African Institute for Mathematical Sciences
RWANDA

AMMI Review sessions

Deep Learning (2)

Machine Learning Basics (2)

14 Feb 2020

Performance evaluation metrics

- Choosing the right metric is crucial while evaluating machine learning (ML) models
- Various metrics are proposed to evaluate ML models in different applications

Performance evaluation metrics

- Choosing the right metric is crucial while evaluating machine learning (ML) models

- Various metrics are proposed to evaluate ML models in different applications

- What is the difference between the loss function and a metric ?

  - Loss functions are functions that show a measure of the model performance and are used to train a machine learning model and are usually differentiable in model's parameters.

  - metrics are used to monitor and measure the performance of a model (during training, and test), and do not need to be differentiable

  - In some tasks the performance metric is differentiable, it can be used both as a loss function and a metric, such as MSE.

Popular Performance evaluation metrics

- Popular evaluation metrics includes but not limited to:

  - *Classification Metrics (accuracy, precision, recall, F1-score, ROC, AUC, …)*
  - *Regression Metrics (MSE, MAE)*
  - *Ranking Metrics (MRR, DCG, NDCG)*
  - *Statistical Metrics (Correlation)*
  - *Computer Vision Metrics (PSNR, SSIM, IoU)*
  - *NLP Metrics (Perplexity, BLEU score)*
  - *Deep Learning Related Metrics (Inception score, Frechet Inception distance)*

Classification Metrics

- Let’s first review the **Confusion Matrix** concept:
- Tabular visualization of the model predictions versus the ground-truth labels

- True positive(TP) — Correct positive prediction
- False positive(FP) — Incorrect positive prediction
- True negative(TN) — Correct negative prediction
- False negative(FN) — Incorrect negative prediction

		Actual Values	
		Positive (1)	Negative (0)
Predicted Values	Positive (1)	TP	FP
	Negative (0)	FN	TN

Classification Metrics

- Classification metrics derived from the confusion matrix:

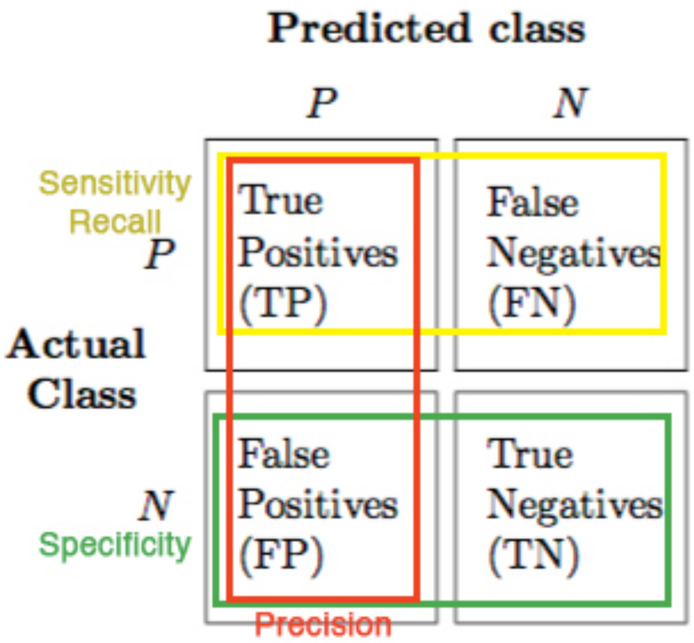
- **Accuracy** = $\frac{TP+TN}{P+N}$ how many samples predicted correctly

- **Precision**(Positive predicted value) = $\frac{true\ positives}{predicted\ postives} = \frac{TP}{TP+FP}$

  can we trust your positive predictions ?

- **Recall**/Sensitivity(True positive rate) = = $\frac{true\ positives}{actual\ postives}\ \frac{TP}{TP+FN}$

