Chapter 1

- Classical statistics and econometrics (including regressions) involve estimation, inference (or testing), prediction, and grouping
- Machine learning
  - Impact of features or explanatory variables may be measured in other metrics, typically not by underlying statistical distributions of the parameters
  - Does not employ underlying theoretical model that provides r/s amongst dependent and explanatory variables
  - Does not assume probability distribution
  - No stats model / hypothesis testing
- Advantage of statistical model – provide policy makers with what (future) decision variables to control
- Why OOS are substandard
  - Randomness of returns create large deviations
  - Estimation of covariance matrix contains errors
  - Errors can be reduced if multi-factor model can explain
- Denoising correlation matrix
  - Positive definite symmetric matrix has strictly positive eigenvalues
  - Matrix is positive definite if symmetric, and all its eigenvalues are strictly positive.
  - Sample/population covariance and correlation matrices of stock returns, that are both symmetric and positive definite, have positive eigenvalues
  - Constant residual eigenvalue method
    - Set a constant eigenvalue for all random eigenvalues that are smaller than or equal to U
    - Preserve total variance
- Diversification is limited in improving results if future returns haves means that are not similar to means of historical returns used to compute optimal weights
- Price drops after dividend is issued, why? Prevent arbitrage, buy before ex d and sell immediately after

Chapter 2

- If the training set performance and validation set performance good, then employing the fitted model on test. Validation also acts as a robustness check.
- Overfitting can occur due to multicollinearity and outliers
- Validation data set prediction performance should return consistent performance of similar accuracy as in the training data, inconsistent performance could be over-fitting or very high prediction accuracy in the training data set but under-fitting or high prediction error in the validation data set.
- If tuning hyperparameters cannot eliminate high prediction error in validation data set, predictive model could be problematic
- Residual noise in ML takes a low profile
- Regularization improve OOS prediction when model is complex too many features against a finite sample size or where high feature correlations that may produce higher prediction errors
- Regularization aka shrinkage estimators – reduce excessive number of features, outsized outliers and high multi-collinearity.
- Lasso – problems with larger set of features (coefficients can go to 0)

$$\text{Min } \sum_{k=1}^{n}\left(Y_k - \sum_{j=0}^{p} b_j X_{jk}\right)^2 + \alpha \sum_{j=0}^{p} |b_j|$$

- Ridge (L2 norm) – reduces correlated features without quickly removing either, useful when shrinking them together without eliminating any would lead to better prediction

$$\text{Min } \sum_{k=1}^{n}\left(Y_k - \sum_{j=0}^{p} b_j X_{jk}\right)^2 + \alpha \left(\sum_{j=0}^{p} b_j^2\right)$$

- CV more robust but more computing time

Chapter 3

- 10-K – annual reports including audited financial statements
- 10-Q – quarterly performance information
- Net working capital = current asset – current liability, +ve able to fund current operations daily basis
- Invested capital = NWC + investments + PPE + intangibles
- Net profits measured against invested capital to see RoR of capital
- LTD vs equity – leverage
- FCF = Operating CF – investing CF
  - Cashflow to repay creditors, dividends, interest
- Type 1 error – null hypothesis that is true is rejected (not accepted)
- Type 2 error – null hypothesis that is false is accepted (not rejected)
- High precision when high FP is a problem such as recommending music (film) of type A (positive class) wrongly to new platform subscribers who dislike music (film) of type A – may cause business loss due to customer dissatisfaction and churning
- High recall when high FN is a problem such as not being able to correctly screen patients for allergies (positive class) to some treatment drugs. Such FN may result in fatal treatments using those medications/drugs.
- Given the predicted probabilities, as threshold increase to 1, less cases are predicted as P, hence TP decreases to zero, and FN increases (since TP + FN add up to a fixed number of actual positive cases). Thus, TPR or sensitivity TP / (TP + FN) goes to 0.
- Threshold is increase to 1, more cases are predicted as N, hence TN increases, and FP decreases (since TN + FP add up to a fixed number of actual negative cases). Thus, FPR or FP / (TN + FP) goes to 0
- Threshold increases, sensitivity (recall) drops while specificity increases, i.e., detection of positive cases becomes less frequent due to more conservatism while detection of negative cases becomes more frequent
- Threshold approaches zero, more cases predicted as P, hence TP increases, and FN decreases (since TP + FN add up to a fixed number of actual positive cases). Thus, TPR or sensitivity TP / (TP + FN) increases to 1.
- Threshold decrease to 0, less cases predicted as N, hence TN decreases, FP increases (since TN + FP add up to a fixed number of actual negative cases). Thus, FPR, FP / (TN + FP), go to 1. As threshold decreases, sensitivity (recall) increases while specificity decreases, i.e., detection of positive cases becomes more frequent due to less conservatism while detection of negative cases becomes less frequent.
- TPR > FPR, high chance model can distinguish positive cases from negative cases
- Kappa value – compares observed accuracy to expected accuracy based on random chance
- True Positive: Predict Positive & Actual Positive
- True Negative: Predict Negative & Actual Negative
- Accuracy = $\frac{TP+TN}{TP+TN+FP+FN}$
- Precision = $\frac{TP}{TP+FP}$
- Recall (TPR / Sensitivity) = $\frac{TP}{TP+FN}$
- Specificity (Actual negative that were predicted correctly) = $\frac{TN}{TN+FP}$
- FPR (actual negative predicted as FP) = 1 – Specificity = $\frac{FP}{TN+FP}$
- F1 = $\frac{2}{\frac{1}{precision}+\frac{1}{recall}}$

