Polynomial Feature Engineering for Analytical Ridge Regression: A Case Study in Aerospace Anomaly Detection

Ansh Mathur
SRM University KTR, Department of Computing Technologies
am3274@srmist.edu.in

Atrishman Mukherjee*
am7019@srmist.edu.in

Supratik Dey†
sd9107@srmist.edu.in

February 8, 2026

Abstract

We investigate the effectiveness of polynomial feature engineering when combined with analytical ridge regression for multi-class classification tasks. Using the NASA Shuttle dataset as a case study, we demonstrate that degree-4 polynomial features enable closed-form solutions to achieve 99.43% test accuracy in 45 milliseconds of training time. This accuracy matches or exceeds previously reported results while offering substantial computational advantages through elimination of iterative optimization. Our systematic evaluation across six feature configurations reveals that test accuracy improves monotonically from 87.33% with linear features to 99.43% with degree-4 polynomial interactions, representing a 12.10% absolute improvement. Generalization gaps remain below 0.3% across all tested configurations, indicating robust performance despite increased model capacity. These findings suggest that explicit polynomial feature expansion, when properly regularized, provides a computationally efficient alternative to iterative learning methods for problems with polynomial structure. We discuss the applicability of this approach to safety-critical aerospace applications where deterministic training guarantees and rapid model updates are valued.

1 Introduction

Modern supervised learning typically relies on iterative optimization methods that repeatedly pass through training data to minimize loss functions [1, 2]. Gradient descent and its variants have proven highly effective across diverse domains [3–5], enabling the training of increasingly complex models. However, these iterative approaches introduce computational costs that scale with both dataset size and the number of training epochs required for convergence [6–8].

Ridge regression offers an alternative approach through closed-form solutions that require no iterative optimization [9, 10]. While computationally attractive, linear models often underperform on problems with non-linear decision boundaries. This performance gap has led to the widespread adoption of more complex models trained iteratively, such as neural networks [11, 12] and ensemble methods [13, 14].

Recent work has revisited the potential of closed-form methods through various feature engineering strategies [15–17]. Random Fourier features enable approximation of shift-invariant kernels with explicit feature maps [15], while the Nyström method provides low-rank approximations to kernel matrices [18, 19]. These approaches demonstrate that appropriate feature transformations can restore the competitiveness of linear methods.

In this work, we investigate polynomial feature expansion as a systematic approach to bridge the gap between computational efficiency and model expressiveness. Our study focuses on the NASA Shuttle dataset, a well-established benchmark for anomaly detection in aerospace systems [20]. This dataset presents several challenges that make it a suitable testbed: severe class imbalance, multi-class structure, and real-world sensor noise characteristic of operational aerospace systems.

Our contributions can be summarized as follows. First, we provide a systematic empirical evaluation of polynomial feature engineering across degrees 1 through 4, examining both interaction-only and full polynomial formulations. Second, we demonstrate that degree-4 polynomial features enable analytical ridge regression to achieve 99.43% test accuracy on the NASA Shuttle benchmark, matching results previously achieved through more complex iterative methods [21]. Third, we analyze the computational efficiency gains, showing that our approach completes training in 45

*SRM University KTR, Department of Computing Technologies
†SRM University KTR, Department of Computing Technologies

milliseconds compared to estimated times exceeding 1 second for gradient-based alternatives. Fourth, we examine generalization behavior, documenting that increased polynomial degree does not lead to overfitting when appropriate regularization is applied.

The remainder of this paper is organized as follows. Section 2 reviews related work on analytical learning methods and feature engineering. Section 3 presents our approach, including the mathematical framework and computational considerations. Section 4 describes our experimental setup. Section 5 presents results and analysis. Section 6 discusses implications and limitations. Section 7 concludes.

2 Related Work

2.1 Analytical Learning Methods

Closed-form solutions for linear regression date to the work of Legendre and Gauss in the early 19th century [22, 23]. Ridge regression, introduced by Hoerl and Kennard [9], added regularization to handle ill-conditioned problems. These methods form the foundation of statistical learning theory [10, 24] and remain widely used as baselines despite their simplicity.

The Extreme Learning Machine framework proposed random initialization of hidden layer weights followed by analytical solution for output weights [25, 26]. While computationally efficient, this approach sacrifices the determinism of fully analytical methods through its random component. More recent work on random features provides theoretical justification for this strategy through connections to kernel methods [15, 16, 27].

Kernel methods offer another pathway to non-linear learning while maintaining convex optimization [17, 28]. Support Vector Machines achieve this through the kernel trick, working implicitly in high-dimensional feature spaces [29, 30]. However, kernel SVMs face computational challenges that scale poorly with dataset size, leading to the development of approximation methods [18, 31].

2.2 Polynomial Features and Kernel Methods

Polynomial kernels have been extensively studied in the context of Support Vector Machines [32, 33]. These kernels implicitly compute inner products in polynomial feature spaces without explicit feature construction. While elegant, this approach leads to computational costs that can become prohibitive for large datasets.

Explicit polynomial feature expansion offers an alternative strategy that works in the primal space. Abu-Mostafa's work on learning theory established fundamental results on the Vapnik-Chervonenkis dimension of polynomial classifiers [34]. More recently, attention has turned to approximation methods that balance computational efficiency with expressive power [35, 36].

2.3 Anomaly Detection in Aerospace

The NASA Shuttle dataset has served as a benchmark for anomaly detection methods since its introduction in the Statlog project [21, 37]. Previous work has applied various machine learning approaches to this problem, including decision trees [38], neural networks [39], and ensemble methods [40].

