Introduction to Machine Learning: Linear and Logistic Regression and Neural Networks Using Python

Chapter 9: Summary and Thank You

Summary and Thank You

Section	Title	Description
9.1	Alternative Approaches	This section has a brief discussion about alternatives to the machine learning approach presented in the previous chapters
9.2	Summary	This section summarizes the course
9.3	Suggestions for Further Study and Thank You	This section provides suggestions for further study and resources and a thank you

9.1 Alternative Approaches

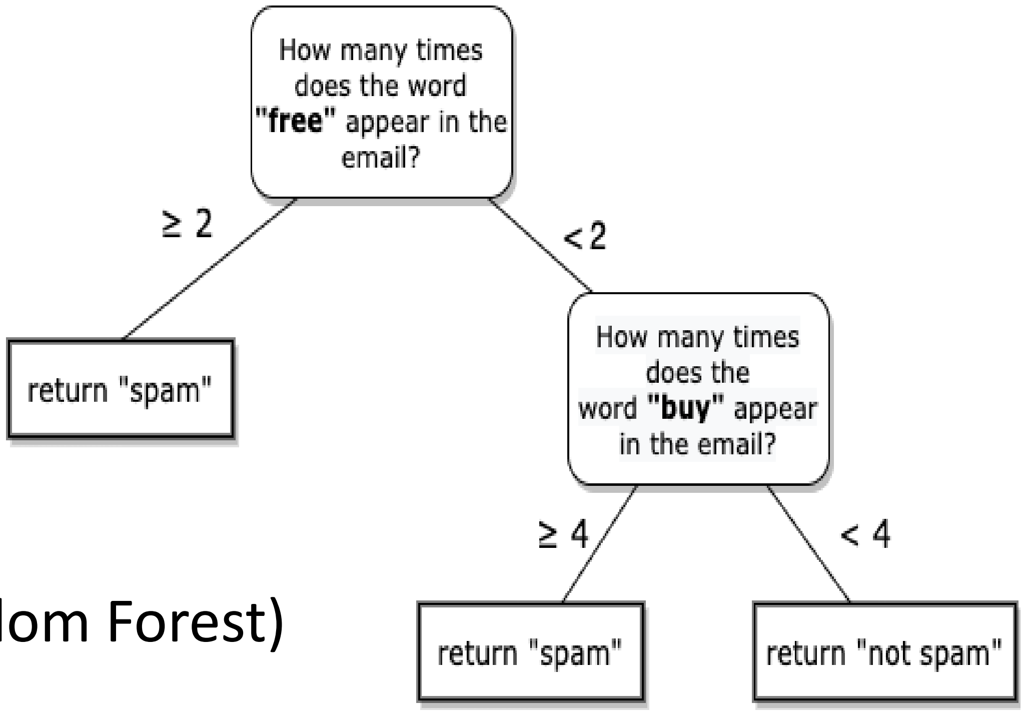
Other Approaches for Supervised Learning

Goal of this Section:

- This section gives a brief description of alternative approaches (tree-based) for supervised learning

