CSCE 590-1: From Data to Decisions with Open Data: A Practical Introduction to AI

Lecture 8: Supervised Machine Learning

PROF. BIPLAV SRIVASTAVA, AI INSTITUTE

4TH FEB, 2021

Carolinian Creed: "I will practice personal and academic integrity."

Organization of Lecture 8

- Introduction Segment
  - Recap of Lecture 7
- Main Segment
  - A plethora of methods
    - Naïve Bayes Method
    - Gradient Tree Boosting
    - Neural Network – MLP
    - Metrics: ROC/ AUC
  - Paper discussion: 10 tips
  - What's next: choosing a method that works
    - Mostly an art
    - Paper: Data-driven advice for applying machine learning to bioinformatics problems
- Concluding Segment
  - About Next Lecture – Lecture 9, 10
  - Ask me anything

Introduction Segment

Recap of Lecture 7

- Reviewed Quiz, Project and Topic Schedule
- Supervised ML
  - Review datasets
  - Review Weka
  - Decision trees/ random forest

Main Segment

Machine Learning – Insights from Data

- Descriptive analysis
  - Describe a past phenomenon
  - **Methods**: classification, clustering, dimensionality reduction, anomaly detection, neural methods
- Predictive analysis
  - Predict about a new situation
  - **Methods**: time-series, neural networks
- Prescriptive analysis
  - What an agent should do
  - **Methods**: simulation, reinforcement learning, reasoning

- New areas
  - Counterfactual analysis
  - Causal Inferencing
  - Scenario planning

Classifier Method Types

- Individual methods
  - Decision Tree
  - Naïve Bayes

- Ensemble
  - Bagging: Aggregate classifiers (“bootstrap aggregation” => bagging)
    - Random Forest
    - Samples are chosen with replacement (bootstrapping), and combined (aggregated) by taking their average
  - Gradient Boosting: aggregate to turn weak learners into strong learners
    - Boosters (aggregators) turn weak learners into strong learners by focusing on where the individual weak models (decision trees, linear regressors) went wrong
    - Gradient Boosting
    - XGBoost: “eXtreme Gradient Boosting.”

Source:
- Data Mining: Concepts and Techniques, by Jiawei Han and Micheline Kamber
- https://towardsdatascience.com/getting-started-with-xgboost-in-scikit-learn-f69f5f470a97

Naïve Bayes Classifier

Notation:
Class variable y and
dependent feature vector x_1 through x_n

Bayes assumption: given the value of the class variable, every pair of features are conditionally independent

$$P(y \mid x_1, \dots, x_n) = \frac{P(y) P(x_1, \dots, x_n \mid y)}{P(x_1, \dots, x_n)}$$

Using the naive conditional independence assumption that

$$P(x_i | y, x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_n) = P(x_i | y),$$

for all i, this relationship is simplified to

$$P(y \mid x_1, \dots, x_n) = \frac{P(y) \prod_{i=1}^{n} P(x_i \mid y)}{P(x_1, \dots, x_n)}$$

Since $P(x_1, \dots, x_n)$ is constant given the input, we can use the following classification rule:

$$P(y \mid x_1, \dots, x_n) \propto P(y) \prod_{i=1}^{n} P(x_i \mid y)$$

$$\Downarrow$$

$$\hat{y} = \arg\max_y P(y) \prod_{i=1}^{n} P(x_i \mid y),$$

Source: https://scikit-learn.org/stable/modules/naive_bayes.html

Boosting Methods

- Concepts
  - **Weak learner**: a classifier that is only slightly correlated with the true classification
    - label examples better than random guessing
  - **Strong learner**: a classifier that is (arbitrarily) well-correlated with the true classification.
- Boosting
  - "Convert weak learners to strong learners"
  - Adapt[at]ive Resampling and Combining algorithm

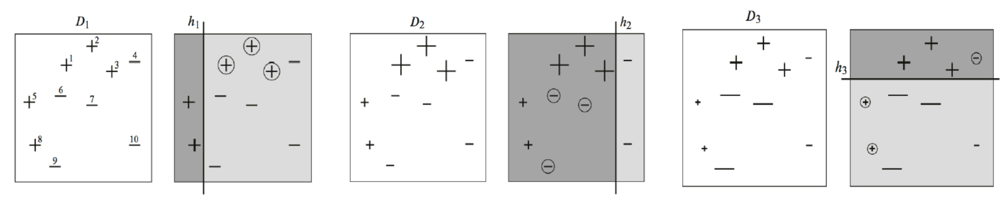


Figure: AdaBoost. Source: Figure 1.1 of [Schapire and Freund, 2012]

