SCHOOL OF COMPUTER SCIENCE AND IT

School of Computer Science & IT

BCA Programme

INTRODUCTION TO DATA ANALYTICS (23BCAD4C01)

MODULE 5: Model Evaluation

Dr. Ananta Charan Ojha, Professor

Session -1

- Assessing Model Accuracy
  - Measuring the quality of fit,
- The Bias-Variance Trade-off.

Supervised Learning Revisited

- Suppose we are observing a *response variable* *Y* (qualitative or quantitative) and *input variable X* having p number of features or columns (X1, X2, ….., Xp) and we assume there is relation between them.
- This relation can be expressed as: ***Y = f(X) + e***
  - Here *f* is some fixed but unknown function of X1, X2, …, Xp, and *e is a random error term, which is independent of X and has mean zero.* In this formulation, *f* represents the systematic information that X provides about Y. **Estimation of this relation or *f*(X) is known as machine learning.**
- The accuracy of the model can be improved by making a more accurate estimate of *f*(X) and therefore reducing the ***reducible error***. But, even if we make a 100% accurate estimate of *f*(X), our model won't be error free, this is known as ***irreducible error***.
- In other terms, the irreducible error can be seen as information that X cannot provide about Y because this information is not present in the training data. Nothing that we can do about it. What we can do is reduce other forms of error (i.e. *reducible error*) to get a near perfect estimation of *f*(X).

Assessing Model Accuracy

- Every data set is different and there is no one machine learning method that works best for all data sets.
- It is important for any given data set to find the machine learning method that produces the best results.
- We need to be able to quantify how well a model's predictions match the observed data. How close are the model's predicted values to the true observed values?

Measuring the Quality of Fit

- To quantify the extent to which the predicted value for a given observation is close to the true observed value for that observation, the most commonly used measure in machine learning is the **mean squared error** (MSE),

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

n	= number of data points
Y_i	= observed values
$\hat{Y}_i$	= predicted values

- As the name goes, *it is the mean of square of the errors or differences in predictions and observed values for all inputs.* It is known as **training MSE** if calculated using training data, and **test MSE** if calculated using testing data.
- There is no guarantee that the model with the lowest *training* MSE will also be the model with the lowest *test* MSE.

Measuring the Quality of Fit contd…

- The expected test MSE, for a given value x0, can always be decomposed into the sum of three fundamental quantities: the variance of *f*(x0), the squared bias of *f*(x0) and the variance of the error terms e. Where, e is the irreducible error, about which we discusses earlier.

$$Err(x) = \mathrm{Bias}^2 + \mathrm{Variance} + \mathrm{Irreducible\ Error}$$

- Irreducible error is the error that can't be reduced by creating good models. So, lets see more about bias and variance.

- **What is bias?**

- Bias is the difference between the average prediction of our model and the correct value which we are trying to predict. Model with high bias pays very little attention to the training data and oversimplifies the model. It always leads to high error on training and test data.

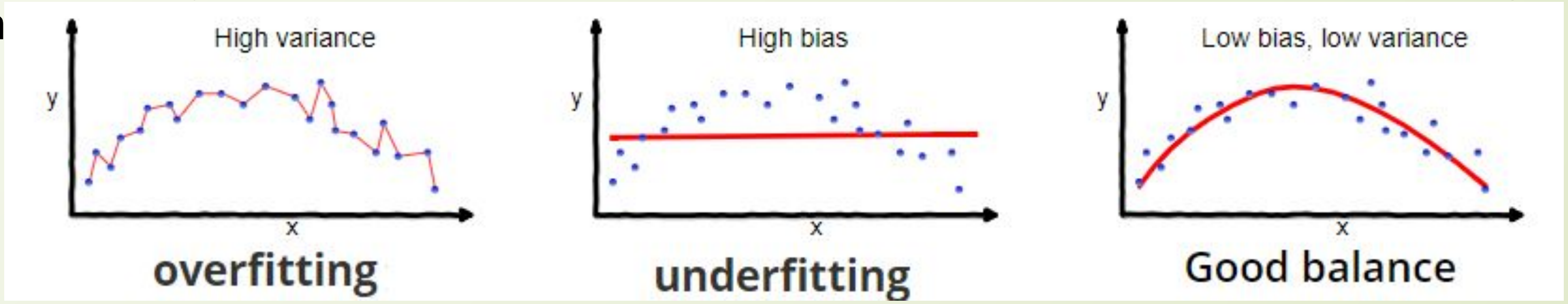
- Bias refers to the error that is introduced by approximating a real-life problem, which may be extremely complicated, by a much simpler model. So, if the true relation is complex and you try to use simple model (e.g. a linear regression), then it will undoubtedly result in some bias in the estimation of *f*(X). No matter how many observations you have, it is impossible to produce an accurate prediction if you are using a restrictive/ simple algorithm, when the true relation is highly complex.