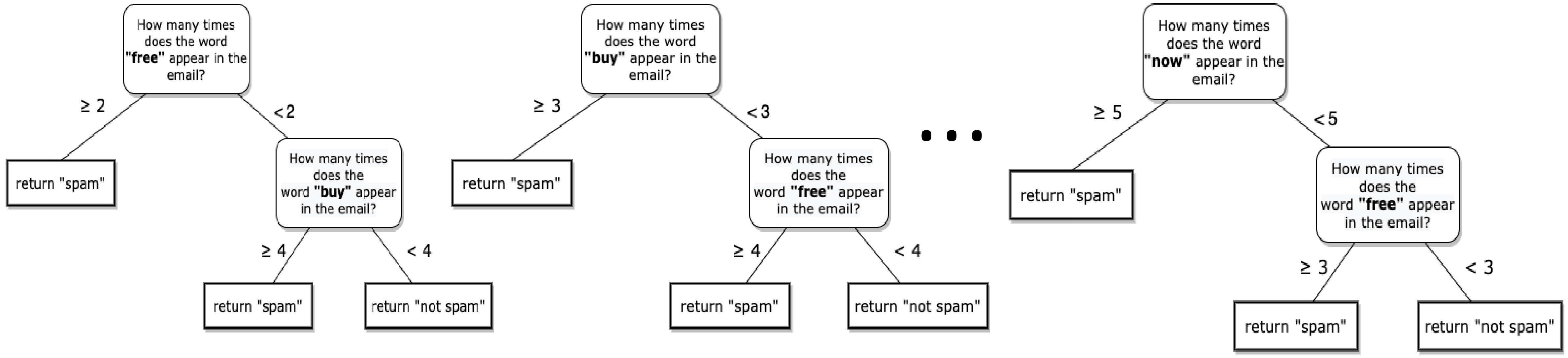
Decision Trees

- Decision trees are most often used for classification but can also be used for regression (regression trees)
- Classification based on decision rules learned from input features
- Advantages:
  - Not sensitive to scale (less preprocessing)
  - Easy to interpret
- Disadvantages:
  - Deep trees tend to have low bias (high accuracy) but high variance (unstable).
  - Often inaccurate compared to other predictors trained with the same data (can improve with Random Forest)


How many times does the word "free" appear in the email?
≥ 2
< 2
return "spam"
How many times does the word "buy" appear in the email?
≥ 4
< 4
return "spam"
return "not spam"

Random Forests

- ”Averages” the result of many deep (low bias) decision trees in order to decrease variance of decision trees
- Decorrelates trees by restricting training of individual trees to random subset of the feature space
    - Results in estimator with low bias (high accuracy) and low variance (stable)

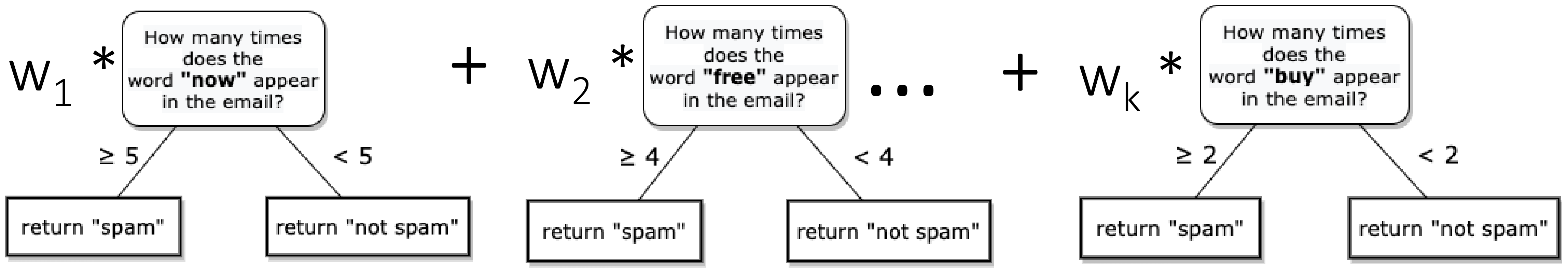


How many times does the word "free" appear in the email?
≥ 2
< 2
return "spam"
How many times does the word "buy" appear in the email?
≥ 4
< 4
return "spam"
return "not spam"
How many times does the word "buy" appear in the email?
≥ 3
< 3
return "spam"
How many times does the word "free" appear in the email?
≥ 4
< 4
return "spam"
return "not spam"
...
How many times does the word "now" appear in the email?
≥ 5
< 5
return "spam"
How many times does the word "free" appear in the email?
≥ 3
< 3
return "spam"
return "not spam"

- **<u>Example</u>**: if result of majority of the trees above is “spam” then the prediction would be ”spam”

AdaBoost

- Weighted combination of the result of multiple “weak” shallow (high bias, low variance) decision trees.
- Adaptive learning: start with all observations weighted equally to obtain first decision tree then put more weight on misclassified observations to obtain second decision tree and so on…
    - The next decision tree will focus more on the cases that the previous decision tree struggled with and focus less on the cases that were already handled well.
    - Results in a “strong” estimator with low bias (high accuracy) and low variance (stable)



W_1 *
How many times does the word **"now"** appear in the email?
≥ 5
< 5
return "spam"
return "not spam"
+ W_2 *
How many times does the word **"free"** appear in the email?
≥ 4
< 4
return "spam"
return "not spam"
...
+ W_k *
How many times does the word **"buy"** appear in the email?
≥ 2
< 2
return "spam"
return "not spam"

Further Resources

- To learn more about Decision Trees, visit:
  - https://en.wikipedia.org/wiki/Decision_tree
  - https://scikit-learn.org/stable/modules/tree.html
- To learn more about Random Forests, visit:
  - https://en.wikipedia.org/wiki/Random_forest#:~:text=Random%20forests%20or%20random%20decision,prediction%20(regression)%20of%20the%20individual
  - https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html
- To learn more about AdaBoost and other Boosting algorithms, visit:
  - https://en.wikipedia.org/wiki/AdaBoost#:~:text=AdaBoost%2C%20short%20for%20Adaptive%20Boosting,learning%20algorithms%20to%20improve%20performance.
  - https://web.stanford.edu/~hastie/TALKS/boost.pdf
  - https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.AdaBoostClassifier.html

9.2 Course Summary

Course Summary

Goal of this Section:

- Present a summary of the course material

What We Have Done!

