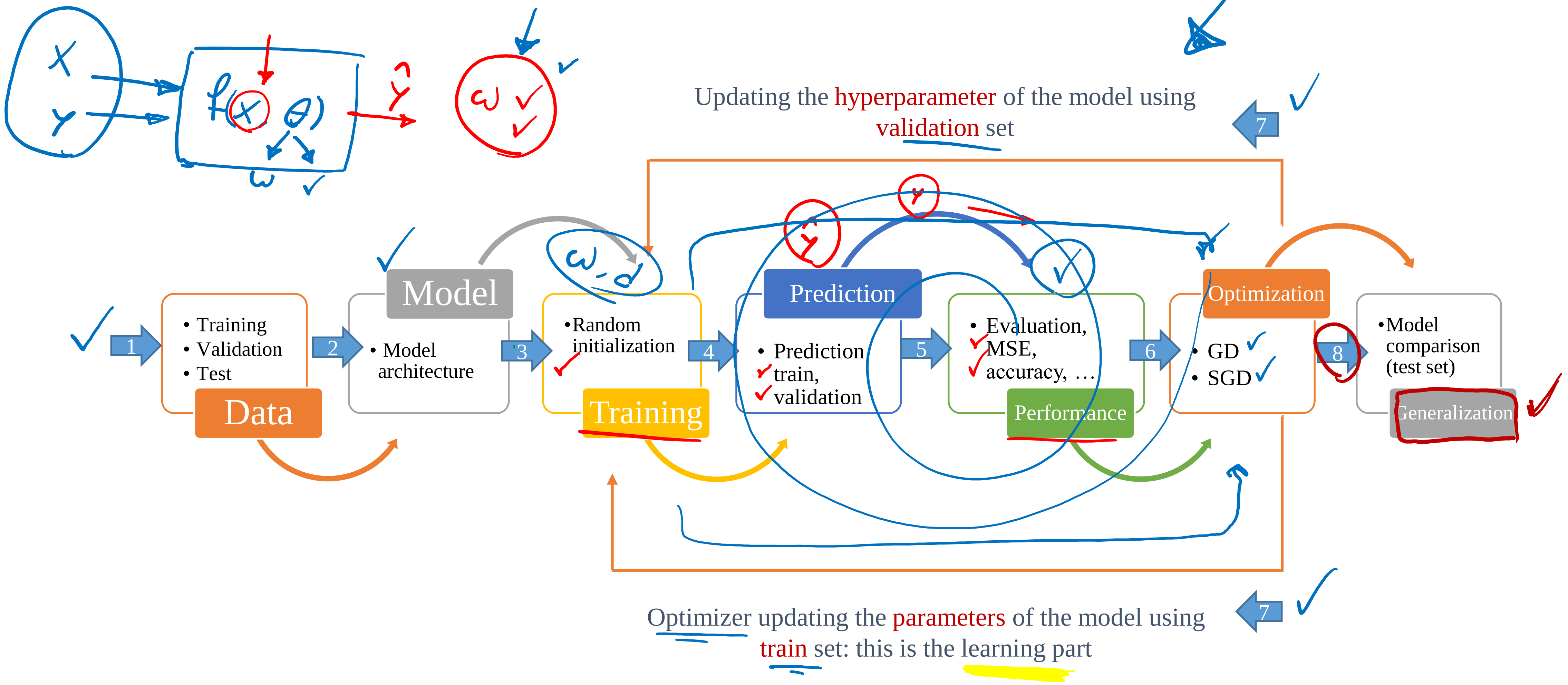
- Loss functions can be difficult to optimize!
- *Visualizing the loss landscape of neural nets*, Li et all, 2018





- Solution: Designing an adaptive learning rate that can adapt to the loss landscape.
- Rather than just looking at the current gradient, consider the previous weight updates.
- This is called, momentum!
- Examples: Adam, Adadelta, Adagrad, RMSProp!

How do machines actually learn?



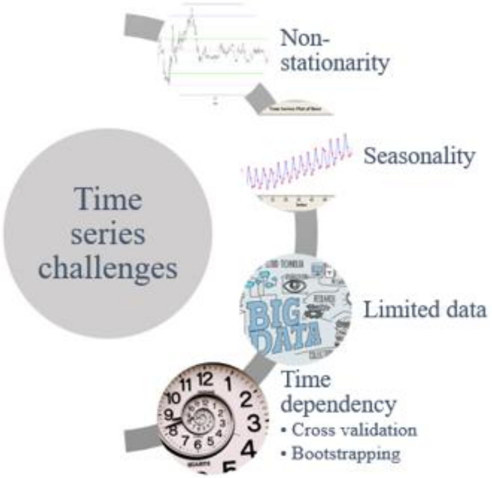
Module 5 – Part I
Machine Learning For timeseries Forecasting

Module 5 – Part II
Machine Learning for timeseries
Decision Tree based Models

dmlc XGBoost CatBoost LightGBM

Aspect	Decision Tree (DT)	Random Forest (RF)	XGBoost	CatBoost	LightGBM
1. Sampling Process	Naïve	Naïve	Histogram-based	Feature Binning (Quantization)	GOSS
Method					Greedy
wth					Leaf-wise
ng Trees					Boosting

Module 5 – Part III
Challenges in Time Series Machine Learning



Module 5 – Part IV
ML for timeseries in Python

scikit learn

PYCARET

Jon M. Huntsman School of Business
UtahStateUniversity

Pedram Jahangiry

Road map!

- ✓ Module 1- Introduction to Deep Forecasting
- ✓ Module 2- Setting up Deep Forecasting Environment
- ✓ Module 3- Exponential Smoothing
- ✓ Module 4- ARIMA models
- ✓ Module 5- Machine Learning for Time series Forecasting
- Module 6- Deep Neural Networks
- Module 7- Deep Sequence Modeling (RNN, LSTM)
- Module 8- Prophet and Neural Prophet

Road map!

- Module 1- Introduction to Deep Forecasting
- Module 2- Setting up Deep Forecasting Environment
- Module 3- Exponential Smoothing
- Module 4- ARIMA models
- Module 5- Machine Learning for Time series Forecasting
- Module 6- Deep Neural Networks
- Module 7- Deep Sequence Modeling (RNN, LSTM)
- Module 8- Prophet and Neural Prophet

✓ Module 5 – Part I
Machine Learning For timeseries Forecasting

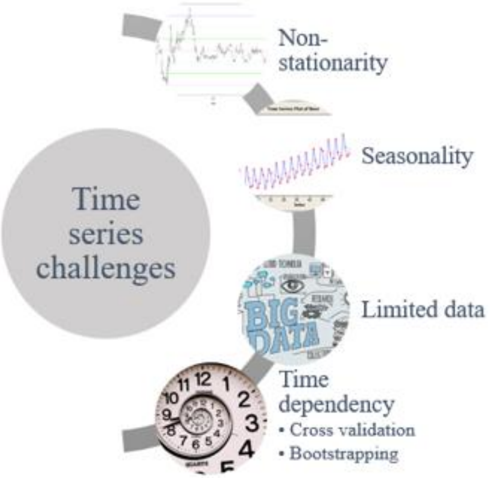
✓ Module 5 – Part II
Machine Learning for timeseries
Decision Tree based Models

dmlc XGBoost ✓ CatBoost ✓ LightGBM ✓

Aspect	Decision Tree (DT) ✓	Random Forest (RF) ✓	XGBoost	CatBoost	LightGBM
1. Sampling Process	Naïve	Naïve	Histogram-based	Feature Binning (Quantization)	GOSS
Method					Greedy
wth					Leaf-wise
ng Trees					Boosting

Module 5 – Part III
Challenges in Time Series Machine Learning

Module 5 – Part IV
ML for timeseries in Python



JON M. HUNTSMAN SCHOOL OF BUSINESS
UtahStateUniversity

Pedram Jahangiry

Module 5 – Part II
Machine Learning for timeseries
Decision Tree based Models

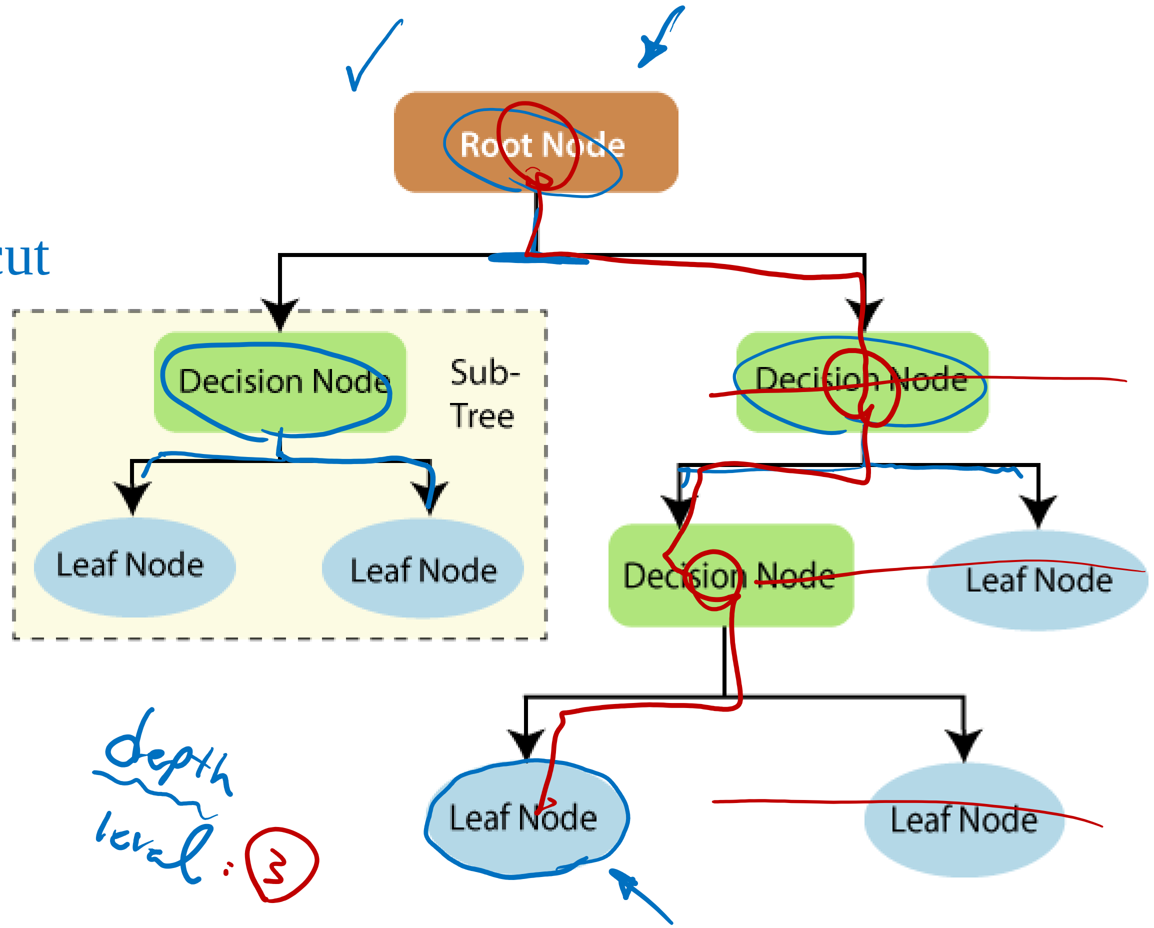
Aspect	Decision Tree (DT)	Random Forest (RF)	XGBoost	CatBoost	LightGBM
1. Sampling Process	Naïve	Naïve	Histogram-based	Feature Binning (Quantization)	GOSS
2. Splitting Method	Greedy	Greedy	Greedy	Greedy	Greedy
3. Tree Growth Strategy	Depth-wise	Depth-wise	Depth-wise (Leaf-wise)	Symmetric	Leaf-wise
4. Combining Trees	Not applicable	Bagging	Boosting	Boosting	Boosting