Predicted Values

Actual Values	Positive	Negative
Positive	**TP** 10	**FN** 10
Negative	**FP** 80	**TN** 900

Macro average of precision = $(prec_A + prec_B)/2$

Macro average of recall = $(rec_A + rec_B)/2$

Macro average of F1-score = $(f1_A + f1_B)/2$

that are based on simple arithmetic averages. The weighted averages of these measures are respectively

$n_A/(n_A+n_B) \times prec_A + n_B/(n_A+n_B) \times prec_B$, $n_A/(n_A+n_B) \times rec_A + n_B/(n_A+n_B) \times rec_B$,

and $n_A/(n_A+n_B) \times f1_A + n_B/(n_A+n_B) \times f1_B$.

- ROC plots TPR against FPR
- Logistic Regression

$$E(Y_i|X_i) = \text{Prob}(Y_i = 1|X_i) = \exp(\beta^T X_i)/[1+\exp(\beta^T X_i)] = 1/[1+\exp(-\beta^T X_i)]$$

  - Loss Function, minimize loss by maximizing likelihood

$$loss = -\sum_{i=1}^{N}\left(Y_i\, logF + (1-Y_i)\, log(1-F)\right)$$

  -
  - Log L is negative, maximizing L = minimizing -LogL
  - Multicollinearity issues
    - VIF – each feature regressed on all other features and VIF is computed, higher R-squared high collinearity
    - Unstable prediction, unstable coefficients, unable to tell which features more important

Chapter 4

- Naïve Bayes
  - Conditional probability assumed independent of any other features
  - Lower computational time, scales up with multi class predictions, works well with many features
  - Common distribution assumptions
    - Categorical
    - Gaussian/normal distributions
      - No need to standardize/normalize features (using all of sample data) before training if test data features also have the same mean and variance as those in the training data
      - Else standardize/normalize features in train and test separately
      - If features do not follow Gaussian => biased predictions
    - Multinomial
      - Probability distribution of features unknown
- KNN
  - transformation is useful so that outliers or features with larger values on a larger scale would not create bias when the distances are measured
  - K needs to be odd
  - K decreases to 1, prediction becomes more unstable
  - K increases, prediction becomes more stable due to majority voting
  - Imbalanced data can cause bias
  - Lazy learning – no active computations done on training data, until test query made
- Linear SVM
  - Solid line boundary (separating hyperplane / optimal decision plane) with the maximum margin (sum of perpendicular distances from the nearest feature points on either class to the line)
  - Kernel trick – transforming feature space with higher dimension than original
  - Non-linear kernels – Gaussian radial bias (overlaps with no clear boundaries) / sigmoid / hyperbolic tangent
  - SVM method does not explicitly compute the probability of a test point Z being in one class/type or another
  - Use SVM score for each sample point as the feature for predicting the class using the logit model
- SVR
  - Minimize errors between the predicted hyperplane and the training data targets but setting the errors to non-zero values only when they exceed a given hyperparameter epsilon > 0.
  - Increasing epsilon would reduce sum of deviations but when the increase is too large and no deviations exist, the fitted slope becomes flat or zero slope.
  - When epsilon shrinks to 0, becomes too sensitive to outliers, approaches linear regression

Chapter 5

- Nonparametric estimation is a statistical method that estimates the functional form of data fits when parametric modelling is absent and where there is no guidance from theory. Examples of nonparametric estimation are kNN, SVM, and kernel estimation
- Due to the non-parametric nature, DT methods can produce unstable outcomes when data features have high variabilities
- Decision Trees
  - Nonparametric estimation
  - How do trees split
    - Gini impurity (empirical probability of being in category), lower better, 0.5 is the worse
    - if the positive case 1 (marksmen) has probability p1 = 200/1000 = 0.2, and the negative case 2 (non-marksmen) has probability p2 = 0.8. The Gini impurity would be $1 - (0.2^2 + 0.8^2) = 0.32$ in this binary classification.

