INTRODUCTION TO NEURAL NETWORKS

Lecture 10. Convolutional Neural Network (2)

Dr. Jingxin Liu

School of AI and Advanced Computing

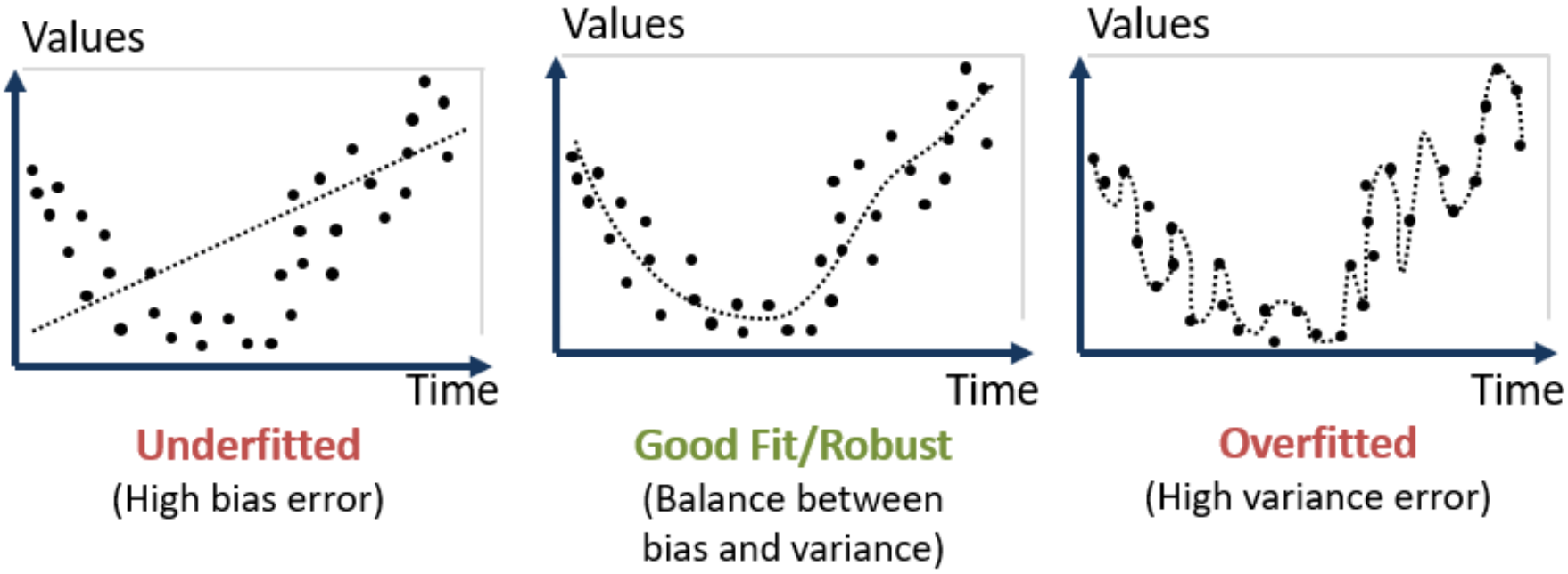
Xi'an Jiaotong-Liverpool University
西交利物浦大学

Table of Contents

Overfitting and Regularization

Overfitting: the model tries to learn too many details in the training data along with the noise from the training data. As a result, the model performance is very poor on unseen or test datasets.

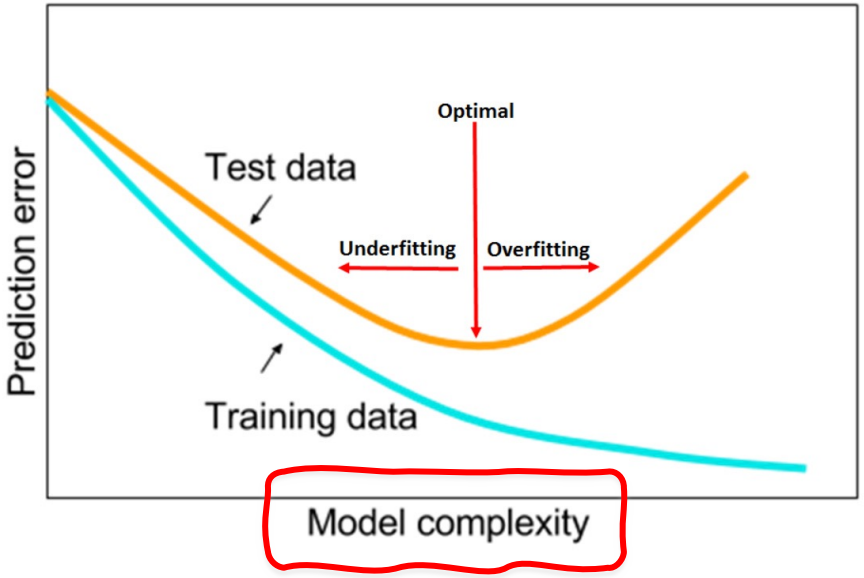
Typical overfitting means that error on the training data is very low, but error on new instances is high.