Decision Trees Fundamental questions

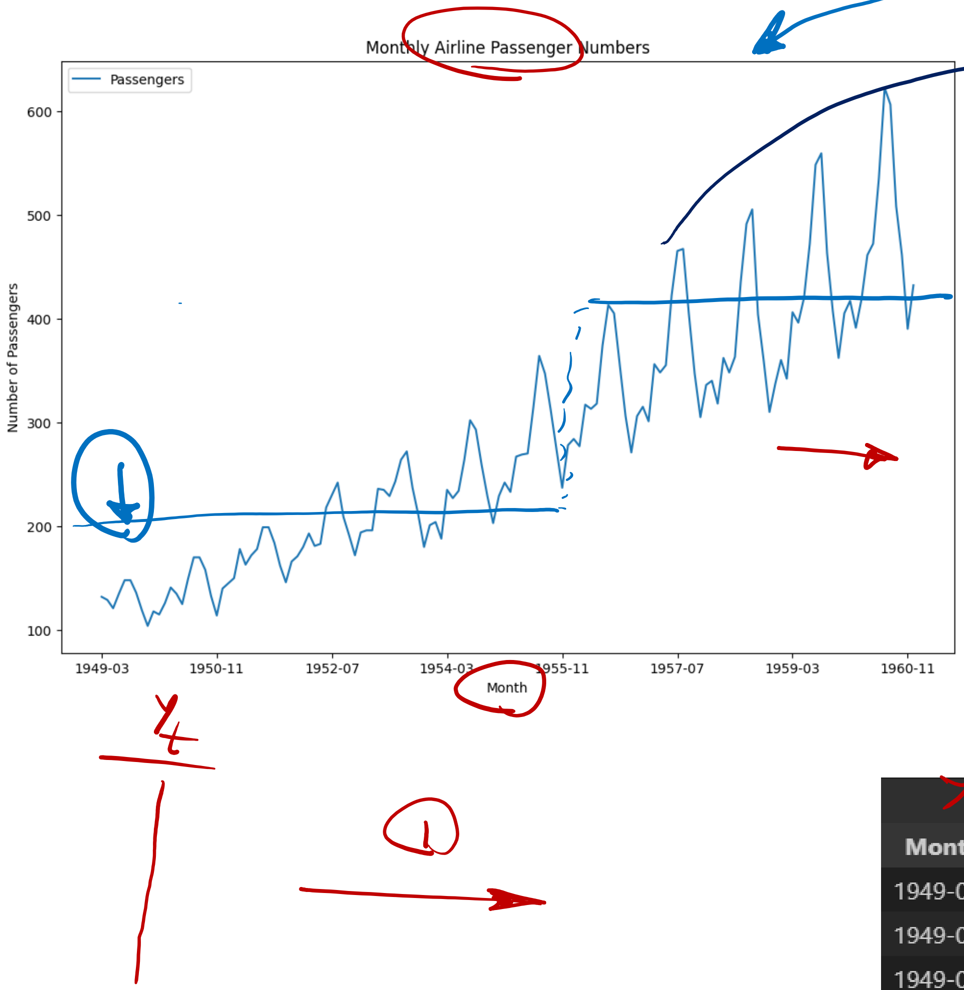
- Four fundamental questions to be answered:

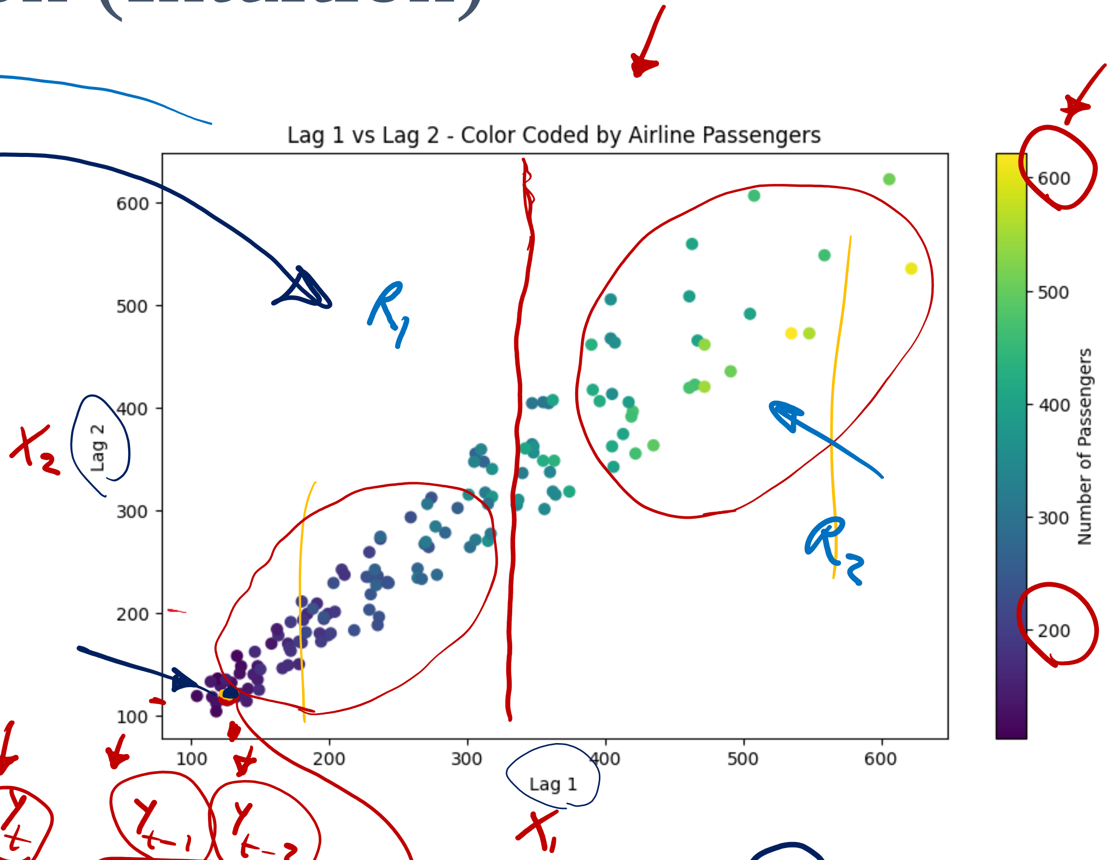
1) How to sample the data for splitting?
2) How to split the samples, What feature and cut off to start with?
3) How to grow a tree?
4) How to combine trees?

DT, RF, Xgboost



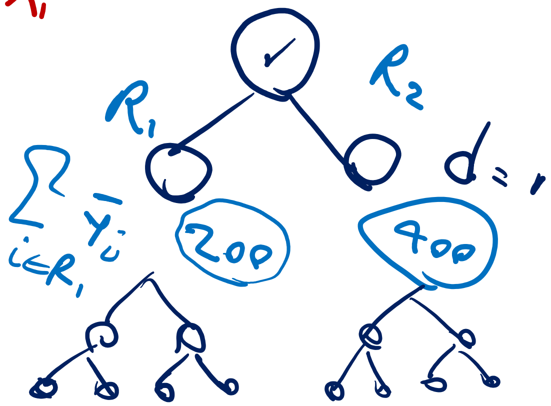
Decision Tree TS regression (Intuition)



