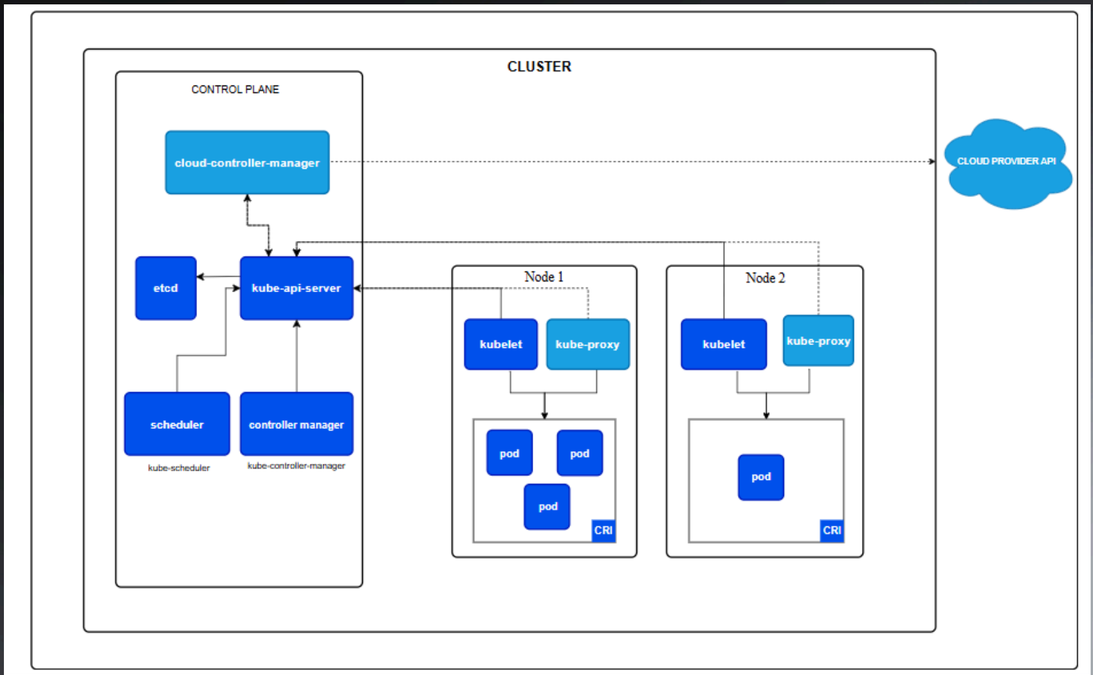
FINAL PROJECT : DATS 6103

Predicting Kubernetes Pod Overload Using Machine Learning

ADITI SHUKLA, ASHLEY GYAPOMAH

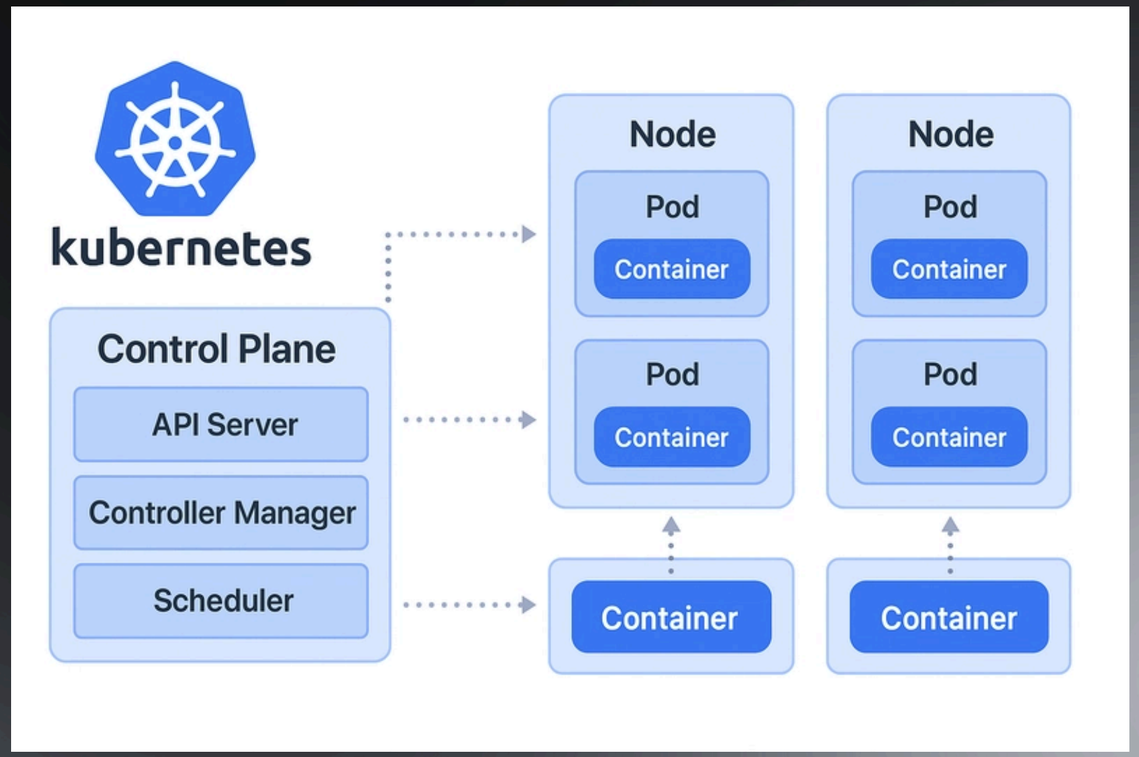
WHAT IS KUBERNATES?



KUBERNATES ARCHITECTURE

- KUBERNETES IS AN APPLICATION USED TO ORCHESTRATE AND MANAGE CONTAINERS ACROSS MANY MACHINES.
- EXAMPLE NETFLIX SCALING AUTOMATICALLY WHEN USERS ARE HIGHER

CORE CONCEPT: CONTAINER ORCHESTRATION



WHAT IS A CONTAINER?

- CONTAINS AND APPLICTAIONS CODE, RUNTIME, SYSTEM LIBRAIES AND SETTINGS

WHAT IS ORCHESTRATION?

- THE PROCESS OF AUTOMATICALLY ORGANIZING AND MANAGING THOUSANDS OF CONTAINERS

PROBLEM STATEMENT AND SMART QUESTION

CAN WE BUILD A MACHINE LEARNING MODEL THAT PREDICTS POD OVERLOAD (NEED_NEW_POD = 1) USING ONLY RESOURCE ALLOCATION METRICS?

CURRENT SYSTEM AUTOMATICALLY CREATES A NEW POD ONLY AFTER THE OVERLOAD HAPPENS.