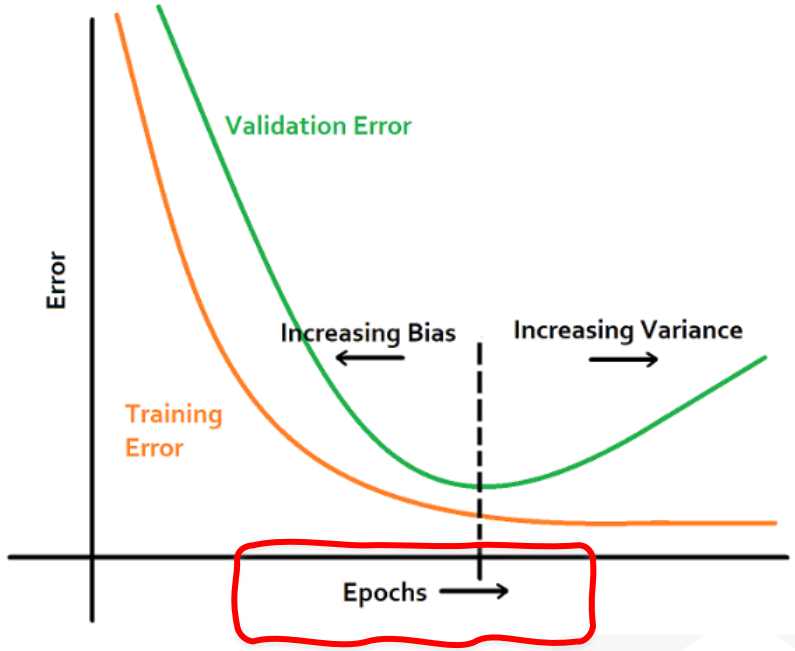


Overfitting and Regularization

Overfitting is a general, HUGELY IMPORTANT problem for all machine learning algorithms.

- The size of the training dataset used is not enough.
- Data used for training is not cleaned and contains noise (garbage values) in it.
- The model is too complex.





