



Enhancing Machine Learning Model Evaluation with Bootstrapping Techniques

Computational Statistics

Jennifer Zhuang

Fall 2024



Introduction

- Motivation
- Core Problem
- Solution Overview

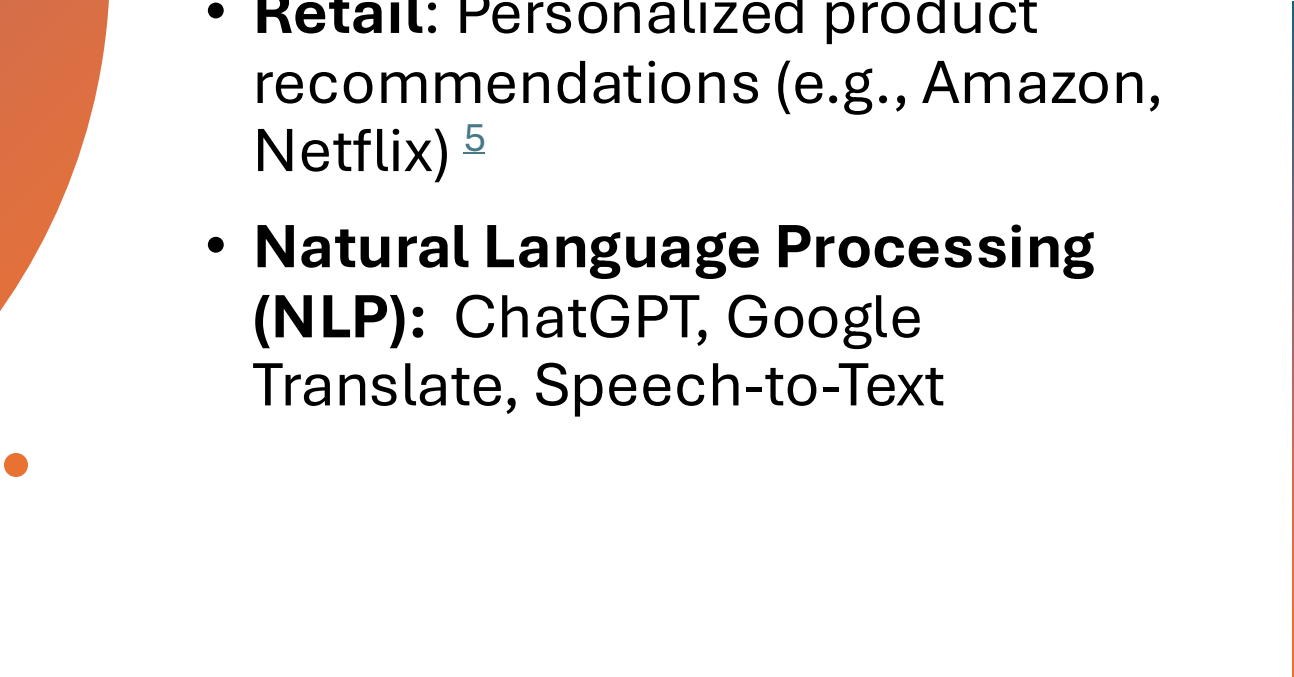


Motivation: What is Machine Learning?

- A subset of artificial intelligence that allows computers to learn from and make predictions or decisions based on data without being explicitly programmed ¹
- **Key Components**
 - **Data:** The foundation for learning patterns and relationships.
 - **Algorithms:** The processes used to analyze data and identify patterns.
 - **Models:** The resulting mathematical representations trained on data
- **What is a ML Model?**
 - Mathematical frameworks or algorithms designed to analyze patterns in data, make predictions, or support decision-making
 - Examples: Linear Regression, Decision Trees, Neural Networks

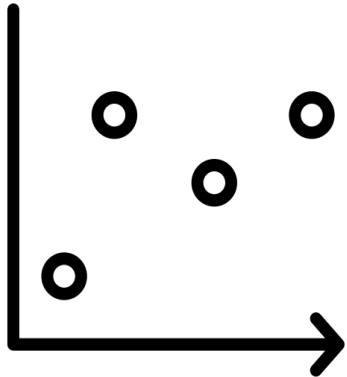


Some Real World Examples

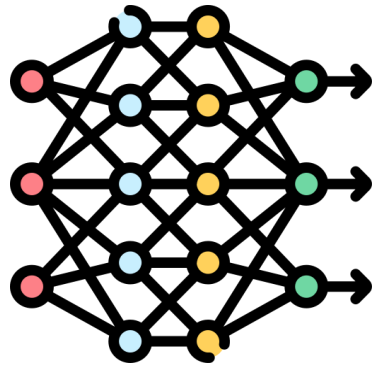
- **Healthcare:** Diagnosing diseases from medical images (e.g., X-rays, MRIs) ²
 - **Finance:** Fraud detection in credit card transactions ³ and predicting stock market trends ⁴
 - **Retail:** Personalized product recommendations (e.g., Amazon, Netflix) ⁵
 - **Natural Language Processing (NLP):** ChatGPT, Google Translate, Speech-to-Text
- 