PROBLEM: THIS REACTION IS SLIGHTLY DELAYED → USERS EXPERIENCE LAG/BUFFERING.
GOAL: PREDICT OVERLOAD BEFORE IT HAPPENS SO SCALING CAN OCCUR PROACTIVELY.

SPECIFIC → CPU & MEMORY METRICS
MEASURABLE → F1, ACCURACY, ROC-AUC
ACHIEVABLE → WE ACHIEVED > 95% F1
RELEVANT → DIRECTLY RELATES TO KUBERNETES AUTOSCALING
TIME-BOUND → COMPLETED WITHIN THE PROJECT TIMELINE

DATASET OVERVIEW

DATASET: KUBERNETES RESOURCE ALLOCATION DATASET FROM KAGGLE

```
0   pod_name
1   namespace
2   cpu_request
3   cpu_limit
4   memory_request
5   memory_limit
6   cpu_usage
7   memory_usage
8   node_name
9   pod_status
10  restart_count
11  uptime_seconds
12  deployment_strategy
13  scaling_policy
14  network_bandwidth_usage
```

RESOURCE METRICS

- CPU_REQUEST – CPU REQUESTED BY THE POD
- CPU_LIMIT – MAX CPU ALLOWED
- CPU_USAGE – ACTUAL CPU USED
- MEMORY_REQUEST – MEMORY REQUESTED
- MEMORY_LIMIT – MAX MEMORY ALLOWED
- MEMORY_USAGE – ACTUAL MEMORY USED