Overfitting and Regularization

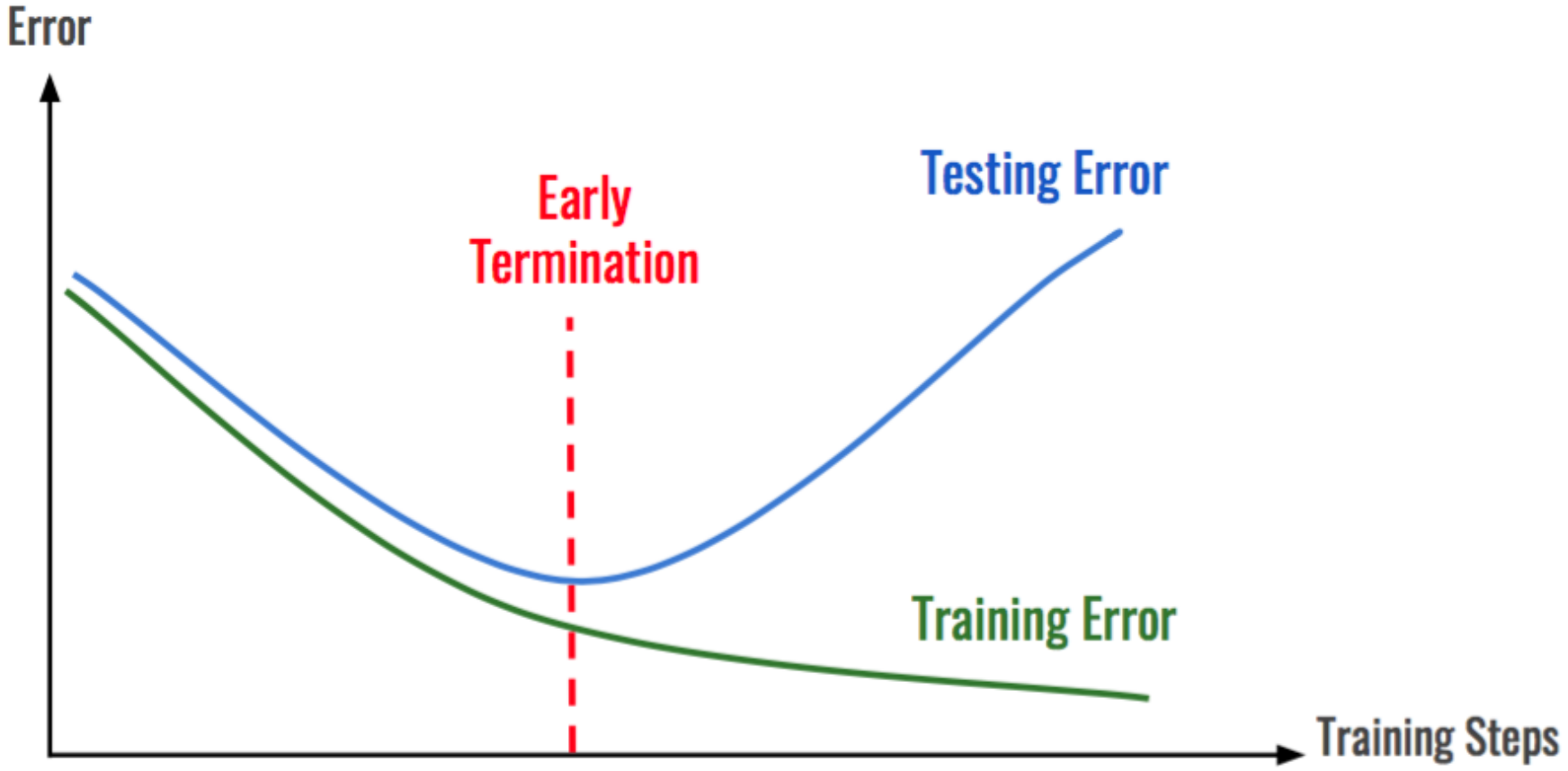
Regularization, in general: any method to prevent overfitting or help the optimization.

Regularization techniques:

- L1 regularization
- L2 regularization
- …….

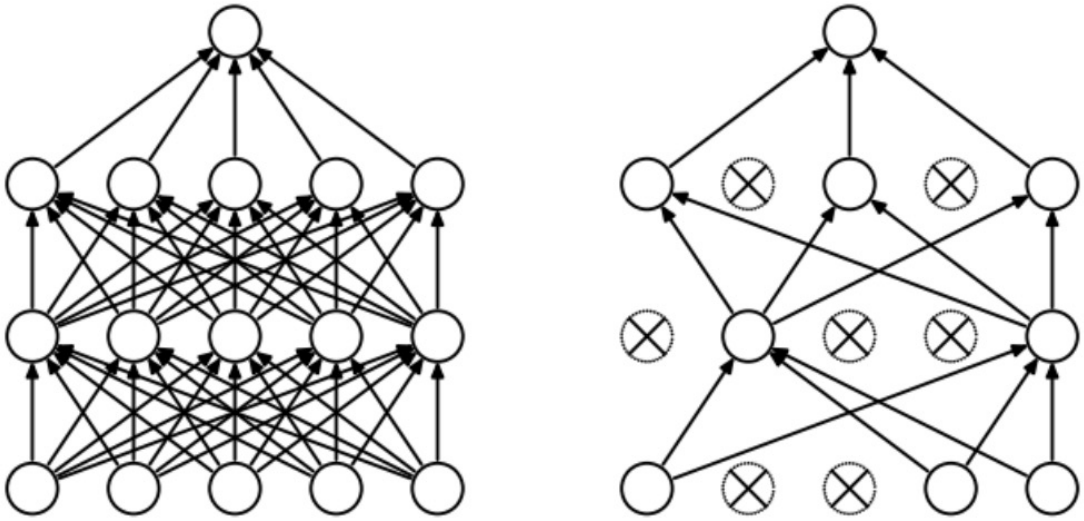
- Simplifying the model
- Data augmentation
- Early stopping
- Dropouts
- Batch normalization

Early Stopping



https://pytorch-lightning.readthedocs.io/en/stable/common/early_stopping.html
https://keras.io/api/callbacks/early_stopping/

Dropout



(a) Standard Neural Net

(b) After applying dropout.