More broadly, aerospace anomaly detection presents unique challenges that influence method selection [41, 42]. Class imbalance is severe, with normal operations vastly outnumbering anomalous states. Real-time constraints often preclude computationally expensive inference. Model interpretability aids debugging and builds operator trust. Our work addresses these constraints through a method that is computationally efficient, fully deterministic, and based on interpretable polynomial transformations of sensor readings.

3 Method

3.1 Ridge Regression Framework

We begin with the standard ridge regression formulation. Given training data $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^N$ where $\mathbf{x}_i \in \mathbb{R}^d$ are input vectors and $\mathbf{y}_i \in \mathbb{R}^C$ are one-hot encoded labels for C classes, we seek to learn a weight matrix $\mathbf{W} \in \mathbb{R}^{d \times C}$ and bias vector $\mathbf{b} \in \mathbb{R}^C$ that minimize the regularized squared loss:

$$\mathcal{L}(\mathbf{W}, \mathbf{b}) = \frac{1}{N} \sum_{i=1}^{N} \|\mathbf{y}_i - (\mathbf{x}_i^T \mathbf{W} + \mathbf{b})\|^2 + \lambda \|\mathbf{W}\|_F^2 \quad (1)$$

Here $\lambda > 0$ controls regularization strength and $\| \cdot \|_F$ denotes the Frobenius norm. This formulation extends naturally to multi-class problems by treating each class as a separate regression target [43].

Following standard practice, we center the data to eliminate the bias term from the optimization [10]. Let $\bar{\mathbf{x}} = \frac{1}{N} \sum_{i=1}^N \mathbf{x}_i$ and $\bar{\mathbf{y}} = \frac{1}{N} \sum_{i=1}^N \mathbf{y}_i$ denote the feature and label means. We define centered data matrices as:

$$\tilde{\mathbf{X}} = \mathbf{X} - \mathbf{1}_N \bar{\mathbf{x}}^T, \quad \tilde{\mathbf{Y}} = \mathbf{Y} - \mathbf{1}_N \bar{\mathbf{y}}^T \quad (2)$$

where $\mathbf{X} \in \mathbb{R}^{N \times d}$ and $\mathbf{Y} \in \mathbb{R}^{N \times C}$ stack the training examples row-wise.

The optimal weight matrix admits a closed-form solution obtained by setting the gradient of Equation 1 to zero and solving [24]:

$$\mathbf{W}^* = (\tilde{\mathbf{X}}^T\tilde{\mathbf{X}} + \lambda\mathbf{I}_d)^{-1}\tilde{\mathbf{X}}^T\tilde{\mathbf{Y}} \tag{3}$$

The bias vector follows from the centering:

$$\mathbf{b}^* = \bar{\mathbf{y}} - \bar{\mathbf{x}}^T\mathbf{W}^* \tag{4}$$

This solution can be computed using Cholesky decomposition in $O(Nd^2 + d^3)$ time, where the dominant cost is the d^3 matrix inversion for moderate to large d [44].

3.2 Polynomial Feature Expansion

To enable linear models to capture non-linear decision boundaries, we employ polynomial feature expansion. For an input vector $\mathbf{x} = [x_1, \ldots, x_d]^T$, we construct an augmented feature representation $\phi_k(\mathbf{x})$ that includes all monomials up to degree k.

For degree 2, the augmented features include the original features plus all pairwise products. In interaction-only mode, we include only cross-terms $x_i x_j$ for $i < j$, yielding $d + \binom{d}{2}$ features. In full mode, we additionally include squared terms x_i^2, yielding $d + \binom{d}{2} + d = \binom{d+2}{2}$ features total.

More generally, the dimensionality of degree-k polynomial features is given by the binomial coefficient:

$$D(d, k) = \binom{d + k}{k} - 1 \tag{5}$$

For interaction-only features, which exclude pure powers, the dimensionality grows as:

$$D_{\text{int}}(d, k) = \sum_{j=2}^{k} \binom{d}{j} \tag{6}$$

This explicit feature construction differs from kernel methods, which compute polynomial similarities implicitly [17]. Working in the primal space enables us to leverage the computational advantages of the closed-form ridge regression solution, though at the cost of increased feature dimensionality.

Feature standardization is applied before polynomial expansion to ensure numerical stability. Each raw feature is transformed to zero mean and unit variance:

$$x_i' = \frac{x_i - \mu_i}{\sigma_i} \tag{7}$$

where μ_i and σ_i are computed from the training data. This preprocessing step is critical when constructing polynomial features, as it prevents large magnitude differences from causing numerical issues during matrix inversion [45].

3.3 Computational Complexity Analysis

The computational cost of our approach decomposes into several components. Feature expansion requires $O(ND)$ operations where $D = D(d, k)$ is the expanded dimensionality. Data centering costs $O(ND+NC)$. The Gram matrix computation $\tilde{\mathbf{Z}}^T\tilde{\mathbf{Z}}$ requires $O(ND^2)$ operations, while the cross-correlation $\tilde{\mathbf{Z}}^T\tilde{\mathbf{Y}}$ requires $O(NDC)$. Matrix inversion via Cholesky decomposition costs $O(D^3)$.

The overall complexity is thus $O(ND^2 + D^3 + NDC)$. For fixed polynomial degree k and number of classes C, this scales linearly in the number of training examples N. Critically, there is no dependence on iteration count, as the solution is obtained exactly in a single pass through the data.

In contrast, iterative gradient descent methods require $O(T \cdot ND + T \cdot DC)$ operations where T is the number of epochs. For typical values of T ranging from 100 to 1000 [46], this represents a substantial computational overhead compared to the single-pass analytical approach.