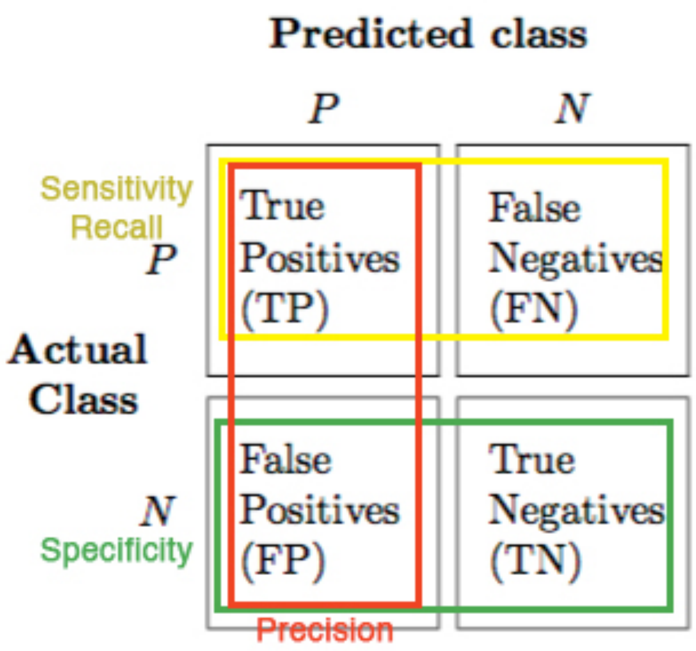
  Of all of the true samples how many of them we were able to predict correctly/ how much we missed ?



Classification Metrics

- Specificity (True negative rate) = $\frac{TN}{N}$

- F1 Score(Harmonic mean of precision and recall) =
$\frac{(1+b)*Precision\ *Recall}{b^2 Percision+Recall}$ where b is commonly 0.5, 1, 2



- Depending on application, you may want to give higher priority to recall or precision

- In medical diagnosis for example you try to avoid false negatives → requires high recall

- For a spam binary classifier you try to avoid false positives → requires high precision

Classification Metrics

- The confusion Matrix for Multi-class

- Let's say you have N classes, then your confusion matrix would be an $N \times N$ matrix

- With the left axis showing the true class, and the top axis showing the class assigned to an item with that true class.

- Each element i,j of the matrix would be the number of items with true class i that were classified as being in class j.

- You then can calculate the precision recall per class, how ?! (exercise)

Target	**Selected**									**Acc**
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	
1	**137**	13	3	0	0	1	1	0	0	0.89
2	1	**55**	1	0	0	0	0	6	1	0.86
3	2	4	**84**	0	0	0	1	1	2	0.89
4	3	0	1	**153**	5	2	1	1	1	0.92
5	0	0	3	0	**44**	2	2	1	2	0.82
6	0	0	2	1	4	**35**	0	0	1	0.81
7	0	0	0	0	0	0	**61**	2	2	0.94
8	0	0	0	1	0	0	0	**69**	3	0.95
9	0	0	0	0	0	0	0	2	**26**	0.93
										0.89

ROC Curve

- The **receiver operating characteristic curve** is plot which shows the performance of a binary classifier as function of its cut-off threshold.

- It essentially shows the true positive rate (TPR) against the false positive rate (FPR) for various threshold values.

- As an example your model may predict the below probabilities for 4 sample images: [0.45, 0.6, 0.7, 0.3].

- Then depending on the threshold values below, you will get different labels:
    - cut-off= 0.5: predicted-labels= [0,1,1,0] (default threshold)
    - cut-off= 0.2: predicted-labels= [1,1,1,1]
    - cut-off= 0.8: predicted-labels= [0,0,0,0]

ROC Curve

- For each of these 4 different thresholds we can construct a confusion matrix and calculate the TRP and FPR.
- ROC curve essentially finds out the TPR and FPR for various threshold values and plots TPR against the FPR