Measuring the Quality of Fit contd…

- **What is variance?**
  - Variance refers to the amount by which the estimate of *f*(X) would change if we estimated it using a different training data set. Since the training data is used to fit the machine learning method, different training data sets will result in a different estimation. But ideally the estimate for *f*(X) should not vary too much between training sets. However, if a method has high variance then small changes in the training data can result in large changes in *f*(X).
  - Variance is the variability of model prediction for a given data point or a value which tells us spread of our data. Model with high variance pays a lot of attention to training data and does not generalize on the data which it hasn't seen before. As a result, such models perform very well on training data but has high error rates on test data.
- **General Rule**
  - A general rule is that, ***as a machine learning method tries to match data points more closely, the bias reduces, but variance increases.***

Measuring the Quality of Fit contd…

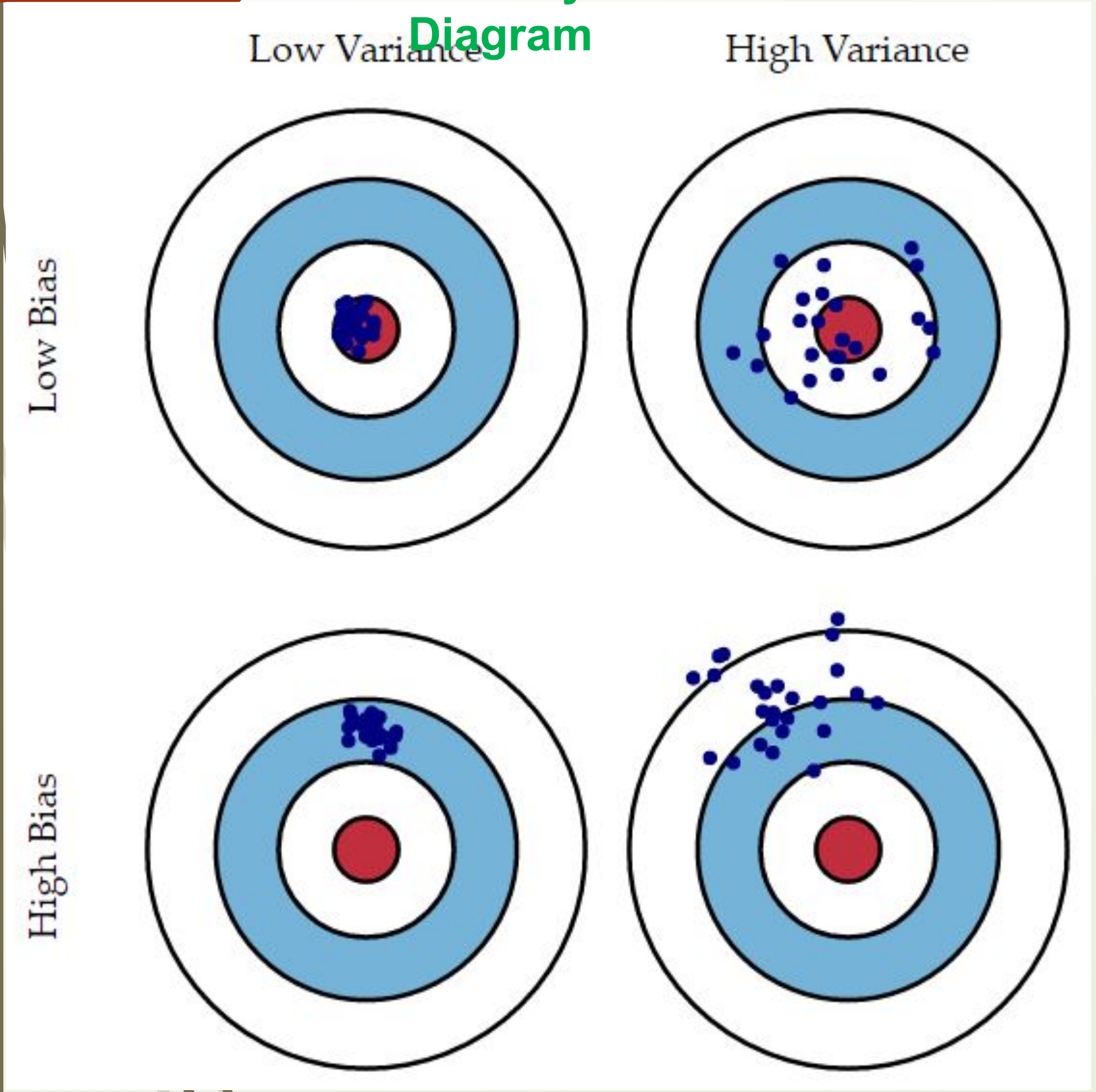
- In supervised learning, **underfitting** happens when a model unable to capture the underlying pattern of the data. These models usually have high bias and low variance. It happens when we have very less amount of data to build an accurate model or when we try to build a linear model with a nonlinear data. Also, these kind of models are very simple to capture the complex patterns in data like Linear and Logistic regression. In, underfitting, a model can neither model the training data nor generalize to test data.
- In supervised learning, **overfitting** happens when our model captures the noise along with the underlying pattern in data. It happens when we train our model a lot over noisy dataset. These models have low bias and high variance. These models are very complex like Decision trees which are prone to overfitting. In overfitting, although the model might fit the train data well, it might not fit the test data well.
- **A Good Fit**: Ideally, we want to select a model at the sweet spot between underfitting and overfitting. This is th



- **Overfitting or underfitting** are the most common causes of **poor performance** for most machine learning (ML) models. Additionally, overfitting is more common than underfitting.
  - **Overfitting** is when the model's error on the training set (i.e. during training) is very low but then, the model's error on the test set (i.e. unseen samples) is large! High Variance!
  - **Underfitting** is when the model's error on **both** the **training** and **test** sets (i.e. during training and testing) is very high. High Bias!
- To overcome these problems, **cross-validation** is usually used in order to estimate the model's performance on **unseen** data.