POD METADATA

- POD_NAME
- NAMESPACE
- NODE_NAME
- POD_STATUS
- DEPLOYMENT_STRATEGY
- SCALING_POLICY
- OTHER SYSTEM FEATURES

RESTART_COUNT

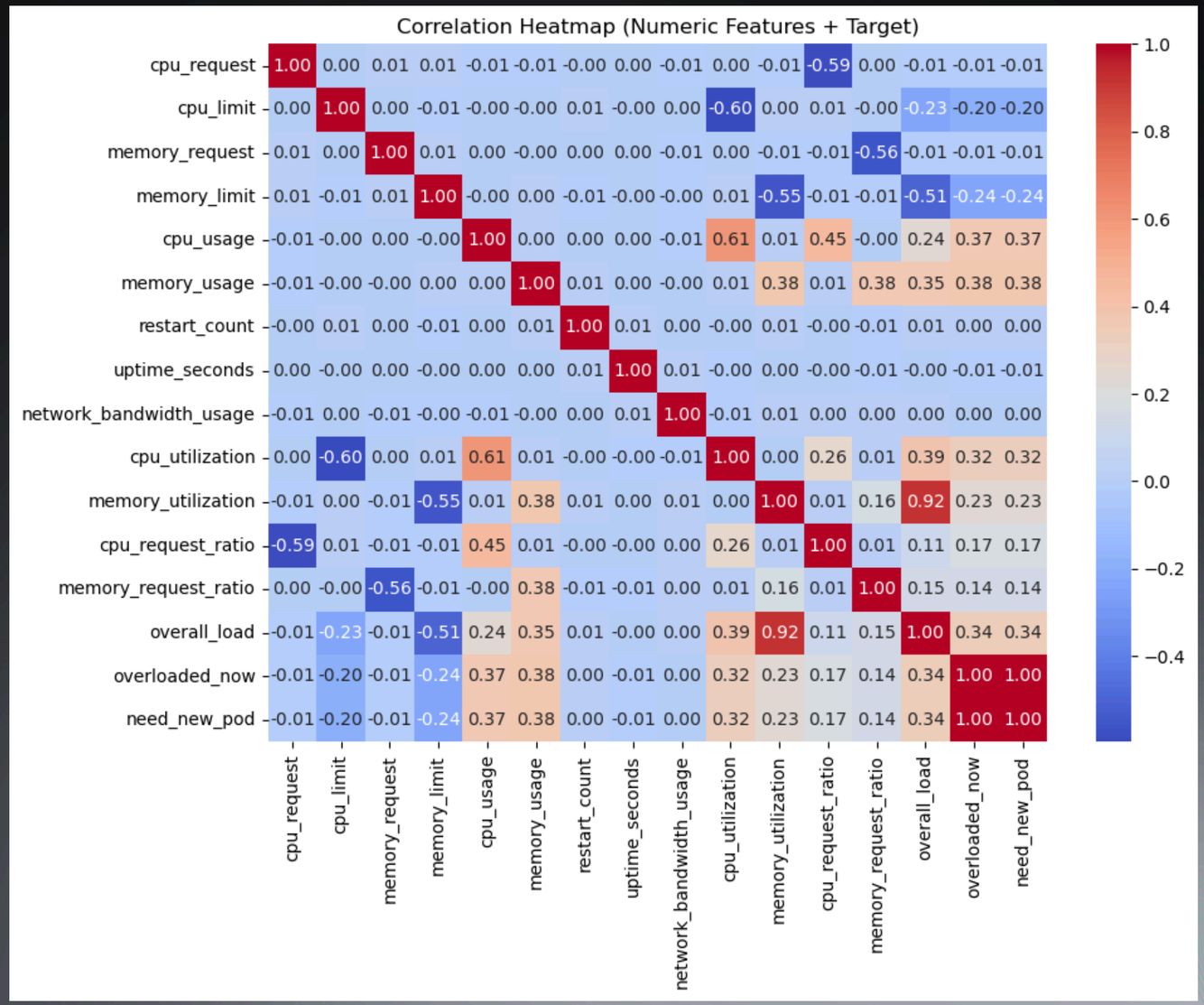
- UPTIME_SECONDS
- NETWORK_BANDWIDTH_USAGE

DATA PREPROCESSING & CLEANING

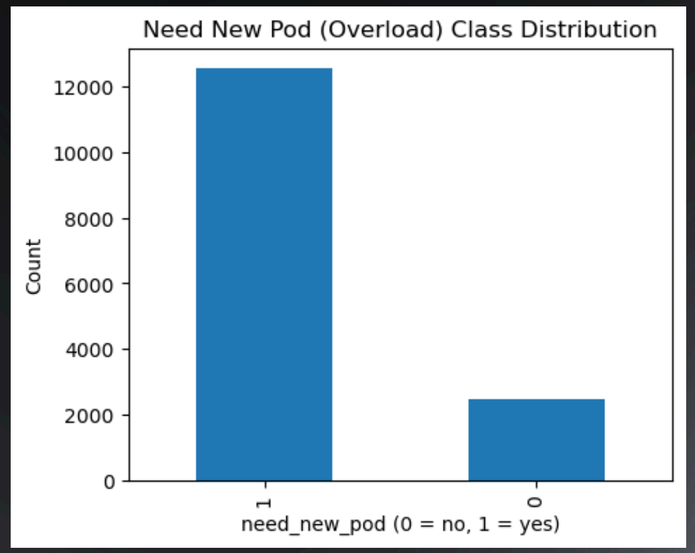
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 15000 entries, 0 to 14999
Data columns (total 15 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   pod_name                15000 non-null  object
 1   namespace               15000 non-null  object
 2   cpu_request             15000 non-null  float64
 3   cpu_limit               15000 non-null  float64