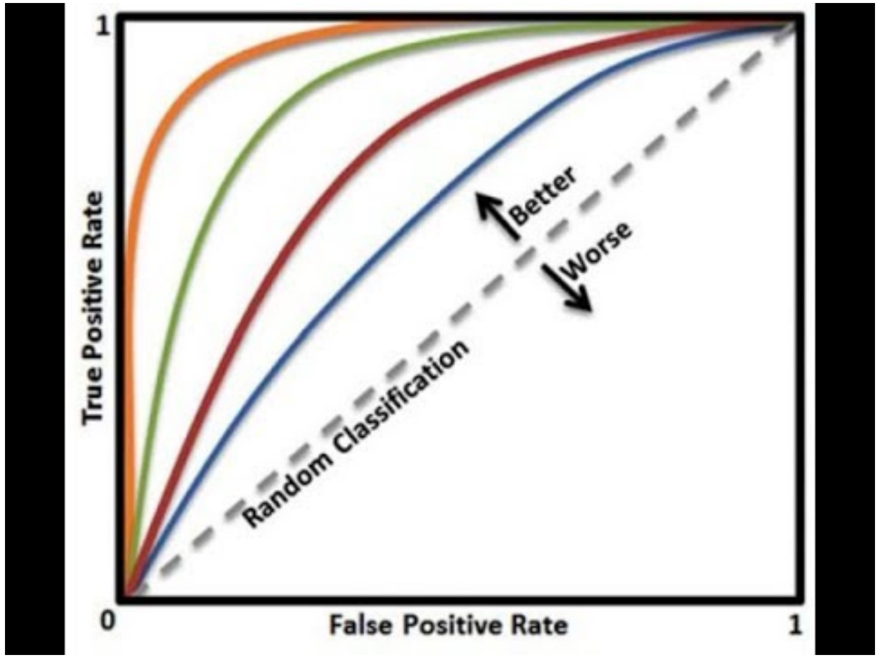
ROC Curve

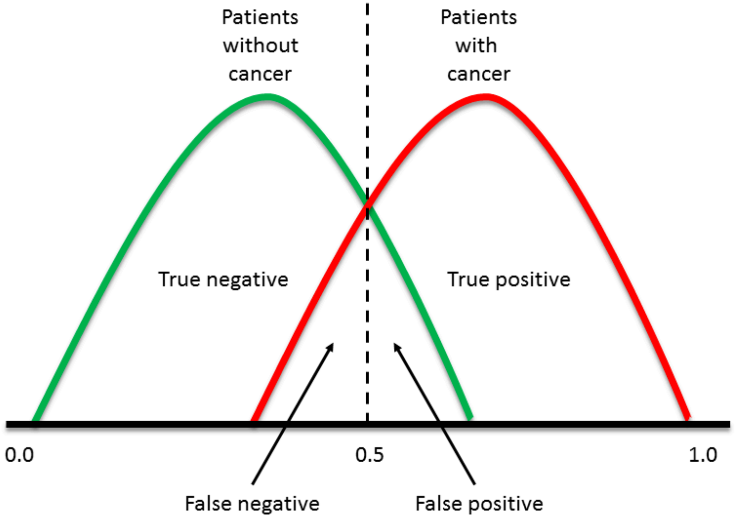
- The lower the cut-off threshold on positive class, the more samples predicted as positive class, i.e. higher true positive rate (recall) and also higher false positive rate

- Therefore, there is a trade-off between how high the recall could be versus how much we want to bound the error (FPR).

- ROC curve is a popular curve to look at overall model performance and pick a good cut-off threshold for the model.

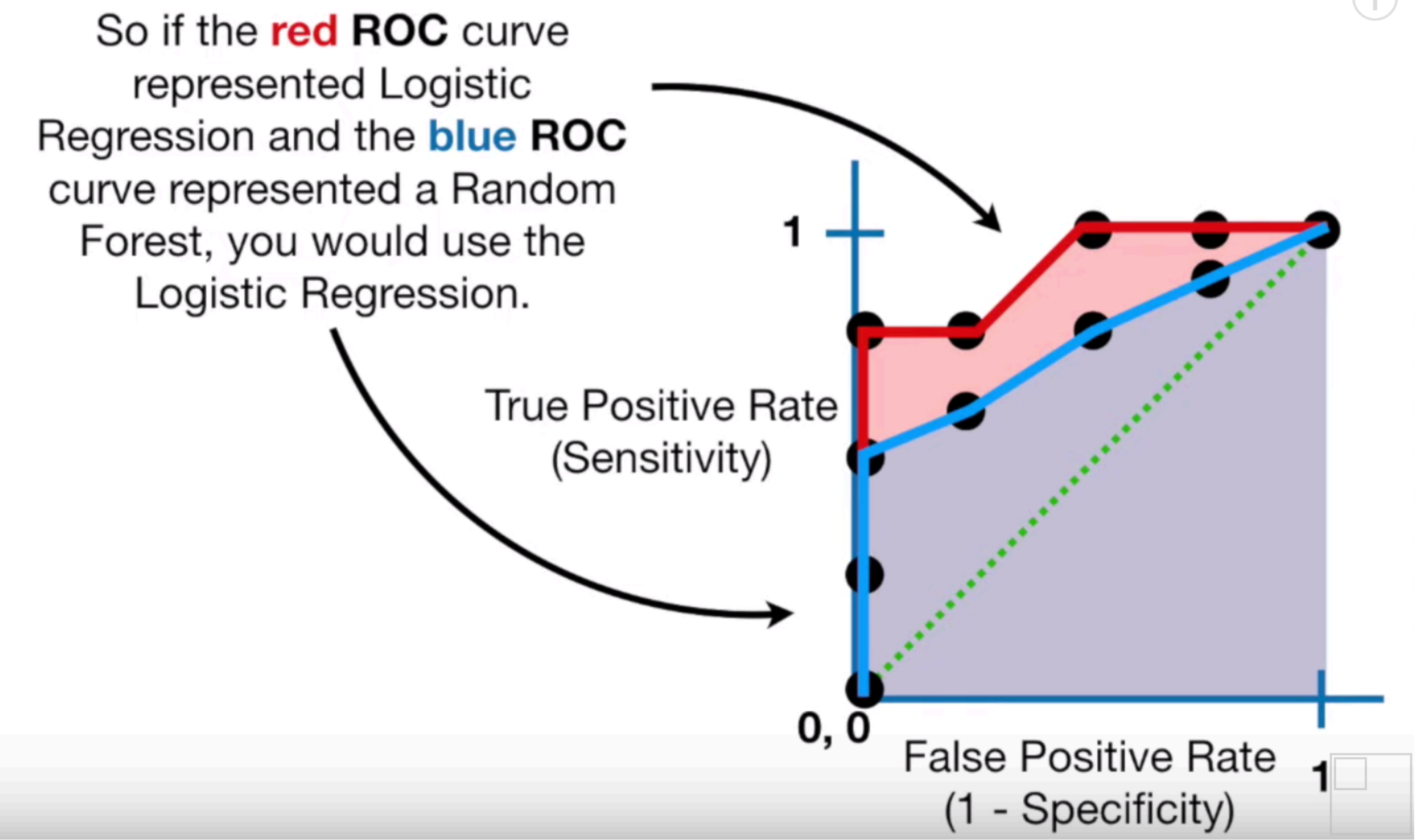
- [Detailed example]





