

DS-UA 301  
Advanced Topics in Data Science  
*Advanced Techniques in ML and Deep  
Learning*

**LECTURE 3**  
**Parijat Dube**

# Today's Agenda

- ML performance metrics
  - Algorithmic
  - System-level

# Performance Metrics

- Accuracy, Loss
- Precision, Recall, F score
- Confusion matrix
- Training time, inference time
- Training cost
- Memory requirement

# Accuracy, Precision, Recall, Specificity

		True value	
		Positive	Negative
Predicted value	Positive	true positive (tp)	false positive (fp)
	Negative	false negative (fn)	true negative (tn)

$$\text{Accuracy} = \frac{tp + tn}{tp + tn + fp + fn}$$

$$\text{Precision} = \frac{tp}{tp + fp}$$

False discovery rate = 1-Precision

$$\text{Recall} = \frac{tp}{tp + fn}$$



Sensitivity, True positive rate

$$\text{True negative rate} = \frac{tn}{tn + fp}$$



Specificity

False positive rate = 1-Specificity

$$\text{Balanced accuracy} = (\text{Sensitivity} + \text{Specificity})/2$$



Considers all entries in the confusion matrix

Value lies between 0 (worst classifier) and 1 (best classifier)

# Performance metrics of a binary classifier

Consider the outcome of a test to detect presence of a radioactive element in drinking water. Given 1000 samples of drinking water from different geographies the test results are summarized as follows (with positive being the sample has radioactive element present or is contaminated). The test results are summarized in the contingency table on the right. Based on the test outcome answer following questions.

1. How many samples did the test correctly classified ?
2. What is the chance (in percentage) that a sample identified as not contaminated is in fact contaminated ?
3. What is the chance (in percentage) that a sample identified as contaminated is in fact contaminated ?
4. What is the precision, recall, and specificity for this test ?

		True value	
		Radioactive present	Radioactive absent
Test outcome	Radioactive present	100	300
	Radioactive absent	250	350

# An Example AI System

- AI-assisted tumor diagnosis for cancer detection
- “The firm’s deep-learning tool was able to correctly distinguish metastatic cancer 99% of the time, a greater accuracy rate than human pathologists.”
- 99% sounds great. What does this actually mean ?

## An Example AI System (contd.)

- Test data for performance evaluation:
  - 1,000,000 tumors in total
  - 999,000 out of 1,000,000 are *actually* benign (Actual Negatives)
  - 1,000 out of 1,000,000 are *actually* malignant (Actual Positives)
- Vendor's performance
  - Accuracy = 99%
  - Precision = 9.02%
  - Recall = 99%
- How many tumors did the system correctly classify (i.e. both True Positives or True Negatives) out of all the tumors? 99%
- What is the chance that a tumor identified by the system as malignant is *in fact* malignant"? 9%

Imbalanced dataset

		Actual Result	
		Malignant (Positive)	Benign (Negative)
Predicted Result	Malignant (Positive)	990 True positive	9,990 False positive
	Benign (Negative)	10 False negative	989,010 True negative

# Importance of Context

- Prioritizing Precision vs Recall depends on problem context
- **Recall should be optimized over precision** when there is a **high cost** associated with a **False Negative**, i.e. **system predicts benign when tumor is in fact malignant**.
- **Precision should be optimized over recall** when there is a **high cost** associated with a **False Positive**, i.e. **spam detection**.



# Balancing Precision and Recall

- F1 score: Harmonic mean of precision and recall; measure of classifier accuracy

$$F_1 = \frac{2}{\text{recall}^{-1} + \text{precision}^{-1}} = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} = \frac{\text{tp}}{\text{tp} + \frac{1}{2}(\text{fp} + \text{fn})}.$$

- $F_\beta$  score:  $\beta = 1$  is F1 score; recall is considered  $\beta$  times as important as precision

$$F_\beta = (1 + \beta^2) \cdot \frac{\text{precision} \cdot \text{recall}}{\beta^2 \cdot \text{precision} + \text{recall}}$$

# Precision and Recall

True negatives are not accounted in Precision and Recall Metrics work in situations where the correct identification of negative class is not important

## Reference



## Clinical Test 1



## Clinical Test 2



## Confusion matrix for the first test

Prediction/Reference	Diseased	Healthy
Diseased	TP = 80	FP = 10
Healthy	FN = 10	TN = 0

## Confusion matrix for the second test

Prediction/Reference	Diseased	Healthy
Diseased	TP = 70	FP = 0
Healthy	FN = 20	TN = 10

## Comparison of the two tests

Let us compare the performance of the two tests:

Measure	Test 1	Test 2
Sensitivity (Recall)	88.9%	77.7%
Specificity	0%	100%
Precision	88.9%	100%