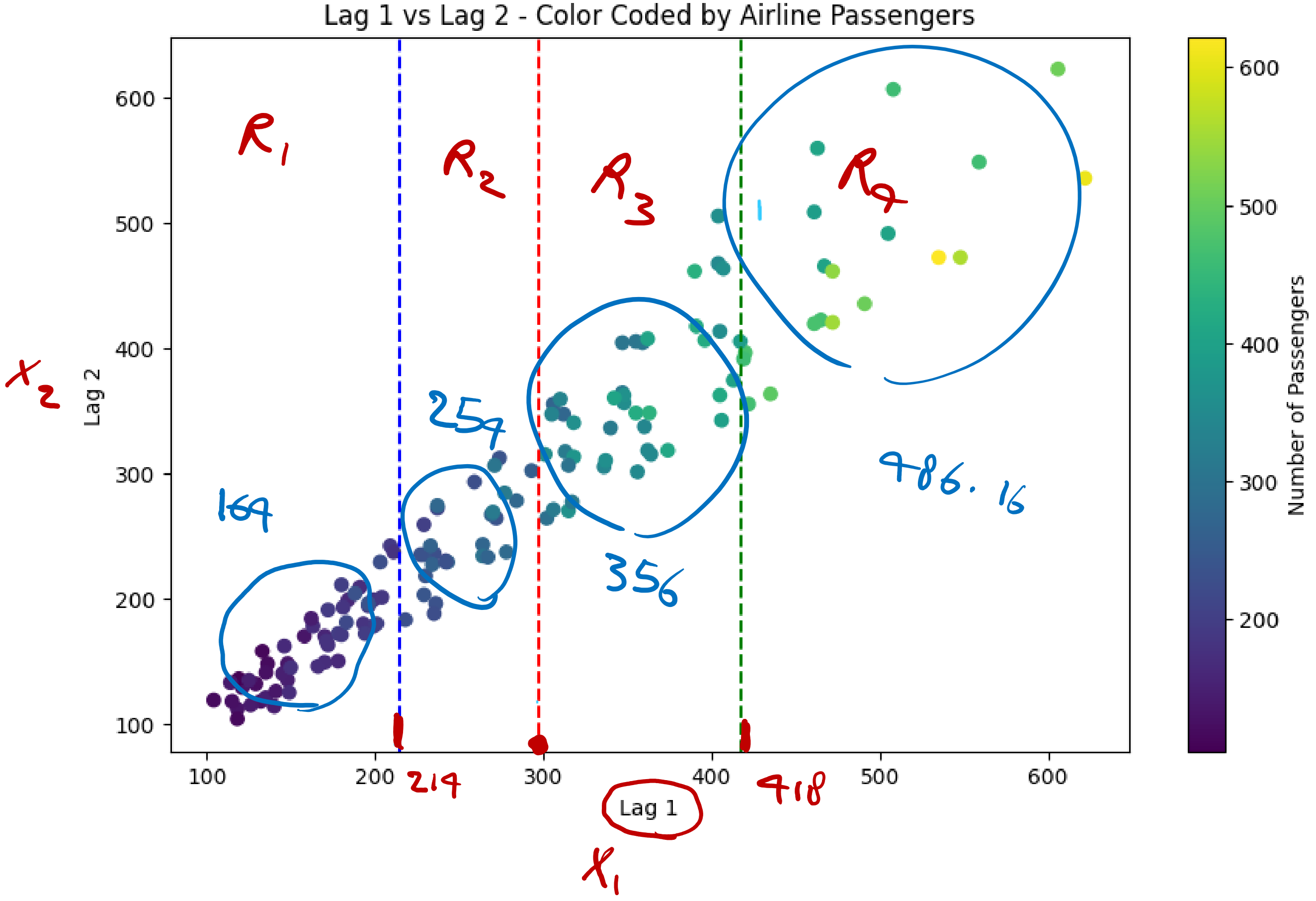


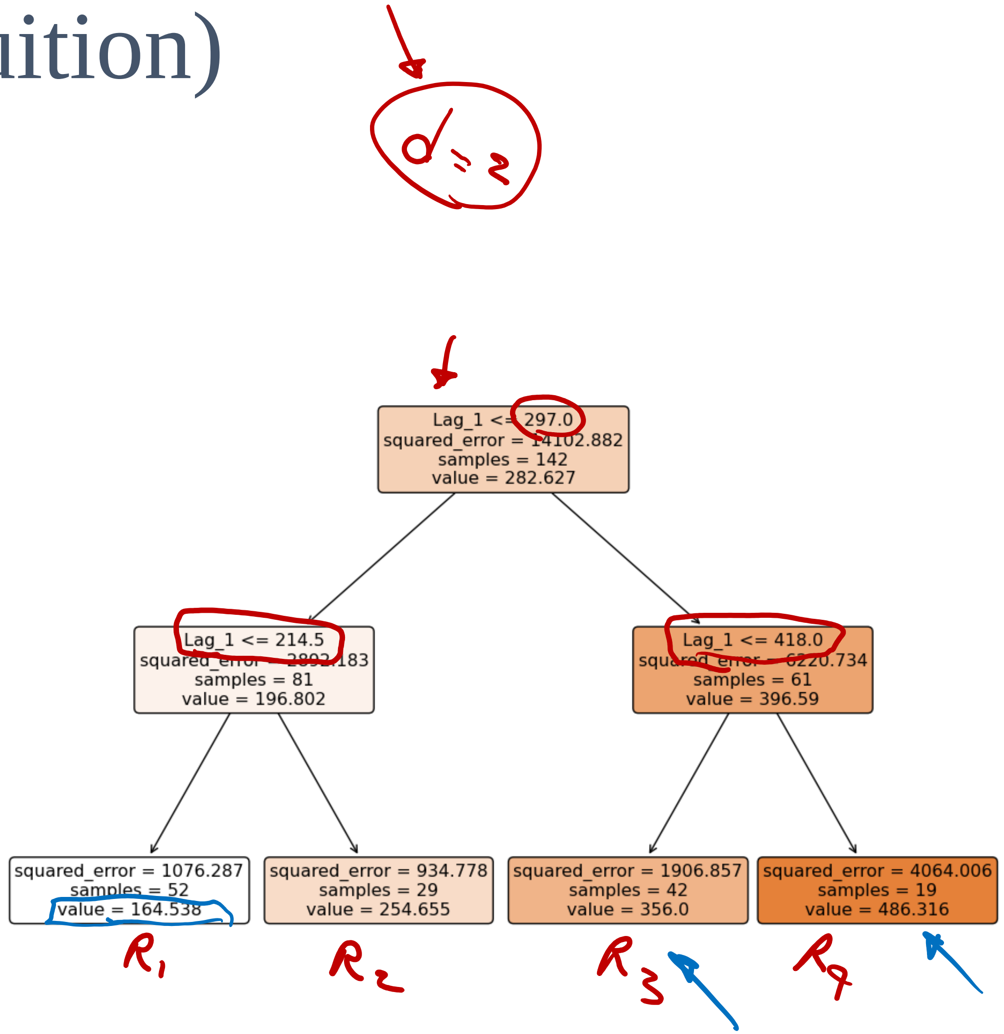
Y_t Y_{t-1} Y_{t-2}

Month	Passengers	Lag_1	Lag_2
1949-01	112	NaN	NaN
1949-02	118	112.0	NaN
1949-03	132	118.0	112.0
1949-04	129	132.0	118.0
1949-05	121	129.0	132.0

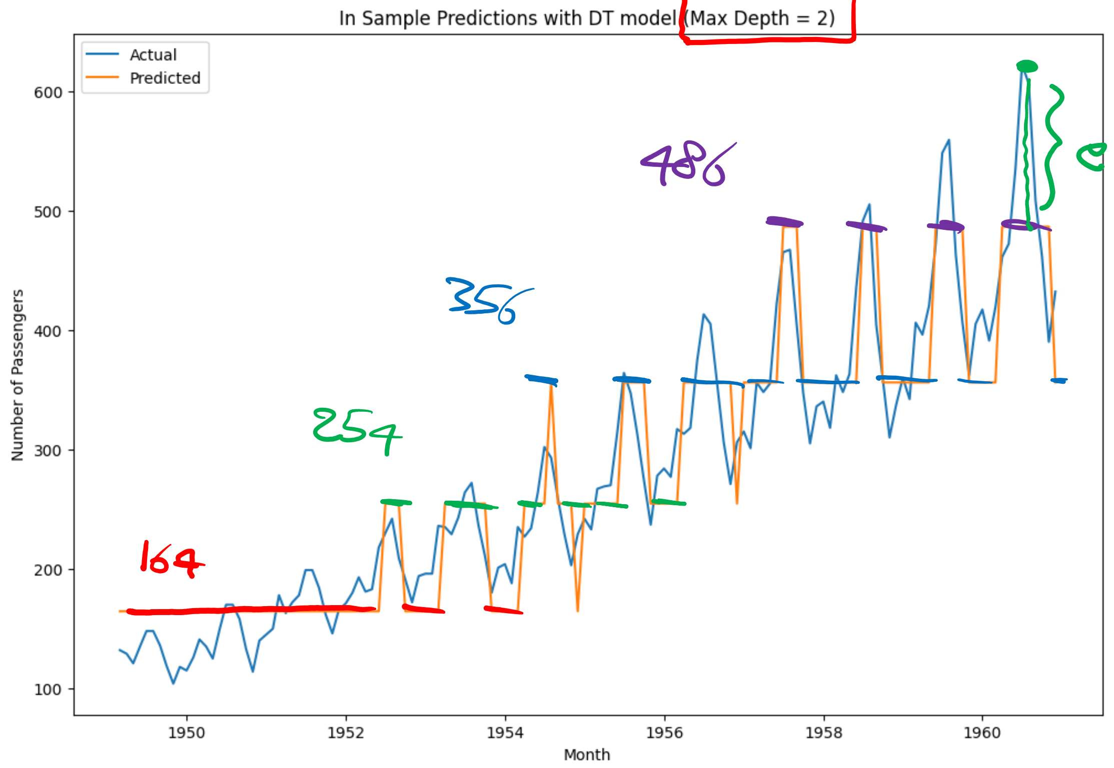


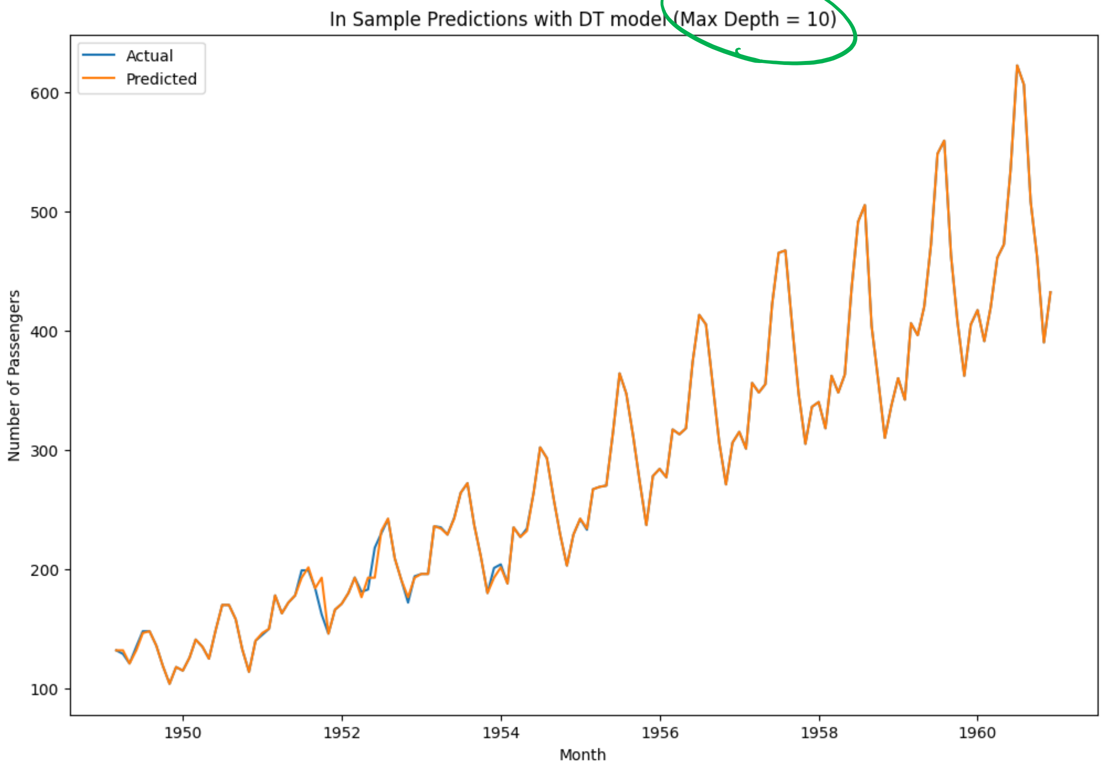
Decision Tree TS regression (Intuition)





Decision Tree TS regression (Intuition)





RMSE !!