Area Under The Curve (AUC)

- An aggregated measure of performance of a binary classifier on all possible threshold values
  - Can be used to compare the performance of two different models

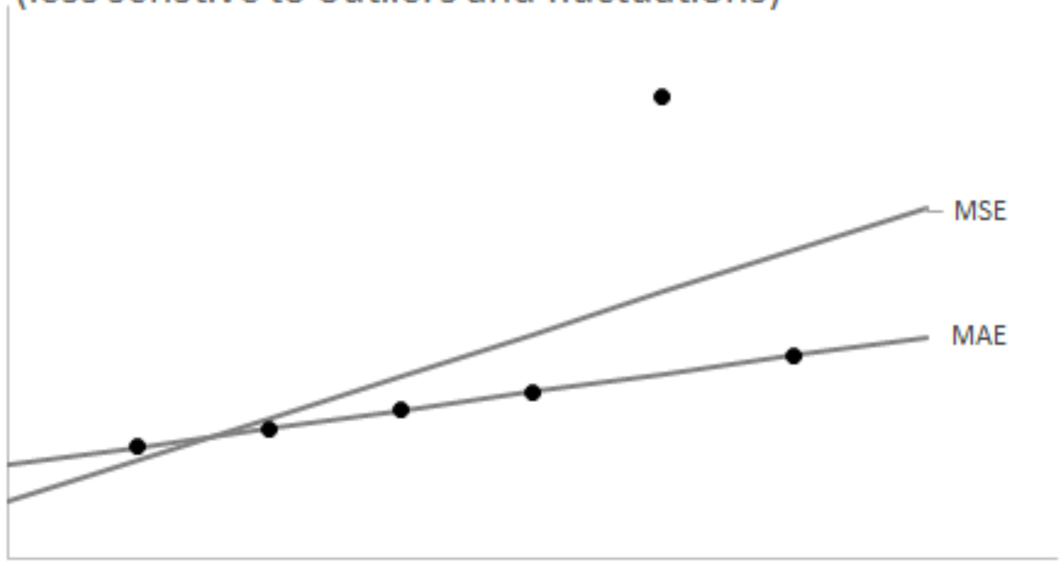


- if your application requires higher recall then you might choose a model using a point the ROC curve even if the the AUC value is not too high

Regression Related Metrics

Mean squared error	$\text{MSE} = \frac{1}{n}\sum_{t=1}^{n} e_t^2$
Root mean squared error	$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{t=1}^{n} e_t^2}$
Mean absolute error	$\text{MAE} = \frac{1}{n}\sum_{t=1}^{n} \lvert e_t \rvert$

The MAE is more robust than the MSE
(less senstive to outliers and fluctuations)



Pearson's Correlation Coefficient

- Correlation is a technique for investigating the relationship between two quantitative, continuous variables, for example, age and blood pressure.