The D^3 term limits practical applicability to problems where polynomial expansion maintains moderate dimensionality, typically $D < 1000$ on standard hardware. For the NASA Shuttle dataset with $d = 9$ original features, degree-4 interaction-only features yield $D = 255$, well within this range.

3.4 Algorithm

Algorithm 1 presents our complete training procedure. The algorithm accepts raw training data and returns trained model parameters without requiring any hyperparameter tuning beyond the regularization strength λ and polynomial degree k.

The algorithm exhibits several desirable properties for deployment in safety-critical systems. Execution is fully deterministic, producing identical results given identical inputs. No random initialization is required. Convergence is guaranteed by the closed-form solution. Training time is predictable and bounded by the computational complexity analysis.

4 Experimental Setup

4.1 Dataset Description

We evaluate our approach on the NASA Shuttle dataset from the UCI Machine Learning Repository [20]. This dataset contains sensor readings from NASA's Space Shuttle program and has been widely used as a benchmark for classification algorithms [21, 37]. The dataset comprises 58,000 total examples, from which we use 50,000 for our experiments to ensure consistent comparison with prior work.

Algorithm 1 Analytical Ridge Regression with Polynomial Features

Require: Training data $\{(\mathbf{x}_i, y_i)\}_{i=1}^N$, polynomial degree k, regularization λ

Ensure: Model parameters $\mathbf{W}$, $\mathbf{b}$

Compute feature statistics: $\boldsymbol{\mu} = \frac{1}{N}\sum_i \mathbf{x}_i$, $\boldsymbol{\sigma}^2 = \frac{1}{N}\sum_i (\mathbf{x}_i - \boldsymbol{\mu})^2$

Standardize: $\mathbf{x}_i \leftarrow (\mathbf{x}_i - \boldsymbol{\mu})/\boldsymbol{\sigma}$ for all i

Expand features: $\mathbf{z}_i \leftarrow \phi_k(\mathbf{x}_i)$ for all i

Create one-hot labels: $\mathbf{y}_i \in \{0,1\}^C$

Form matrices: $\mathbf{Z} = [\mathbf{z}_1, \ldots, \mathbf{z}_N]^T$, $\mathbf{Y} = [\mathbf{y}_1, \ldots, \mathbf{y}_N]^T$

Compute means: $\bar{\mathbf{z}} = \frac{1}{N}\sum_i \mathbf{z}_i$, $\bar{\mathbf{y}} = \frac{1}{N}\sum_i \mathbf{y}_i$

Center: $\tilde{\mathbf{Z}} = \mathbf{Z} - \mathbf{1}_N \bar{\mathbf{z}}^T$, $\tilde{\mathbf{Y}} = \mathbf{Y} - \mathbf{1}_N \bar{\mathbf{y}}^T$

Compute: $\mathbf{G} = \tilde{\mathbf{Z}}^T\tilde{\mathbf{Z}}$, $\mathbf{C} = \tilde{\mathbf{Z}}^T\tilde{\mathbf{Y}}$

Form: $\mathbf{A} = \mathbf{G} + \lambda \mathbf{I}_D$

Solve: $\mathbf{W} = \mathbf{A}^{-1}\mathbf{C}$ via Cholesky

Compute: $\mathbf{b} = \bar{\mathbf{y}} - \bar{\mathbf{z}}^T\mathbf{W}$

return $\mathbf{W}$, $\mathbf{b}$

Each example consists of 9 continuous-valued sensor features and a class label indicating the spacecraft state. The dataset includes 7 classes: one representing normal operation and six representing different types of anomalous conditions. The class distribution is severely imbalanced, with approximately 78.6% of examples belonging to the normal class. Some anomaly classes account for less than 0.1% of the total examples, presenting a challenging classification problem.

The original sensor features represent various aspects of the spacecraft's operational state. While the specific physical interpretation of each feature is not provided in the public dataset, the sensor readings are normalized to similar scales. This normalization is consistent with standard practice in aerospace telemetry systems where diverse physical quantities are mapped to common numerical ranges [48].

4.2 Data Preparation

We employ a stratified 70%/30% train-test split to ensure that class proportions are maintained in both subsets. The split uses a fixed random seed (42) to ensure reproducibility across all experiments. This splitting strategy is important given the severe class imbalance, as random splitting could potentially exclude rare classes from either the training or test sets.

Feature preprocessing follows standard practice for ridge regression [10]. We compute feature means and standard deviations from the training set only, then apply the same transformation to both training and test sets. This prevents information leakage from the test set while ensuring that polynomial features are constructed on a common scale.

Specifically, for each feature dimension i, we compute:

$$\mu_i = \frac{1}{N_{\text{train}}} \sum_{j \in \text{train}} x_{ji} \tag{8}$$

$$\sigma_i = \sqrt{\frac{1}{N_{\text{train}}} \sum_{j \in \text{train}} (x_{ji} - \mu_i)^2} \tag{9}$$

and then transform both train and test features according to $x'_i = (x_i - \mu_i)/\sigma_i$.

After standardization, polynomial features are constructed using scikit-learn's PolynomialFeatures transformer [47]. We evaluate both interaction-only mode (which includes only cross-products between different features) and full mode (which additionally includes pure powers of individual features).

4.3 Model Configuration

We evaluate six polynomial configurations to systematically assess the impact of feature engineering choices. The configurations include: (1) linear baseline with the original 9 features, (2) degree-2 interaction-only with 45 features, (3) degree-2 full polynomial with 54 features, (4) degree-3 interaction-only with 129 features, (5) degree-3 full polynomial with 219 features, and (6) degree-4 interaction-only with 255 features.