The Bias-Variance Trade-Off

Bulls-eye Diagram



- The **bias-variance tradeoff** is a particular property of all (supervised) machine learning models, that enforces a tradeoff between how "flexible" the model is and how well it performs on unseen data.
- The center of the target is a model that perfectly predicts the correct values. As we move away from the bulls-eye, our predictions get worse and worse.
- we can repeat our entire model building process to get a number of separate hits on the target, such that each blue dot represents different realizations of our model based on different data sets for same problem.
- *It displays four different cases representing combinations of both high and low bias and variance. High bias is when all dots are far from bulls eye and high variance is when all dots are scattered.*
- A model will have a high error if it has very high bias and low variance i.e. when it is not able to adapt at all to the data points in a sample set. On the other extreme, a model will also have a high error if it has very high variance and low bias i.e. when it adapts too well to all the data points in a sample set(with an incomplete representation of true data) and hence fails to generalize the true dataset.
- A model that strikes a balance between the bias and variance is able to minimize the error.
- **In order to minimize the expected test error, we need to select a machine learning method that simultaneously achieves low variance and low bias.**

Conclusion

- When you are working with machine learning models, you need to understand all the different sources of error. You can then attempt to minimize them so that your final model looks great!
- We want low bias and low variance, but it's difficult to attain. So the bias vs. variance analysis is a way of understanding our algorithm's abilities to generalize well.
- Using this analysis, we can understand our algorithm's expected generalization error.
- It's an attempt to find an answer to questions like "Will my algorithm perform well on unknown data?" and "Will my algorithm be consistent across different types of data points?"
- This tradeoff usually applies to all supervised learning problems like classification, regression, etc.

THANK YOU

Any questions…?

JAIN DEEMED-TO-BE UNIVERSITY | SCHOOL OF COMPUTER SCIENCE AND IT

School of Computer Science & IT

BCA Programme

INTRODUCTION TO DATA ANALYTICS (21BCAD4C01)

MODULE 5: Model Evaluation

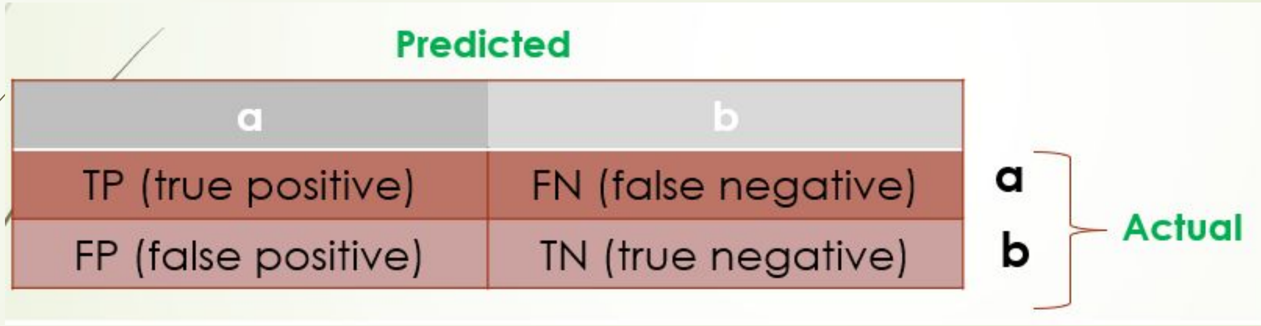
Dr. Ananta Charan Ojha, Professor

Session -2

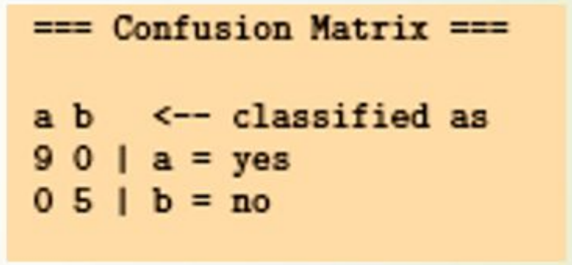
- Evaluation Metrics
  - Class Imbalance Problem
- Methods for Evaluation

Metrics for Evaluating Performance of Classifiers

- Several evaluation measures for assessing how "accurate" your classifier is at predicting the class label of tuples : accuracy, sensitivity, specificity, precision, recall, F1.
- **Confusion Matrix**
- Shows how many instances have been assigned to each class.
- Elements in the matrix show the number of instances whose actual class is the row and whose predicted class is the column.