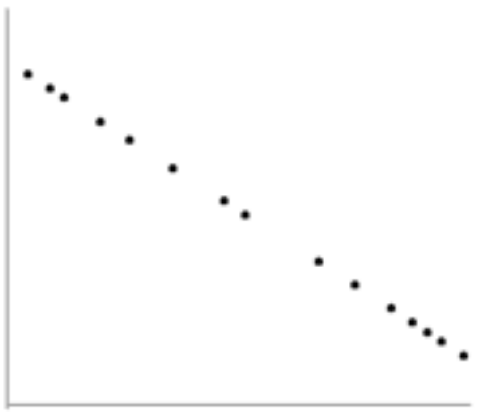
- Pearson's correlation coefficient (r) is a **measure of the strength of the association** between the two variables.

- The first step in studying the relationship between two continuous variables is to draw a scatter plot of the variables to check for linearity.

- The nearer the scatter of points is to a straight line, the higher the strength of association between the variables.

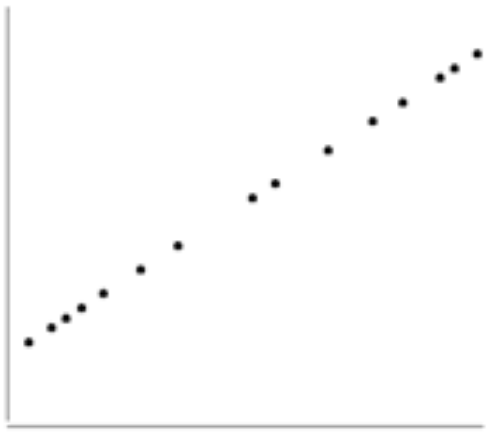
Pearson's Correlation Coefficient

- r = -1
- data lie on a perfect straight line with a negative slope



- r = 0
- no linear relationship between the variables

- r = +1
- data lie on a perfect straight line with a positive slope



Cross validation

Split	Exp. A	Exp. B	Exp. C
Train | Valid	Performance A1	Performance B1	Performance C1

Test

K-fold cross-validation

Splits	Exp. A	Exp. B	Exp. C
Train Valid	Performance A_1	Performance B_1	Performance C_1
	Performance A_2	Performance B_2	Performance C_2
	Performance A_K	Performance B_K	Performance C_K
	Average perf. A StdError perf. A	Average perf. B StdError perf. B	Average perf. C StdError perf. C

Test

K-fold cross-validation

Splits	Exp. A	Exp. B	Exp. C
Train Valid	Performance A_1	Performance B_1	Performance C_1
	Performance A_2	Performance B_2	Performance C_2
	Performance A_K	Performance B_K	Performance C_K
	Average perf. A StdError perf. A	Average perf. B StdError perf. B	Average perf. C StdError perf. C

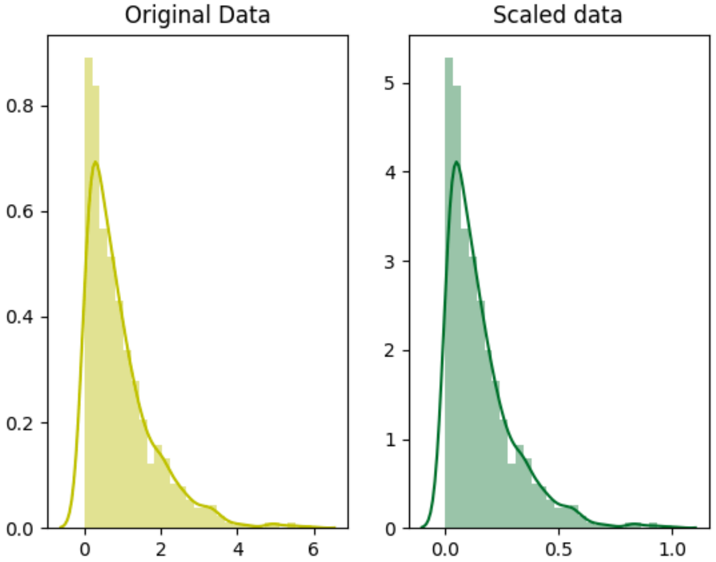
Test

- K-fold is less biased but it’s computationally expensive, in the above example we train the model 9 times and validate it 9 times and test 1 time!