F1 score

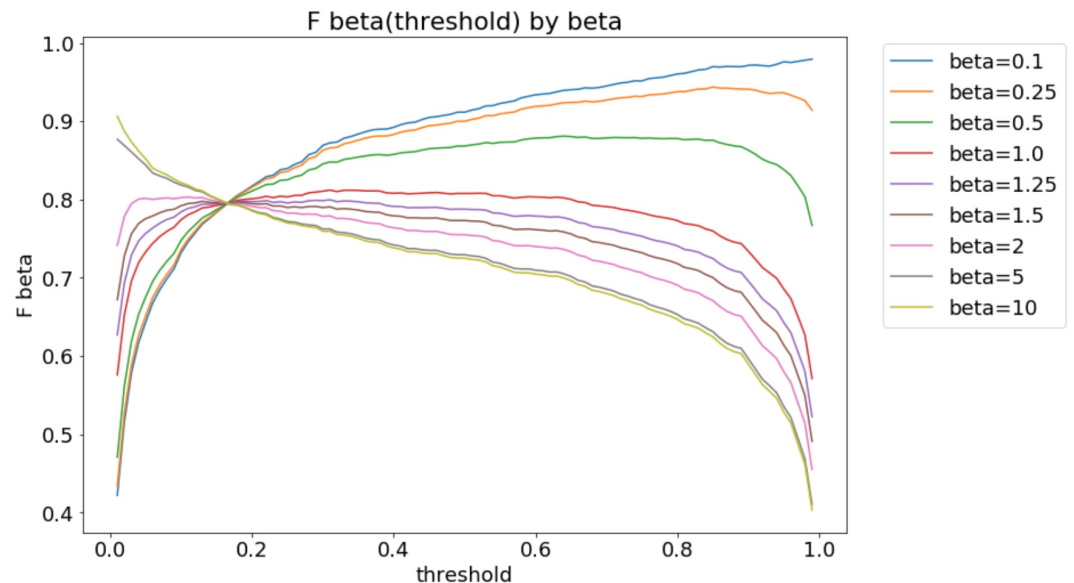
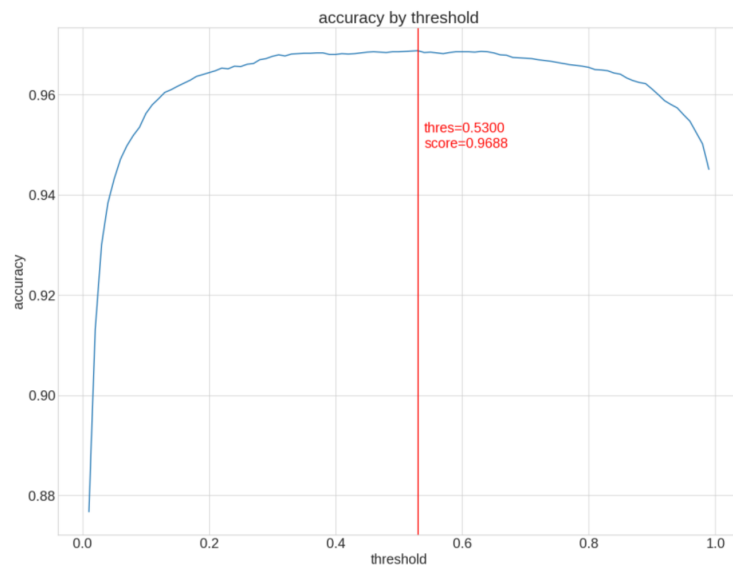
0.889

0.87

If we prefer Test 1 with higher F1 score  
 → No one healthy is identified as healthy (Specificity = 0%)

## Performance dependence on positive class threshold

- Classifier gives prediction score (the probability of belonging to positive class) and then we apply a threshold on the score to predict the class
- Metrics like accuracy, precision, recall, F-beta score are all calculated on the predicted classes and not on prediction scores
  - Metric value depends on the threshold



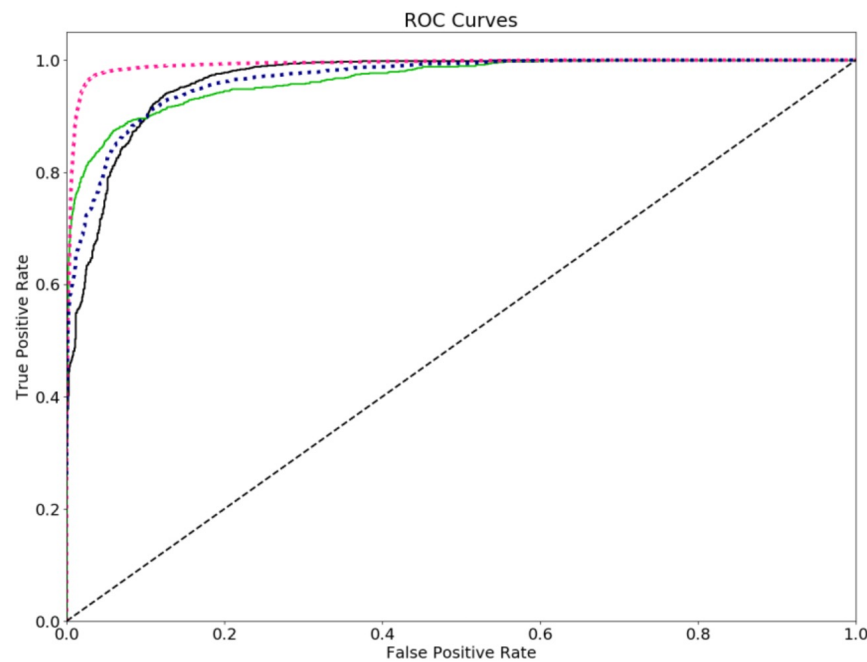
# Performance dependence on positive class threshold