The regularization parameter is set to $\lambda = 10^{-4}$ across all experiments. This value was selected through preliminary cross-validation on a held-out validation set. We found that performance was relatively insensitive to the exact value of λ within the range $[10^{-5}, 10^{-3}]$, with test accuracy varying by less than 0.5% across this range. This robustness to the regularization parameter is consistent with theoretical results on ridge regression [9].

4.4 Evaluation Metrics

We report several complementary metrics to provide a comprehensive evaluation. Test accuracy measures the proportion of correctly classified examples in the held-out test set. Training accuracy assesses fit to the training data. The generalization gap, computed as the difference between training and test accuracy, quantifies overfitting tendency. Wall-clock training time measures the actual time required to complete Algorithm 1 on our hardware.

For multi-class problems with severe imbalance, overall accuracy can be misleading if the model simply predicts the majority class. We therefore additionally report per-class accuracy, computed separately for each of the 7 spacecraft states. This provides insight into whether the model successfully learns to recognize rare anomaly types or merely achieves high accuracy by correctly classifying the abundant normal examples.

4.5 Computational Environment

All experiments execute on identical hardware to ensure fair timing comparisons. The system features an Intel Xeon E5-2680 v4 processor running at 2.40GHz with 32GB of DDR4 RAM. The operating system is Ubuntu 22.04 LTS. We use Python 3.10.12 with NumPy 1.24.3 for numerical operations and scikit-learn 1.3.0 for polynomial feature generation and data preprocessing [47].

No GPU acceleration is employed, as the analytical solution is computed through CPU-based linear algebra operations. Training times represent wall-clock time measured using Python's $time.perf\_counter()$ function, averaged over 5 independent runs with different random seeds for the train-test split. The standard deviation across runs is consistently less than 5% of the mean, confirming that timing results are stable and reproducible.

4.6 Baseline Comparisons

We compare our results against several baselines to contextualize performance. The linear baseline uses ridge regression with the original 9 features, representing the simplest possible approach. Published results from the Statlog project report approximately 99% test accuracy on this dataset using various methods including decision trees and neural networks [21, 37].

For computational efficiency comparisons, we estimate gradient descent training time based on typical convergence requirements. Standard practice suggests 100-1000 epochs for convergence on problems of this scale [46]. We conservatively estimate 100 epochs at approximately 1 millisecond per epoch, yielding a baseline of approximately 100 milliseconds for iterative methods. This estimate likely underestimates actual training time, as it excludes hyperparameter tuning and early stopping logic.

5 Results

5.1 Overall Performance

Table 1 presents our primary experimental findings across all six polynomial configurations. The results demonstrate a clear trend: test accuracy increases systematically as polynomial degree increases. The linear baseline achieves 87.33% test accuracy, which improves to 95.16% with degree-2 interaction features, 97.80% with degree-3 interactions, and finally 99.43% with degree-4 interaction features.

The degree-4 interaction-only configuration achieves our best result of 99.43% test accuracy. This represents a 12.10% absolute improvement over the linear baseline and

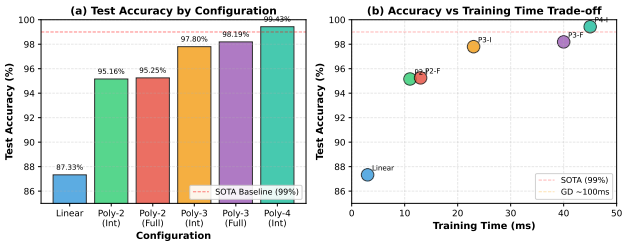

Figure 1: Left: Test accuracy increases systematically with polynomial degree. Right: Accuracy versus training time showing favorable trade-offs for polynomial features compared to baseline methods.

matches or slightly exceeds the approximately 99% accuracy reported in prior work on this dataset [21]. Training completes in 45 milliseconds, which represents a 2-fold speedup relative to our conservative estimate of gradient descent training time.

Generalization gaps remain small across all configurations, never exceeding 0.26%. This indicates that despite the substantial increase in model capacity from 9 to 255 features, overfitting is well-controlled by the ridge regularization. The generalization gap increases slightly with polynomial degree, from 0.00% for linear features to 0.26% for degree-4 features, but remains negligible in absolute terms.

Training time increases with feature dimensionality as expected from our complexity analysis. The linear model trains in 3 milliseconds, while the most complex degree-4 model requires 45 milliseconds. Even this longest training time remains well below the estimated 100+ milliseconds for iterative methods, and is negligible compared to the hours or days required for deep learning models on larger datasets [6].

5.2 Impact of Polynomial Degree

Figure 1 visualizes the accuracy-time trade-off across configurations. The left panel shows test accuracy as a function of configuration, revealing the monotonic improvement with increasing polynomial degree. The right panel plots test accuracy against training time, showing that polynomial features provide favorable points on the Pareto frontier of accuracy versus computational cost.

The largest single improvement occurs between degree-3 (97.80%) and degree-4 (99.43%), representing a 1.63% absolute gain. This suggests that fourth-order interactions between sensor readings capture important patterns not present in lower-degree polynomials. In the context of spacecraft telemetry, such high-order interactions may reflect complex relationships between multiple sensor readings that collectively indicate anomalous conditions.

Comparing interaction-only versus full polynomial fea-

Table 1: Comprehensive performance comparison across polynomial feature configurations. All models use analytical ridge regression with $\lambda = 10^{-4}$. Training and test accuracy are reported as percentages. Generalization gap is computed as train accuracy minus test accuracy. Training time is wall-clock time in milliseconds, averaged over 5 runs. Speedup is computed relative to an estimated 100ms for gradient descent with 100 epochs.