Normalization

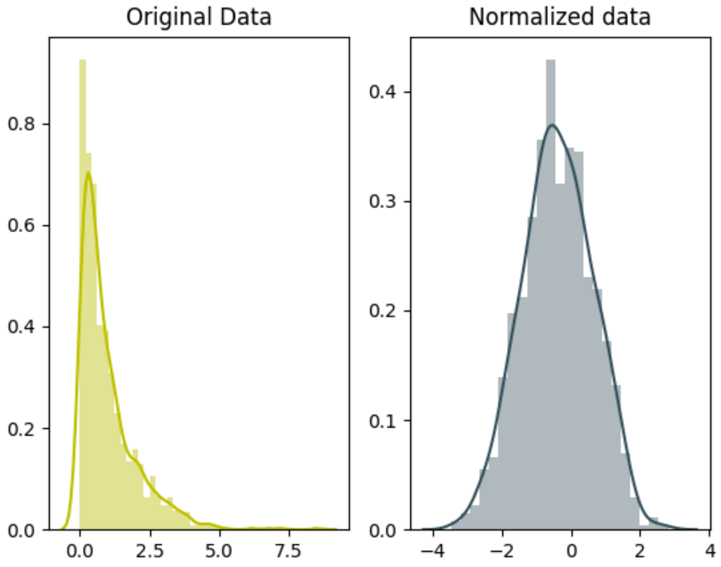
- Feature scaling

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}}$$



- Standardization

$$z = \frac{x - \mu}{\sigma}$$



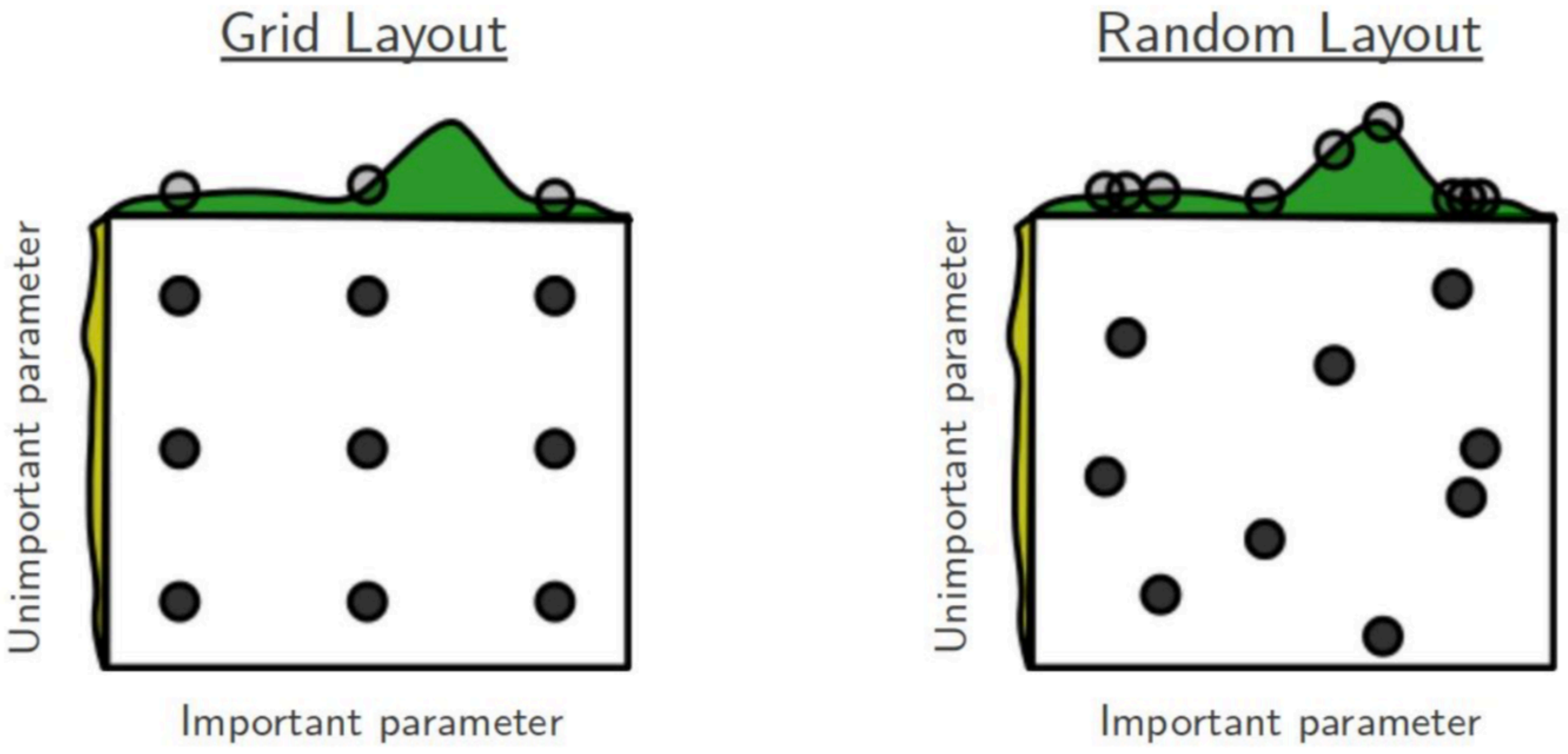
- In scaling, you’re changing the range of your data while in normalization you’re mostly changing the shape of the distribution of your data.