(Values) [Gini Impurity]	Yes	No	Weighted Gini Impurity
If < 10.9 min	(100,320) [0.36281]	(100,480) [0.28358]	(420/1000) × 0.36281 + (580/1000) × 0.28358 = 0.31790
If < 11 min	(120,300) [1 – (120/420)² – (300/420)² = 0.40816]	(80,500) [1 – (80/580)² – (500/580)² = 0.23781]	(420/1000) × 0.40816 + (580/1000) × 0.23781 = 0.30936
If < 11.1 min	(110,340) [0.369383]	(90, 460) [0.273719]	(450/1000) × 0.369383 + (550/1000) × 0.273719 = 0.31677

    - Find split that produces lowest weighted gini impurity lower than the previous node (largest gain in information)
  - Issues
    - Stability: too few features / resampling
    - Overfitting
- Random Forest – run independently, bootstrap aggregation (bagging, sampling with replacement)
  - Reduces variance of outcomes
  - Handles high dimensionality well, reduce use of all features
  - Handles missing value amounts
- Feature importance
  - Total gini importance contributed by all features / sum of gini importance
  - Ratios of all features sum to 100%.
- Not as sensitive to multicollinearity

Chapter 6

- Gradient Boosting
  - RF combines trees at the end while GBM combines trees as each subsequent tree is generated
  - Subsequent trees may have different set of target values, GB solves for the net error let from the past tree

- Constructs weak predictor, compute error, increases predictor by error, build tree to fit/predict error using features of each sample points, compute error (should be smaller here), increases predictor by smaller error hence ideally should converge
- Splitting criterion of minimizing weighted MSE in tree
- More weight to sample points in previous trees with larger errors

Gradient Boosting

- We provide a numerical example. Suppose training sample size N = 4 and the set of target values, initial predictors (sample mean), and initial fitting errors is shown.

Y_i	0.5	1.0	1.2	1.3
F_0	1.0	1.0	1.0	1.0
$e_{(0)i}$	-0.5	0.0	0.2	0.3

- Next, DT(1) computes h_{1j}, hence F_{1j}. Next error is $e_{(1)i} = e_{(0)i} - F_{1j}$.

h_{1i}	-0.8	-0.5	0.4	0.3
F_{1i}	-0.08	-0.05	0.04	0.09
$e_{(1)i}$	-0.42	0.05	0.16	0.21

- Next, DT(2) computes h_{2j}, hence F_{2j}. Next error is $e_{(2)i} = e_{(1)i} - F_{2j}$.

h_{2i}	-0.5	0.4	0.7	1.5
F_{2i}	-0.05	0.04	0.07	0.15
$E_{(2)i}$	-0.37	0.01	0.09	0.06

- We can see that if the algorithm works, e(n) gets smaller toward zero.

- Continuous is predict error term, classification is predict pseudo residuals
- Classification
  - MSE doesn't work as log loss is non-convex (does not allow global minimum and local minimum may not lead to effective predictors)
  - If terminal nodes used updated estimate as empirical probability without going through gradient involving log odds then tree becomes ordinary, replacing gini index with probability residual
  - May use random sampling of smaller set of sample points but without replacement
- Bagging decreases variance in the prediction but may not decrease the bias
- GBM approach reduces bias in trees toward zero, but the variability may be larger than in RF
- LightGBM
  - tree grows not level wise but leaf wise
  - nodes only extended down from previous node with largest information gain
  - if grow as long as possible, similar results with regular GradientBoosting
- XGBoost
  - Offers regularization techniques
  - employs tree pruning algorithms to control the size of decision trees, reducing overfitting and improving computational efficiency. Tree pruning techniques such as depth based pruning (to reduce too much depth) and weight-based (to reduce nodes with too little weights) pruning remove unnecessary branches from decision trees
  - support for handling missing values, automatically estimates best imputation strategy
- Feature importance does not explicitly measure impact of a feature on predicted outcome
- Shaply for feature importance
  - Measure how much each feature in an algorithm contributes to predicted outcome
  - efficiency (no leakage), symmetry, nullity, and linear additivity (or linear scaling in total coalition output)
  - Shapley value of a feature value is the average change in the prediction when the feature value joins an existing coalition of features.
  - Cannot be done for every X so use approximation (monte carlo sampling)
  - Can be done on train and test
    - Prefer test because if good model want to know what contributed to it
  - High Shap means contribute more to probability Y=1
  - For classification expected value is % of predicted value then see what contributed to the predicted 1 / 0