Motivation: The Machine Learning Workflow

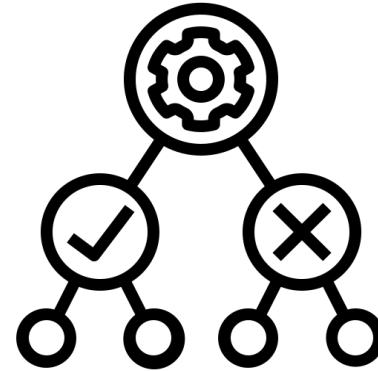
Input Data



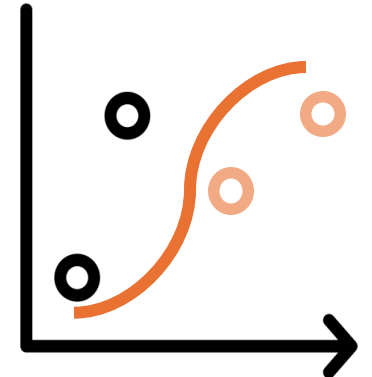
Model Training



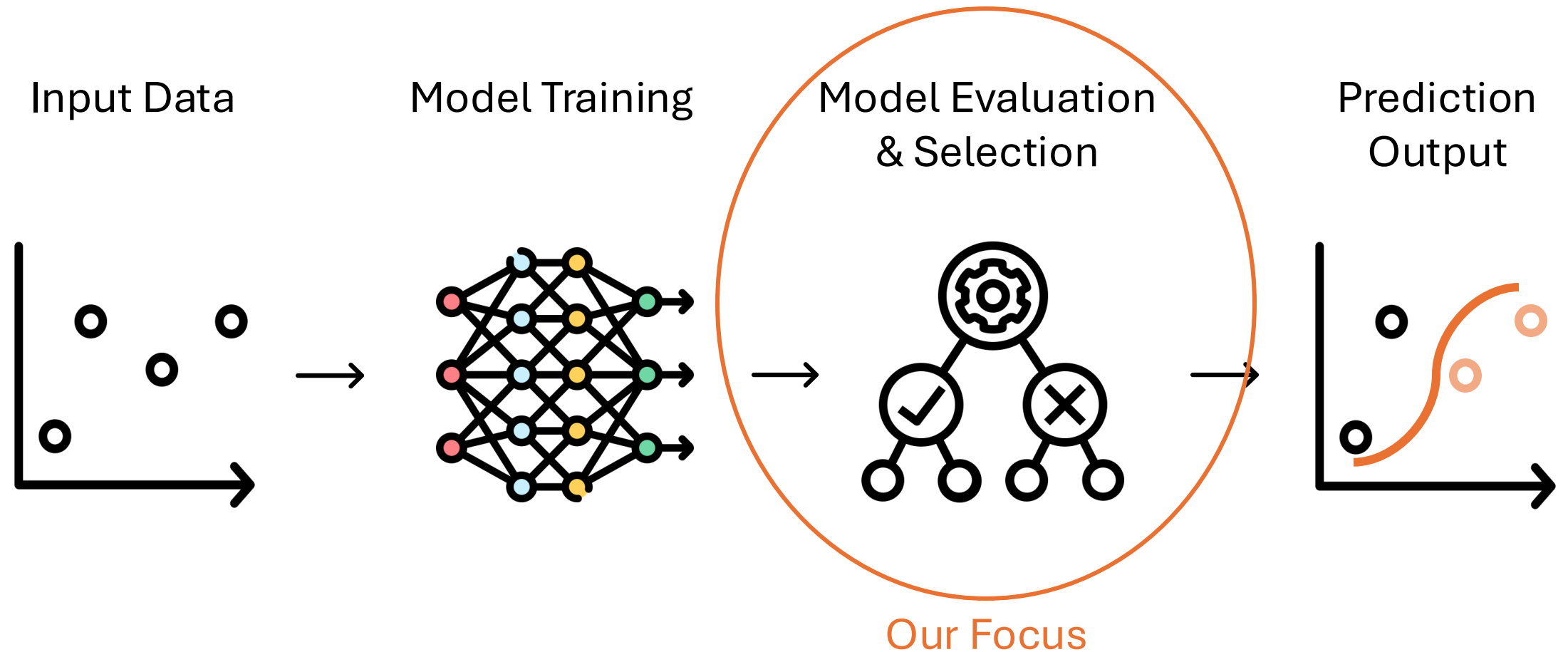
Model Evaluation
& Selection



Prediction
Output

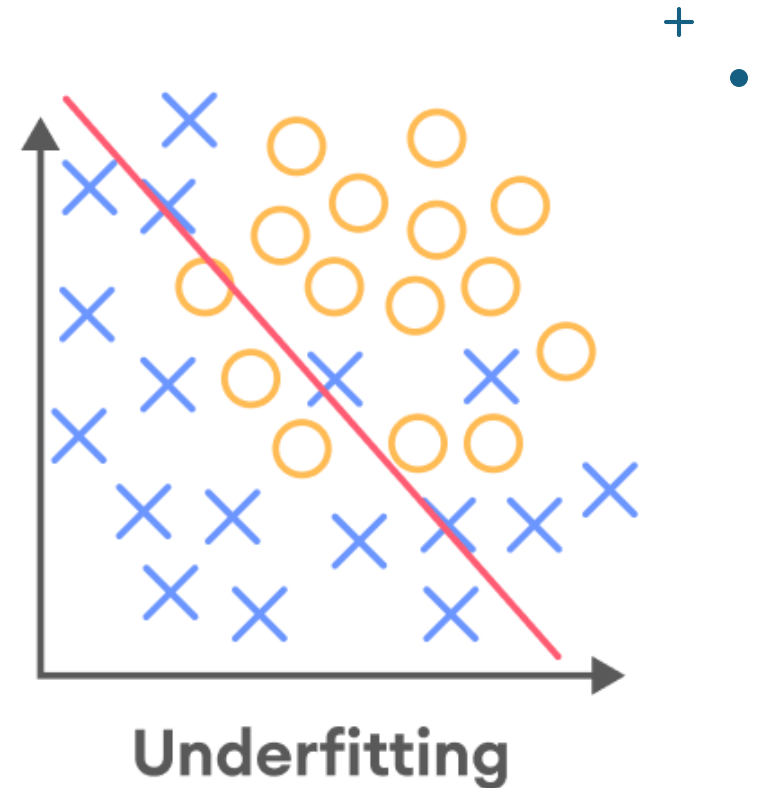


Motivation: The Machine Learning Workflow

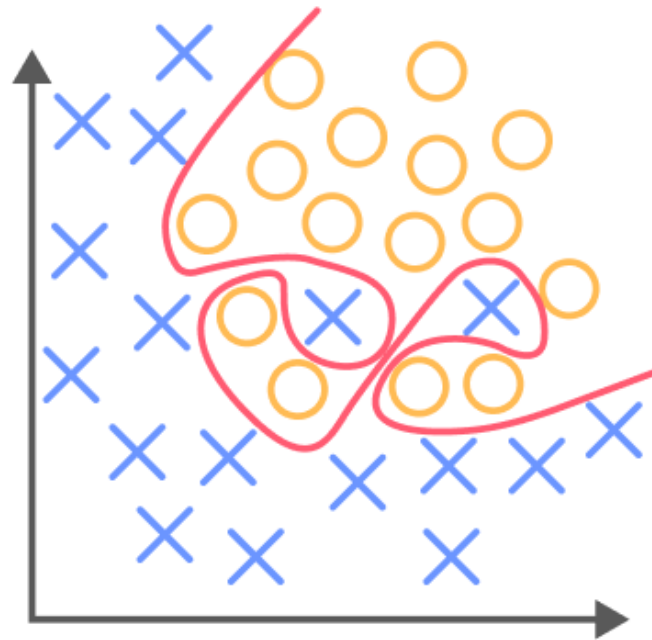


Motivation: Why Model Evaluation Matters

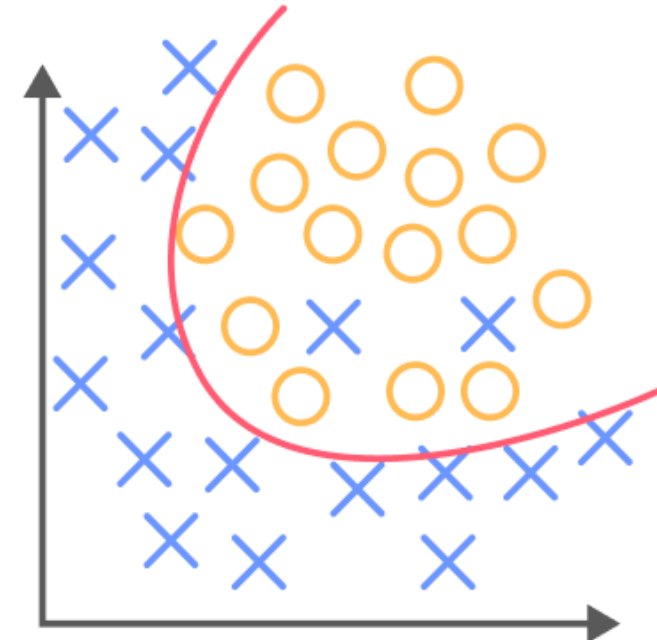
- **Definition:** The process of assessing a model's performance to ensure it generalizes well to unseen data ⁵