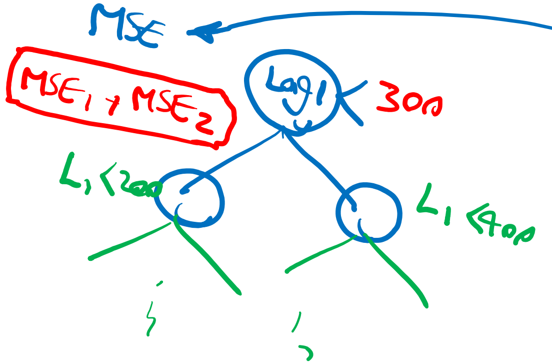
train set

RMSE → 0

What feature and cut off to start with?

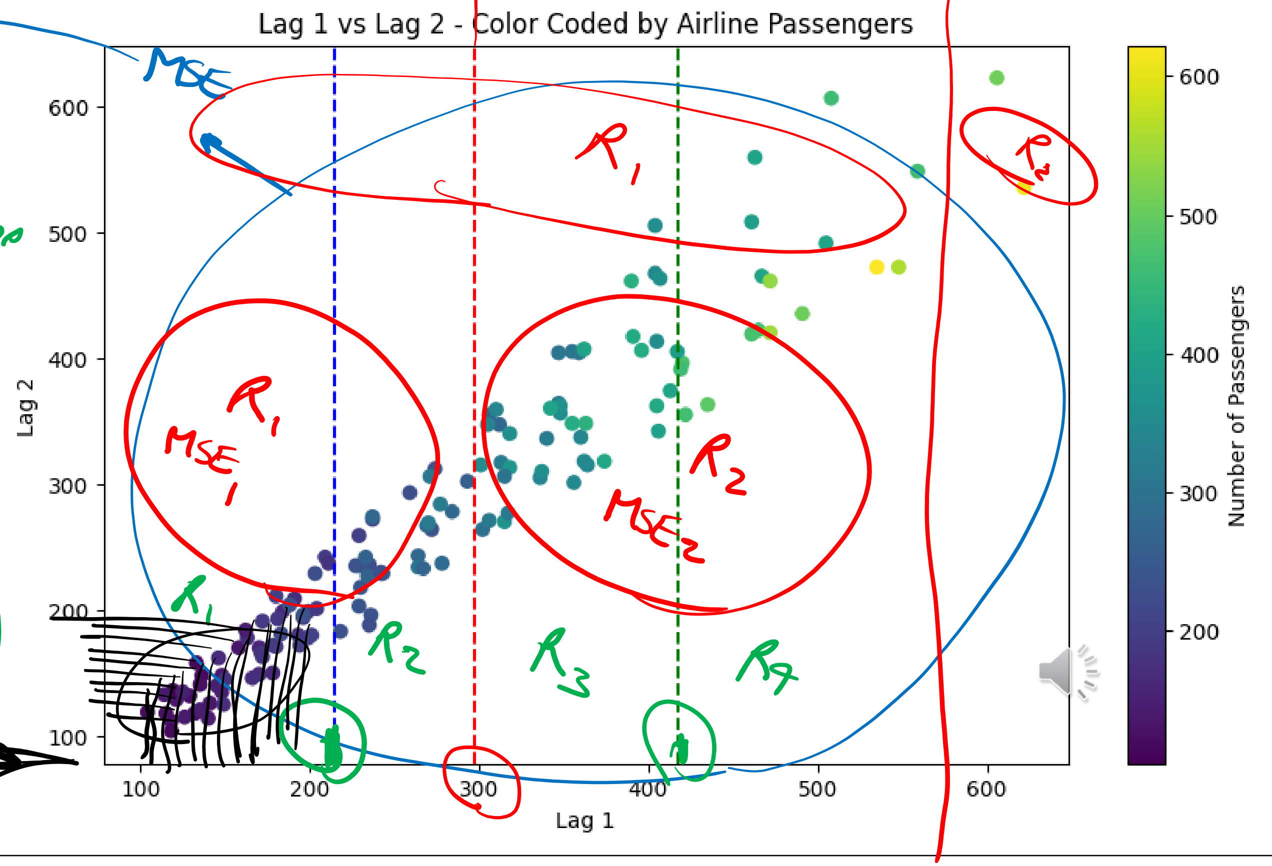
- Which feature and cut off adds the most information gain (minimum impurity)?
  - Regression trees: MSE
  - Classification trees:
    1. Error rate
    2. Entropy
    3. Gini Index

Control how a Decision Tree decides to split the data



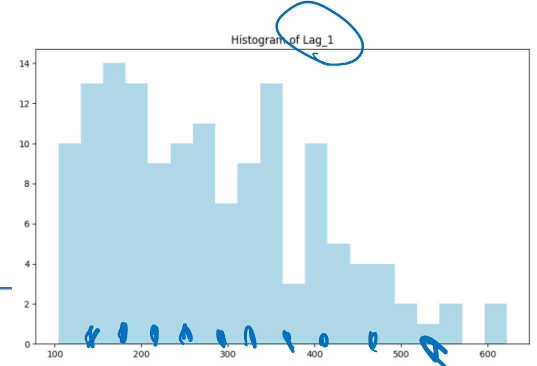
$$\sum_{j=1}^{J}\sum_{i\in R_j}(y_i-\hat{y}_{R_j})^2$$

$$\sum_{j=1}^{4} MSE_j$$



How to sample the data for splitting?

x_1 Lag1 (112, —— 800)
x_2 Lag2 (112, —— 800)



Method	Description
Pre-sorted Naive DT RF	Sorts all data for each feature to evaluate every possible split point. Data is pre-sorted to expedite split evaluation. + Simple and accurate - Computationally intensive
Histogram-Based Xgboost	Bins continuous feature values into discrete intervals (histograms) to reduce the number of potential split points evaluated. + Reduces computational load - May miss optimal splits within bins
GOSS (Gradient-based One-Side Sampling) for Gradient Boosting models only lightGBM	Focuses on samples with large gradients (errors) by retaining them and under-sampling those with small gradients. Adjusts sample weights accordingly. + Reduces computational cost - Adds complexity to the implementation.

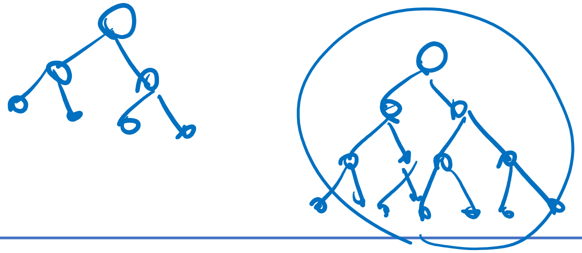
JON M. HUNTSMAN SCHOOL OF BUSINESS
UtahStateUniversity

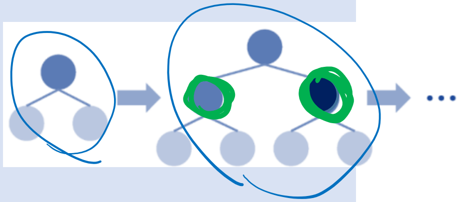
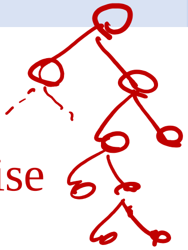
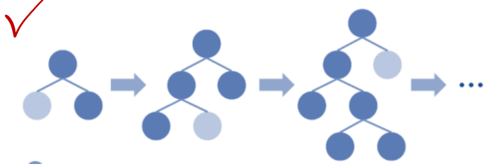
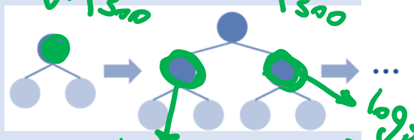
Pedram Jahangiry

How to Split the Samples? (What Features and Cutoffs to Start With)

Method	Description
Greedy DT RF xgb CAT light	At each node, selects the feature and cutoff that provide the greatest immediate reduction in the loss function (e.g., MSE). Decisions are made based solely on local optimization without considering future splits. + Simple and computationally efficient - May not lead to the globally optimal tree
Non-Greedy	Considers the global impact of splits by evaluating future nodes or using global optimization techniques. Methods include optimal decision trees, lookahead splits, evolutionary algorithms, Bayesian trees, and etc. + More accurate and generalizable models by avoiding local optima - Computationally intensive and complex

How to grow a tree?



Algorithm	Description	
Depth-Wise Level-Wise	This strategy grows the tree level by level (one level at a time), all nodes are expanded simultaneously before moving to the next level. This results in a balanced tree structure.	
 Leaf-wise	This strategy, rather than growing by levels, focuses on expanding the tree by adding nodes to the leaves, specifically those that result in the highest decrease in impurity or error. This can lead to a more unbalanced tree but potentially more efficient learning.	
Symmetric	This strategy attempts to maintain balance not just in the depth of the tree but also in how the features are split, aiming for a tree that grows evenly across all paths. This results in faster compute.	