Recall definition of Supervised Machine Learning:

- Process of learning a function that maps input information to labelled output information. The labelled input/output information is called the training data. The learned function is then used to predict outputs when new input information is provided.

This course has focused on Linear Regression, Logistic Regression and Neural Network approaches for Supervised Machine Learning

- Each of these approaches assumes a different structure for the learned function
- We have covered the math and algorithms
- We have developed a basic machine learning framework in Python to apply these approaches

Components of Supervised Learning

(1) Training Data

- Input information
- Labelled Output information

(2) Function Structure

- Defines general form of the function to be learned
- Will have unknown parameters
- Process of applying function structure is called Forward Propagation

(3) Loss Function

- Used to measure effectiveness of function structure and choice of parameters

(4) Training Phase

- Uses optimization to determine function parameters that minimize loss function for training data
- Process of computing derivatives used in optimization is called Back Propagation

(5) Prediction Phase

- Applies forward propagation using parameters determined in Training Phase to predict outputs when new input data is provided

Training Data - Summary

Case	Details
Input/Output	Assume m data points Data point j: Input information: feature vector: $\begin{bmatrix} X_{0,j} \\ X_{1,j} \\ \dots \\ X_{d-1,j} \end{bmatrix}$ assume d features Labelled output information: Y_j Define the feature matrix (dxm) and output vector (1xm): $X = \begin{bmatrix} X_{00} & \dots & X_{0,m-1} \\ \dots & \dots & \dots \\ X_{d-1,0} & \dots & X_{d-1,m-1} \end{bmatrix} \quad Y = [Y_0 \quad \dots \quad Y_{m-1}]$
Linear Regression	Y_j is a real value
Binary Classification (Logistic Regression/ Neural Network)	Y_j is 0 or 1
Multiclass Classification	Y_j is one of 0,1,…,c-1 (assume c classes)

Function Structure – Summary

Component	Details
Linear Regression	Input: X is feature matrix (dxm) Parameters: W is row vector of length d, b is scalar $Z = WX + b$ (row vector of length m) Activation function: $f(z) = z$ $A = f(Z)$ (row vector of length m)
Logistic Regression	Input: X is feature matrix (dxm) Parameters: W is row vector of length d, b is scalar $Z = WX + b$ (row vector of length m) Activation function: $f(Z) = \frac{1}{1+e^{-Z}}$ (sigmoid activation) $A = f(Z)$ (row vector of length m)
Neural Network	Input: X is feature matrix (dxm) Assume N layers Parameters: $W^{[k]}, b^{[k]}$ for layers k=1,…,N For each layer k =1, …, N Linear: $Z^{[k]} = W^{[k]}A^{[k-1]} + b^{[k]}$ $A^{[0]} = X$ Activation: $A^{[k]} = f^{[k]}(Z^{[k]})$ (use sigmoid/softmax activation in layer N for binary/multiclass classification)