- Poor evaluation leads to models that fail to capture patterns (underfitting) or over-specialize to training data (overfitting)
- Robust evaluation ensures models are reliable and their predictions can be trusted
- Helps select the best ML models by comparing performance metrics



By using robust evaluation methods, we can avoid common pitfalls and ensure our ML models perform well on unseen data [6](#)



Overfitting



Appropriate fitting

+

•

○

Core Problem: Limitations of Traditional Evaluation Methods

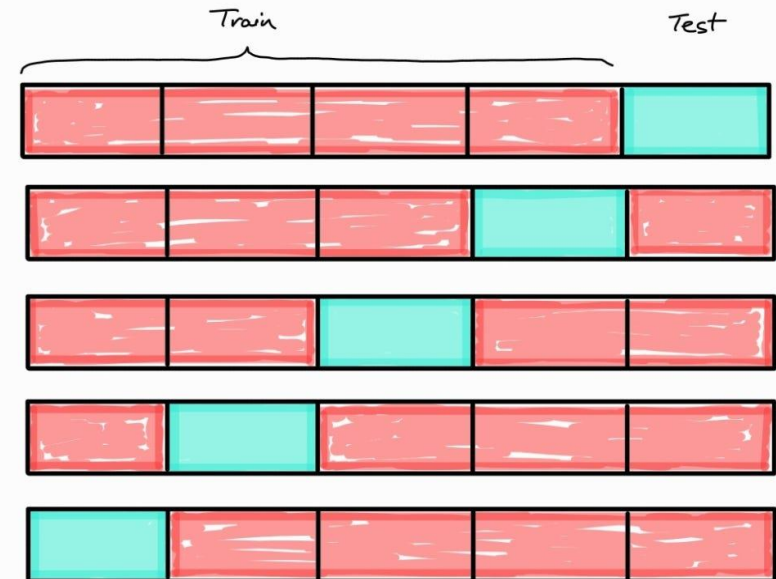
- **Performance Metrics:**
Accuracy, F1 Score, Precision, Recall, etc.