ID	Actual	Prediction Probability	>0.6	>0.7	> 0.8	Metric
1	0	0.98	1	1	1	
2	1	0.67	1	0	0	
3	1	0.58	0	0	0	
4	0	0.78	1	1	0	
5	1	0.85	1	1	1	
6	0	0.86	1	1	1	
7	0	0.79	1	1	0	
8	0	0.89	1	1	1	
9	1	0.82	1	1	1	
10	0	0.86	1	1	1	
			0.75	0.5	0.5	TPR
			1	1	0.66	FPR
			0	0	0.33	TNR
			0.25	0.5	0.5	FNR

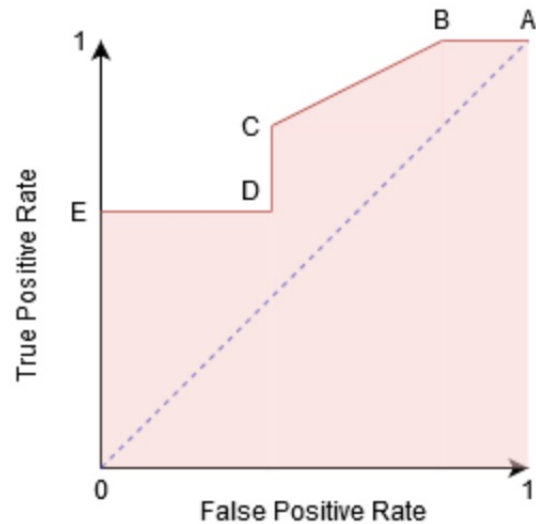
- The metrics change with the changing threshold values.
- Which threshold to choose?

# Receiver Operating Characteristics (ROC)

- Plots True positive rate (Recall) vs False positive rate at different thresholds