All the methods, repeatedly split the data along the feature with the highest information gain and process continues until a stopping criterion is met (max depth, min samples at nodes, etc)

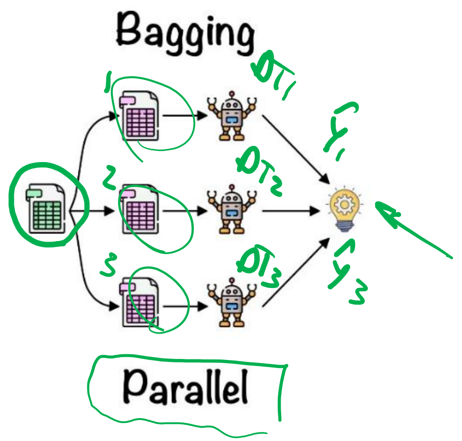
Jon M. Huntsman School of Business
Utah State University

Pedram Jahangiry

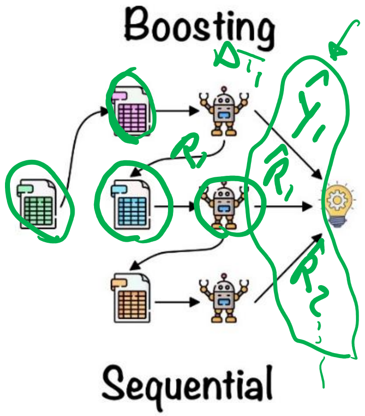
How to combine trees?

RF

- Bagging consists of creating many “copies” of the training data at the same time (each copy is slightly different from another) and then apply the weak learner to each copy to obtain multiple weak models and then combine them. DT
- In bagging, the bootstrapped trees are independent from each other.



- Boosting consists of using the “original” training data and iteratively (sequentially) creating multiple models by using a weak learner. Each new model tries to “fix” the errors which previous models make.
- In boosting, each tree is grown using information from previous tree.



JON M. HUNTSMAN SCHOOL OF BUSINESS
UtahStateUniversity

Pedram Jahangiry

Decision-Tree based models

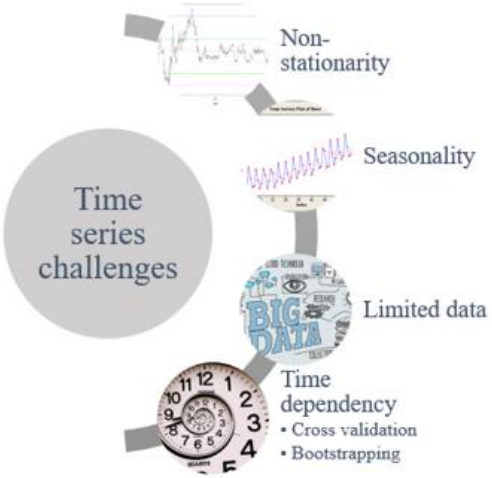
Aspect	Decision Tree (DT)	Random Forest (RF)	XGBoost	CatBoost	LightGBM
1. Sampling Process	Naïve	Naïve	Histogram-based	Feature Binning (Quantization)	GOSS
2. Splitting Method	Greedy	Greedy	Greedy	Greedy	Greedy
3. Tree Growth Strategy	Depth-wise	Depth-wise	Depth-wise (Leaf-wise)	Symmetric	Leaf-wise
4. Combining Trees	Not applicable	Bagging	Boosting	Boosting	Boosting

Module 5 – Part I ✓
Machine Learning For timeseries Forecasting

✓ Module 5 – Part II
Machine Learning for timeseries Decision Tree based Models

Aspect	Decision Tree (DT)	Random Forest (RF)	XGBoost	CatBoost	LightGBM
1. Sampling Process	Naïve	Naïve	Histogram-based	Feature Binning (Quantization)	GOSS
Method					Greedy
wth					Leaf-wise
ng Trees					Boosting

✓ Module 5 – Part III
Challenges in Time Series Machine Learning



Module 5 – Part IV
ML for timeseries in Python

Road map!

- ✓ Module 1- Introduction to Deep Forecasting
- ✓ Module 2- Setting up Deep Forecasting Environment
- ✓ Module 3- Exponential Smoothing
- ✓ Module 4- ARIMA models
- ✓ Module 5- Machine Learning for Time series Forecasting
- Module 6- Deep Neural Networks
- Module 7- Deep Sequence Modeling (RNN, LSTM)
- Module 8- Prophet and Neural Prophet

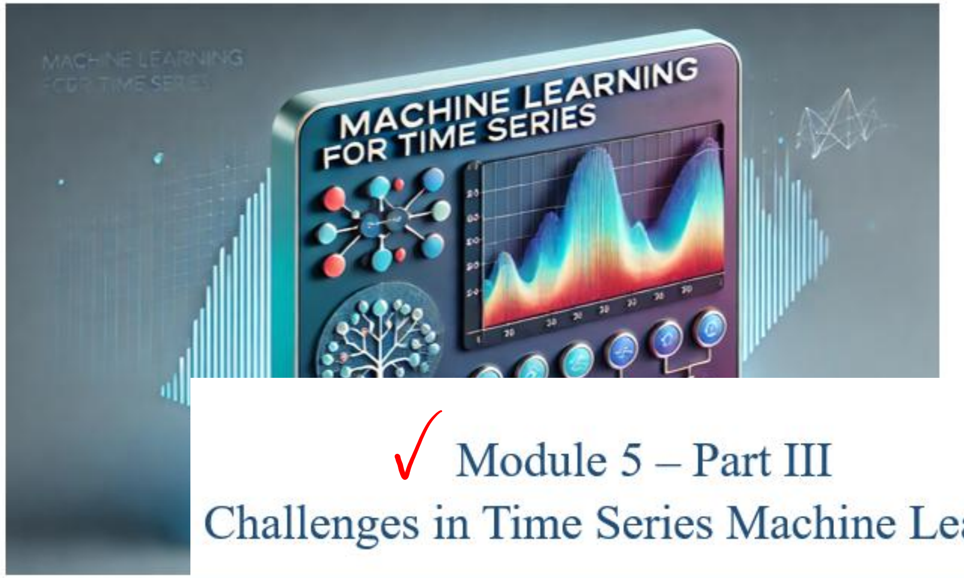
Jon M. Huntsman School of Business
Utah State University

Pedram Jahangiry

Road map!

- Module 1- Introduction to Deep Forecasting
- Module 2- Setting up Deep Forecasting Environment
- Module 3- Exponential Smoothing
- Module 4- ARIMA models
- Module 5- Machine Learning for Time series Forecasting
- Module 6- Deep Neural Networks
- Module 7- Deep Sequence Modeling (RNN, LSTM)
- Module 8- Prophet and Neural Prophet

Module 5 – Part I
Machine Learning For timeseries Forecasting

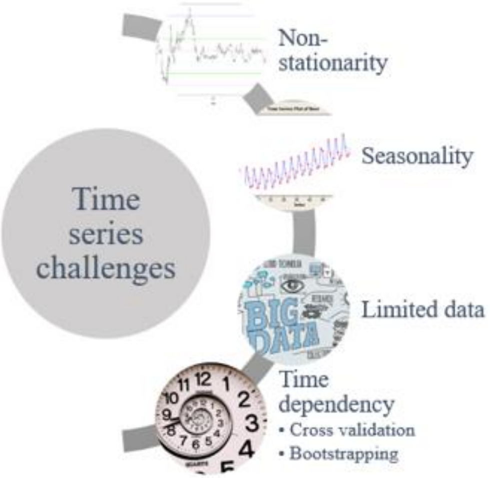


Module 5 – Part II
Machine Learning for timeseries Decision Tree based Models

dmlc XGBoost CatBoost LightGBM

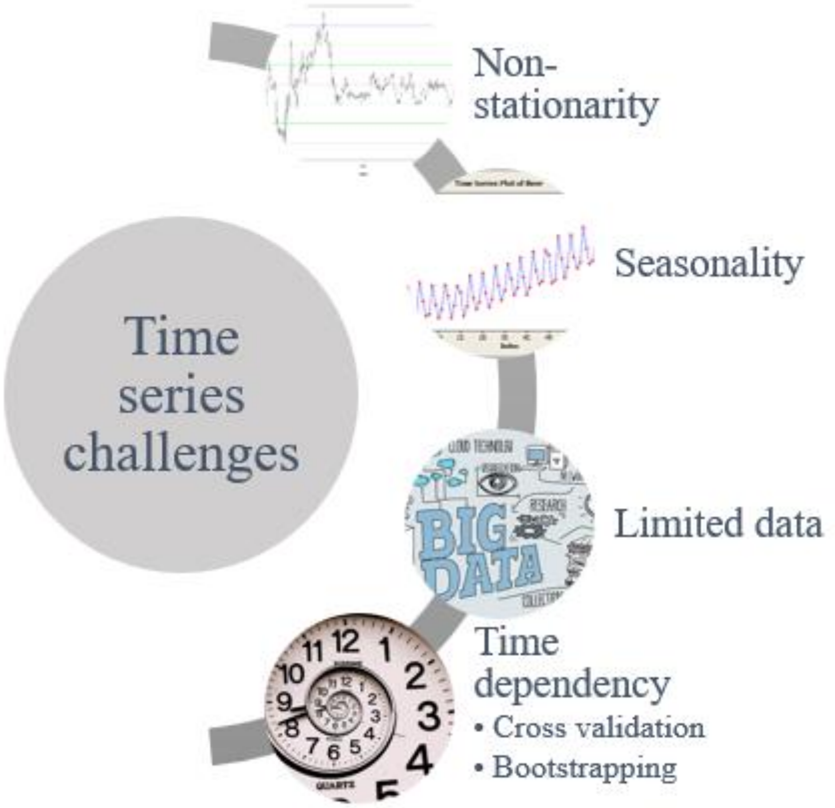
Aspect	Decision Tree (DT)	Random Forest (RF)	XGBoost	CatBoost	LightGBM
1. Sampling Process	Naïve	Naïve	Histogram-based	Feature Binning (Quantization)	GOSS
Method					Greedy
wth					Leaf-wise
ng Trees					Boosting