- **Approach:**

- Single Train-Test Split



- K-Folds Cross Validation



Performance Metrics

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}} = \frac{TP + TN}{TP + TN + FP + FN}$$

$$\text{Precision} = \frac{\text{True Positives (TP)}}{\text{True Positives (TP)} + \text{False Positives (FP)}}$$

$$\text{Recall} = \frac{\text{True Positives (TP)}}{\text{True Positives (TP)} + \text{False Negatives (FN)}}$$

$$\text{F1 Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

Core Problem: Limitations of Traditional Methods



- Performance metrics are highly dependent on how data is divided
- Can lead to biased or misleading results
- Single metrics may not capture the full range of performance across different data samples
- Cross-validation assumes data points in each fold are independent, which is not always true
- Many statistical evaluation methods (t-tests, ANOVA, CI's) assume data follows a normal distribution, which may not hold in real-world scenarios



Solution Overview:

Bootstrapping


*A resampling technique used to
enhance model evaluation
reliability*



- **Solves Core Problems**

- Reduces bias caused by single train-test splits
- Captures a full range of performance across resampled datasets
- Avoids assumptions about data independence or normality
- Provides a clearer picture of model stability and generalization

- **New Key Benefits**

- Provides confidence intervals for performance metrics
 - Maximizes the use of small datasets
 - Generates a distribution of metrics for deeper insights
- 

Bootstrapping

- Bootstrapping Explained
- Applying Bootstrapping to Performance Evaluation
- Statistical & Mathematical Foundations
- Leveraging Bootstrap Metrics



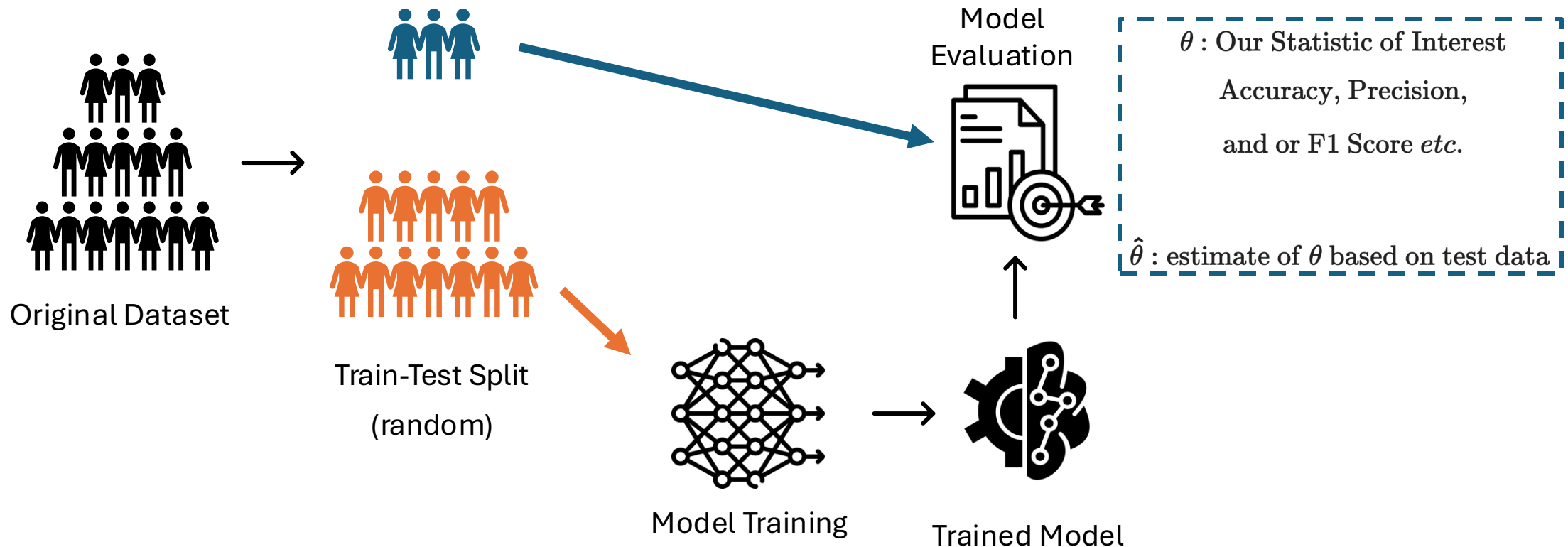
Bootstrapping Explained

+

- Definition: A statistical resampling method that creates multiple datasets by sampling with replacement from the original data
- Steps 7:
 - **Sampling with Replacement:** Randomly draw data points from the original dataset to create a new "bootstrap" sample
 - **Model Training:** Train the model on the bootstrap sample
 - **Model Testing:** On the remaining data, evaluate the performance of our model (our statistic of interest θ)
 - **Repeat:** Repeat the process multiple times to create an empirical distribution of θ