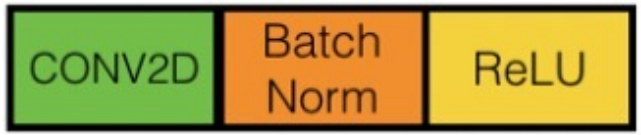
```
vgg16.add(Flatten())
vgg16.add(Dense(4096))
vgg16.add(Activation('relu'))
vgg16.add(Dropout(0.5))
vgg16.add(Dense(4096))
vgg16.add(Activation('relu'))
vgg16.add(Dropout(0.5))
vgg16.add(Dense(n_classes))
vgg16.add(Activation('softmax'))
```

```
x = x.reshape(x.shape[0], -1)
x = F.relu(self.fc1(x))
x = F.dropout(x, 0.5) #dropout was included to combat overfitting
x = F.relu(self.fc2(x))
x = F.dropout(x, 0.5)
x = self.fc3(x)
```

https://keras.io/api/layers/regularization_layers/dropout/
https://pytorch.org/docs/stable/generated/torch.nn.Dropout.html

Batch Normalization

Batch normalization (also known as **batch norm**) is a method used to make artificial neural networks faster and more stable through normalization of the layers' inputs by re-centering and re-scaling.




```
x = layers.Conv2D(filters1, (1, 1), strides=strides,
                  kernel_initializer='he_normal',
                  name=conv_name_base + '2a')(input_tensor)
x = layers.BatchNormalization(axis=bn_axis, name=bn_name_base + '2a')(x)
x = layers.Activation('relu')(x)

x = layers.Conv2D(filters2, kernel_size, padding='same',
                  kernel_initializer='he_normal',
                  name=conv_name_base + '2b')(x)
x = layers.BatchNormalization(axis=bn_axis, name=bn_name_base + '2b')(x)
x = layers.Activation('relu')(x)
```

Ioffe, Sergey, and Christian Szegedy. "Batch normalization: Accelerating deep network training by reducing internal covariate shift." *International conference on machine learning*. PMLR, 2015.

Evaluation Metrics for Classification

Evaluation metrics are used to measure the quality of the statistical or machine learning model.

It is very important to use multiple evaluation metrics to evaluate your model. This is because a model may perform well using one measurement from one evaluation metric, but may perform poorly using another measurement from another evaluation metric. Using evaluation metrics are critical in ensuring that your model is operating correctly and optimally.

Evaluation metrics for **classification tasks**:

- Accuracy
- Confusion Matrix

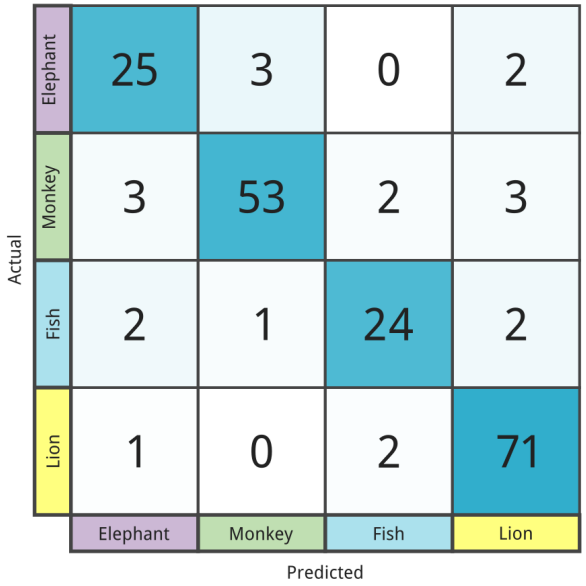
- Precision and Recall
- F1 Score
- ROC curve
- Precision-Recall curve
- ……

Confusion Matrix

Confusion Matrix, also known as an **Error Matrix**, the Confusion Matrix is a two-dimensional matrix that allows visualization of the algorithm's performance. While this isn't an actual metric to use for evaluation, it's an important starting point.