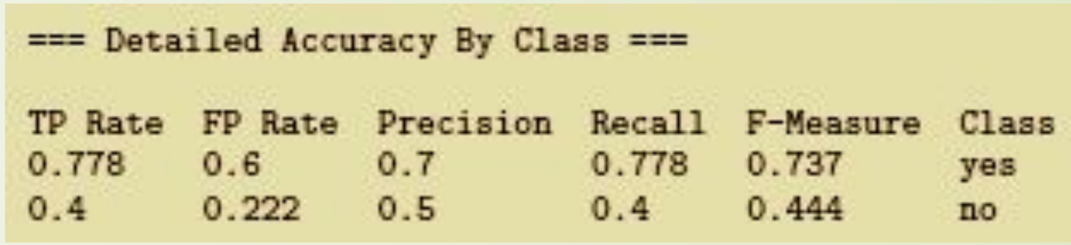
	Predicted a	Predicted b
Actual a	TP (true positive)	FN (false negative)
Actual b	FP (false positive)	TN (true negative)



```
=== Confusion Matrix ===

a b   <-- classified as
9 0 | a = yes
0 5 | b = no
```

True Positives (TP): positive tuples that are correctly labeled by the classifier.
True Negatives (TN): negative tuples that are correctly labeled by the classifier.
False Positives (FP): negative tuples that are incorrectly labeled as positive class by the classifier
False Negatives (FN): positive tuples that are incorrectly labeled as negative class by the classifier



```
=== Detailed Accuracy By Class ===

TP Rate   FP Rate   Precision   Recall   F-Measure   Class
0.778     0.6       0.7         0.778    0.737       yes
0.4       0.222     0.5         0.4      0.444       no
```

- **Classifier Accuracy**(or recognition rate): percentage of tuples that are correctly classified

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

- **Error rate:** *1 – accuracy*, or **Error rate = (FP + FN) / (TP+TN+FP+FN)**
- The *precision* and *recall* measures are also widely used in classification.
- **Precision** can be thought of as a measure of *exactness* (that is, what percentage of tuples labeled as positive are actually such).

  Precision = TP / (TP + FP)

- **Recall** is a measure of *completeness* (what percentage of positive tuples are labeled as such).

  Recall = TP / (TP + FN)

- An alternative way to use precision and recall is to combine them into a single measure. This is the approach of the **F measure** (also knows as the F_1 score or Fscore) and the F_β measure. They are defined as:

$$F = \frac{2 \times precision \times recall}{precision + recall}$$

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

where β is a non-negative real number.

Class Imbalance Problem:

- When there is class imbalance, the classifier tend to be biased towards the majority class. In that case sensitivity and specificity measures are used to evaluate the classifier performance.
- The sensitivity measure shows how well, the classifier can recognize the positive tuples

  Sensitivity = TP / (Total no of positive classes)

- The specificity measure shows how well the classifier can recognize the negative tuples

  Specificity = TN / (Total no of negative classes)

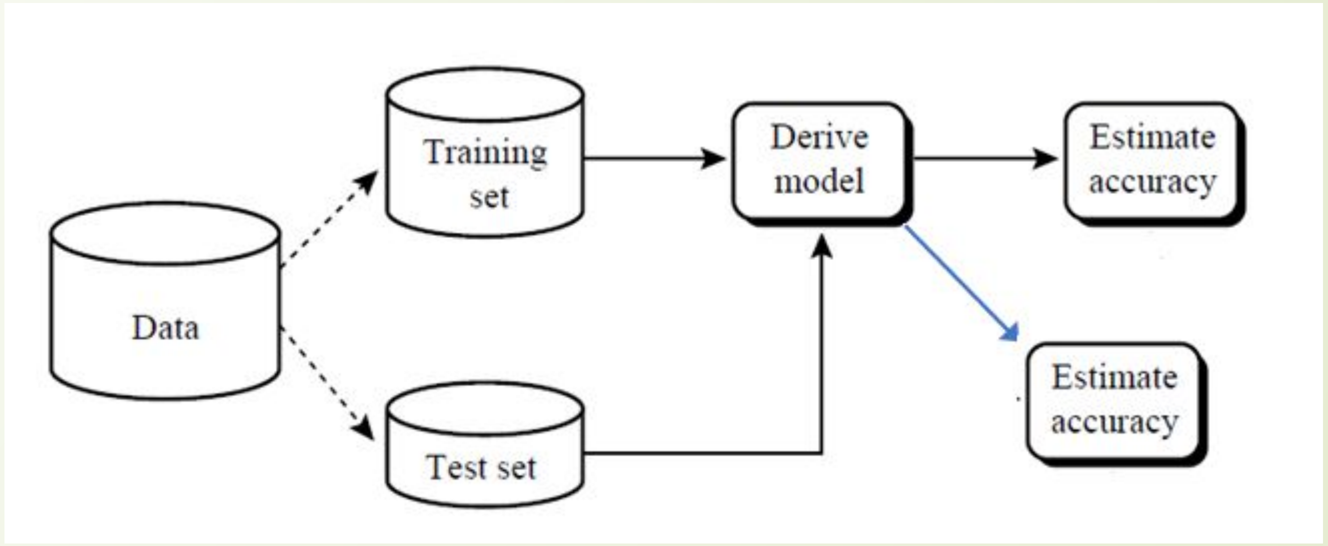
- It can be shown that accuracy is a function of sensitivity and specificity (P = positive classes, N = negative classes):

$$accuracy = sensitivity \frac{P}{(P+N)} + specificity \frac{N}{(P+N)}$$

Additional Measures:

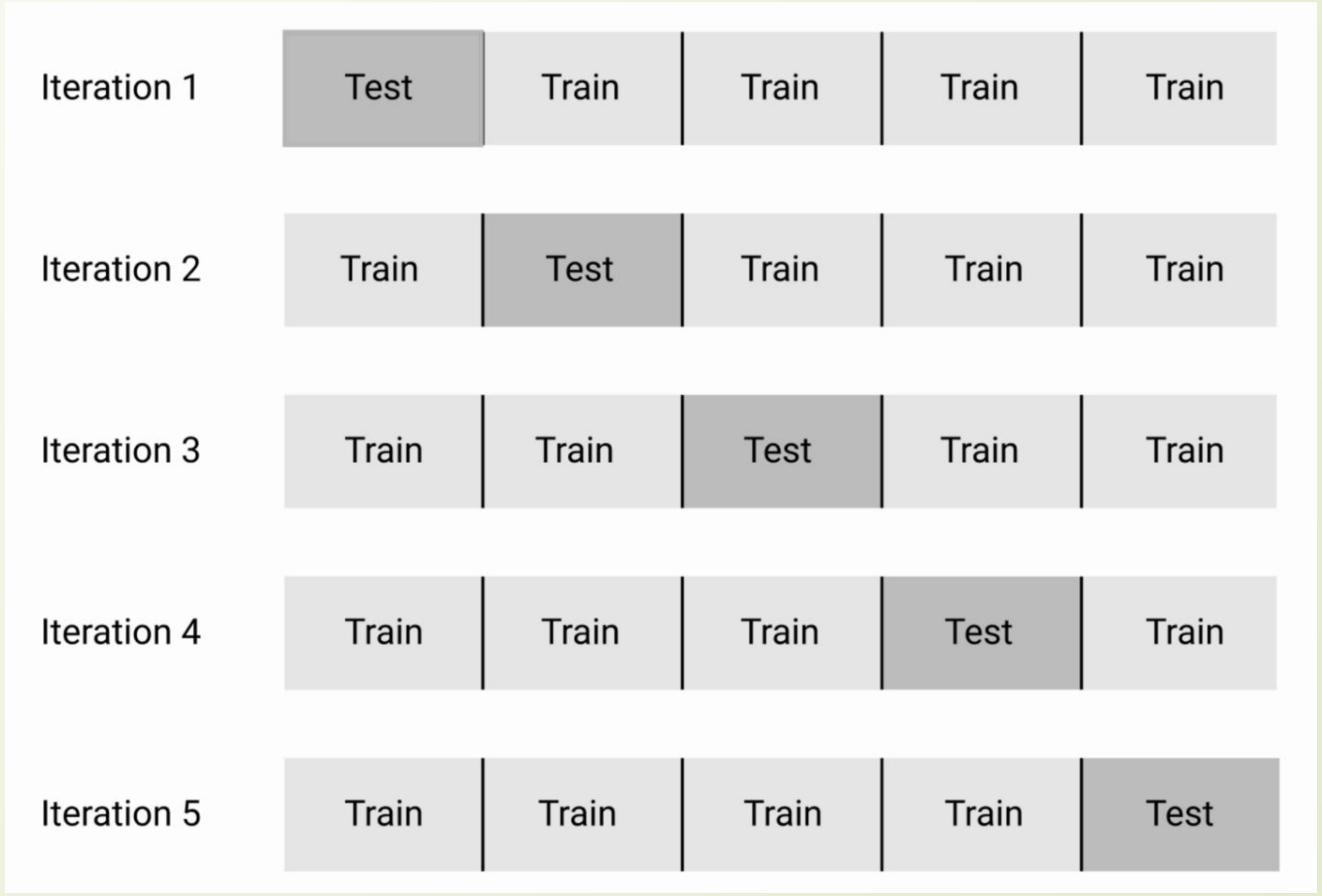
- Speed:
  - This refers to the computational costs involved in generating and using the given classifier.
- Robustness:
  - This is the ability of the classifier to make correct predictions given noisy data or data with missing values.
  - Robustness is typically assessed with a series of synthetic datasets representing increasing degrees of noise and missing values.
- Scalability:
  - Scalability is typically assessed with a series of datasets of increasing size.
- Interpretability:
  - This refers to the level of understanding and insight that is provided by the classifier or predictor.
  - Interpretability is subjective and therefore more difficult to assess.
  - Decision trees and classification rules can be easy to interpret, yet their interpretability may diminish the more they become complex.

Methods for Evaluating Classifiers Accuracy



- **Holdout method**
  - Given data is randomly partitioned into two independent sets (sampling without replacement/ with replacement)
    - Training set (e.g., 2/3, 60%) for model construction
    - Test set (e.g., 1/3, 40%) for accuracy estimation
- **Random sampling**: a variation of holdout
  - The holdout method is repeated k times. The overall accuracy estimate is taken as the average of the accuracies obtained from each iteration.