# AUC-ROC



A

	Actual	
	+ve	-ve
Predicted +ve	5	5
Predicted -ve	0	0

B

	Actual	
	+ve	-ve
Predicted +ve	5	4
Predicted -ve	0	1

C

	Actual	
	+ve	-ve
Predicted +ve	4	2
Predicted -ve	1	3

TPR/Sensitivity = 0.8  
FPR = 0.4  
Specificity = 1-FPR = 0.6

D

	Actual	
	+ve	-ve
Predicted +ve	3	2
Predicted -ve	2	3

TPR/Sensitivity = 0.6  
FPR = 0.4  
Specificity = 1-FPR = 0.6

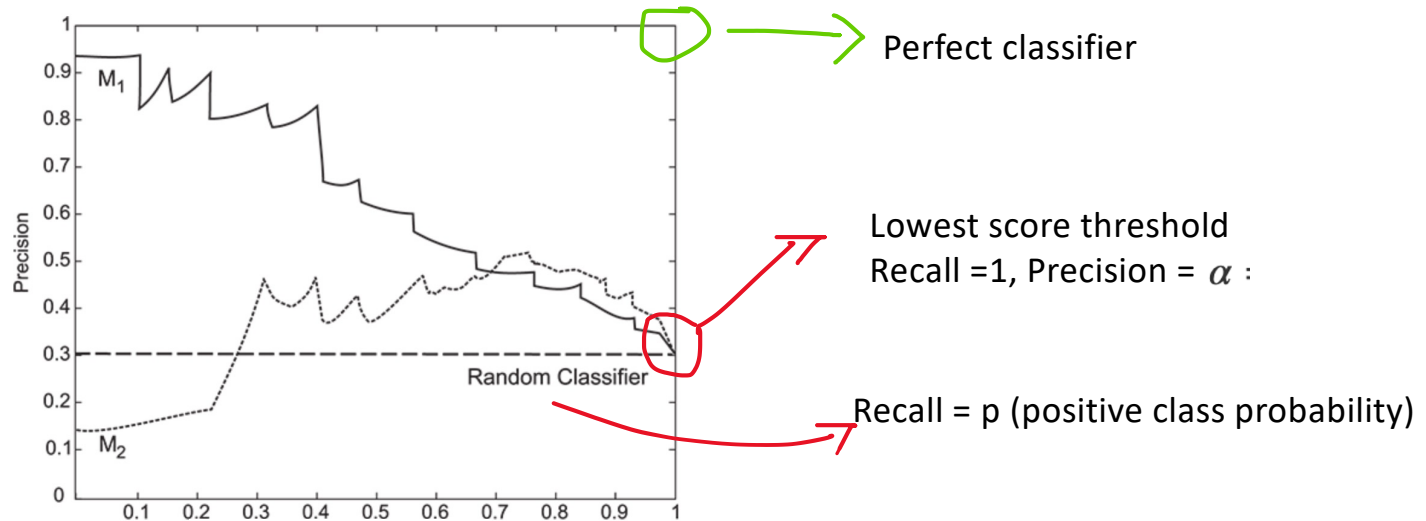
E

	Actual	
	+ve	-ve
Predicted +ve	3	0
Predicted -ve	2	5

TPR/Sensitivity = 0.6  
FPR = 0  
Specificity = 1-FPR = 1

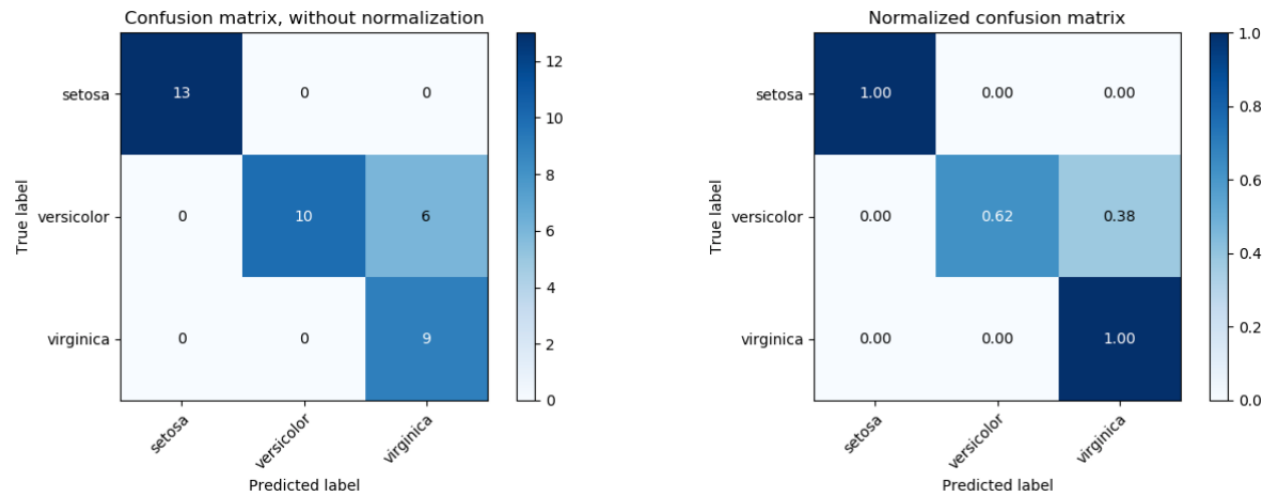
<https://www.analyticsvidhya.com/blog/2020/06/auc-roc-curve-machine-learning/>

# Precision Recall Curve (PRC)



- Sensitive to skew factor  $\alpha = P/(P + N)$ ,
- Recall monotonically declines as score threshold increases
- Precision may increase or decrease for the same value of recall
- Random baseline in PR curve depends on the skew; ROC it's the fixed diagonal line independent of skew
- AUC-PR of random classifier =  $\alpha$  ; of perfect classifier = 1

# Confusion matrix example

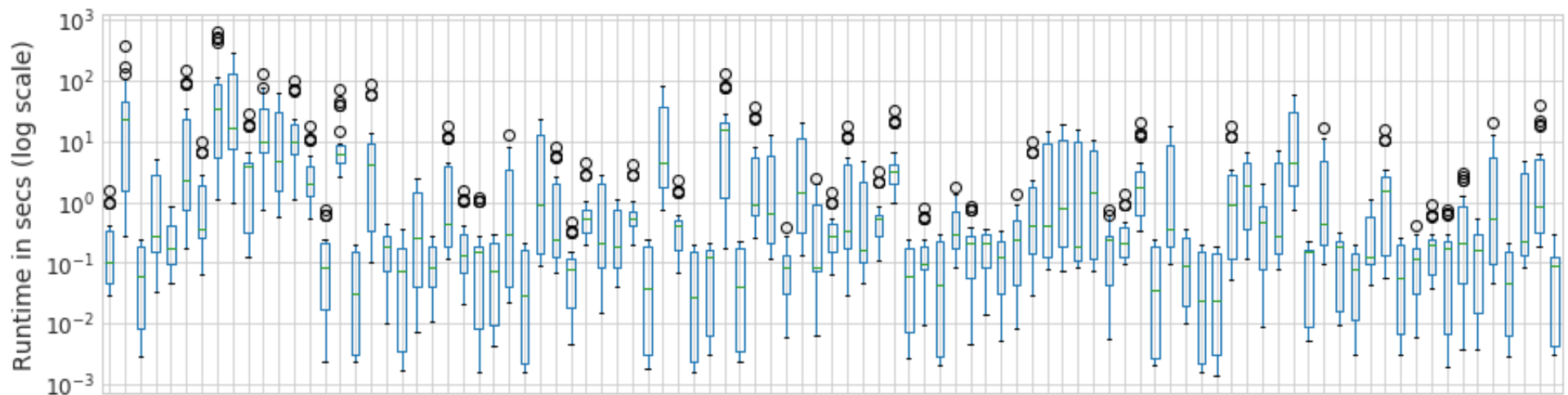


- Heat map visualization
- Not symmetric: *versicolor* confused with *virginica* not vice-versa
- Prediction is most accurate for *setosa* and *virginica*
- Helps calculate Precision and Recall for multiclass classification