 4   memory_request          15000 non-null  float64
 5   memory_limit            15000 non-null  float64
 6   cpu_usage               15000 non-null  float64
 7   memory_usage            15000 non-null  float64
 8   node_name               15000 non-null  object
 9   pod_status              15000 non-null  object
 10  restart_count           15000 non-null  int64
 11  uptime_seconds          15000 non-null  int64
 12  deployment_strategy     15000 non-null  object
 13  scaling_policy          15000 non-null  object
 14  network_bandwidth_usage  15000 non-null  float64
dtypes: float64(7), int64(2), object(6)
...
deployment_strategy        0
scaling_policy             0
network_bandwidth_usage    0
dtype: int64
```

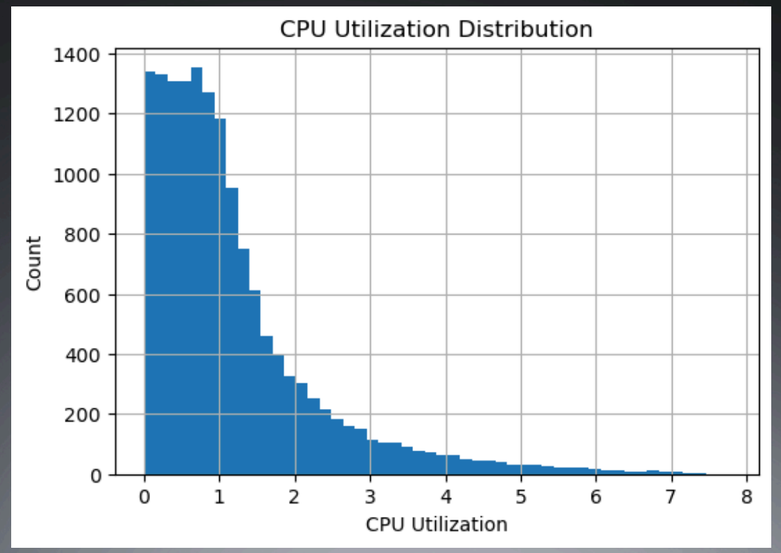
- ALL COLUMNS HAVE 15,000 NON-NULL ENTRIES
- 0 MISSING VALUES
- CHECKED FOR NEGATIVE VALUES IN COLUMNS BECAUSE IT WOULD BE INVALID
- NORMALIZED COLUMN NAMES TO LOWER CASE
- DATASET IS CLEAN AND COMPLETE

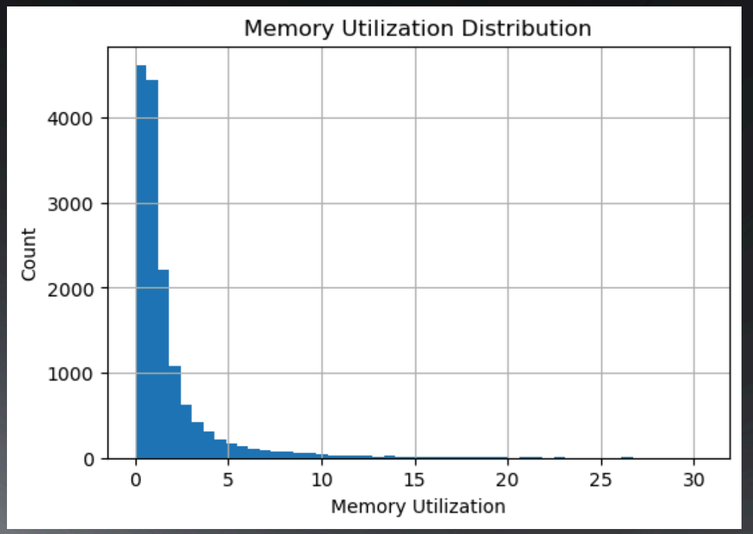
EXPLORATORY DATA ANALYSIS



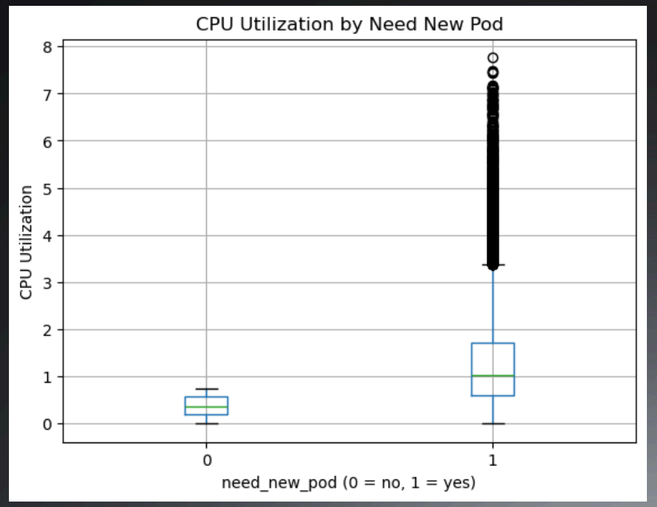
EXPLORATORY DATA ANALYSIS

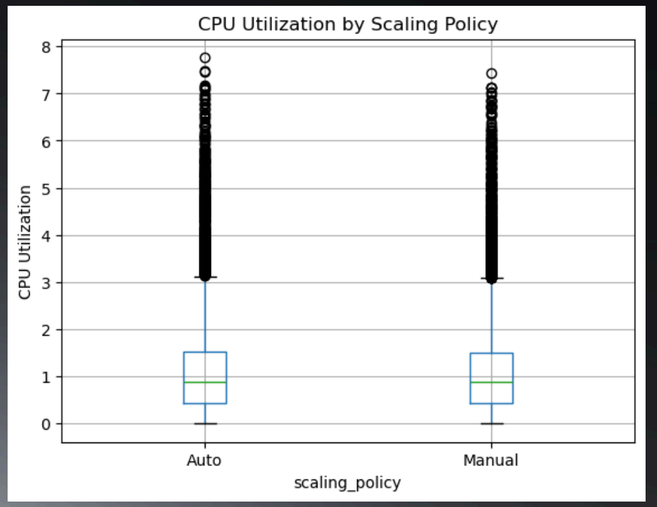


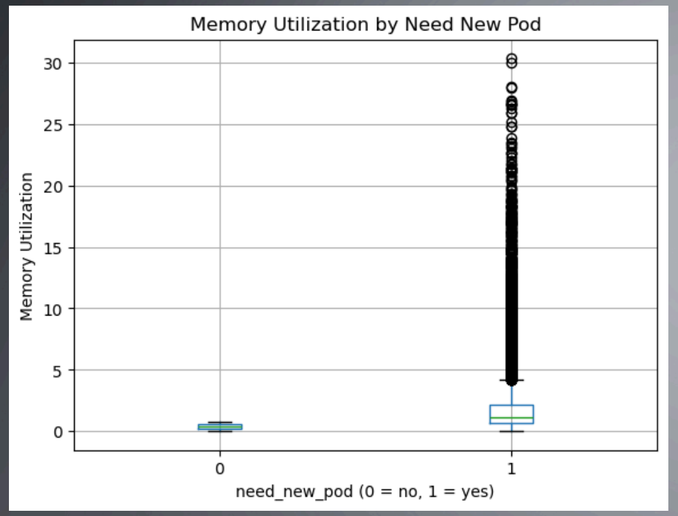


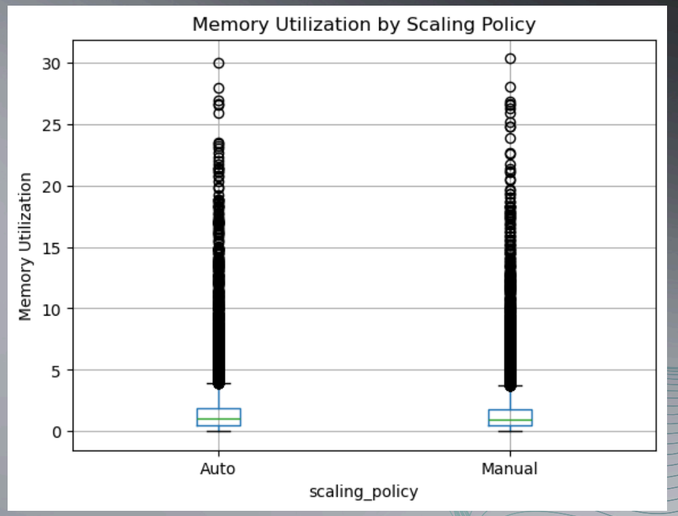


EXPLORATORY DATA ANALYSIS









STATISTICAL TESTING