Source: https://en.wikipedia.org/wiki/Boosting_(machine_learning)

Image Courtesy: Prof. Cheng Li

Boosting Methods

- Fit an additive model (ensemble) $\sum_t \rho_t h_t(x)$ in a forward stage-wise manner.
- In each stage, introduce a weak learner to compensate the shortcomings of existing weak learners.
- In Adaboost, "shortcomings" are identified by high-weight data points.

Gradient Boosting = Gradient Descent + Boosting

Adaboost

$$H(x) = \sum_t \rho_t h_t(x)$$

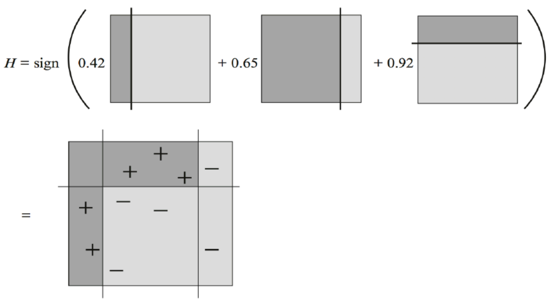


Figure: AdaBoost. Source: Figure 1.2 of [Schapire and Freund, 2012]

Content and Image Courtesy: Prof. Cheng Li
https://www.ccs.neu.edu/home/vip/teach/MLcourse/4_boosting/slides/gradient_boosting.pdf

Boosting Methods

AdaBoost,
Illustration: for binary classification, images

1.Form a large set of simple features
2. Initialize weights for training images
3. For T rounds
    1. Normalize the weights
    2. For available features from the set, train a classifier using a single feature and evaluate the training error
    3. Choose the classifier with the lowest error
    4. Update the weights of the training images: increase if classified wrongly by this classifier, decrease if correctly

4. Form the final strong classifier as the linear combination of the T classifiers (coefficient larger if training error is small)

Source: https://en.wikipedia.org/wiki/Boosting_(machine_learning)

Gradient Boosting = Gradient Descent + Boosting

Adaboost

$$H(x) = \sum_t \rho_t h_t(x)$$

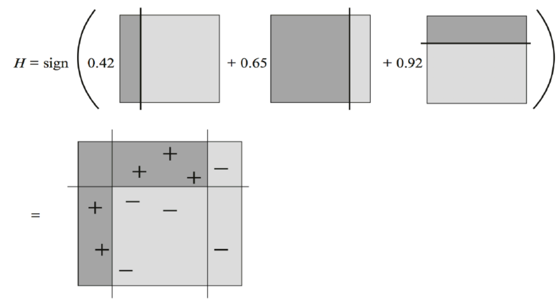


Figure: AdaBoost. Source: Figure 1.2 of [Schapire and Freund, 2012]

Image Courtesy: Prof. Cheng Li
https://www.ccs.neu.edu/home/vip/teach/MLcourse/4_boosting/slides/gradient_boosting.pdf

Boosting Methods

Gradient Boosting = Gradient Descent + Boosting

Gradient Boosting

- Fit an additive model (ensemble) $\sum_t \rho_t h_t(x)$ in a forward stage-wise manner.
- In each stage, introduce a weak learner to compensate the shortcomings of existing weak learners.
- In Gradient Boosting, "shortcomings" are identified by gradients.
- Recall that, in Adaboost, "shortcomings" are identified by high-weight data points.
- Both high-weight data points and gradients tell us how to improve our model.