•

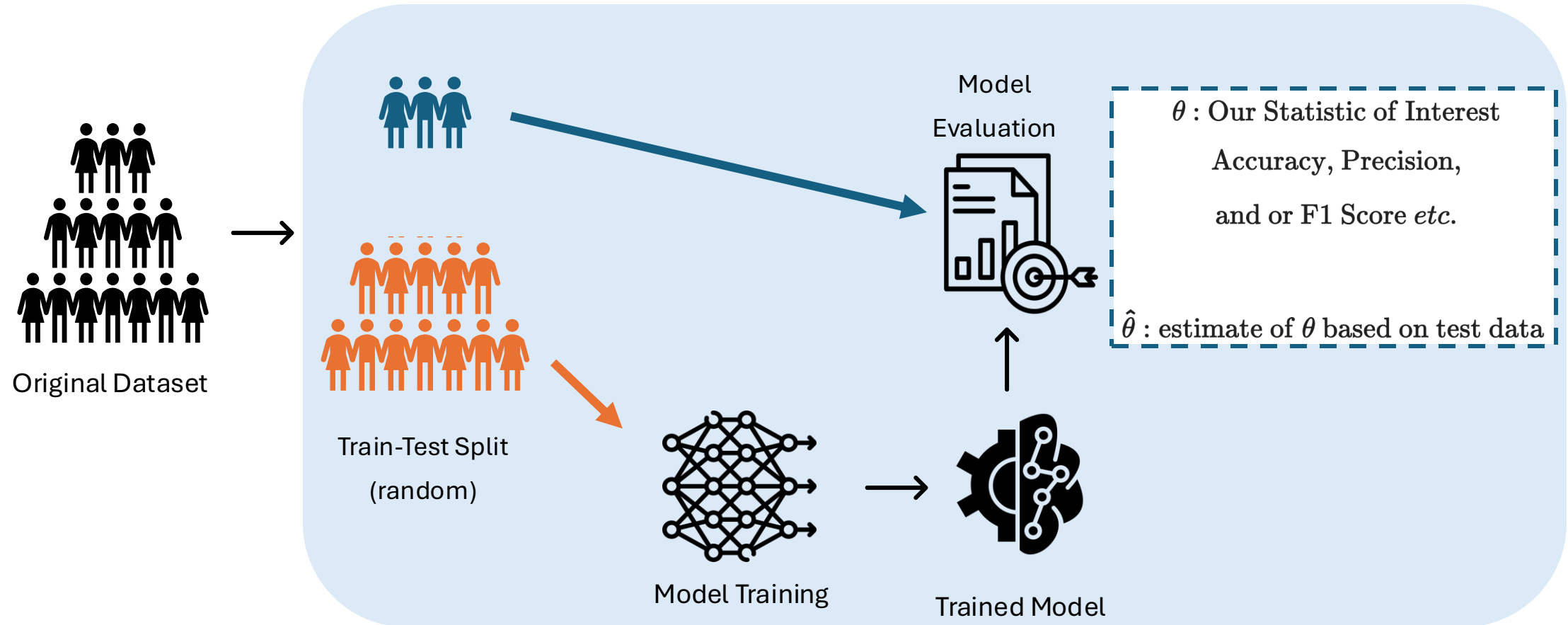
Performance Evaluation *Without* Bootstrapping



Performance Evaluation *With* Bootstrapping

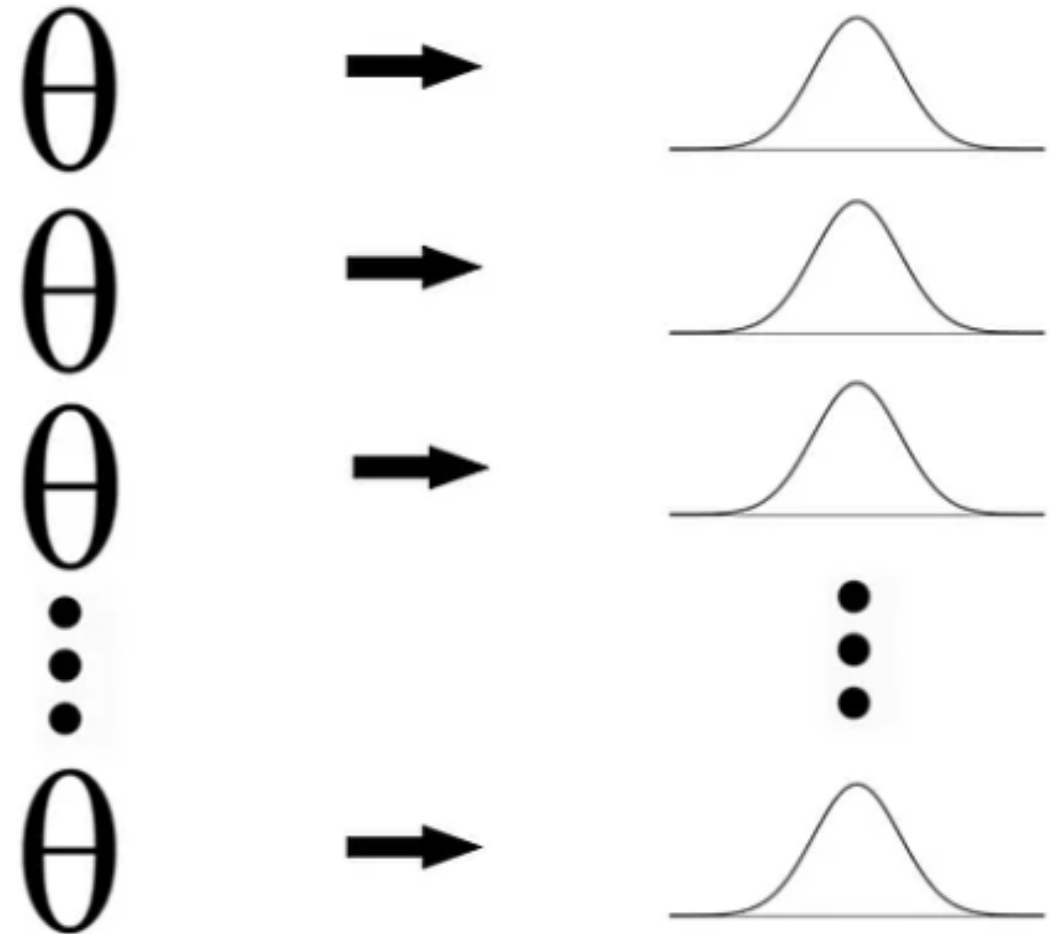


Repeat **Many** Times, Recording θ for each iteration



The Result: N Bootstrap Estimates of θ

- Repeated resampling produces N estimates of the statistic of interest θ and an empirical sampling distribution of θ ⁸
- Serves as a simulation of the underlying population distribution⁸
- We do not need to know the true underlying population distribution (which we rarely have access to)⁸
- From these simulations, we can estimate important properties (variance or CI's) about the population⁸



Statistical & Mathematical Foundations

1. Bootstrap Resampling

Given a dataset $D = \{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$, create bootstrap samples D_b^* by sampling with replacement:

$$D_b^* = \{(x_1^*, y_1^*), (x_2^*, y_2^*), \dots, (x_n^*, y_n^*)\}, \quad b = 1, \dots, B$$

where B is the total number of bootstrap samples.