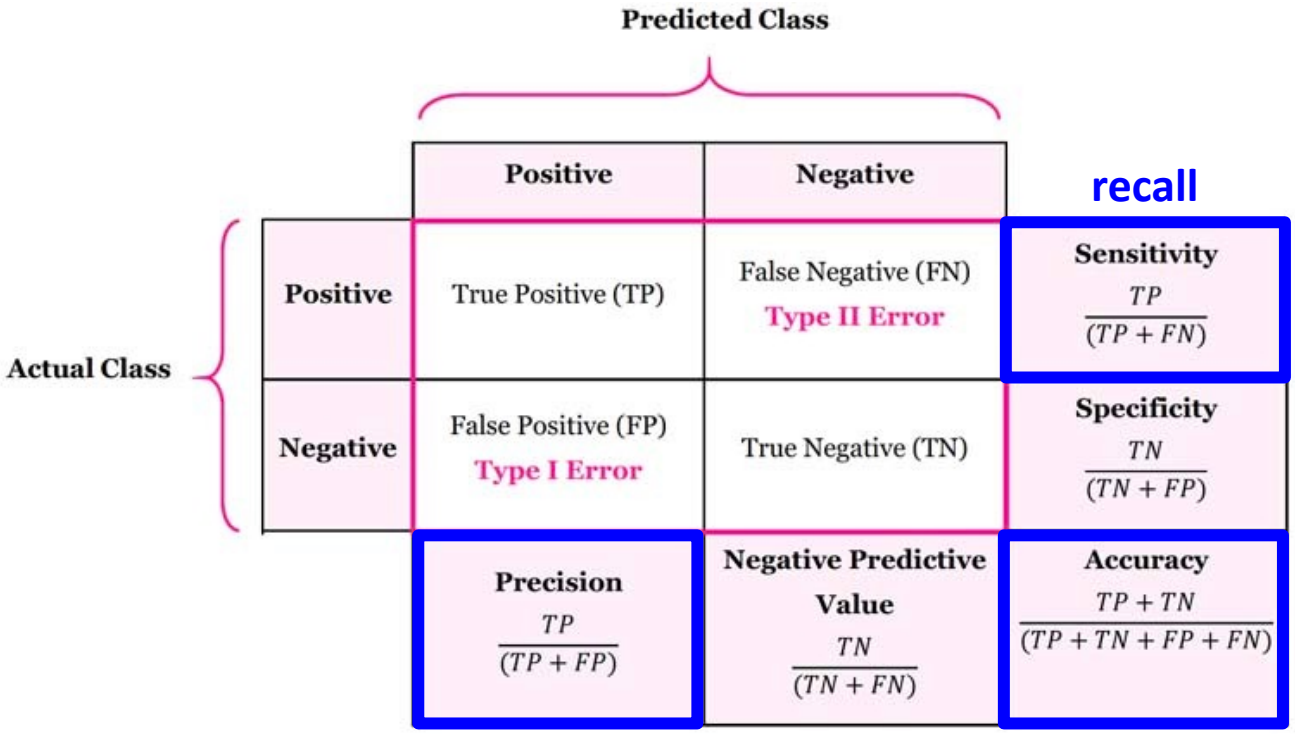
In a binary classification, the matrix will be 2x2. If there are 3 classes, the matrix will be 3x3, and so on.

n=165	**Predicted: NO**	**Predicted: YES**
Actual: NO	50	10
Actual: YES	5	100



https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html

Confusion Matrix



Accuracy

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

$$\frac{number\ of\ correctly\ classified\ examples}{total\ number\ of\ examples}$$

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F1\text{-}score = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

Precision and Recall

Define patients who have Covid-19 as Positive and those who do not have Covid-19 as Negative

- **Precision**：Out of the bunch of people diagnosed with Covid-19, how many actually have Covid-19?
- **Recall**：Out of a bunch of patients with Covid-19, how many people can be successfully detected Covid-19?

Spam filtering system, spam is Positive, normal mail is Negative

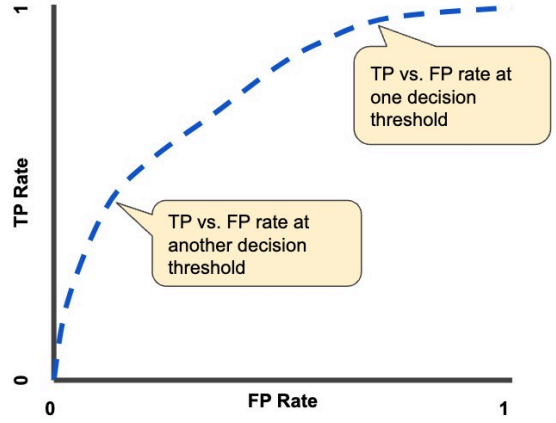
- **Precision**： How much of the blocked email is really spam?
- **Recall**： How much of all spam is blocked?

ROC Curve

An **ROC curve** (**receiver operating characteristic curve**) is a graph showing the performance of a classification model at all classification thresholds. This curve plots two parameters:

- True Positive Rate
- False Positive Rate

An ROC curve plots TPR vs. FPR at different classification thresholds. Lowering the classification threshold classifies more items as positive, thus increasing both False Positives and True Positives.