Gradient Boosting = Gradient Descent + Boosting

Adaboost

$$H(x) = \sum_t \rho_t h_t(x)$$

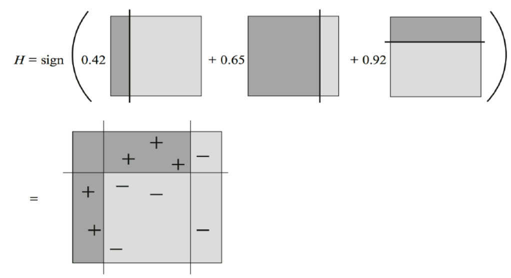
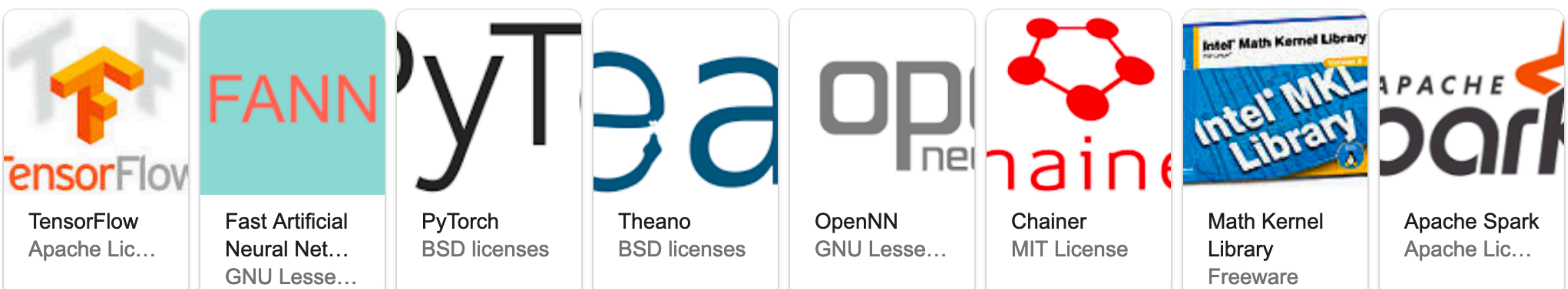


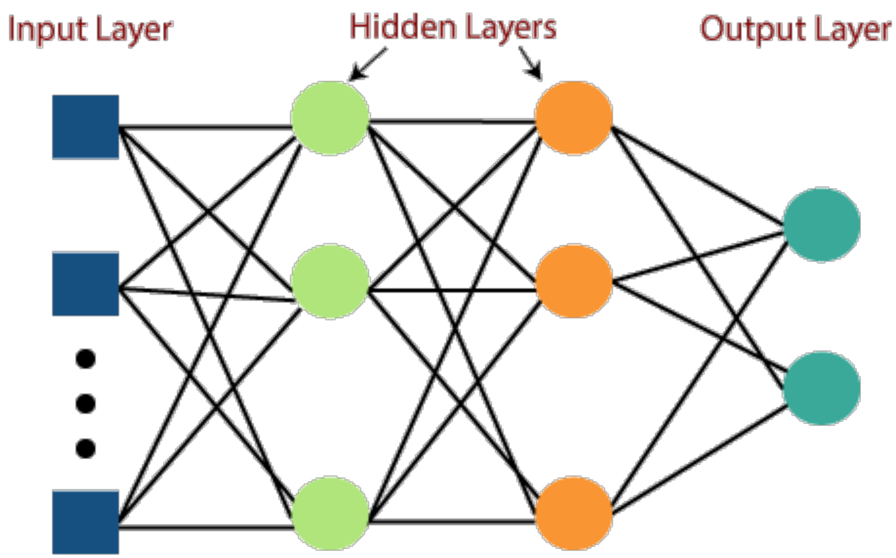
Figure: AdaBoost. Source: Figure 1.2 of [Schapire and Freund, 2012]

Content and Image Courtesy: Prof. Cheng Li
https://www.ccs.neu.edu/home/vip/teach/MLcourse/4_boosting/slides/gradient_boosting.pdf

Neural Network Methods



NN – Multi Layer Perceptron



Content and Image Courtesy:
https://github.com/Thanasis1101/MLP-from-scratch

Logistic Regression in a Slide

<u>Function estimate (linear)</u>
W: weight, b: bias

$$f(X_j) = X_j W + b$$

<u>Update Weight</u>

$$W^* = W - \eta \frac{dL}{dW}$$

<u>Error Term (mean squared error)</u>

$$MSE = \frac{1}{n} \sum_{j=1}^{n} \left[f(X_{j\cdot}) - y_j \right]^2$$

<u>Common Code Pattern</u>

```
y = tf.matmul(x, W) + b
loss = tf.reduce_mean(tf.square(y - y_label))
```