```
ANOVA F-statistic: 0.7010766503854831
ANOVA P-value: 0.5512752358978207
No significant difference between groups.
```

```
T-statistic: 65.85529426785459
P-value: 0.0
The means are significantly different.
```

```
Chi-square: 3.5775707777555152
P-value: 0.05856473401330775
Variables are independent.
```

FEATURE ENGINEERING

- CPU & MEMORY UTILIZATION (USAGE ÷ LIMIT) :ROW 4 → CPU_UTILIZATION = 1.67 → CPU USAGE EXCEEDS THE ALLOWED LIMIT → RISK OF OVERLOAD
- USAGE VS REQUEST RATIO (USAGE ÷ REQUEST) : ROW 3 → CPU_REQUEST_RATIO = 12.04 → USED 12× MORE CPU THAN REQUESTED → VERY UNSTABLE POD
- COMBINED LOAD METRIC (OVERALL LOAD):ROW 1 → OVERALL_LOAD = 1.37 → HEAVY LOAD, LIKELY TO REQUIRE A NEW POD SOON.

	cpu_request	cpu_limit	cpu_usage	memory_request	memory_limit	memory_usage
0	1.569542	3.679152	3.345496	3174.582783	5134.413852	2135.310365
1	0.343119	3.722716	2.758188	3551.459173	3698.349366	7442.200271
2	0.249271	1.318147	1.319703	1578.313253	7418.271122	5142.897754
3	0.311497	2.852595	3.752312	1392.962372	3628.480705	2952.449331
4	1.532775	0.521618	0.874224	2660.192655	5091.497752	3382.299355

cpu_utilization	memory_utilization	cpu_request_ratio	memory_request_ratio	overall_load
0.909311	0.415882	2.131510	0.672627	0.662597
0.740907	2.012303	8.038545	2.095533	1.376605
1.001180	0.693274	5.294233	3.258477	0.847227
1.315403	0.813687	12.046042	2.119547	1.064545
1.675984	0.664303	0.570353	1.271449	1.170143

TARGET VARIABLE

Target Variable: scaling_event

Binary Target (0/1):
1 → Scaling needed (pod was overloaded / near limits)
0 → No scaling needed (pod running normally)