- You need to normalize our data if you’re going use a machine learning or statistics technique that assumes that data is normally distributed

Source: scaling vs normalization by Harshit Kumar

Hyperparameter search

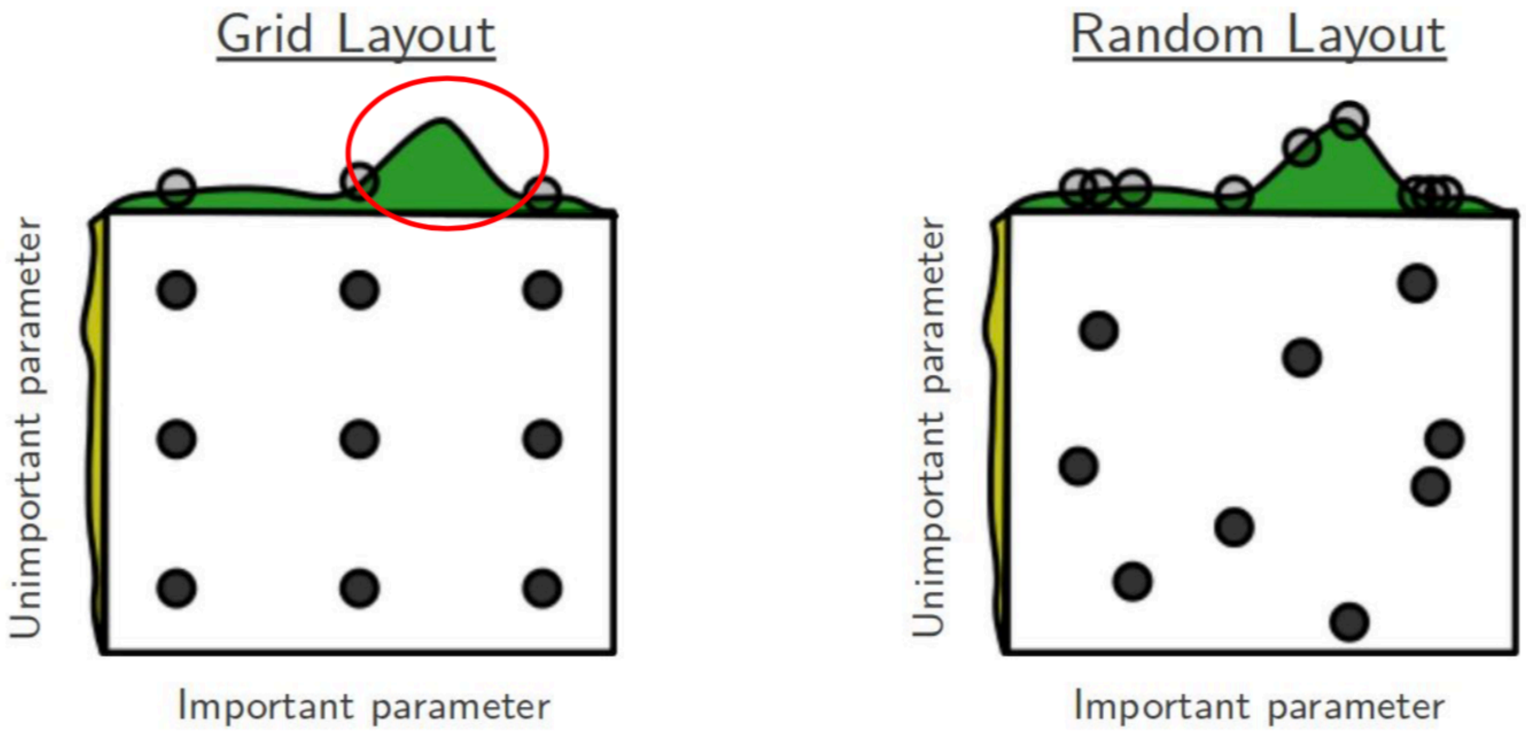
Random search with good candidates is already a strong baseline.



Source:Bergstra, James, and Yoshua Bengio. "Random search for hyper-parameter optimization." Journal of Machine Learning Research 13, no. Feb (2012): 281-305.