Keras and TensorFlow

- By Example:
  - https://github.com/biplav-s/course-nl/blob/master/l9-ml-review/Basic%20TensorFlow%20and%20Keras.ipynb
- TensorFlow's NMIST tutorial
  - https://www.tensorflow.org/tutorials/quickstart/beginner
- More examples
  - Number Addition by sequence learning: https://keras.io/examples/nlp/addition_rnn/
  - AutoEncoder: https://machinelearningmastery.com/lstm-autoencoders/

Review: Code on GitHub

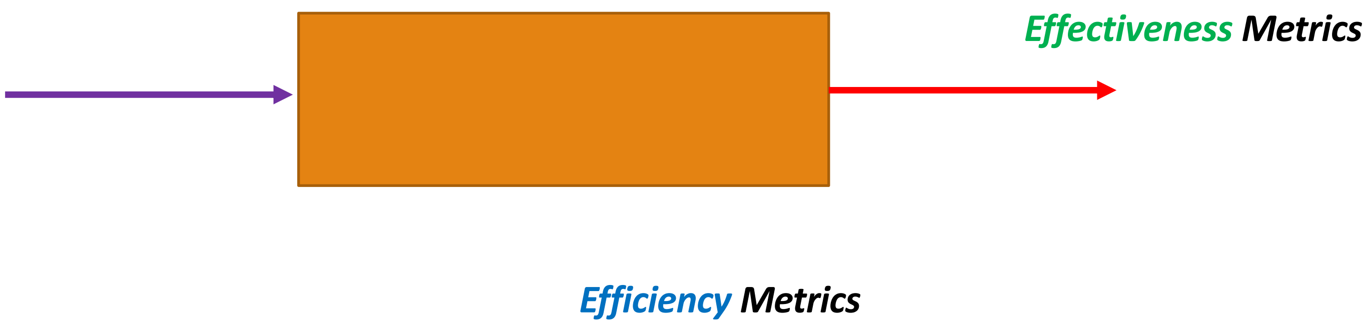
- Notebook: https://github.com/biplav-s/course-d2d-ai/blob/main/sample-code/l6-l7-l8-supervised-ml/Supervised-NaiveBayes-GradientBoost-NN-Classification.ipynb

Activity: Try Weka and Classifiers

- Naïve Bayes Method
- Gradient Tree Boosting
- Neural Network – MLP

Metric Types

- **Effectiveness**: what the **user** of a system sees, primarily cares about
- **Efficiency**: what the **executor** in a system sees, primarily cares about



Metrics: Accuracy, Precision, Recall

	Predicted class		
Actual Class		Class = Yes	Class = No
	Class = Yes	True Positive	False Negative
	Class = No	False Positive	True Negative

Accuracy =
(TP+TN)/
(TP+FP+FN+TN)

Precision =
(TP)/
(TP+FP)

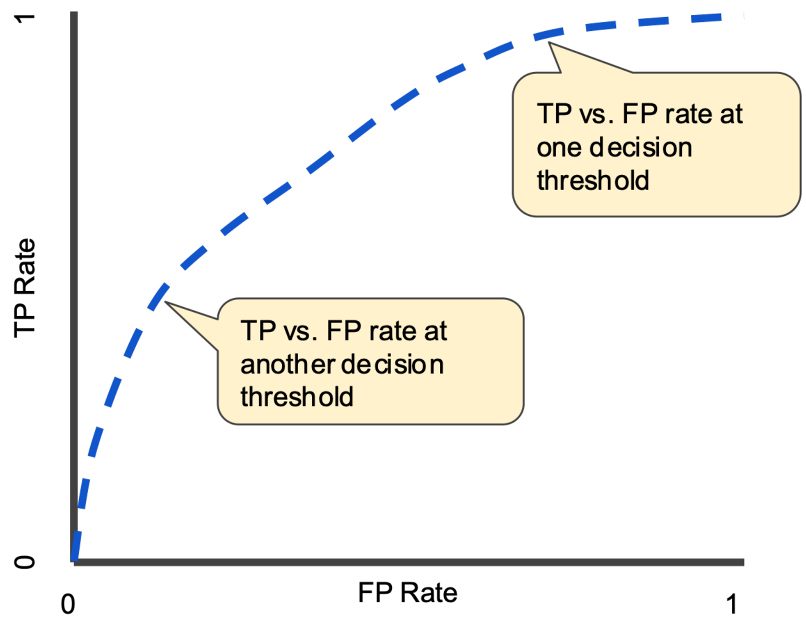
Recall =
(TP)/
(TP+FN)