2. Train and Evaluate Model

For each bootstrap sample D_b^* :

1. Train the model f_b on D_b^* .
2. Evaluate the model on the **out-of-bag (OOB)** data, D_{OOB}^b , and calculate the performance metric T_b^* (e.g., accuracy, precision).

$$T_b^* = g(D_{\text{OOB}}^b, f_b)$$

where g is the function that computes the performance metric.

Statistical & Mathematical Foundations



3. Empirical Distribution of the Metric

After B bootstrap samples, the empirical distribution of the performance metric is:

$$\{T_1^*, T_2^*, \dots, T_B^*\}$$

This approximates the sampling distribution of the performance metric (e.g., accuracy).

4. Bootstrapped Mean and Variance

Mean (average performance metric):

$$\bar{T}^* = \frac{1}{B} \sum_{b=1}^B T_b^*$$

Variance (variability of the metric):

$$\text{Var}(T^*) = \frac{1}{B-1} \sum_{b=1}^B (T_b^* - \bar{T}^*)^2$$

Statistical & Mathematical Foundations

5. Confidence Interval (Percentile Method)

To construct a $100(1 - \alpha)\%$ confidence interval for the performance metric:

$$\text{CI} = \left[T_{\alpha/2}^*, T_{1-\alpha/2}^* \right]$$

where $T_{\alpha/2}^*$ and $T_{1-\alpha/2}^*$ are the $\alpha/2$ -th and $(1 - \alpha/2)$ -th percentiles of the bootstrap distribution.



Leveraging Bootstrap Metrics

- **Mean:** Gauges the average performance of ML model across bootstrap samples
- **Variance:** Reveals the stability and consistency of the performance metric ⁸[9](#)
- **Confidence Interval (CI):** Provides a range for the expected performance, aiding in comparison and selection of the most reliable ML model ⁸

Coding Demonstration



- Our Dataset
- Our ML Models
- Traditional Methods
- Bootstrapping Methods
- Visualizations

+

•

○

Our Dataset: Breast Cancer Diagnosis

- A binary classification dataset used to predict whether a tumor is malignant or benign
- Derived from digitized images of biopsies/cell samples from abnormal areas of the body
- Our Goal: Predict whether a tumor is malignant (0) or benign (1).
- Motivation:
 - Widely recognized and benchmarked in machine learning, making it an ideal reference
 - The features are meaningful and easy to understand
 - Demonstrates how ML can assist in real world decision making

#	Column	Non-Null Count	Dtype
0	mean radius	569 non-null	float64
1	mean texture	569 non-null	float64
2	mean perimeter	569 non-null	float64
3	mean area	569 non-null	float64
4	mean smoothness	569 non-null	float64
5	mean compactness	569 non-null	float64
6	mean concavity	569 non-null	float64
7	mean concave points	569 non-null	float64
8	mean symmetry	569 non-null	float64
9	mean fractal dimension	569 non-null	float64
10	radius error	569 non-null	float64
11	texture error	569 non-null	float64
12	perimeter error	569 non-null	float64
13	area error	569 non-null	float64
14	smoothness error	569 non-null	float64
15	compactness error	569 non-null	float64
16	concavity error	569 non-null	float64
17	concave points error	569 non-null	float64
18	symmetry error	569 non-null	float64
19	fractal dimension error	569 non-null	float64
20	worst radius	569 non-null	float64
21	worst texture	569 non-null	float64
22	worst perimeter	569 non-null	float64
23	worst area	569 non-null	float64
24	worst smoothness	569 non-null	float64
25	worst compactness	569 non-null	float64
26	worst concavity	569 non-null	float64
27	worst concave points	569 non-null	float64
28	worst symmetry	569 non-null	float64
29	worst fractal dimension	569 non-null	float64
30	Target	569 non-null	int64

A Quick Look

- Contains 569 samples and 30 numerical features.
- Features:
 - 30 numeric features derived from cell nuclei measurements
 - Mean radius, texture, area, smoothness, compactness, etc.
- Target Class Counts:
 - Benign: 357 (62.7%)
 - Malignant: 212 (37.3%)

Our Machine Learning Models

- The following are the machine learning models we will use for analysis and a brief description:
 - **Logistic Regression:** A simple and interpretable linear model that predicts the probability of a binary outcome.
 - **Random Forest:** An ensemble learning method that combines multiple decision trees for robust and accurate predictions.
 - **K Nearest Neighbors:** A non-parametric model that classifies data points based on the majority class of their nearest neighbors.
 - **Neural Network:** A multi-layer model capable of capturing complex patterns in data, often used for high-dimensional and non-linear problems.