Hyperparameter search

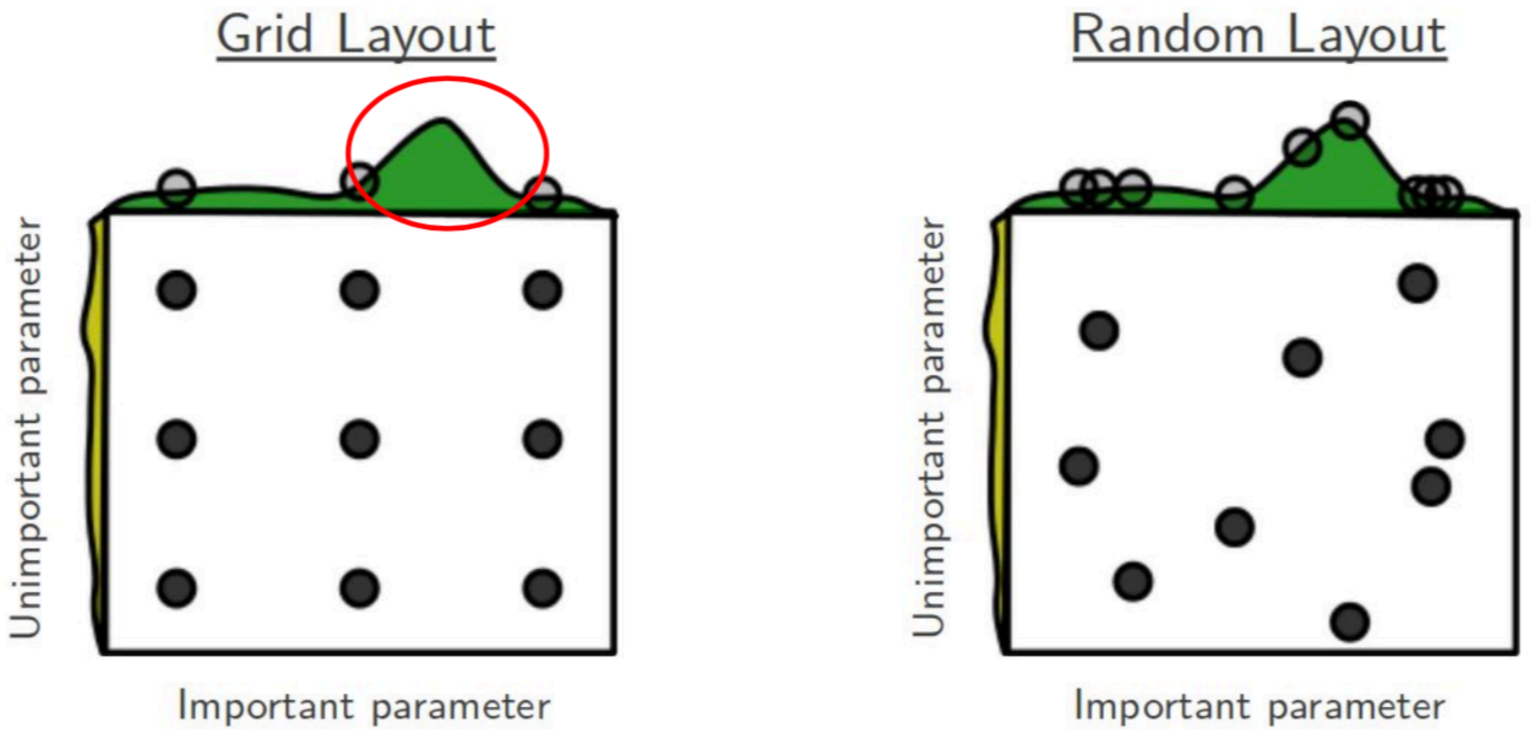
Random search with good candidates is already a strong baseline.



Source:Bergstra, James, and Yoshua Bengio. "Random search for hyper-parameter optimization." Journal of Machine Learning Research 13, no. Feb (2012): 281-305.

Hyperparameter search

Random search with good candidates is already a strong baseline.



Source:Bergstra, James, and Yoshua Bengio. "Random search for hyper-parameter optimization." Journal of Machine Learning Research 13, no. Feb (2012): 281-305.

References

- Ian Goodfellow, Yoshua Bengio, and Aaron Courville. “**Deep learning”,** MIT press, 2016.

- Shervin Minaee 20 Popular Machine Learning Metrics

- Ivado-mila Deep learning summer school 2019