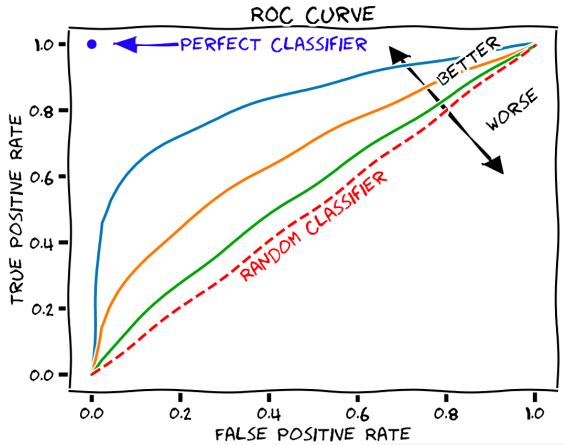


Predicted Class

Actual Class	Positive	Negative	
Positive	True Positive (TP)	False Negative (FN) Type II Error	Sensitivity $\frac{TP}{(TP+FN)}$
Negative	False Positive (FP) Type I Error	True Negative (TN)	Specificity $\frac{TN}{(TN+FP)}$
	Precision $\frac{TP}{(TP+FP)}$	Negative Predictive Value $\frac{TN}{(TN+FN)}$	Accuracy $\frac{TP+TN}{(TP+TN+FP+FN)}$

True Positive Rate

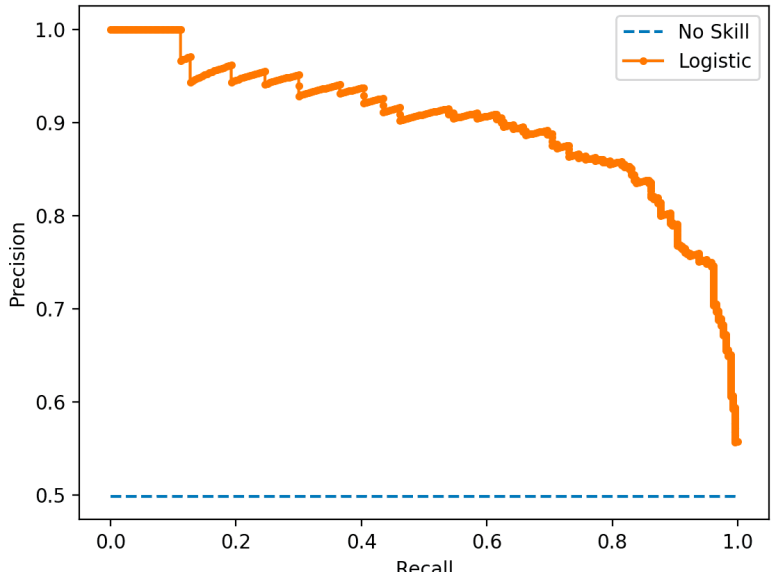
$$FPR = \frac{FP}{FP+TN}$$

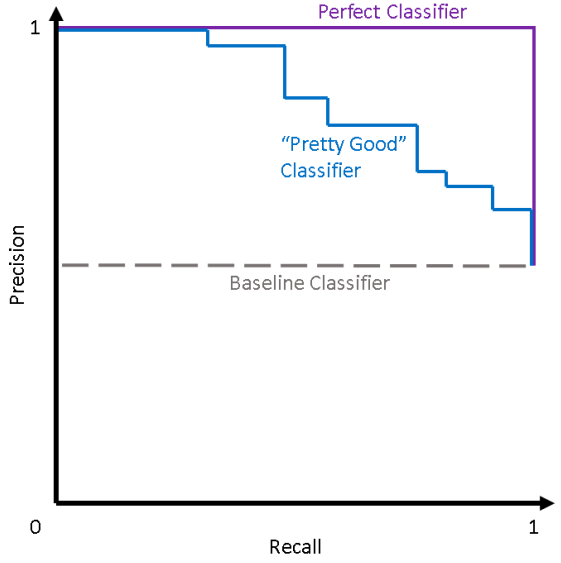


https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc#:~:text=An%20ROC%20curve%20(receiver%20operating,False%20Positive%20Rate

Precision-Recall Curve

The **precision-recall curve** is used for evaluating the performance of binary classification algorithms. It is often used in situations where classes are heavily imbalanced. Also like ROC curves, precision-recall curves provide a graphical representation of a classifier's performance across many thresholds, rather than a single value.