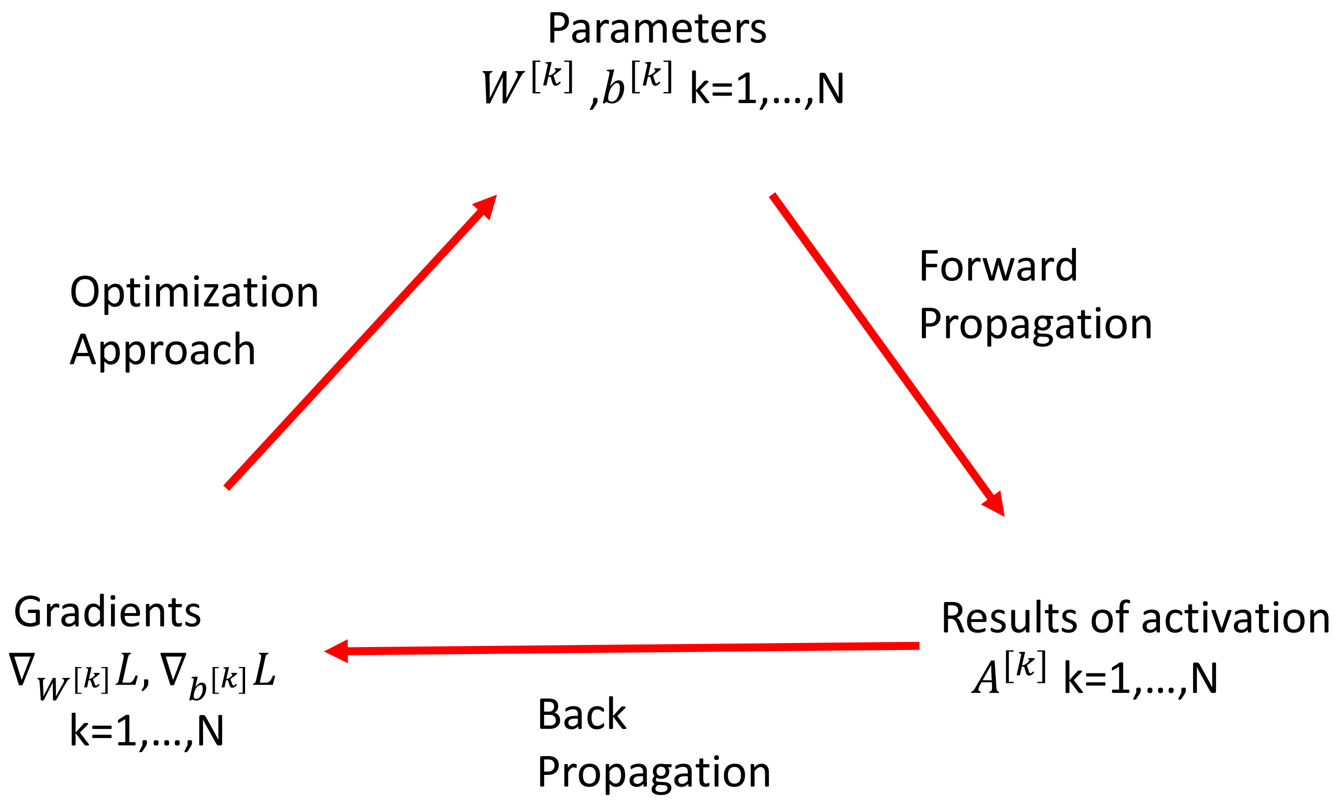
Loss Function – Summary

Component	Details
General	$Y = [Y_0 \quad \dots \quad Y_{m-1}]$ is training output information (m data points) $Y^h = [Y_0^h \quad \dots \quad Y_{m-1}^h]$ is the one-hot matrix (m one-hot vectors for mult-iclass classification) $A = [A_0 \quad \dots \quad A_{m-1}]$ is output of forward propagation for Linear Regression/Logistic Regression $A^{[N]} = \left[A_0^{[N]} \quad \dots \quad A_{m-1}^{[N]}\right]$ is output for the final layer of forward propagation for Neural Network (binary) $A_{ij}^{[N]}$ i =0,..,c-1 and j=0,...,m-1 is output for the final layer of forward propagation for Neural Network (multi-class)
Linear Regression Mean Squared Error	$L = \frac{1}{m}\sum_{j=0}^{m-1}(A_j - Y_j)^2$
Logistic Regression/ Neural Network: Binary Cross Entropy	$L = -\frac{1}{m}\sum_{j=0}^{m-1} Y_j \ln A_j^{[N]} + (1 - Y_j)\ln(1 - A_j^{[N]})$
Neural Network: Cross Entropy (for multi-class classification)	$L = -\frac{1}{m}\sum_{j=0}^{m-1}\sum_{i=0}^{n^{[N]}-1} Y_{ij}^h \ln A_{ij}^{[N]}$
L2 Regularization Penalty Term	$P = \sum_{k=1}^{N} \lambda^{[k]} \sum_{i=0}^{n^{[k]}-1} \sum_{j=0}^{n^{[k-1]}-1} \left(W_{ij}^{[k]}\right)^2$ added to loss function