# Runtime: ML

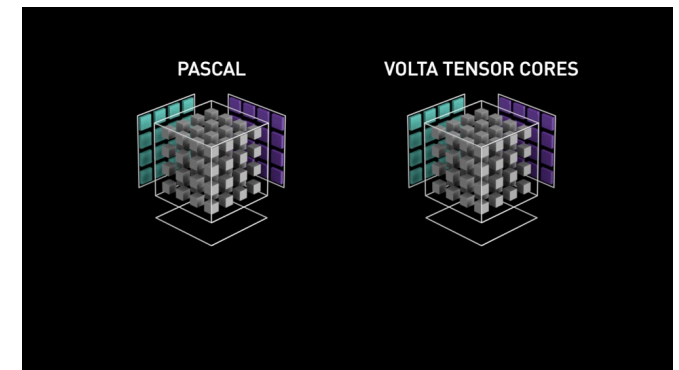
95 OpenML datasets, 7 classification algorithms, fit time



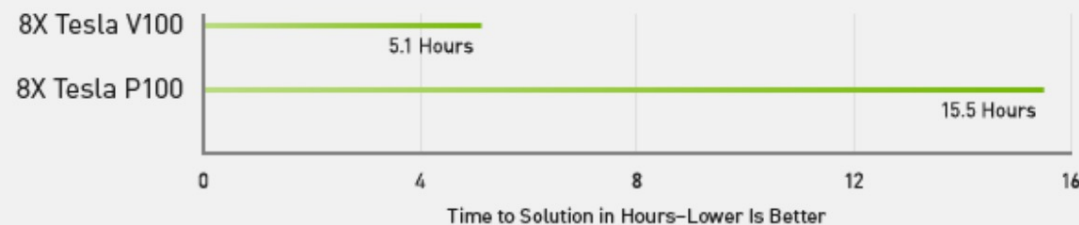
- For the same dataset, depending on the choice of classifier, the runtime can differ by orders of magnitude
- Runtime has both algorithmic, dataset, and system level dependency
- Runtime depends on how the model scales with different features of input: training data size, linear separability, complexity (number of output classes for classification problems), algorithmic hyperparameters, statistical meta-features of data (mean size per class, log number of features)

# DL Training Time

- DL performance is closely tied to the hardware
  - compute power, memory, network
  - Tesla V100: 640 tensor cores (> 100 TFLOPS), 16 GB
  - NVIDIA NVLink: 300 GB/s
  - Volta optimized CUDA libraries



## Deep Learning Training in Less Than a Workday

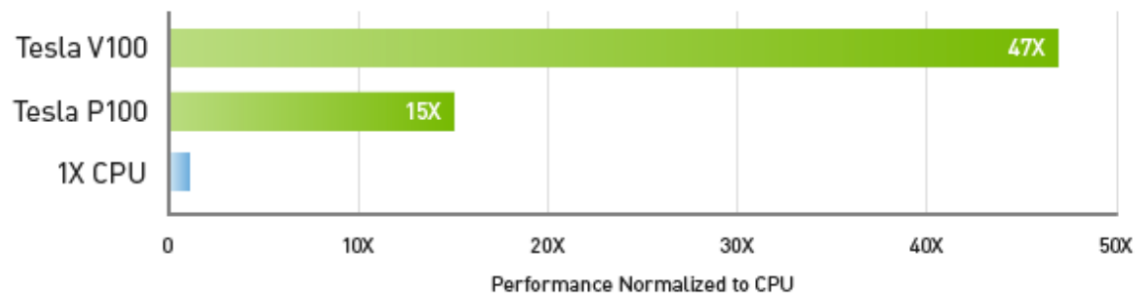


Server Config: Dual Xeon E5-2699 v4 2.6 GHz | 8X NVIDIA® Tesla® P100 or V100 | ResNet-50 Training on MXNet for 90 Epochs with 1.28M ImageNet Dataset.

- 32x faster training throughput than a CPU
- 24 faster inference throughput than a CPU