https://scikit-learn.org/stable/auto_examples/model_selection/plot_precision_recall.html

Summary

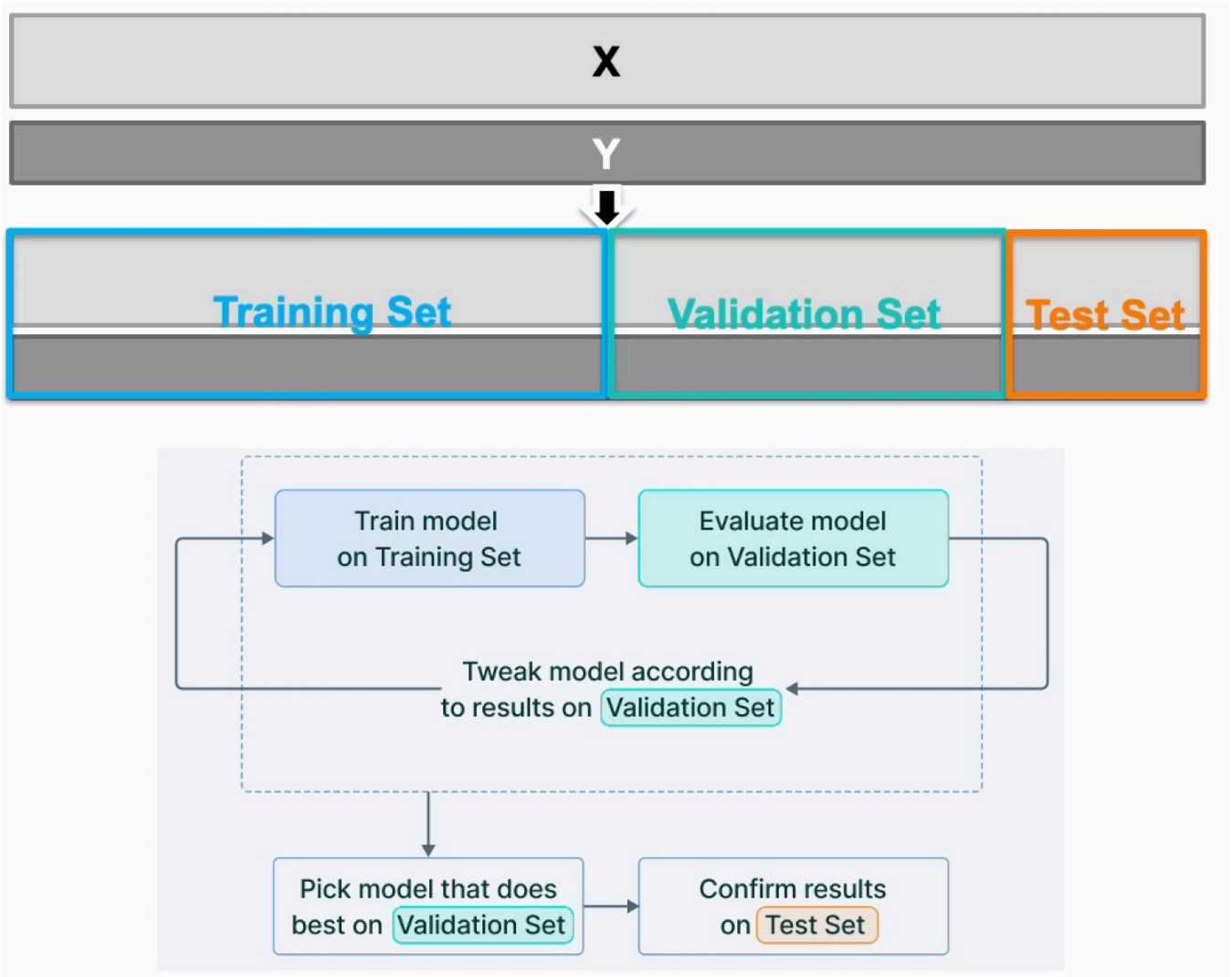
- Data Preprocessing
- Data Augmentation
- ……