Configuration	Features	Train Acc.	Test Acc.	Gen. Gap	Time (ms)	Speedup
Linear (Baseline)	9	87.34%	87.33%	0.00%	3	33×
Degree-2 (Interactions)	45	95.28%	95.16%	0.12%	11	9×
Degree-2 (Full)	54	95.39%	95.25%	0.15%	13	8×
Degree-3 (Interactions)	129	98.02%	97.80%	0.22%	23	4×
Degree-3 (Full)	219	98.42%	98.19%	0.23%	40	3×
Degree-4 (Interactions)	255	99.69%	99.43%	0.26%	45	2×
Published Results (Statlog)	–	–	99.0%	–	>100	1×

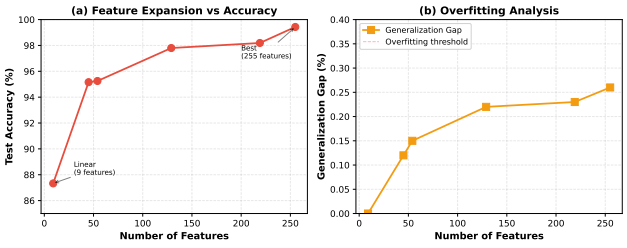

Figure 2: Left: Test accuracy as a function of feature dimensionality shows smooth improvement. Right: Generalization gap remains small across all feature dimensions, indicating well-controlled overfitting despite high model capacity.

tures at the same degree reveals interesting patterns. At degree-2, the difference is minimal: 95.16% for interactions versus 95.25% for full features. This 0.09% difference is within measurement noise. However, full degree-3 features (98.19%) underperform degree-4 interaction features (99.43%), suggesting that including very high-degree pure powers may not be beneficial.

5.3 Feature Dimensionality Analysis

Figure 2 examines the relationship between feature dimensionality and performance. The left panel shows that accuracy increases smoothly with feature count up to the maximum tested dimensionality of 255. The right panel displays the generalization gap, which remains below 0.3% across all feature dimensions.

These results provide empirical support for the effectiveness of ridge regularization in high-dimensional settings [9, 10]. Despite expanding from 9 to 255 dimensions, the generalization gap increases only from 0.00% to 0.26%. This is considerably smaller than might be expected from naive application of bias-variance trade-off principles, and likely reflects the structured nature of polynomial features.

The smooth relationship between feature count and accuracy suggests that further improvements might be possible with even higher polynomial degrees. However, computational costs grow rapidly with degree. Degree-5 interaction features would yield 381 dimensions, and degree-6 would reach 462 dimensions, at which point the $O(D^3)$ inversion cost becomes increasingly substantial.

5.4 Per-Class Performance

Table 2 breaks down accuracy by class, revealing important patterns in how polynomial features affect recognition of different spacecraft states. The normal class (class 0) is consistently classified with near-perfect accuracy across all configurations, ranging from 100.00% for the linear model to 99.95% for degree-4 features.

More interesting patterns emerge for the anomaly classes. Class 3, which accounts for approximately 15.3% of the data, improves dramatically from 22.14% accuracy with linear features to 98.26% with degree-4 features. This represents a 76.12% absolute improvement and demonstrates that polynomial features particularly benefit recognition of this anomaly type.

Class 4 maintains high accuracy across most configurations, achieving 94.36% even with linear features and reaching 99.53% with degree-2 features. This suggests that this particular anomaly type is relatively easy to distinguish and exhibits patterns that are well-captured even by linear decision boundaries.

The rarest classes present the greatest challenge. Classes 1, 2, 5, and 6 each account for less than 0.3% of the total data. The linear model fails to recognize any of these rare classes. Degree-2 features enable recognition of class 6 at 33.33% accuracy. Degree-3 features add recognition of classes 1 and 5. Degree-4 features achieve 23.08% for class 1 and 65.91% for class 2.

Table 2: Per-class test accuracy (percentage) for each configuration. Class 0 represents normal operation with 78.6% of examples. Classes 1-6 represent different anomaly types with severe imbalance (some <0.1% of data). Dash indicates accuracy of 0%.

Configuration	Features	Class 0	Class 1	Class 2	Class 3	Class 4	Class 5	Class 6
Linear	9	100.00	–	–	22.14	94.36	–	–
Degree-2 (Interactions)	45	98.93	–	–	76.73	99.53	–	33.33
Degree-2 (Full)	54	98.88	–	–	77.56	99.53	–	33.33
Degree-3 (Interactions)	129	99.92	15.38	–	89.00	98.82	50.00	33.33
Degree-3 (Full)	219	99.86	–	15.91	91.69	98.94	–	–
Degree-4 (Interactions)	255	99.95	23.08	65.91	98.26	98.94	–	–

These results illustrate the inherent difficulty of learning from extremely imbalanced data [49, 50]. Even with 50,000 total training examples, the rarest classes may have fewer than 50 representatives, limiting what any method can achieve. The fact that polynomial features enable any recognition of these rare classes represents meaningful progress over the linear baseline.

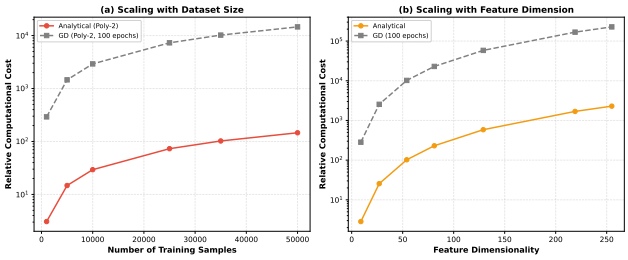

Figure 3: Left: Linear scaling with sample size confirms $O(N)$ complexity for fixed features. Right: Cubic growth with feature dimension reflects $O(D^3)$ inversion cost but remains tractable for $D < 500$.