Back Propagation – Summary

Component	Details
Input	Feature matrix: X Value vector: Y or Y^h (one-hot matrix in case of multiclass classification)
Compute gradients of loss with respect to $W^{[k]}$ and $b^{[k]}$	Perform Forward Propagation to compute $A^{[k]}$ for k=1,…,N Compute $\nabla_{A^{[N]}} L$ For k = N,…,1 • Compute $\nabla_{Z^{[k]}} L$ • $\nabla_{W^{[k]}} L = \nabla_{Z^{[k]}} L A^{[k-1]^T} + 2\lambda^{[k]} W^{[k]}$ (includes penalty term for L2 regularization) • $\nabla_{b^{[k]}} L_i = \sum_{j=0}^{m-1} \nabla_{Z^{[k]}} L_{ij}\,,\quad i = 0, \dots, n^{[k]} - 1$ • If k>1:$\nabla_{A^{[k-1]}} L = W^{[k]T} \nabla_{Z^{[k]}} L$

Training Algorithm



- Training Algorithm is used to find parameters $W^{[k]}, b^{[k]}$ k=1,…,N that minimize Loss function
- With initial $W^{[k]}, b^{[k]}$ k=1,…,N, use Forward Propagation to compute $A^{[k]}$ k=1,…,N
- Use Back Propagation to compute gradients
- Use Optimization Approach to update parameters:
  - Update is a function of gradients and depends on choice of optimizer
  - $W^{[k]} \leftarrow W^{[k]} + Update(W^{[k]})$
  - $b^{[k]} \leftarrow b^{[k]} + Update(b^{[k]})$
- Process is repeated

Prediction Algorithm – Summary

Component	Details
Input	New input feature matrix $\tilde{X}$ (d features x p samples)
Linear Regression	Perform Forward Propagation to compute $\tilde{A}$ (1xp), the prediction for values
Binary Classification Logistic Regression/ NeuralNetwork	Perform Forward Propagation to compute $\tilde{A}^{[N]}$ (1xp) Round values to 0 if less than 0.5 and to 1 if greater than or equal to 0.5 to get predicted labels
Multiclass Classification	Perform Forward Propagation to compute $\tilde{A}^{[N]}$(c classes x p samples) Predicted class label for each sample is row index of $\tilde{A}^{[N]}$ with largest value

Overview of Machine Learning Approach

Component	Details
Data	Split data into training, validation, and test sets
Set up Neural Network	Pick number of layers Units per layer Activation functions
Train	Perform training - monitoring accuracy for the training and validation data sets
Underfitting	In case of underfitting: increase complexity of neural network (add more layers/units)
Overfitting	In case of overfitting: add regularization or additional training data or reduce complexity of neural network
Hyperparameter Search	Perform training and monitor accuracy for range of hyperparameters (regularization parameters, number of units and layers, optimization parameters) to determine best parameter set
Final test	Train using "optimal" parameters from hyperparameter search and compute accuracy using test set
Prediction	Use results of training with optimal parameters for prediction

9.3 Suggestions for Further Study and Thank You

Suggestions for Further Study and Thank You

Goal of this Section:

- Present suggestions for further study and thank you

Further Study: Other Areas of Machine Learning

- This course has focused on Supervised Machine Learning and has looked in detail at applications:
  - House Price Prediction (regression)
  - Spam Filter (binary classification)
  - Digits Classification (multiclass classification)
- Unsupervised Machine Learning
  - Clustering
  - Data Mining
- Reinforcement Learning
  - Game Playing
  - Industrial Control

Further Study: Supervised Learning

- Data
  - Feature Preprocessing and Engineering
  - Natural Language Processing - how to convert words to feature matrix
  - Image Classification – data augmentation
- Function Structures
  - Recurrent Neural Networks for Natural Language Processing
  - Convolutional Neural Networks for Image Classification
- Training Algorithms
  - Stopping criterion
  - Regularization
  - Optimization techniques
- Applications:
  - Classification and Regression
- Machine Learning Frameworks and Packages
  - Tensorflow, Pytorch, FastAI, sklearn, etc

Machine Learning Resources

- Online Courses
  - Coursera
  - Udemy
- Online Resources – Many Blogs and Tutorials
  - Medium
  - Machine Learning Mastery
  - Towards Data Science
  - Analytics Vidhya
- Sources of Data for Practice
  - Kaggle website (Data Science Competitions)
    - www.kaggle.com
    - Many datasets
    - Collaborative community – many tutorials and publicly available Jupyter notebooks available for learning purposes
  - UCI (University of California, Irvine) Machine Learning Repository
    - https://archive.ics.uci.edu/ml/index.php
    - 100s of datasets

Thank You

- Thank you for taking this course
- I hope you feel that it was a worthwhile use of your time
- I hope it has inspired you to further your education in and practice of machine learning