Chapter 7

- Single layer perceptron
  - SUM(input * weight + bias)
  - Weights initialized randomly
- MultiLayer Perceptron
  - Many-to-one – one layer (for number or binary)
  - Many-to-many – output more than 1, for multiclass classification
  - Each input is fed into each neuron
- Wider – larger number of input nodes / neurons
- Deeper – larger number of hidden layers
- Notation: W (weight), j (current layer neuron), k (next layer neuron), superscript (layer number)
- ANN computes outputs in the output layer in forward pass process (forward propagation), errors measured by a loss function to be minimized, reduction in fitting errors requires revision of weights and biases (backward propagation)
- 1 forward prop + 1 backward prop = 1 epoch
- Mini batches => more iterations within one epoch
- Activation functions
  - Logistic $= (1 + e^{-z})^{-1}$, Hyperbolic Tangent $= \frac{(e^z - e^{-z})}{(e^z + e^{-z})}$, RELU$= \max(z, 0)$
- Loss function
  - MAE more robust in presence of some outliers, more stable results
  - MSE better convergence properties when finding optimal weights
  - Binary cross entropy (for binary classification)
    - If y = 1 loss function is reduced if probability of 1 is predicted higher
  - Log-loss measures "distance" of the predicted probability p to the target
- Backward propagation – mechanism of training the NN toward an acceptable loss
- If derivative of loss wrt weight/bias > (<) 0, lower loss can be attained by decreasing (increasing) weight/bias
- Using mini batches reduces computational memory burden of each iteration
- Iterations = epochs * training size / batch size
- To update weights to minimize loss, use gradient descent methods
  - Gradient descent
  - Stochastic gradient descent – slow convergence, more iterations, more volatile
  - Adam (adaptive moment estimation) – compute EMA of gradients and past squared gradients
- When cases are independent, then computation in single iteration can be done in parallel

Chapter 8

- RNN
  - Weights and biases at each neuron level do not change from one hidden layer to another
  - when the hidden layer has m neurons, then the total number of parameters to be fitted is (m+2)(m+1) – 1
  - Each of the hidden layer may be viewed as a recurrent cell since the computations are recurrent on the updated hidden states as the sequence progresses until the output
  - Sequence with T datapoints. The T sample points are divided into cases or packs each with R, e.g., 5, number of recurrent cells/hidden layers. The number of cases or packs is T/R. A certain number of cases/packs are grouped into batches with batch size S. So, there are T/(RS) number of mini batches (each size S).
- Advantage
  - Recognize input of lagged information
  - Take long sequence of inputs
  - Longer computation time during back propagation
- Disadvantage
  - Backpropagation can collapse to 0 or explode to infinity (vanishing / exploding gradient)
    - Pruning, reducing recurrent cells, make model less complex, use variant RNN LSTM
  - Cannot consider future / forward input for training since time sequence uses information from past
    - Use features at current time but are other market-based forecasts of future state
- LSTM
  - Hidden states store short term memory
  - Cell state stores long term memory effects of past inputs
  - In general, for N neurons or units in the first hidden layer of LSTM, and J number of inputs/features, there are 4 * [(N +J) * N + N] parameters to be optimized, where the (N+J) denotes the number of previous hidden state and feature inputs. This is multiplied into N, the number of neurons in the hidden layer.
- Dropout
  - Reduce overfitting.
  - In general, it is used in deep learning NN with many neurons and layers. If the dropout rate is set to say 0.20 at a particular layer (including input layer), then at that layer, 20% of the neurons will be randomly selected to "disappear" in that iteration of forward and backward propagation through that layer.
  - No forward pass from the temporarily dropped out neurons
  - No updates or revisions in weights connected to those neurons during the backward propagation
  - In the next iteration, other neurons will be dropped, revisions of weights continue
  - When dropout rate is added to the training, it is sometimes suggested that the width of the layer be scaled up by 1/(1 – dropout rate) so that the sum of all effective neurons in each iteration remains the same.
- Bidirectional RNN/LSTM – sequences that are NOT time based
- CNN – image identification/classification, can also be applied to time series
- Caveats
  - Transaction cost, bid ask spread, liquidity risk, slippage, trend chasing
- Lecture 5 and 6

Sample MCQs

- Dimensionality reduction reduces collinearity
- Limitations of backpropagation
  - Slow convergence
  - Scaling (increasing size of NN, width and depth)
  - Local minima problem
- Logistic is non-linear
- Best ML method = fast + accurate + scalable
- Stacking – ensemble multiple classifications / regression
- Feature importance DOES NOT show size and direction of effect of predictor on outcome
- Training data linearly separable use linear hard-margin SVM
- Effectiveness of SVM depends on selection of kernel
- Kernel trick transforms nonlinearly separable to linearly separable