Module 5 – Part III
Challenges in Time Series Machine Learning



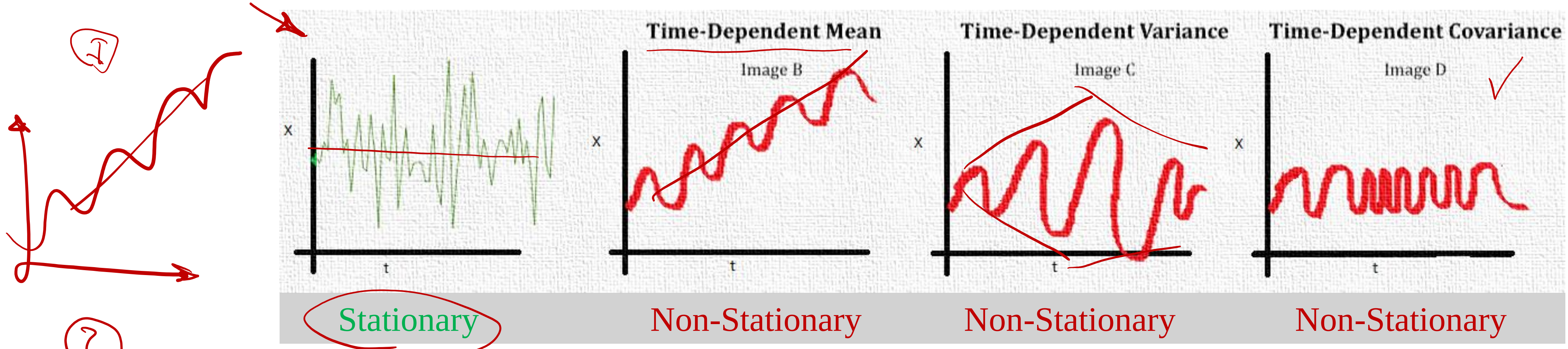
Module 5 – Part IV
ML for timeseries in Python

Module 5 – Part III
Challenges in Time Series Machine Learning



Stationarity

- Stationary vs Non-Stationary Data. What makes a data set Stationary?
- In a stationary timeseries, the statistical properties do not depend on the time



- Data with trend and seasonality are NOT stationary!

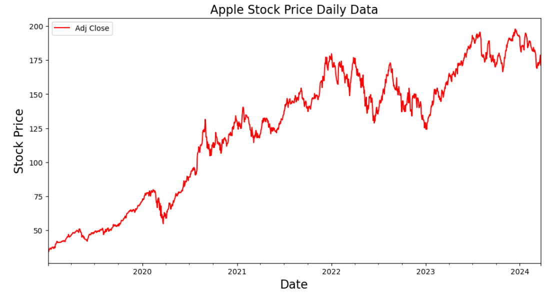
Jon M. Huntsman School of Business
UtahStateUniversity

Pedram Jahangiry

Time Series Stationarity & ML: Well-Behaved Non-Stationary Data

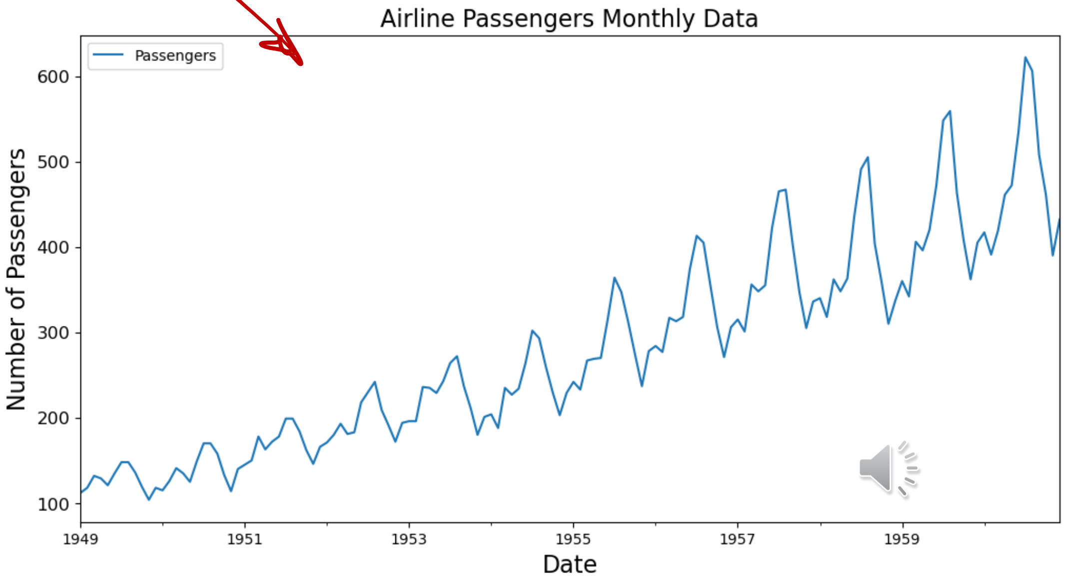
Example: Airline Passenger Data

- Shows a clear upward trend and seasonal pattern
- Non-stationary but predictable structure



Methods That Can Handle It:

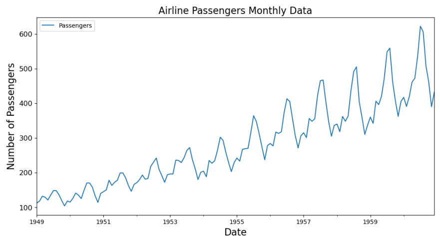
- ETS: Handles trend and seasonality directly
- ARIMA: Differencing (the "I") to remove trend
- Machine Learning: Lag features and time-based feature engineering



Time Series Stationarity & ML: Complex Non-Stationary Data

Example: Apple Stock Price

- Driven by external/random factors (markets, news, events)
- May have sudden jumps, volatility, and regime shifts



Challenges:

- Simple differencing or trend/seasonality modeling may **not** suffice. Need more advanced techniques:

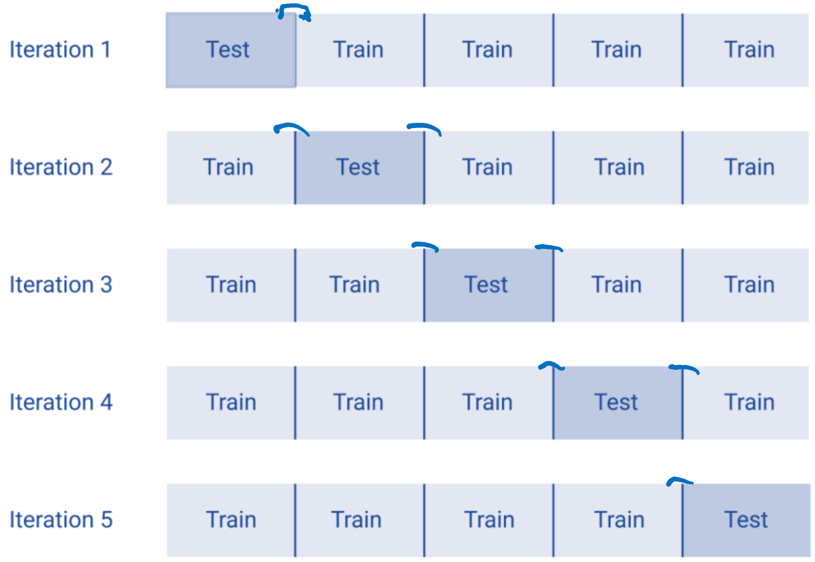
✓ Additional feature engineering (e.g., volatility measures, external economic indicators)

✓ Regime-switching models (to capture structural breaks)

✓ Deep learning / advanced ML with more sophisticated patterns

Time Series Cross Validation

- With time series data, we cannot shuffle the data! TS data is not IID.
- We also need to avoid data leakage!



- The main time series CV methods are:
  1) Purged K-Fold CV
  2) Walk forward rolling / expanding window
  3) Combinatorial purged CV

Purged K-Fold CV

t, t_{+1}

- Leakage takes place when the training set contains information that also appears in the testing set.
- Leakage will enhance the model performance
- Solution: Purging and Embargoing

- Purged K-Fold CV: Adding purging and embargoing whenever we produce a train/test split in K-Fold CV.

Iteration 1	Test	Train	Train	Train	Train
Iteration 2	Train	Test	Train	Train	Train
Iteration 3	Train	Train	Test	Train	Train
Iteration 4	Train	Train	Train	Test	Train
Iteration 5	Train	Train	Train	Train	Test