# DL Inference Throughput

47X Higher Throughput Than CPU Server on Deep Learning Inference



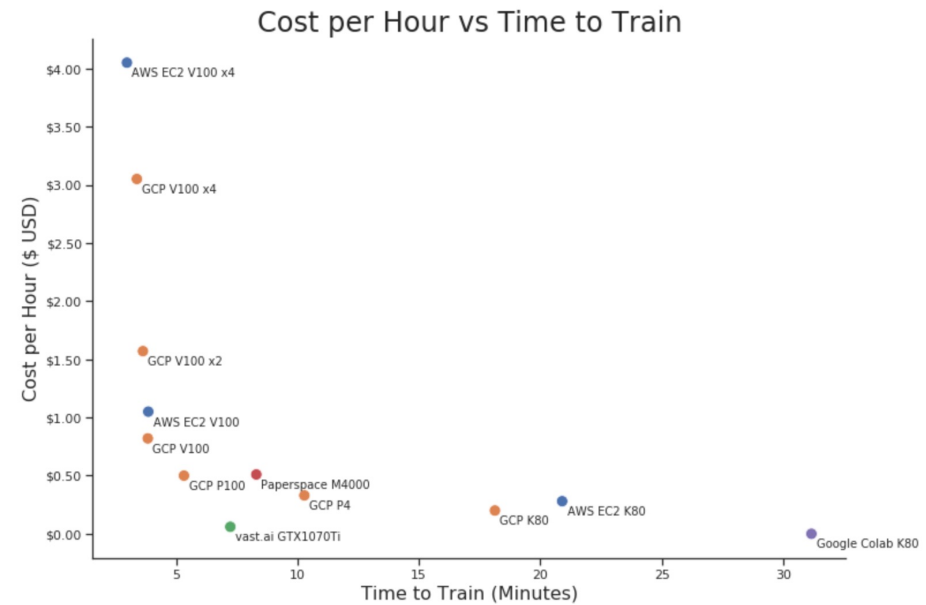
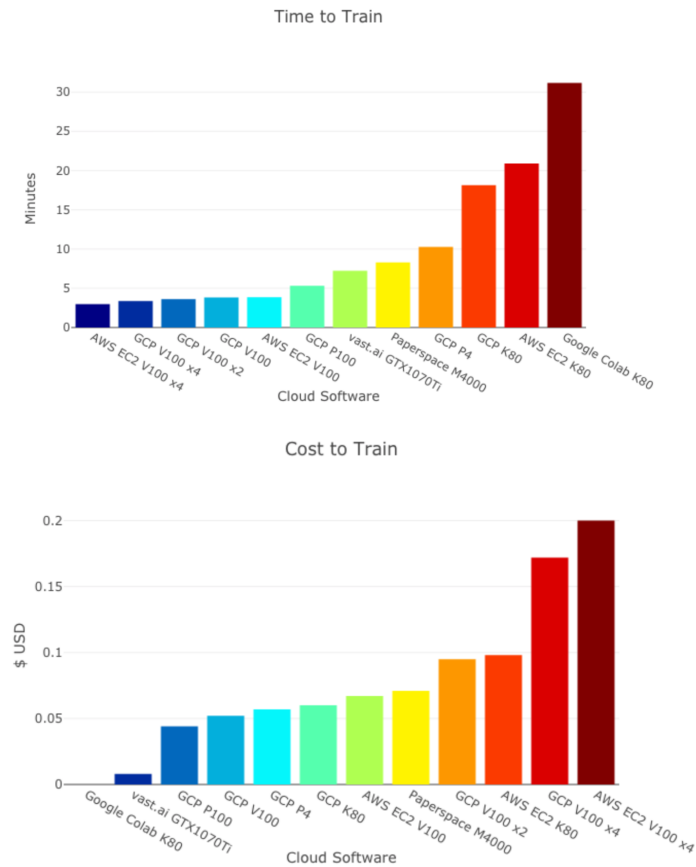
Workload: ResNet-50 | CPU: 1X Xeon E5-2690v4 @ 2.6 GHz | GPU: Add 1X Tesla P100 or V100

# Training time and cost: An example

Cloud Service	NVIDIA GPU	CPUs	GPU RAM	CPU RAM	Cost Per Hour	Wall Time	Cost to Train
Google Colab	K80	1	12	13	0.00	31.17	0.000
Google Cloud Compute Engine	P100	6	16	20	0.50	5.32	0.044
Google Cloud Compute Engine	K80	6	12	17	0.20	18.13	0.060
Google Cloud Compute Engine	V100	8	16	20	0.82	3.83	0.052
Google Cloud Compute Engine	P4	4	8	26	0.33	10.28	0.057
Google Cloud Compute Engine	V100 x 2	8	32	30	1.57	3.63	0.095
Google Cloud Compute Engine	V100 x 4	8	64	30	3.05	3.38	0.172
AWS EC2	K80 (p2.xlarge)	4	12	61	0.28	20.90	0.098
AWS EC2	K80 x 8 (p2.8xlarge)	32	96	488	2.35	16.12	0.631
AWS EC2	V100 (p3.2xlarge)	8	16	61	1.05	3.85	0.067
AWS EC2	V100 x 4 (p3.8xlarge)	64	128	488	4.05	2.97	0.200
vast.ai	GTX 1070 Ti	4	8.1	16	0.06	7.23	0.008
Paperspace	Quadro M4000	8	8	30	0.51	8.30	0.071

- Training time and cost are important when doing DL on cloud
- While runtime (Wall Time) is mostly governed by GPU type, different cloud platforms show differences (18.13 on GCP vs 20.90 on AWS EC2)
- GCP is cheaper than AWS EC2 for same GPU (compare cost with 1 K80 and 1 V100)
- Job does not scale linearly with increasing compute

# Training time and cost tradeoffs



# What Performance Metrics are Important ?

- Depends on the use case
- Model deployment on edge devices: model size is very important; model should fit into device (limited) memory
- Model deployment as a cloud service: model robustness, de-biasing, inference time
- Model development and training: training time, training cost, accuracy
- Instead of a single performance objective, performance optimization of ML models is often a *multi-objective problem*