Traditional Methods

Single Train-Test Split

- Perform an 80/20 split of the data (80% goes to training, 20% goes to testing)
- Models are trained and tested once, providing one set of performance metrics

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42, stratify=y)
```

```
from sklearn.model_selection import cross_validate, StratifiedKFold
cv = StratifiedKFold(n_splits=5, shuffle=True, random_state=42)
```

5-Fold Cross Validation

- Splits the dataset into 5 folds
- Each iteration the models are trained on 4 of the folds, and tested on the remaining 1
- Produces performance metrics for each iteration

Traditional Methods: Results

Single Train-Test Split

- Perform an 80/20 split of the data (80% goes to training, 20% goes to testing)

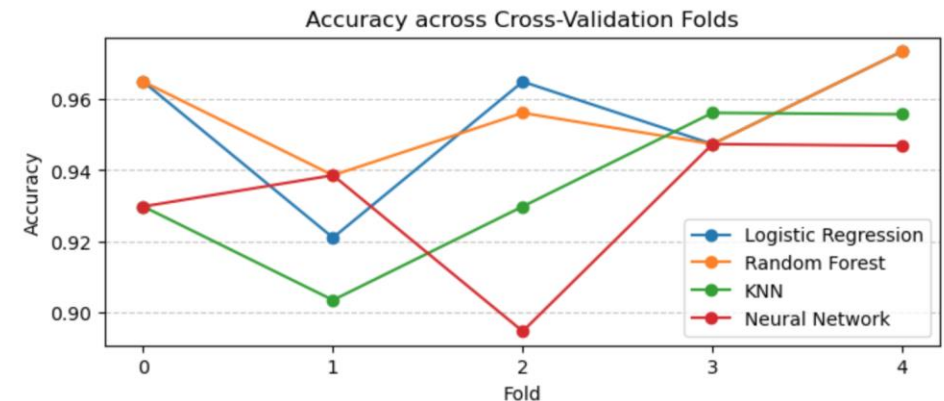
Model	Accuracy	Precision	Recall	F1-Score
Logistic Regression	0.964912	0.959459	0.986111	0.972603
Random Forest	0.956140	0.958904	0.972222	0.965517
KNN	0.956140	0.958904	0.972222	0.965517
Neural Network	0.964912	0.985714	0.958333	0.971831

- Models are trained and tested once, providing one set of performance metrics

5-Fold Cross Validation

- Performance Metrics Mean & SD

Accuracy (Avg \pm SD)	Precision (Avg \pm SD)	Recall (Avg \pm SD)	F1-Score (Avg \pm SD)
0.95 \pm 0.02	0.96 \pm 0.03	0.97 \pm 0.03	0.96 \pm 0.01
0.96 \pm 0.01	0.97 \pm 0.03	0.97 \pm 0.03	0.97 \pm 0.01
0.94 \pm 0.02	0.94 \pm 0.03	0.96 \pm 0.02	0.95 \pm 0.01
0.93 \pm 0.02	0.93 \pm 0.01	0.96 \pm 0.03	0.95 \pm 0.02



Traditional Methods: Results

Single Train-Test Split

- Perform an 80/20 split of the data (80% goes to training, 20% goes to testing)

Model	Accuracy	Precision	Recall	F1-Score
Logistic Regression	0.964912	0.959459	0.986111	0.972603
Random Forest	0.956140	0.958904	0.972222	0.965517
KNN	0.956140	0.958904	0.972222	0.965517
Neural Network	0.964912	0.985714	0.958333	0.971831

- Models are trained and tested once, providing one set of performance metrics

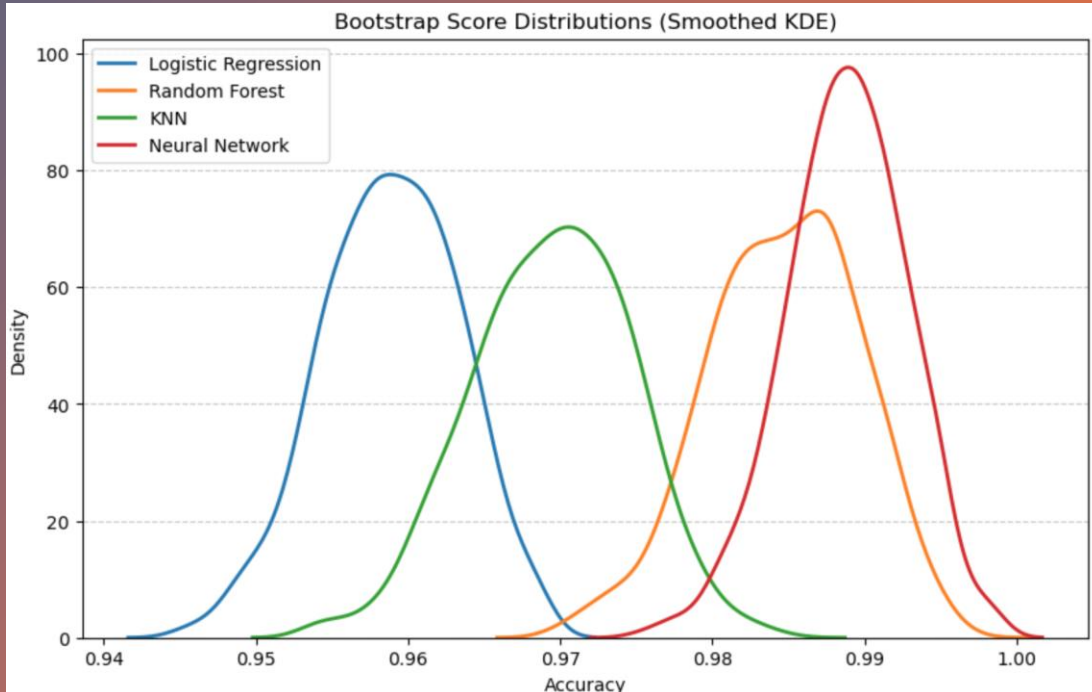
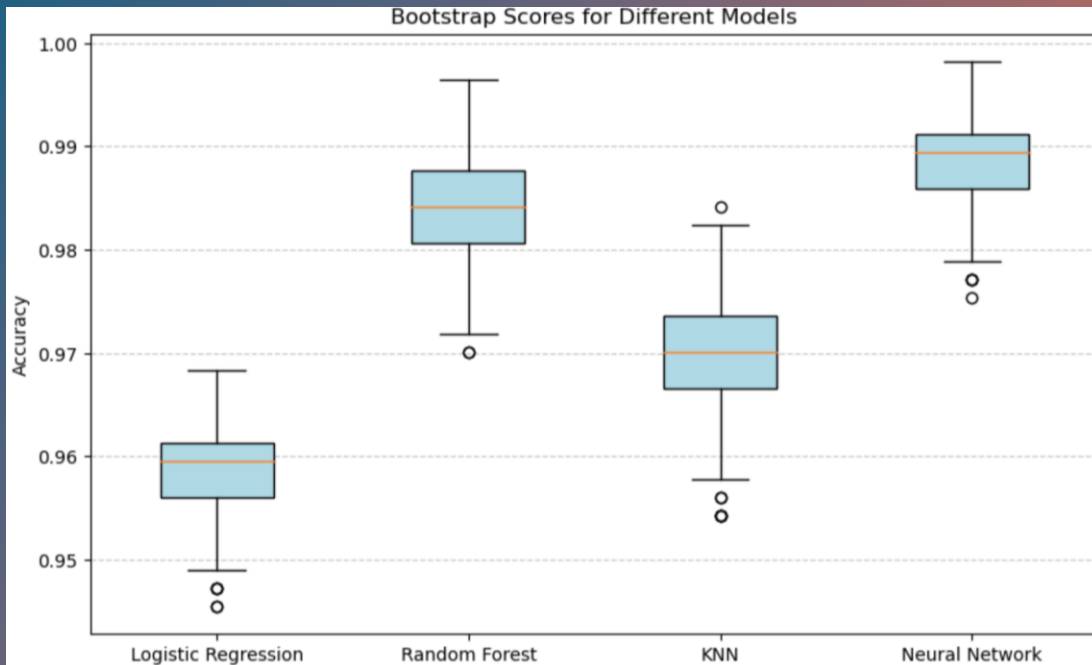
Comments:

- A train-test split can either over or underestimate a model's performance due to a favorable or unfavorable division of the data
- That is the fluctuations in score we see
- Also, while CV provides a mean and standard deviation of performance metrics, these values are descriptive and cannot be used to make formal statistical inferences
- i.e. We cannot establish whether the model's performance is statistically significant compared to another



Bootstrapping Results

- Next, we performed bootstrapping with $n = 500$ resamples
- Displayed are the boxplots and smoothed kernel density estimates of the models
- Summary Statistics:



Model	Mean Accuracy	Std Deviation	Min	Q1	Q2	Q3	Max
Logistic Regression	0.96	0.00	0.95	0.96	0.96	0.96	0.97