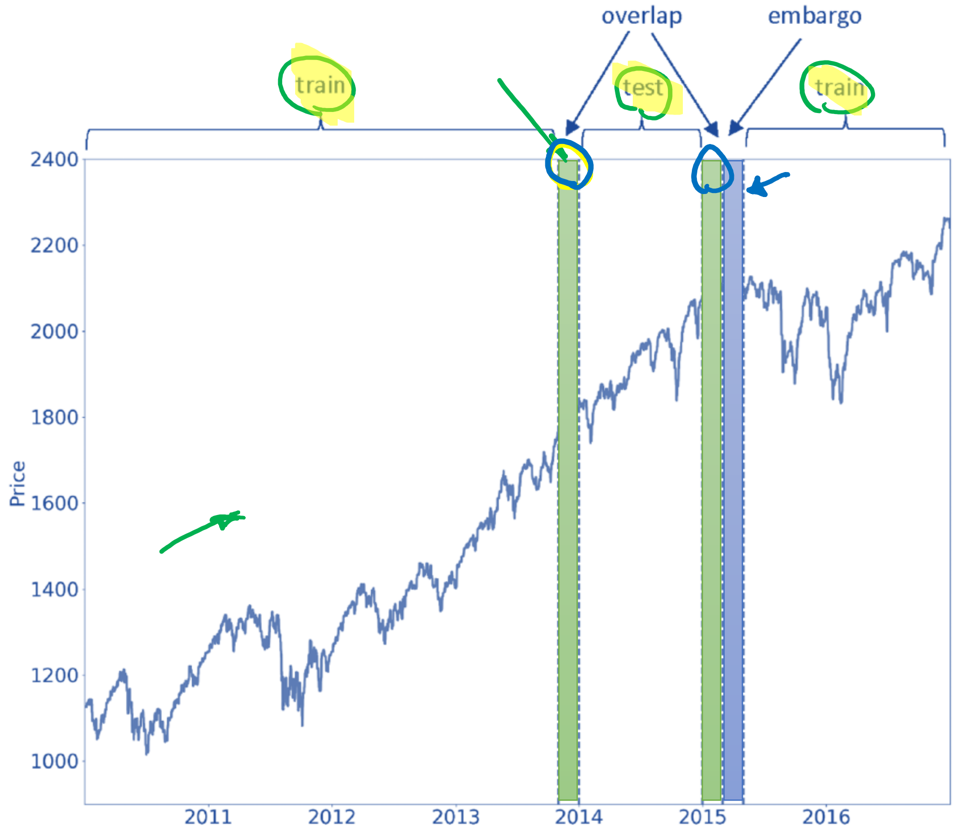


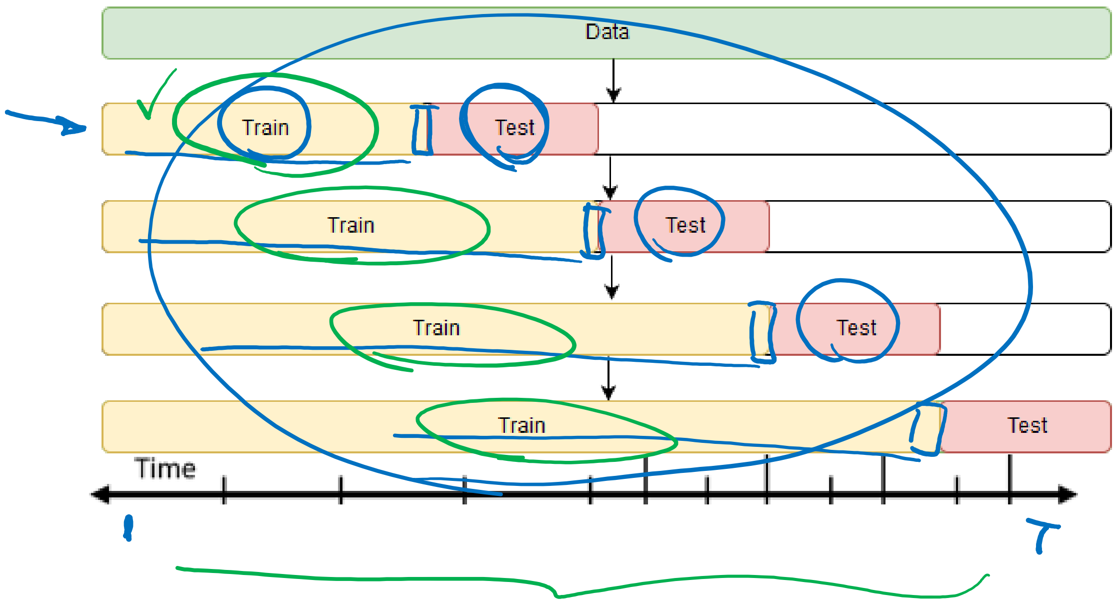
FIGURE 7.3 Embargo of post-test train observations

Jon M. Huntsman School of Business
UtahStateUniversity

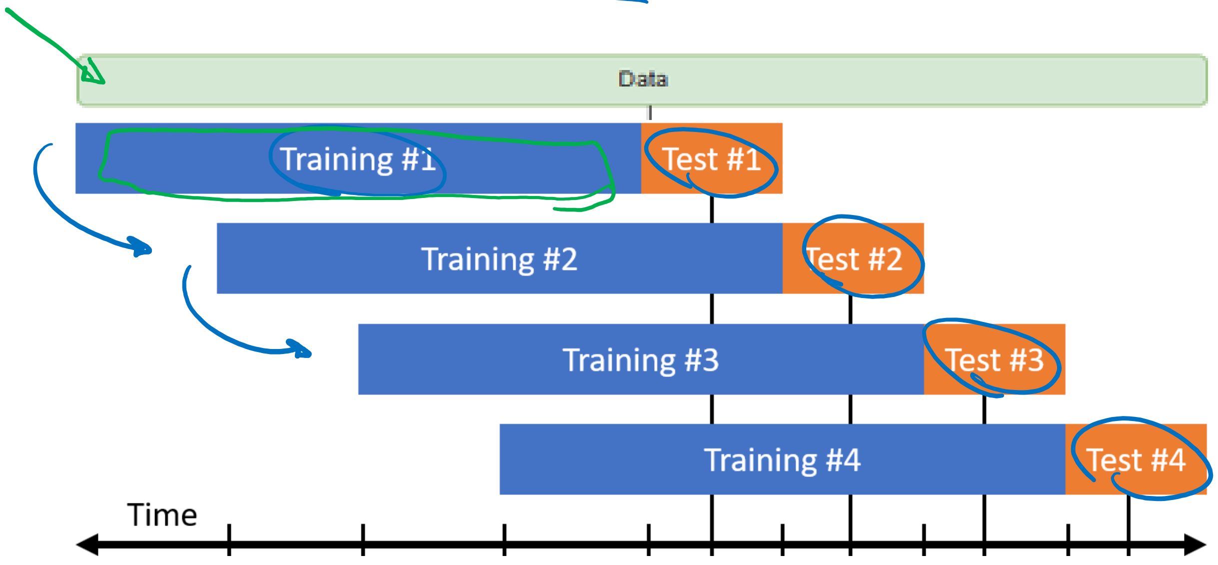
Pedram Jahangiri

Walk Forward Cross Validation

Walk forward cross validation
Expanding windows



Walk forward cross validation
Rolling windows



Combinatorial Purged Cross Validation (CPCV)

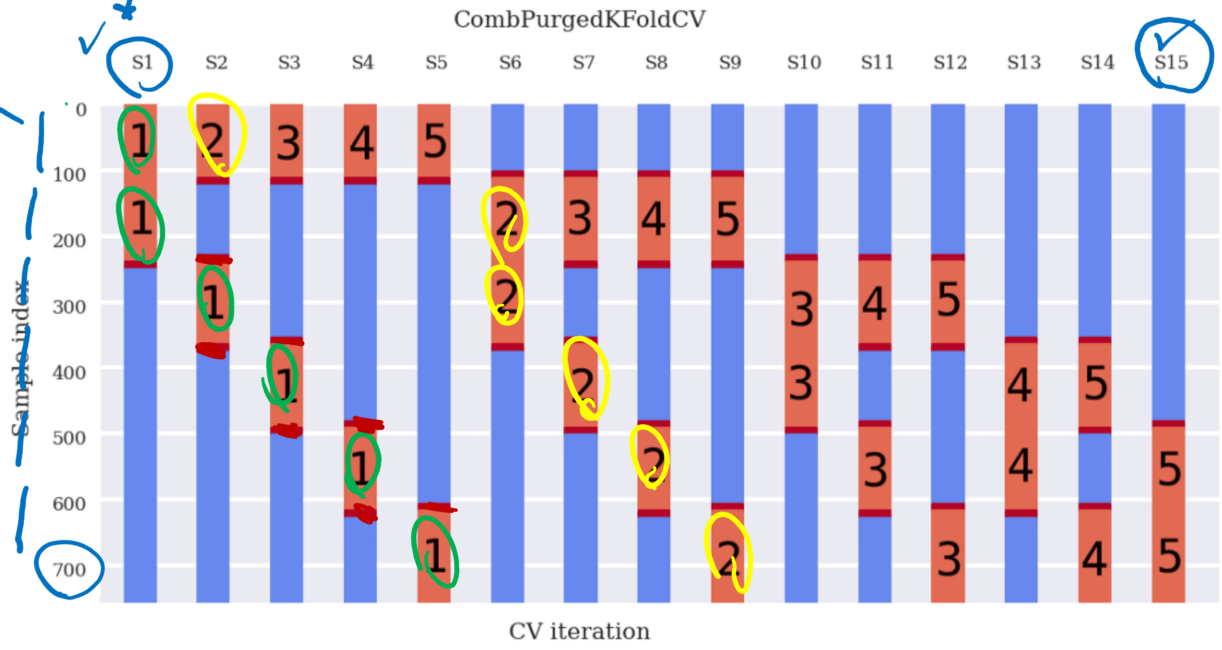
- The goal is to generate multiple unique back-test path that span the entire data set.
- In each path, we can look at the model's OOS performance for the entire time period.

Groups: Month

$G = 6$

test 2

$\binom{6}{2} = 15$



5

unique paths

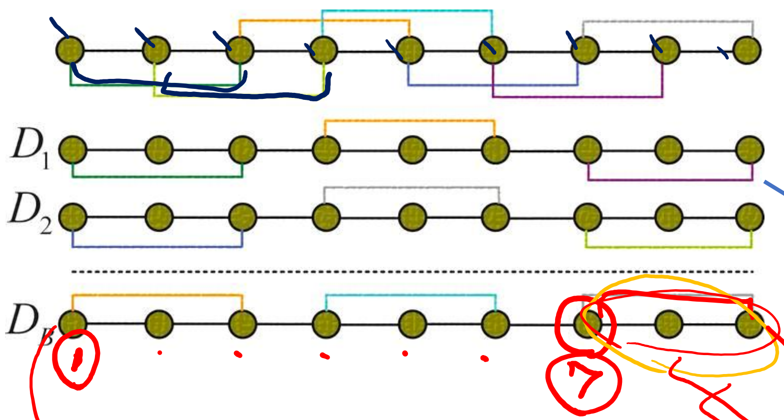
Time Series Bootstrapping

- IID bootstrapping (random sample with replacement) does not work for time series data with temporal dependency.
- Time series Bootstrapping methods:
  - Parametric (based on models with iid residuals and resampling from residuals. Example: ARIMA bootstrap)
  - Non-parametric block bootstrap (data is directly resampled. Assumption: blocks can be samples so that they are approximately iid)
    - Moving Block Bootstrap (MBB)
    - Circular Block Bootstrap (CBB)
    - Stationary Bootstrap (SB)

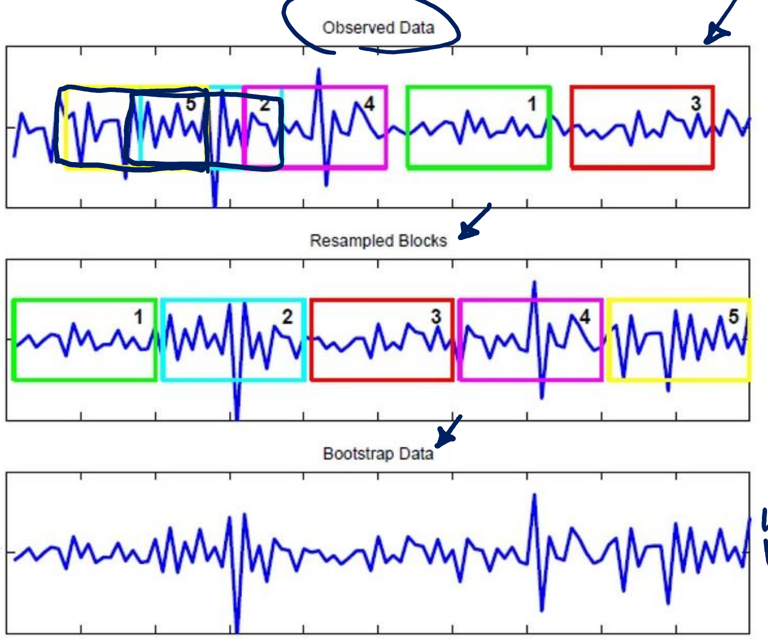
Moving Block Bootstrap (MBB)