- **Cross-validation** (*k*-fold, where k=5, k = 10 is the most popular)
  - Randomly partition the data into *k mutually exclusive* subsets, each approximately equal size
  - At *i*-th iteration, use D_i as test set and others as training set



Iteration 1	Test	Train	Train	Train	Train
Iteration 2	Train	Test	Train	Train	Train
Iteration 3	Train	Train	Test	Train	Train
Iteration 4	Train	Train	Train	Test	Train
Iteration 5	Train	Train	Train	Train	Test

THANK YOU

Any questions…?

SCHOOL OF COMPUTER SCIENCE AND IT

School of Computer Science & IT

BCA Programme

INTRODUCTION TO DATA ANALYTICS (21BCAD4C01)

MODULE 5: Model Evaluation

Dr. Ananta Charan Ojha, Professor

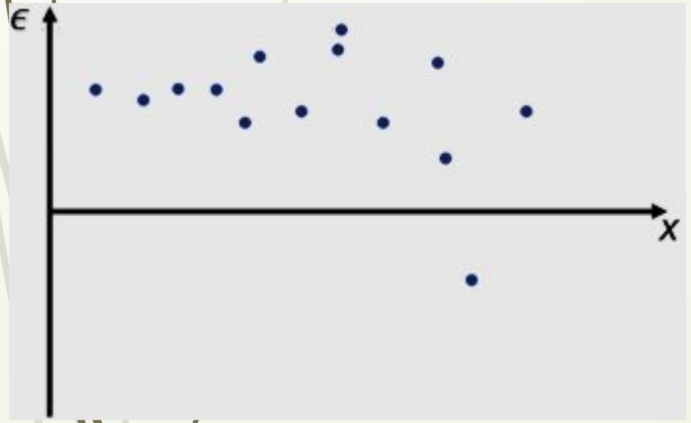
Session -3

- Model Evaluation using Visualization
  - Residual Plot
  - ROC Curve
- Model Selection
- Model Optimization

Residual Plot

$Residual\ (\epsilon) = y - \hat{y}$

- **Residuals:** A residual is a measure of how far away a point is vertically from **the regression line**. Simply, it is the error between a predicted value and the observed actual value. **Residual = Observed – Predicted**
- **Residual Plots:** A typical residual plot has the residual values on the Y-axis and the independent variable on the x-axis. Residual plots can be used to assess the quality of a regression.
- **Characteristics of Good Residual Plots**
  - A few characteristics of a good residual plot are as follows:
  1. It has a high density of points close to the origin and a low density of points away from the origin
  2. It is symmetric about the origin



1. This plot has high density far away from the origin and low density close to the origin.
2. Also, we can see the distribution curve is not normal.