# Multi-objective optimization

- Example: Control complexity of the model while minimizing MSE
- Control complexity:
  - Improve model Interpretability: In general, the lower the complexity, the easier it is to understand the model.
  - Prevent overfitting
- Scalarized multiobjective learning  $\min f = E + \lambda \Omega$
- Pareto based multiobjective learning

$$\begin{aligned} \min \{f_1, f_2\} \\ f_1 = E \\ f_2 = \Omega. \end{aligned} \quad \Omega = \sum_{i=1}^M w_i^2$$

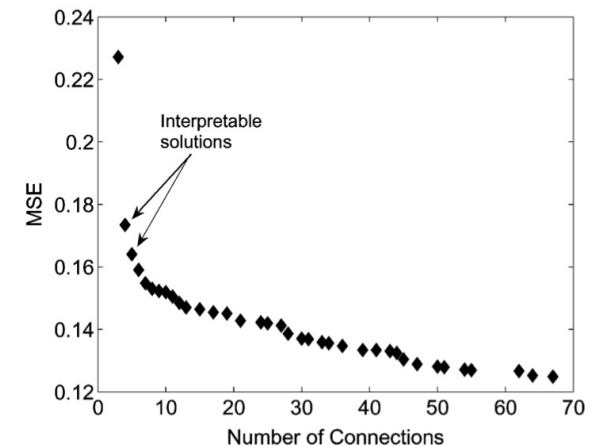
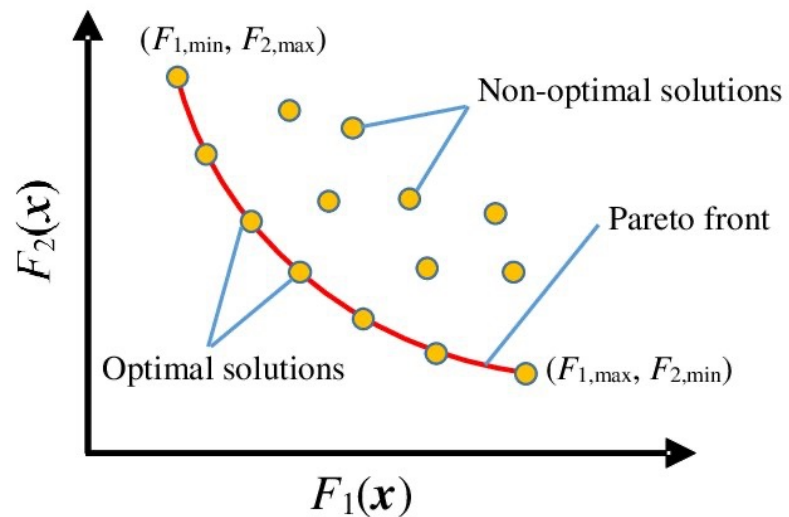


Fig. 9. Typical Pareto-front obtained for the diabetes data composed of 37 solutions.

# Pareto front

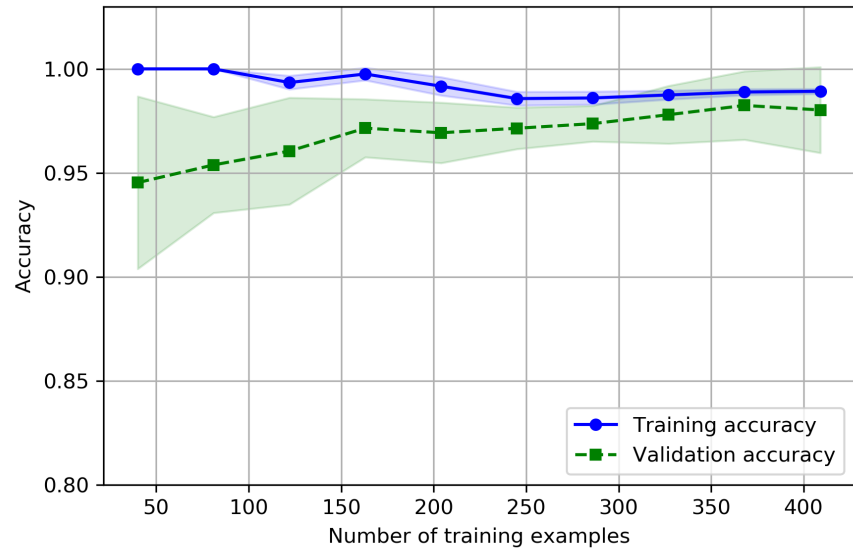
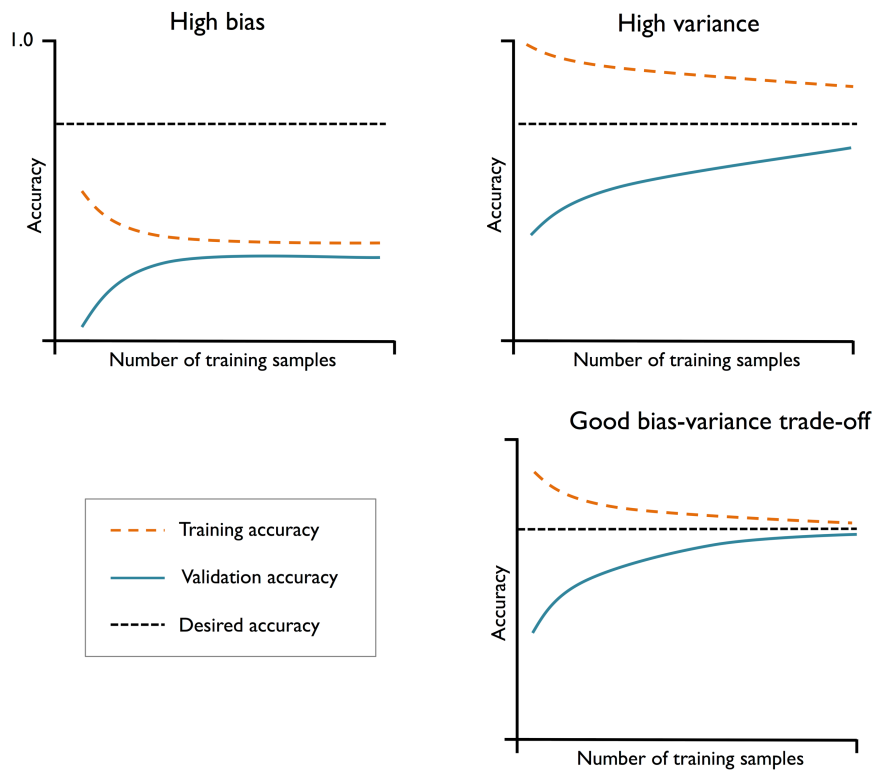