Random Forest	0.98	0.00	0.97	0.98	0.98	0.99	1.00
KNN	0.97	0.01	0.95	0.97	0.97	0.97	0.98
Neural Network	0.99	0.00	0.98	0.99	0.99	0.99	1.00

Bootstrap Inference: Percentile Method



- A bootstrap $1 - \alpha$ confidence interval (CI) based on the percentile method could be constructed by finding the $[(1 - \alpha/2)100]$ th and $[(\alpha/2)100]$ th percentiles in the empirical distribution¹⁰

- Below are the **95% confidence intervals (CI)** for accuracy for each model:

Model	95% CI Lower	95% CI Upper
Logistic Regression	0.949	0.967
Random Forest	0.974	0.993
KNN	0.960	0.979
Neural Network	0.981	0.996

- Interpretation:
 - Using these CI, we can make statements like
 - "With 95% confidence, the classification accuracy of our logistic regression model on unseen data lies between 94.9% and 96.7%"
 - If the confidence intervals (CIs) models do not overlap, we can infer that there is likely a difference in performance
 - "With 95% confidence, we can say that our neural network performed better than logistic regression"

Bootstrap Inference: Variance Analysis

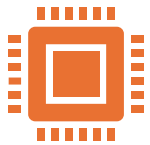


- Bootstrap resampling provides insights into the variability of model accuracy, offering a measure of stability and consistency [9](#)
- Variance analysis helps identify robust models that generalize well to unseen data, avoiding overconfidence in performance metrics derived from single splits
- Below are the variance in accuracy for each model:

Model	Variance
Logistic Regression	0.000020
Random Forest	0.000024
KNN	0.000027
Neural Network	0.000016

- Interpretation:
 - Our neural network exhibits low variance in accuracy, indicating consistent performance across different resampled datasets

Implementation Challenges to Note



Computational Overhead

Challenge: High run time and memory usage, especially for large datasets.

Example: Bootstrapping a 1-million-row dataset with 1000 iterations can take hours without optimized computing resources.



3. Percentile Method Drawbacks

Challenge: Prone to bias and inaccurate confidence interval coverage.

Example: Bootstrapping variance estimates without a variance-stabilizing transformation can result in overly wide or inaccurate confidence intervals.



4. Small Dataset Limitations

Challenge: Bootstrapping does not solve the problem of insufficient data.

Example: If the dataset is too small, bootstrap samples may not capture the true variability, leading to unreliable inferences.



5. General Challenges

Sensitive to Outliers:

- Example:** A single extreme value in the dataset can appear repeatedly in bootstrap samples, skewing the results.

Complexity with Non-Location Parameters:

- Example:** Metrics like correlation coefficients or skewness require additional transformations to stabilize their distributions