Fig.: Bad Residual plot

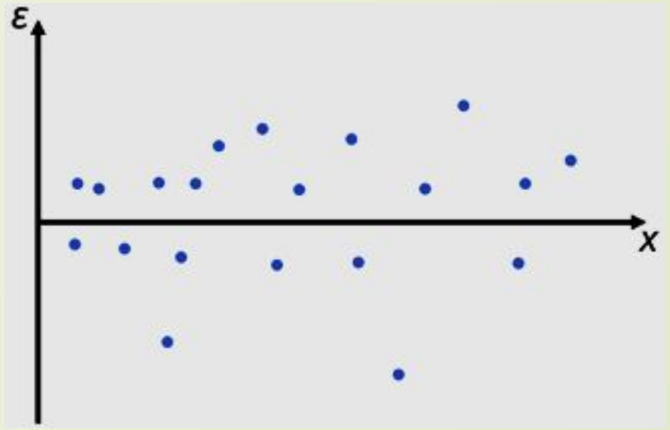
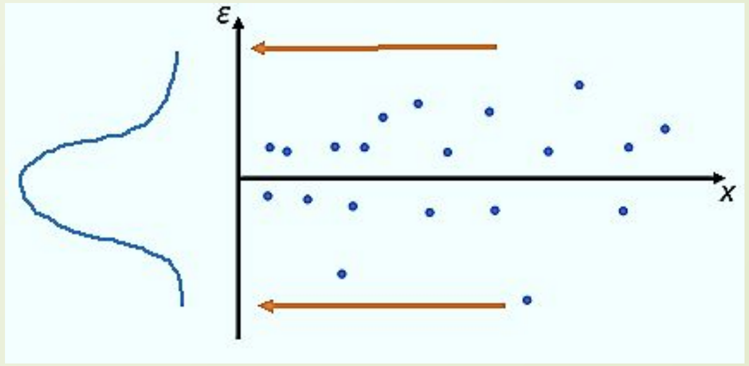
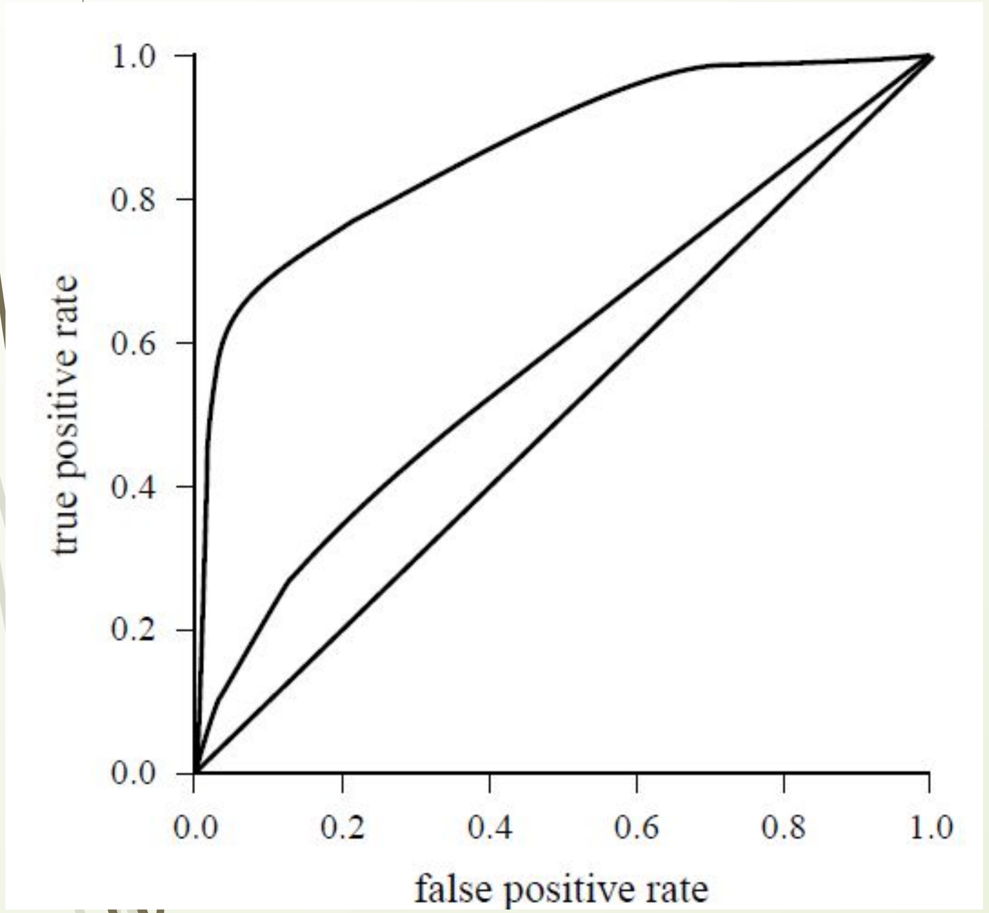


Fig.: Good Residual Plot



ROC Curve



- **Using ROC** (Receiver Operating Characteristics) curves: used for visual comparison of classification models. It was originated from signal detection theory.
- Shows the trade-off between the true positive rate and the false positive rate
  - Vertical axis represents the true positive rate
  - Horizontal axis represents the false positive rate
  - The plot also shows a diagonal line
- To assess the accuracy of a model, we can measure the area under the curve. Several software packages are able to perform such calculation. The closer the area is to 0.5 (diagonal line), the less accurate the corresponding model is. A model with perfect accuracy will have an area of 1.0.
- For each class (positive/negative), you can have a ROC curve.

Model Selection

- Suppose we have 2 classifiers, M_1 and M_2, which one is better?
- Use 10-fold cross-validation to obtain mean errors for both the models.
- These mean error rates are just *estimates* of error on the true population of *future* data cases.
- Although the mean error rates obtained for M_1 and M_2 may appear different, that difference may not be statistically significant.
- What if the difference between the 2 error rates is just attributed to *chance*?
- Use a **test of statistical significance**
- ✓ Perform Hypothesis testing: **Null Hypothesis**: M_1 & M_2 are the same
- If we can **reject** null hypothesis, then
  - we conclude that the difference between M_1 & M_2 is **statistically significant**
  - Chose model with lower error rate

Hyperparameter Tuning & Model Optimization

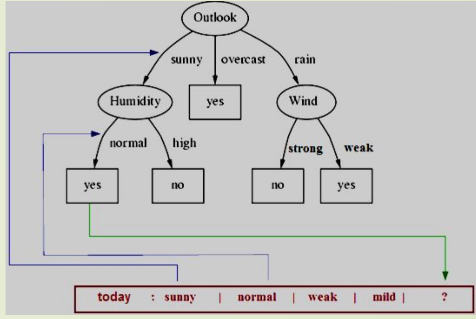
❑ Model Parameters Versus Hyperparameters

- A **model parameter** is a configuration variable that is internal to the model and whose value can be estimated from data automatically during training.
- Examples:
  - The weights in an ANN.
  - The support vectors in a SVM.
  - The coefficients in a linear regression.
- A **hyperparameter** is a configuration that is external to the model and whose value cannot be estimated from data. They are set manually and tuned
- Some examples of model hyperparameters include:
  - The learning rate for training a neural network.
  - The C and sigma hyperparameters for support vector machines.
  - The k in k-nearest neighbors.
- ***Search for the best value of hyperparameter by trial and error***