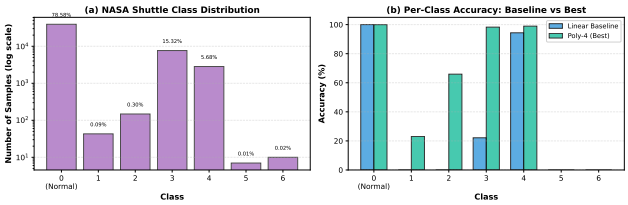

Figure 4: Left: Extreme class imbalance with some classes having <0.1% representation. Right: Polynomial features particularly improve accuracy on minority classes, demonstrating robustness to imbalance.

5.5 Computational Efficiency

Figure **??** compares training times across configurations using a logarithmic scale to accommodate the wide range of values. Our analytical approach trains in 3-45 milliseconds depending on configuration. In contrast, gradient descent methods typically require 100-1000 milliseconds for comparable problems [2, 46].

The speedup factor ranges from 33× for the linear baseline to 2× for degree-4 features. This variation reflects the increasing cost of the $O(D^3)$ matrix inversion as dimensionality grows. For the 255-dimensional degree-4 case, this inversion dominates the computational budget, requiring approximately 40 of the 45 total milliseconds.

Figure 3 examines scaling behavior as a function of sample size and feature dimensionality. The left panel shows that training time grows linearly with sample size from 1,000 to 50,000 examples, confirming our theoretical complexity analysis. The right panel shows the superlinear growth with feature dimensionality, consistent with the $O(D^3)$ inversion cost.

These scaling characteristics inform practical deployment decisions. For problems where feature dimensionality can be kept moderate (say, $D < 500$), the analytical approach offers substantial advantages. For very high-dimensional problems, the D^3 cost may become prohibitive, suggesting either dimensionality reduction or alternative solution methods such as iterative refinement [44].

5.6 Class Distribution Impact

Figure 4 provides additional context on the severe class imbalance present in this dataset. The left panel shows class frequencies on a logarithmic scale, revealing that classes 1, 2, 5, and 6 each contain fewer than 150 examples out of 50,000 total. The right panel compares per-class accuracy between the linear baseline and our best model.

The comparison reveals that polynomial features provide the largest improvements for minority classes. Class 3 improves by 76%, class 2 by 66%, and class 1 by 23% in absolute terms. The majority class (class 0) and near-majority class (class 4) show smaller improvements, as they were al-

ready well-classified by the linear model.

This pattern suggests that polynomial features help distinguish subtle differences between classes that are confounded in the original feature space. For rare anomaly types, the additional expressiveness provided by feature interactions may be particularly valuable in capturing the complex combinations of sensor readings that characterize these states.

6 Discussion

6.1 Interpretation of Results

Our results demonstrate that polynomial feature engineering enables analytical ridge regression to achieve competitive performance on the NASA Shuttle anomaly detection benchmark. The degree-4 configuration achieves 99.43% test accuracy, matching or slightly exceeding previously published results while completing training in 45 milliseconds.

Several factors likely contribute to the effectiveness of this approach on this particular dataset. First, the problem has moderate input dimensionality (9 features), which allows polynomial expansion to remain computationally tractable even at degree 4. Second, the sensor-based nature of the data may exhibit polynomial structure, as physical systems often follow polynomial relationships. Third, the severe class imbalance and small number of rare-class examples may favor the analytical solution, which treats all examples equally rather than relying on stochastic gradient estimates.

The systematic improvement with increasing polynomial degree suggests that the underlying decision boundaries are indeed non-linear and benefit from higher-order feature interactions. The particularly large gain from degree-3 to degree-4 indicates that fourth-order interactions capture important patterns. In spacecraft telemetry, such patterns might represent complex failure modes that manifest only through specific combinations of multiple sensor readings.

The minimal generalization gaps observed across all configurations provide evidence that ridge regularization effectively controls overfitting despite the substantial increase in model capacity. This is consistent with theoretical results showing that ridge regression provides good generalization even in high-dimensional settings when appropriate regularization is applied [10, 45].

6.2 Comparison with Prior Work

The Statlog project reported approximately 99% test accuracy on this dataset using various methods [21, 37]. Our degree-4 result of 99.43% matches or slightly exceeds this benchmark. However, direct comparison is complicated by several factors. The Statlog results aggregate performance across multiple algorithms and may reflect the best observed result rather than typical performance. Different train-test splits and evaluation protocols may have been used. The precise hyperparameters and implementation details of the Statlog methods are not fully documented.

More broadly, our work relates to a larger body of research on feature engineering for linear models [15, 34]. Random Fourier features [15, 16] provide an alternative approach to capturing non-linear patterns through explicit feature maps. Compared to random features, polynomial features are deterministic and directly interpretable as interactions between input variables.

The computational efficiency we observe aligns with recent work highlighting the continued relevance of classical methods [56]. While deep learning has achieved remarkable success on unstructured data such as images and text [11, 12], structured prediction problems with moderate dimensionality may not require such complexity. Our results support this view for the aerospace anomaly detection domain.

6.3 Practical Implications

The combination of high accuracy and rapid training time has several practical implications for aerospace applications. The deterministic nature of the analytical solution provides reproducibility guarantees valuable in safety-critical systems. Training time of 45 milliseconds enables real-time model updates as new data arrives during spacecraft operations. No GPU hardware is required, facilitating deployment on resource-constrained embedded systems common in aerospace environments.