- Moving Block Bootstrap, samples overlapping fixed size blocks of m consecutive observations.
- Blocks starts at indices 1, … , T-m+1

m = 3



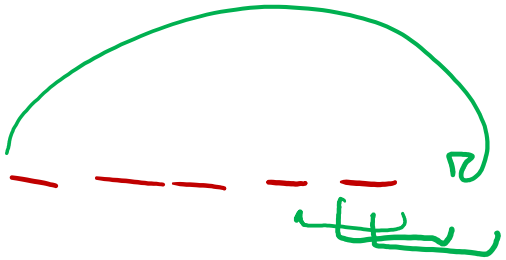
T-m+1 = 9-3+1 = 7

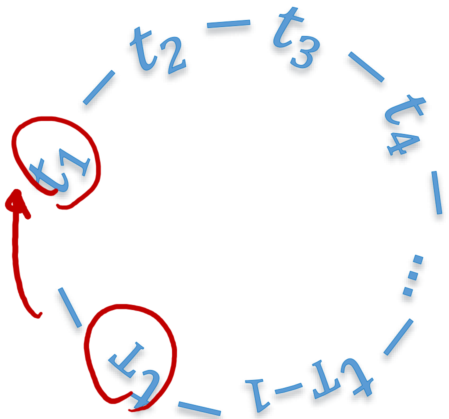


Circular Block Bootstrap (CBB)

- CBB is a simple extension of MBB which assumes the data live on a circle so that $y_{T+1} = y_1, \; y_{T+2} = y_2$, etc.

- CBB has better finite sample properties since all data points get sampled with equal probability.



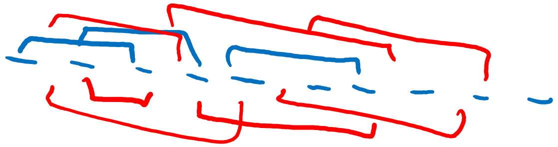


Jon M. Huntsman School of Business
Utah State University

Pedram Jahangiry

Stationary Bootstrap (SB)

$\bar{m} = 3$



- In SB, the block size is no longer fixed.
- Chooses an average block size of m rather than an exact block size.
- Popularity of SB stems from difficulty in determining optimal m
- Once applied to stationary data, the resampled pseudo time series by SB are stationary. This is not the case for MBB and CBB.

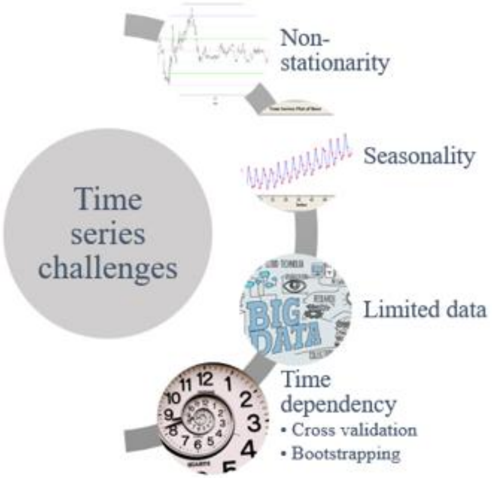
Module 5 – Part I
Machine Learning For timeseries Forecasting

Module 5 – Part II
Machine Learning for timeseries
Decision Tree based Models

dmlc XGBoost CatBoost LightGBM

Aspect	Decision Tree (DT)	Random Forest (RF)	XGBoost	CatBoost	LightGBM
1. Sampling Process	Naïve	Naïve	Histogram-based	Feature Binning (Quantization)	GOSS
Method					Greedy
wth					Leaf-wise
ng Trees					Boosting

Module 5 – Part III
Challenges in Time Series Machine Learning



Module 5 – Part IV
ML for timeseries in Python

scikit learn

PYCARET

Jon M. Huntsman School of Business
Utah State University

Pedram Jahangiry

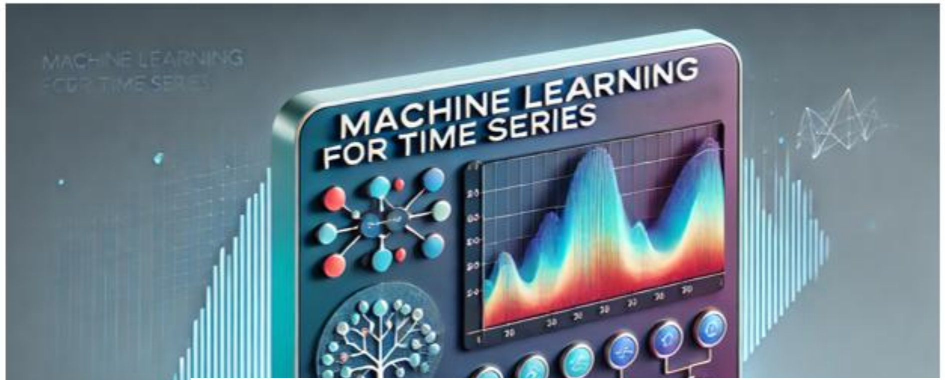
Road map!

- ✓ Module 1- Introduction to Deep Forecasting
- ✓ Module 2- Setting up Deep Forecasting Environment
- ✓ Module 3- Exponential Smoothing
- ✓ Module 4- ARIMA models
- ✓ Module 5- Machine Learning for Time series Forecasting
- Module 6- Deep Neural Networks
- Module 7- Deep Sequence Modeling (RNN, LSTM)
- Module 8- Prophet and Neural Prophet

Road map!

- Module 1- Introduction to Deep Forecasting
- Module 2- Setting up Deep Forecasting Environment
- Module 3- Exponential Smoothing
- Module 4- ARIMA models
- Module 5- Machine Learning for Time series Forecasting
- Module 6- Deep Neural Networks
- Module 7- Deep Sequence Modeling (RNN, LSTM)
- Module 8- Prophet and Neural Prophet
- Miscellaneous topics

Module 5 – Part I
Machine Learning For timeseries Forecasting

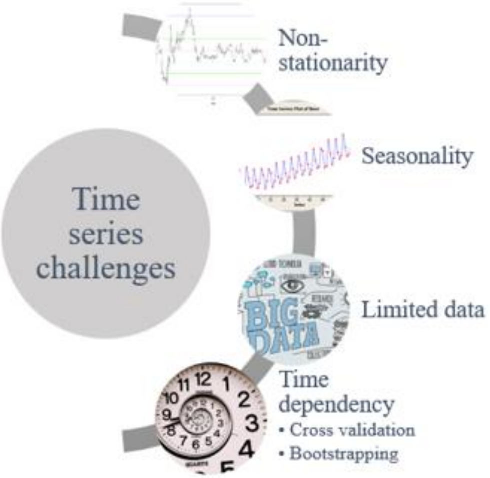


Module 5 – Part II
Machine Learning for timeseries Decision Tree based Models

dmlc XGBoost CatBoost LightGBM

Aspect	Decision Tree (DT)	Random Forest (RF)	XGBoost	CatBoost	LightGBM
1. Sampling Process	Naïve	Naïve	Histogram-based	Feature Binning (Quantization)	GOSS
Method					Greedy
wth					Leaf-wise
ng Trees					Boosting

Module 5 – Part III
Challenges in Time Series Machine Learning



Module 5 – Part IV
ML for timeseries in Python

python scikit learn PYCARET

Jon M. Huntsman School of Business
UtahStateUniversity

Pedram Jahangiry

Module 5 – Part IV

ML for timeseries in Python

Jon M. Huntsman School of Business
Utah State University

Pedram Jahangiry

Single-Output

	series
	y1
	y2
	y3
	y4
	y5
Train	.
	.
	.
	y130
	y131
	y132
	y133
	y134
	.
Test	.
	.
	y142
	y143
	y144

Tx = 2
Ty = 1

X1	X2	Single output y
y1	y2	y3
y2	y3	y4
y3	y4	y5
y4	y5	y6
y5	y6	y7
.	.	.
.	.	.
y129	y130	y131
y130	y131	y132
y131	y132	y133
y132	y133	y134
.	.	.
.	.	.
.	.	.
.		
y141	y142	y143
y142	y143	y144

	X1	X2	y
	y143	y144	y145
Forecast	y144	y145	y146
	y145	y146	y147

train 132
test 12
total 144

train 130
test 12
total 142

Multi-Output

	series
	y1
	y2
	y3
	y4
	y5
Train	.
	.
	.
	y130
	y131
	y132
	y133
	y134
	.
Test	.
	.
	y142
	y143
	y144

Tx = 2
Ty = 12

		Multi output				
X1	**X2**	**y**	**y**	**y**	**...**	**y**
y1	y2	y3	y4	y5	...	y14
y2	y3	y4	y4	y4	..	y15
y3	y4	y5	y5	y5	..	y16
.	.	.	.	.	..	.
.	.	.	.	.	..	.
.	.	.	.	.	..	.
.	.	.	.	.	..	.
y130	y131	y132	y133	y134	..	y143
y131	y132	y133	y134	y135	..	y144

Forecast	y143	y144	y145	y146	y147	...	y156

train 130
test 1
total 131

JON M. HUNTSMAN SCHOOL OF BUSINESS
UtahStateUniversity

Pedram Jahangiry

Road map!

- ✓ Module 1- Introduction to Deep Forecasting
- ✓ Module 2- Setting up Deep Forecasting Environment
- ✓ Module 3- Exponential Smoothing
- ✓ Module 4- ARIMA models
- ✓ Module 5- Machine Learning for Time series Forecasting
- Module 6- Deep Neural Networks
- Module 7- Deep Sequence Modeling (RNN, LSTM)
- Module 8- Prophet and Neural Prophet
- Miscellaneous topics

Jon M. Huntsman School of Business
Utah State University

Pedram Jahangiry