```
# Thresholds for overload
cpu_thresh = 0.75
mem_thresh = 0.75
```

MODEL TRAINING AND EVALUATION

```
Final modeling dataset shape: (15000, 19) (15000,)

Target distribution:
need_new_pod
1    12552
0     2448
Name: count, dtype: int64

Train/Test Split Complete
Training samples: 12000
Testing samples : 3000
```

Feature & Target Selection

- Selected key numeric features (CPU/memory usage, limits, ratios, load)
- Selected categorical features (namespace, status, scaling policy)

Final Modeling Dataset

- No missing values
- X = selected features
- y = need_new_pod

Train/Test Split

- 80% train, 20% test
- Stratified to preserve target class ratio
- 12,000 training samples
- 3,000 testing samples

- Scaled all the numeric values
- used ‘onehotencoder ‘ to convert categorical values in to numeric
- combined it using ‘columntransformer’

MODEL TRAINING AND EVALUATION

```
===== Training LogisticRegression =====
Accuracy : 0.9343
Precision: 0.9591
Recall   : 0.9625
F1-score : 0.9608
ROC-AUC : 0.9796

Classification report:
              precision    recall  f1-score   support

           0       0.80      0.79      0.80       490
           1       0.96      0.96      0.96      2510

    accuracy                           0.93      3000
   macro avg       0.88      0.88      0.88      3000
weighted avg       0.93      0.93      0.93      3000
```

```
===== Training DecisionTree =====
Accuracy : 0.9983
Precision: 0.9984
Recall   : 0.9996
F1-score : 0.999
ROC-AUC : 0.9969

Classification report:
              precision    recall  f1-score   support

           0       1.00      0.99      0.99       490
           1       1.00      1.00      1.00      2510

    accuracy                           1.00      3000
   macro avg       1.00      1.00      1.00      3000
weighted avg       1.00      1.00      1.00      3000
```

MODEL TRAINING AND EVALUATION

```
===== Training RandomForest =====
Accuracy : 0.9987
Precision: 0.9984
Recall   : 1.0
F1-score : 0.9992
ROC-AUC : 1.0

Classification report:
              precision    recall  f1-score   support

           0       1.00      0.99      1.00       490
           1       1.00      1.00      1.00      2510

    accuracy                           1.00      3000
   macro avg       1.00      1.00      1.00      3000
weighted avg       1.00      1.00      1.00      3000
```

```
===== Training GradientBoosting =====
Accuracy : 0.9997
Precision: 0.9996
Recall   : 1.0
F1-score : 0.9998
ROC-AUC : 0.9999

Classification report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       490
           1       1.00      1.00      1.00      2510

    accuracy                           1.00      3000
   macro avg       1.00      1.00      1.00      3000
weighted avg       1.00      1.00      1.00      3000
```

MODEL TRAINING AND EVALUATION

```
===== Training XGBoost =====
Accuracy : 0.9993
Precision: 0.9996
Recall   : 0.9996
F1-score : 0.9996
ROC-AUC : 1.0

Classification report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       490
           1       1.00      1.00      1.00      2510

    accuracy                           1.00      3000
   macro avg       1.00      1.00      1.00      3000
weighted avg       1.00      1.00      1.00      3000
```

MODEL TRAINING AND EVALUATION