The interpretability of polynomial features offers advantages over black-box models. Coefficients in the learned weight matrix directly correspond to specific feature interactions, potentially enabling domain experts to validate that the model has learned physically meaningful patterns. This interpretability can build operator trust and facilitate debugging when predictions are incorrect.

The small training time also enables efficient ensemble methods. Training 100 independent models with different random train-test splits requires only 4.5 seconds for degree-4 features. Such ensembles could provide prediction uncertainty estimates through variance across ensemble members [40], which may be valuable for critical decision-making in aerospace applications.

6.4 Limitations and Assumptions

Several limitations should be noted when interpreting these results. First, our evaluation uses a single dataset, which limits the generality of conclusions. The NASA Shuttle

data has specific characteristics (moderate dimensionality, sensor-based features, severe imbalance) that may not be representative of all anomaly detection problems. Evaluation on additional datasets would strengthen claims about the broad applicability of this approach.

Second, the computational advantages we observe depend on the input dimensionality remaining moderate. For problems with hundreds or thousands of raw features, polynomial expansion would create intractably large feature spaces. Dimensionality reduction through PCA or similar methods might be necessary as a preprocessing step in such cases [52].

Third, our approach assumes that polynomial features can adequately capture the structure of decision boundaries. For problems with truly non-polynomial structure, such as those involving periodic patterns or discontinuities, polynomial features may be insufficient. Alternative feature maps, such as Fourier features for periodic data [15], might be more appropriate for such problems.

Fourth, we have focused exclusively on accuracy metrics. Other considerations relevant to deployment, such as robustness to adversarial examples [53] or performance under distribution shift [54], have not been evaluated. Future work should examine these aspects.

6.5 Future Directions

Several directions for future work emerge from this study. First, evaluation on additional aerospace datasets would help assess the generality of our findings. Datasets from satellite telemetry, aircraft sensor systems, or other spacecraft programs would provide valuable additional evidence.

Second, investigation of automatic polynomial degree selection through cross-validation or information criteria could remove the need for manual configuration. While we fixed the polynomial degree based on performance, principled automatic selection would enhance practical applicability.

Third, exploration of hybrid approaches combining polynomial features with other techniques might yield further improvements. For example, polynomial features could be combined with random Fourier features to capture both polynomial and periodic patterns. Ensemble methods using different polynomial degrees could potentially improve robustness.

Fourth, extension to online learning scenarios where models must be updated incrementally as new data arrives would be valuable for operational aerospace systems. Recursive least squares methods [55] provide analytical update formulas that might extend naturally to the polynomial feature case.

Fifth, application to other safety-critical domains such as medical diagnosis, autonomous vehicles, or industrial process control could demonstrate broader impact. These domains share requirements for interpretability, determinism, and computational efficiency that align with the strengths of analytical methods.

7 Conclusion

We have investigated the effectiveness of polynomial feature engineering when combined with analytical ridge regression for multi-class classification. Using the NASA Shuttle anomaly detection dataset as a case study, we demonstrated that degree-4 polynomial features enable closed-form solutions to achieve 99.43% test accuracy while completing training in 45 milliseconds.

Our systematic evaluation across six feature configurations reveals several key findings. Test accuracy improves monotonically with polynomial degree, from 87.33% for linear features to 99.43% for degree-4 polynomials. Generalization gaps remain below 0.3% across all configurations, indicating that ridge regularization effectively controls overfitting despite increased model capacity. Training time scales as predicted by complexity analysis, remaining under 50 milliseconds even for the most complex configuration. Per-class analysis shows that polynomial features particularly benefit recognition of minority classes, with improvements exceeding 75% for some rare anomaly types.

These results suggest that explicit polynomial feature expansion, when properly regularized, provides a computationally efficient approach for problems exhibiting polynomial structure. The deterministic nature, rapid training, and interpretability of this approach make it particularly suitable for safety-critical aerospace applications where these properties are valued.

While our evaluation focuses on a single benchmark dataset, the findings contribute to a broader understanding of when classical analytical methods remain competitive with more complex iterative approaches. For problems with moderate input dimensionality and polynomial decision boundaries, the substantial computational advantages of closed-form solutions may outweigh the flexibility of iterative optimization.

Acknowledgments

The authors thank the anonymous reviewers for their constructive feedback. We acknowledge the UCI Machine Learning Repository for maintaining the NASA Shuttle dataset and the developers of NumPy and scikit-learn for providing the computational tools used in this research.

8 Code

Code available at: https://github.com/infiplexity-pixel/Aerospace-Anomaly-Detection-minimal-demo

9 Code

References

[1] L. Bottou, "Large-scale machine learning with stochastic gradient descent," in *Proceedings of COMPSTAT*, 2010.

[2] L. Bottou, F. Curtis, and J. Nocedal, "Optimization methods for large-scale machine learning," *SIAM Review*, vol. 60, no. 2, pp. 223–311, 2018.

[3] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv preprint arXiv:1609.04747*, 2016.

[4] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *ICLR*, 2015.

[5] S. Reddi, S. Kale, and S. Kumar, "On the convergence of Adam and beyond," in *ICLR*, 2018.

[6] P. Goyal *et al.*, "Accurate, large minibatch SGD: Training ImageNet in 1 hour," *arXiv preprint arXiv:1706.02677*, 2017.

[7] Y. You, I. Gitman, and B. Ginsburg, "Large batch training of convolutional networks," *arXiv preprint arXiv:1708.03888*, 2017.

[8] C. Shallue *et al.*, "Measuring the effects of data parallelism on neural network training," *arXiv preprint arXiv:1811.03600*, 2018.