Image Classification
- Lenet
- AlexNet
- VGG
- ResNet
- ……

Image Classification
- Confusion matrix
- Accuracy
- ……

- Learning rate
- Dropout rate
- Optimization methods
- ……

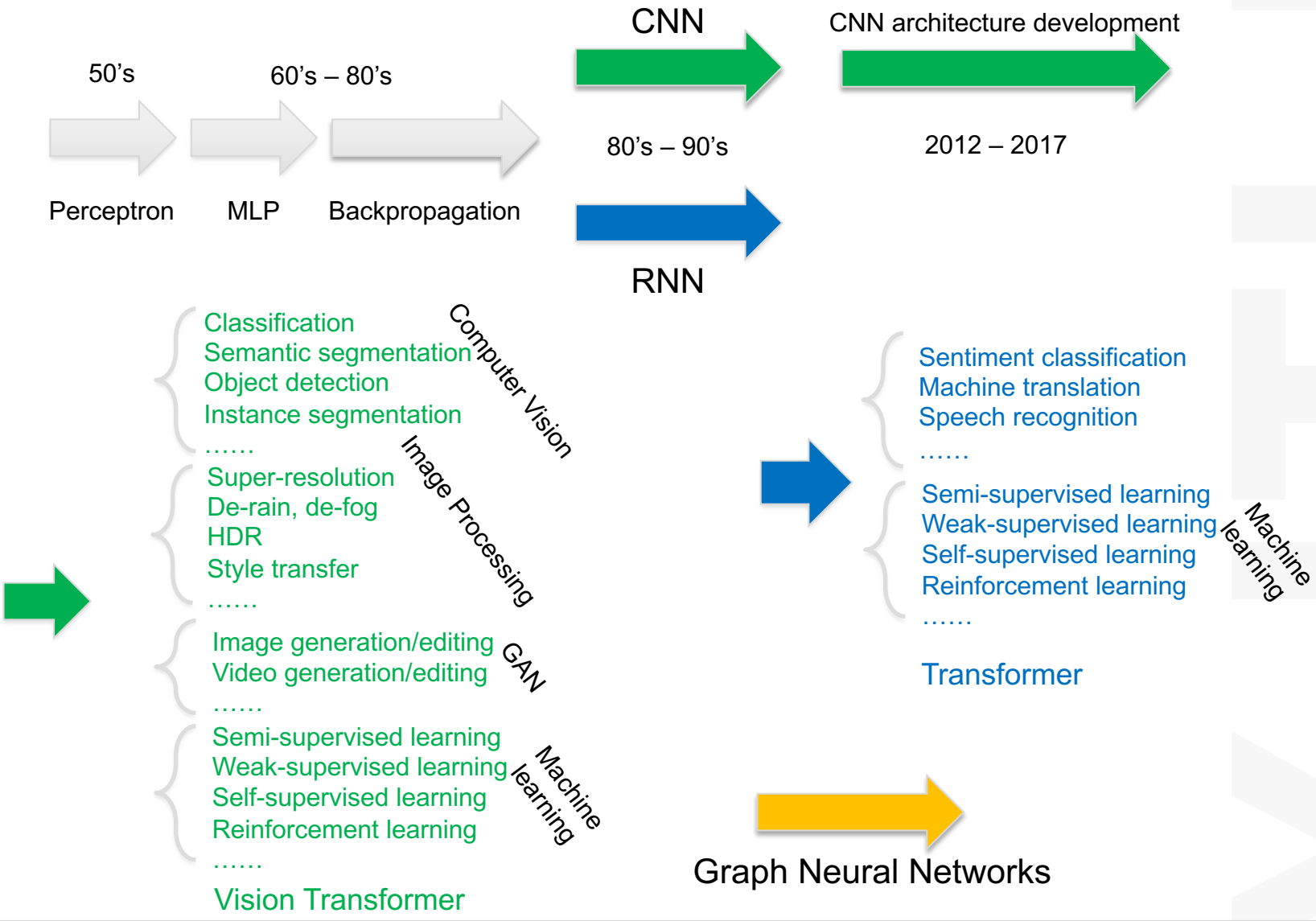
Summary


X
Y
Training Set
Validation Set
Test Set
Train model on Training Set
Evaluate model on Validation Set
Tweak model according to results on Validation Set
Pick model that does best on Validation Set
Confirm results on Test Set

Summary

Lecture 1	Course Specification, AI, ML Introduction
Lecture 2	Linear Regression and Logistic Regression
Lecture 3	ANN Overview, Application, History, etc.
Lecture 4	Perceptron
Lecture 5	feed forward network, Forward Propagation
Lecture 6	Backpropagation
Lecture 7	Introduction to Recurrent Network
Lecture 8	Image Processing Fundamental
Lecture 9	Convolutional Neural Networks - 1
Lecture 10	Convolutional Neural Networks - 2

Future Works



Thank you !