F1 Score: *Harmonic Mean*

1/F1 = 1/Precision + 1/Recall

F1 = 2*(Recall * Precision) /
(Recall + Precision)

ROC – Receiver Operating Characteristic curve



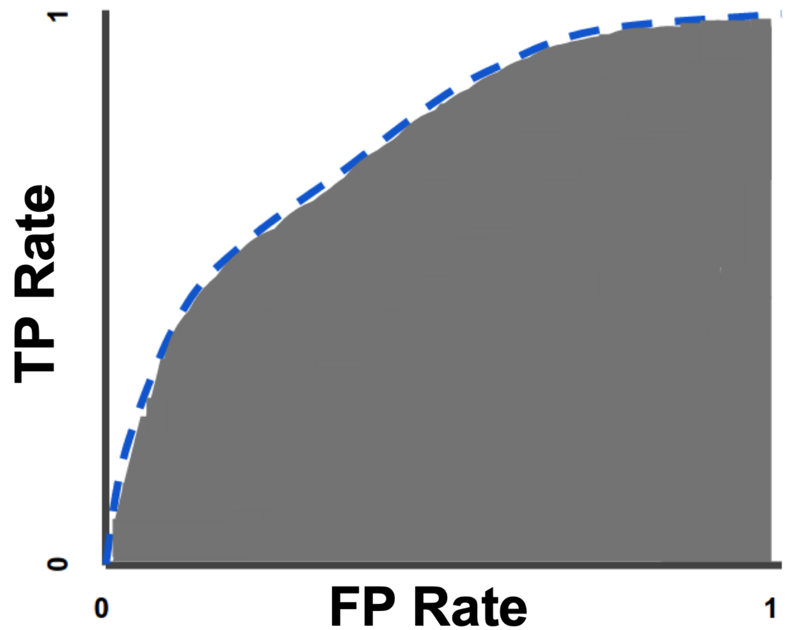
True Positive Rate = Recall =
(TP)/
(TP+FN)

False Positive Rate =
(FP)/
(FP+TN)

	Predicted class	
Actual Class	Class = Yes	Class = No
Class = Yes	True Positive	False Negative
Class = No	False Positive	True Negative

Source: https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc

AUC – Area Under the ROC Curve



- Aggregate measure of performance across all possible classification thresholds.
- Interpretation: probability that the model ranks a random positive example more highly than a random negative example

Source: https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc

References

- Blogs: https://blog.exsilio.com/all/accuracy-precision-recall-f1-score-interpretation-of-performance-measures/
- Google: https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc

Discussion: 10 Tips Paper

- Access: https://biodatamining.biomedcentral.com/articles/10.1186/s13040-017-0155-3
- Chicco, D. Ten quick tips for machine learning in computational biology. *BioData Mining* **10,** 35 (2017). https://doi.org/10.1186/s13040-017-0155-3

The Tips

- Tip 1: Check and arrange your input dataset properly
- Tip 2: Split your input dataset into three independent subsets (training set, validation set, test set), and use the test set only once you complete training and optimization phases
- Tip 3: Frame your biological problem into the right algorithm category
- Tip 4: Which algorithm should you choose to start? The simplest one!
- Tip 5: Take care of the imbalanced data problem
- Tip 6: Optimize each hyper-parameter
- Tip 7: Minimize overfitting
- Tip 8: Evaluate your algorithm performance with the Matthews correlation coefficient (MCC) or the Precision-Recall curve
- Tip 9: Program your software with open source code and platforms
- Tip 10: Ask for feedback and help to computer science experts, or to collaborative Q&A online communities

Lecture 8: Concluding Comments

- We looked at taxonomies

Concluding Segment

About Next Lecture – Lecture 9

Lecture 9: Paper Reading

- Paper reading in pairs (Which ML to use)
- Implement a couple of methods and check

Lecture 10: Unsupervised Learning

- Structured Data: Unsupervised Methods
  - Setting and characteristics
- Methods: k-means
- Working with Weka