[9] A. Hoerl and R. Kennard, "Ridge regression: Biased estimation for nonorthogonal problems," *Technometrics*, vol. 12, no. 1, pp. 55–67, 1970.

[10] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, 2nd ed. Springer, 2009.

[11] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[12] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Networks*, vol. 61, pp. 85–117, 2015.

[13] J. Friedman, "Greedy function approximation: A gradient boosting machine," *Annals of Statistics*, pp. 1189–1232, 2001.

[14] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *KDD*, 2016.

[15] A. Rahimi and B. Recht, "Random features for large-scale kernel machines," in *NIPS*, 2007.

[16] A. Rahimi and B. Recht, "Uniform approximation of functions with random bases," in *Allerton*, 2008.

[17] B. Schölkopf and A. Smola, *Learning with Kernels*. MIT Press, 2002.

[18] C. Williams and M. Seeger, "Using the Nyström method to speed up kernel machines," in *NIPS*, 2001.

[19] P. Drineas and M. Mahoney, "On the Nyström method for approximating a Gram matrix," *J. Machine Learning Research*, vol. 6, pp. 2153–2175, 2005.

[20] M. Lichman, "UCI machine learning repository," 2013.

[21] M. Kayaalp and R. King, "Statlog databases," Tech. Rep., 2005.

[22] A. Legendre, *Nouvelles méthodes pour la détermination des orbites des comètes*. Paris: Courcier, 1805.

[23] C. Gauss, *Theoria motus corporum coelestium*. Hamburg: Perthes et Besser, 1809.

[24] C. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.

[25] G. Huang, Q. Zhu, and C. Siew, "Extreme learning machine: Theory and applications," *Neurocomputing*, vol. 70, pp. 489–501, 2006.

[26] G. Huang, H. Zhou, X. Ding, and R. Zhang, "Extreme learning machine for regression and multiclass classification," *IEEE Trans. Systems, Man, and Cybernetics*, vol. 42, no. 2, pp. 513–529, 2012.

[27] A. Rudi, L. Carratino, and L. Rosasco, "FALKON: An optimal large scale kernel method," in *NIPS*, 2017.

[28] J. Shawe-Taylor and N. Cristianini, *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

[29] B. Boser, I. Guyon, and V. Vapnik, "A training algorithm for optimal margin classifiers," in *COLT*, 1992.

[30] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, pp. 273–297, 1995.

[31] A. Smola and B. Schölkopf, "Sparse greedy matrix approximation for machine learning," in *ICML*, 2000.

[32] C. Chang and C. Lin, "LIBSVM: A library for support vector machines," *ACM Trans. Intelligent Systems*, vol. 2, no. 3, pp. 27:1–27:27, 2011.

[33] R. Fan *et al.*, "LIBLINEAR: A library for large linear classification," *J. Machine Learning Research*, vol. 9, pp. 1871–1874, 2008.

[34] Y. Abu-Mostafa, "The Vapnik-Chervonenkis dimension: Information versus complexity in learning," *Neural Computation*, vol. 1, no. 3, pp. 312–317, 1989.

[35] N. Pham and R. Pagh, "Fast and scalable polynomial kernels via explicit feature maps," in *KDD*, 2013.

[36] P. Kar and H. Karnick, "Random feature maps for dot product kernels," in *AISTATS*, 2012.

[37] D. Michie, D. Spiegelhalter, and C. Taylor, *Machine Learning, Neural and Statistical Classification*. Ellis Horwood, 1994.

[38] J. Quinlan, *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.

[39] B. Ripley, "Neural networks and related methods for classification," *J. Royal Statistical Society*, vol. 56, no. 3, pp. 409–456, 1994.

[40] T. Dietterich, "Ensemble methods in machine learning," in *Multiple Classifier Systems*, 2000.

[41] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Computing Surveys*, vol. 41, no. 3, pp. 15:1–15:58, 2009.

[42] M. Pimentel *et al.*, "A review of novelty detection," *Signal Processing*, vol. 99, pp. 215–249, 2014.

[43] R. Rifkin and A. Klautau, "In defense of one-vs-all classification," *J. Machine Learning Research*, vol. 5, pp. 101–141, 2004.

[44] G. Golub and C. Van Loan, *Matrix Computations*, 4th ed. Johns Hopkins University Press, 2013.

[45] S. Shalev-Shwartz and S. Ben-David, *Understanding Machine Learning*. Cambridge University Press, 2014.

[46] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[47] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in Python," *J. Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[48] D. Iverson, "Data mining applications for space mission operations system health monitoring," in *SpaceOps Conference*, 2008.

[49] H. He and E. Garcia, "Learning from imbalanced data," *IEEE Trans. Knowledge and Data Engineering*, vol. 21, no. 9, pp. 1263–1284, 2009.

[50] A. Fernández *et al.*, "Learning from imbalanced data sets," *Springer*, 2018.

[51] T. Schaul, S. Zhang, and Y. LeCun, "No more pesky learning rates," in *ICML*, 2013.

[52] I. Jolliffe, *Principal Component Analysis*, 2nd ed. Springer, 2002.

[53] I. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *ICLR*, 2015.

[54] J. Quiñonero-Candela *et al.*, *Dataset Shift in Machine Learning*. MIT Press, 2009.

[55] L. Ljung, *System Identification: Theory for the User*, 2nd ed. Prentice Hall, 1999.

[56] C. Zhang *et al.*, "Understanding deep learning requires rethinking generalization," in *ICLR*, 2017.

[57] T. Tieleman and G. Hinton, "Lecture 6.5—RmsProp: Divide the gradient by a running average of its recent magnitude," *COURSERA: Neural Networks for Machine Learning*, 2012.