A solution  $x_1$  is said to dominate  $x_2$  if

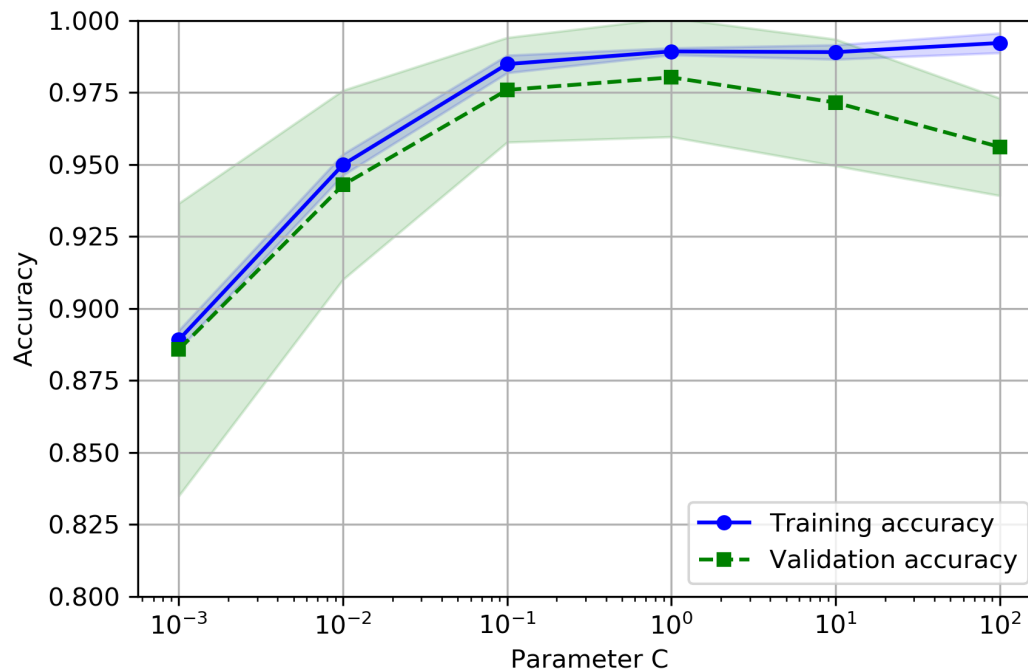
- $x_1$  is better or equal to  $x_2$  in all objectives
- $x_1$  is **strictly** better than  $x_2$  in at least one objective



# Diagnosing bias and variance problems with learning curve



# Addressing over- and under-fitting with validation curves



# Reading List

- **Model Selection**

- Sebastian Raschka. [Model Evaluation, Model Selection, and Algorithm Selection in Machine Learning](#). 2018

- **Practical ML**

- Pedro Domingos, [A few useful things to Know about machine Learning](#)
- Jeff Dale, [Best Deals in Deep Learning Cloud Providers](#), *medium article*

- **Precision-Recall**

- Jesse Davis, Mark Godarich [The Relationship Between Precision-Recall and ROC Curves](#), ICML 2006
- Blog. [The case against precision as a model selection criterion](#)
- Blog. [Beyond Accuracy: Precision and Recall](#)

- **Application Paper**

- [A prediction model of outcome of SARS-CoV-2 pneumonia based on laboratory findings](#)

# Code Links

- Model Selection: Underfitting, Overfitting, and the Bias-Variance Tradeoff

<https://dustinstansbury.github.io/theclevermachine/bias-variance-tradeoff>

- Methods for testing linear separability in Python

<http://www.tarekatwan.com/index.php/2017/12/methods-for-testing-linear-separability-in-python/#fnref-102-6>

- Jupyter notebook comparing cloud providers

<https://www.kaggle.com/discdiver/best-values-in-deep-learning-cloud-providers/>