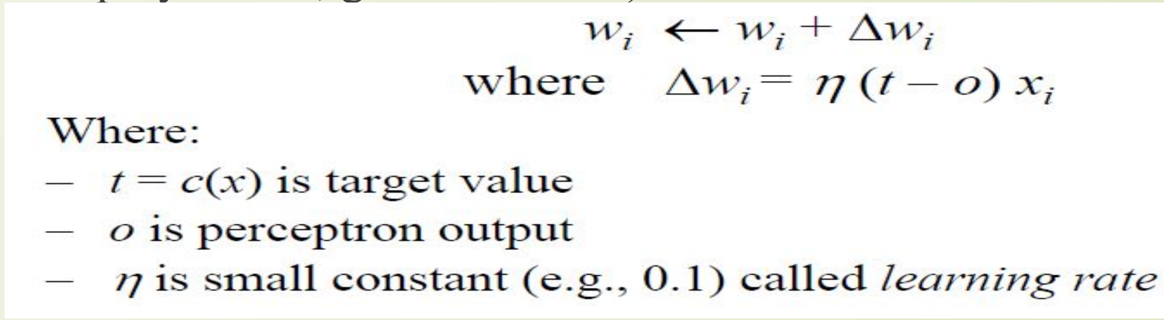
❖ **Baseline model with default parameters**

❖ **Optimized model with the best hyperparameters**

Hyperparameters: Examples



- Support Vector Machine
  - Regularization/complexity constant (If the complexity parameter C is high, we get a hard margin. Otherwise, if C is close to 0, the SVM margin is very soft and almost not caring about proper division of data.)
  - Kernel Function and its parameters (e.g. exponent if polynomial; gamma if RBF)
- Neural Network
  - No of hidden layers
  - Learning Rate



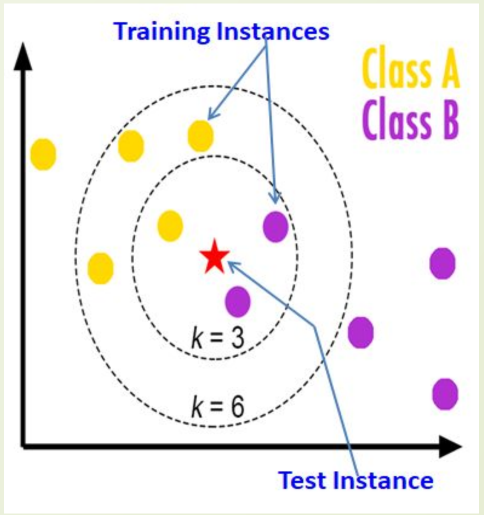
$$w_i \leftarrow w_i + \Delta w_i$$

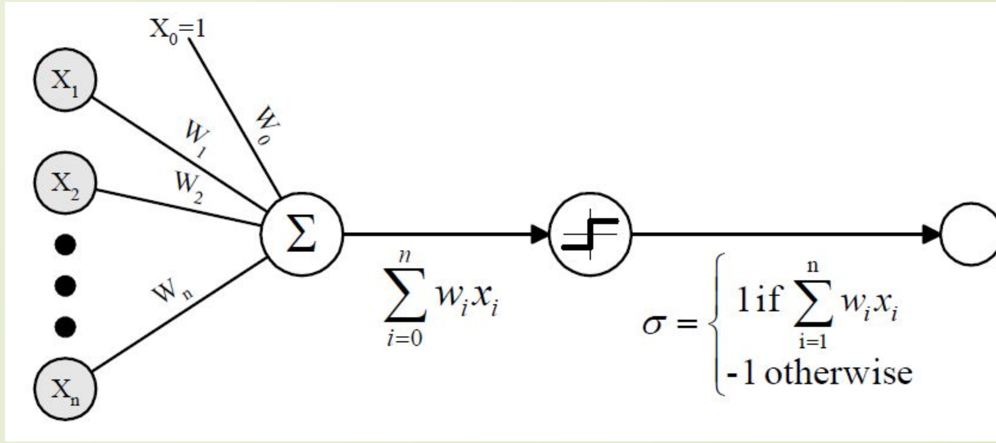
$$\text{where} \quad \Delta w_i = \eta\,(t - o)\,x_i$$

Where:

– $t = c(x)$ is target value
– o is perceptron output
– η is small constant (e.g., 0.1) called *learning rate*

- Decision Tree
  - Depth of tree
  - Minimum number of instances required at a leaf node
  - Confidence Factor for Pruning Tree (smaller value more pruned tree)
- Random Forest
  - Depth of a tree
  - No of trees in the Forest
- K-Nearest Neighbor
  - K value and Distance measure




$X_0=1$
X_1
X_2
X_n
W_0
W_1
W_2
W_n
Σ
$\sum_{i=0}^{n} w_i x_i$
$\sigma = \begin{cases} 1 \text{ if } \sum_{i=1}^{n} w_i x_i \\ -1 \text{ otherwise} \end{cases}$

THANK YOU

Any questions…?